모바일 응용 S.W GUI의 자동화 테스트 및 관리를 위한 도구 설계 및 구현

채현철*, 이정주*, 황선명*, 정양재**
*대전대학교 컴퓨터공학과
**한국전자통신연구원 임베디드 S/W연구단
e-mail:chaehc@gmail.com, cornor@etri.re.kr

Design & Implementation for Automatic Test & Management of Mobile Application S.W GUI

Hyeonl-Cheol Chae *, Jeong-Joo Lee*, Sun-Myung Hwang* ,Yang-Jae Jung
*Dept of Computer Engineering, Dae-Jeon University
Embedded S.W Research Division, ETRI

요 약

현재 업계에서 사용하고 있는 GUI 테스트 방법은 개발자 혹은 테스터가 매뉴얼을 보며 화면의 변화를 관찰할 뿐이다. 이 방법은 정확한 비교를 할 수 없을 뿐만 아니라 테스트에 걸리는 시간이 매우 오래 걸린다. 생명주기가 짧은 모바일 시장에서는 빠른 출시가 기업의 흥망을 좌우하기 때문이다. 본 논무에서는 이러한 문제를 해결하기 위하여 GUI 테스트 자동화 지원기를 제시하고자 한다.

1. 서론

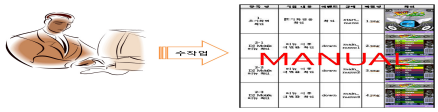
모바일 어플리케이션 소프트웨어의 시장은 여러 소프트웨어 시장 중에서 가장 많은 어플리케이션 소프트웨어를 출시하고 있다. 모바일 소프트웨어는 짧은 개발주기와 짧은 생명주기로 인해 어느 누구보다 먼저 어플리케이션 소프트웨어를 출시해야 소비자를 확보하는 전쟁 아닌 전쟁을 치르고 있다. 모바일 어플리케이션에서 가장 많은 사용자를 확보하고 있는 컨텐츠 분야는 게임, 이미지 등 많은 컨텐츠로 구성 되어져 있다. 이러한 컨텐츠들은 GUI를 기본으로 하여 사용자와의 정보를 교환하는 역할을 한다.

GUI는 Graphics User Interface의 약자로서 사용자가 그래픽을 통해 정보를 교환하고자 하는 대상과 정보를 교환하는 작업환경을 일컫는다. 나날이 변화하는 모바일 어플리케이션 소프트웨어의 GUI는 중요 요소로 자리 잡았고, 모바일 어플리케이션의 중요 테스트 대상으로 보여지고 있다. 하지만 현재 모바일 업체들에서 하는 테스트 방법으로는 테스터가 단계별로 매뉴얼을 보며 변환되는 화면을 체크하는 수동적이고 부정확한 방법으로 테스트를 수행하고 있다.

따라서 본 논문에서는 효율적인 GUI 테스트를 지원하는 자동화 도구의 설계 기법을 제시하고자 한다.

2. 기존 모바일 S.W GUI 테스트 방법

현재 업계에서 사용하고 있는 GUI 테스트 방법은 개발자 혹은 테스터가 매뉴얼을 보며 화면의 변화를 관찰할 뿐이다. 그러나 사람의 눈으로 테스트를 할 때에는 한계가 따른다. 정확한 비교를 할 수 없을 뿐만 아니라 테스트에 걸리는 시간이 매우 오래 걸린다. 생명주기가 짧은 모바일 시장에서는 빠른 출시가 기업의 흥망을 좌우하기 때문이다. (그림 1)은 기존의 모바일 GUI 테스트 방법이다.

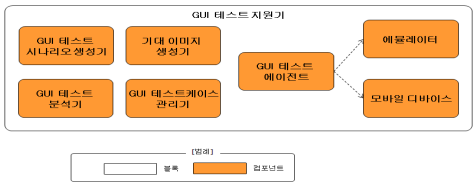


(그림 1) 기존 매뉴얼 방식의 테스팅

3. GUI 테스트 지원기

GUI 테스트 지원기는 모바일 S.W의 GUI를 테스트 하기 위한 것으로 시나리오에 따라 테스트 케이스를 정의한 후 에뮬레이터 또는 모바일 디바이스에서 동작하는 모바일 S.W를 동작시켜 테스트케이스의 내용과 비교하여 GUI를 테스트 한다.

3.1 GUI 테스트 지원기 구조와 기능



(그림 2) GUI 테스트 지원기 구성도

본 연구는 지식경제부 및 정보통신연구진흥원의 IT신성장동력핵심 기술개발사업의 일환으로 수행하였음. [2007-S-032-02, 다중 플랫폼 지원 모바일 응용 S/W 개발환경 기술 개발]

(그림 2)은 GUI 테스트 지원기 블록의 구조이다. 블록을 구성하는 컴포넌트의 기능은 다음과 같다.

■ **GUI 테스트 시나리오 생성기** : GUI를 테스트 위한 시나리오를 작성한다. 테스트 시나리오에는 테스트 개요, 테스트 목적, 주의 사항과 함께 . GUI 테스트케이스로 구성된다.

■ **기대이미지 생성기** : GUI 테스트케이스에 포함되는 기대이미지를 만든다.

■ **GUI 테스트 분석기** : GUI 테스트케이스의 기대이미지와 실행 결과 화면을 분석한다.

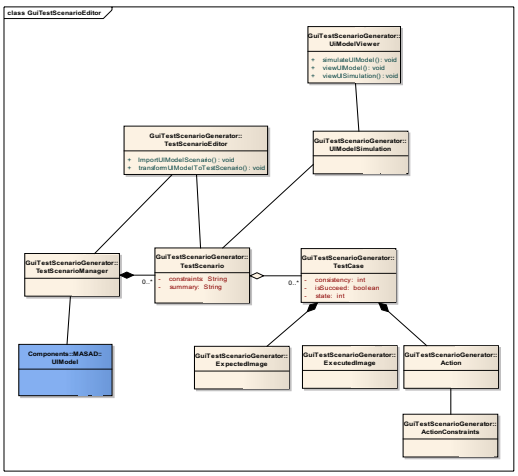
■ **GUI 테스트케이스 관리기** : GUI 테스트 시나리오와 GUI테스트케이스를 관리한다. 에뮬레이터 또는 모바일 디바이스에서 동작하는 모바일 SW와 연결하여 SW를 실행시키고 실행 결과를 입력받는다.

■ **GUI 테스트 에이전트** : 모바일 SW를 GUI 테스트케이스에 따라 실행하도록 한다. 모바일 SW와 함께 빌드하여 에뮬레이터 또는 모바일 디바이스 상에서 실행된다.GUI 테스트케이스 관리기에서 보내는 이벤트를 받아서 모바일 SW를 동작시킨다.

3.2 GUI 테스트 시나리오 생성기

GUI 테스트 시나리오 생성기는 GUI 테스트를 위한 테스트 시나리오를 생성하도록 하는 컴포넌트이다.

테스터가 GUI 테스트를 위해 실행경로를 수행하기 위한 이벤트 순서, 예상화면 등을 만드는 작업은 거의 불가능한 작업니다. 이에 대한 해결책으로 설계도구에서 만든 UI 모델을 이용한다. (그림 2)는 시나리오 생성기의 클래스 다이어그램이다.

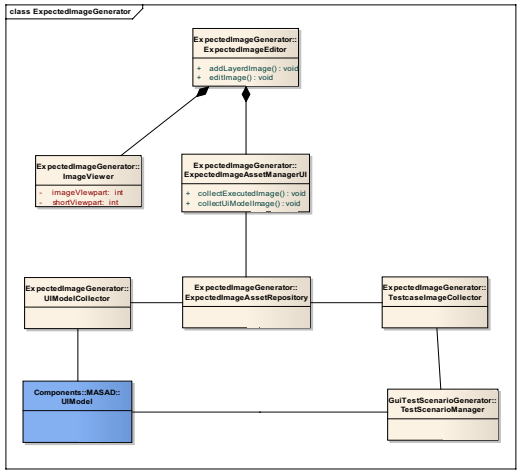


(그림 3) GUI 테스트 시나리오 생성기 클래스 다이어그램

시나리오에는 테스트 개요, 목적, 주의 사항과 함께 GUI 테스트케이스로 구성된다.

3.3 기대이미지 생성기

기대이미지 생성기는 GUI 테스트케이스에 포함되는 이개이미지를 만든다. 기대이미지는 모바일 S.S 실행 이후 결과 화면과 비교하여 GUI 테스트 성공 여부를 결정하는데 사용된다.



(그림 4) 기대이미지 생성기의 클래스 다이어그램

■ ExpectedImageAssetManagerUI : 후보 기대이미지를 관리하기 위한 관리기 화면이다. 기대이미지 자산리파지토리 내용을 사용자에게 보여주고 선택할 수 있도록 한다.

■ ExpectedImageAssetRepository : 기대이미지 자산을 관리한다. 기대이미지 자산은 UI모델로부터 정의한 화면과 실행화면, 기존에 만든 기대이미지들로 구성된다.

■ ExpectedImageEditor : 기대이미지를 생성하기 위한 메인화면이다. 기대이미지를 편집하기 위한 화면과 쇼컷화면, 기대이미지 후보 리스트로 구성된다.

■ ImageViewer : 기대이미지를 편집하기 위한 화면으로 편집창과 쇼컷화면으로 구성되어 있다. 편집기능은 레이어 기능으로 여러 이미지를 조합하여 기대이미지를 만든다.

3.4 GUI 테스트 분석기

GUI 테스트 분석기는 GUI 테스트케이스의 기대이미지와 실행 결과 화면을 분석한다.

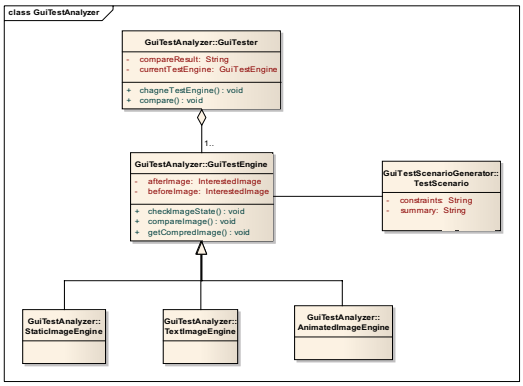
■ AnimatedImageEngine : 동적으로 움직이는 화면을 비교하기 위한 비교엔진이다. 동적 화면 비교는 여러 정지화면들의 리스트로 이루어지며 많은 비교 화면으로 테스트 시간에 영향을 준다. 옵션에 따라 비교 횟수를 조절하여 테스트 시간을 조절한다.

■ GuiTestEngine : 테스트엔진의 상위클래스. 화면의 형태에 따라 테스트엔진을 특화하여 테스트시간과 테스트성능을 향상시킨다. 이미지 수행 상태를 확인하고 2개의 이미지를 모은 후 비교한다.

■ GuiTester : GUI 분석을 테스트엔진을 관리하고 기대 화면과 실제 화면을 비교한다.

■ StaticImageEngine : 일반적인 정적인 화면을 비교하기 위한 테스트엔진이다. 비트맵방식으로 이미지를 비교한다.

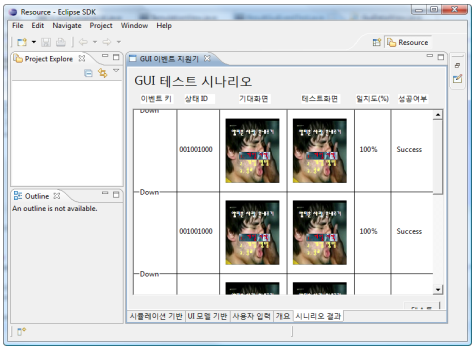
■ TextImageEngine : 화면 중에서 텍스트로 이루어진 부분을 비교하기 위한 비교 엔진이다. 텍스트의 내용을 추출하여 비교한다.



(그림 5) GUI 테스트 분석기의 클래스 다이어그램

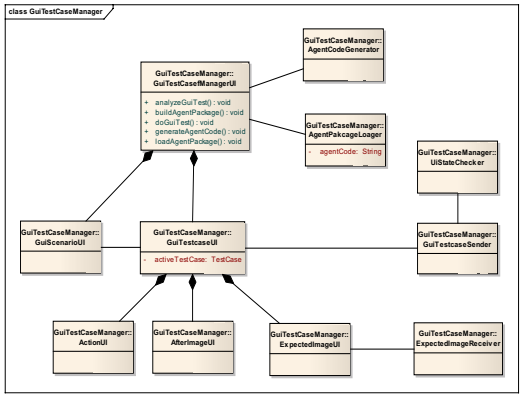
3.5 GUI 테스트 관리기

GUI 테스트케이스 관리기는 GUI 테스트 시나리오를 편집할 수 있는 화면을 제공하고 모바일 S.W에 이벤트를 보내고 실행 결과 화면을 수집한다.



(그림 6) GUI 테스트케이스 관리기 화면

또한 GUI 테스트 시나리오와 GUI 테스트케이스를 관리한다. 에뮬레이터 또는 모바일 디바이스에서 동작하는 모바일 S.W와 연결하여 S.W를 실행시키고 실행결과를 입력받는다.



(그림 7) GUI 테스트케이스 관리기의 클래스 다이어그램

■ ActionUI : 데이터입력, 사용자의 액션을 추가하기 위한 화면을 제공한다.

■ AfterImageUI : 이벤트 발생 이후에 예상되는 화면을 등록하기 위한 인터페이스를 제공한다.

■ AgentCodeGenerator : 모바일 SW에 이벤트를 보내고 실행결과 화면을 받기 위한 에이전트 코드를 생성한다.

■ AgentPakcageLoager : 에이전트 코드와 개발자가 개발한 모바일 SW를 함께 빌드하여 에뮬레이터 또는 모바일디바이스에 탑재한다.

■ ExpectedImageReceiver : 이벤트 이후 모바일 SW의 화면 결과를 전송받는다.

■ ExpectedImageUI : 이벤트 발생 이후 예상되는 화면을 편집하기 위한 인터페이스를 제공한다.

■ GuiScenarioUI : GUI 테스트를 위한 테스트 시나리오를 편집하기 위한 화면을 제공한다. GUI 테스트케이스 생성기에서는 테스트 시나리오를 위한 자료구조를 만든다. GuiScenarioUI 클래스는 자료구조를 읽어서 사용자가 사나리오를 편집할 수 있도록 한다.

■ GuiTestCasefManagerUI : GUI 테스트를 수행하기 위한 화면을 제공한다. GUI 테스트 시나리오를 보여주고 그에 따른 GUI 테스트케이스를 편집할 수 있는 화면을 제공한다. 실행 버튼을 누르면 자동으로 모바일 SW와 연결하여 테스트 결과를 사용자에게 제공한다.

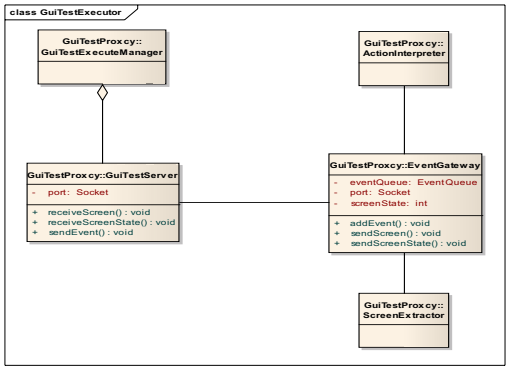
■ GuiTestcaseUI : GUI 테스트 시나리오에 포함되는 테스트케이스를 편집하기 위한 화면을 제공한다.

■ UiStateChecker : 이벤트 수행을 위한 조건을 확인하여 GUI 테스트 시나리오를 수행하는 도중 오류 상황이 발생할 경우 테스트를 중단한다.

3.6 GUI 테스트 에이전트

GUI 테스트 에이전트는 모바일 SW를 GUI 테스트케이스에 따라 실행하도록 한다. 모바일 SW와 함께 빌드하여 에뮬레이터 또는 모바일 디바이스 상에서 실행된다. GuiTestCaseManager에서 보내는 이벤트를 받아서 모바

일 SW를 동작시킨다.



(그림 8) GUI 테스트 에이전트의 클래스 다이어그램

■ ActionInterpreter : Data 입력이나 사용자 액션을 해석하여 플랫폼에서 지원하는 이벤트로 변경한다.

■EventGateway : Socket을 통해 GUI 테스트시나리오를 수행하기 위한 이벤트를 받고 실행 결과 화면을 보낸다. 데이터입력 등의 액션이 전달될 경우 액션을 분석하여 모바일플랫폼에서 사용하는 이벤트로 변경해준다.

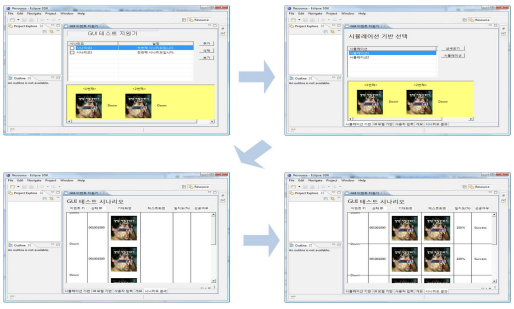
■ GuiTestExecuteManager : GUI테스트지원기와 에뮬레이터 또는 모바일디바이스에서 동작하는 모바일 SW의 연결부분을 관리한다. GUI테스트 시나리오에 따른 이벤트를 소켓을 통해 모바일 SW에 보낸다.

■ GuiTestServer : Socket을 통해 모바일SW에 이벤트와 실행화면을 주고받는다.

■ ScreenExtractor : 에뮬레이터 또는 모바일 기기에서 동작하는 SW의 실행 결과를 수집한다.

4. GUI 테스트 도구 수행 과정 및 결과 출력

(그림 8)은 3장에서 설계한 정보를 바탕으로 실제 GUI 테스트 지원기를 구현 한 것이다. 설계 도구에서 UI 시뮬레이션을 위하여 만든 정보를 활용하여 GUI 테스트 시나리오를 만드는 과정이며 또한 UI 시뮬레이션으로 만든 시나리오를 테스트하고 결과를 출력하기까지의 과정이다.



(그림 9) 모바일 GUI 테스트 과정 및 결과

기존에 저장되어 있던 시나리오나 새로 작성한 시나리오를 선택하면 해당 시나리오의 이벤트 및 화면의 흐름을 보여준다. 이 시나리오를 테스트 하면 에뮬레이터가 자동으로 실행된다. 이때 실행되는 각종 정보들을 서버로 다운받아 분석한 후이벤트, 이벤트에 따른 화면 변화, 원본 이미지, 그리고 테스트한 결과를 퍼센트(%)로 표시해주며 성공유무를 출력하게 된다. 또한 결과를 저장하여 추후에 재사용이 가능하도록 하였다.

5. 결론 및 향후과제

모바일 업계에서 가장 많은 방법으로 사용되고 있는 베타 테스트 방법은 많은 예산과 시간을 투자하여 전문성이 인증되지 않은 비인증 테스트들로서 테스트들을 대상으로 시행하는 베타 테스트보다 짧은 개발주기와 생명주기에 맞는 자동화 도구가 더 좋은 방법이라 생각된다.

가장 많은 사용자를 확보하고 있는 이동통신사가 제공하는 컨텐츠형 어플리케이션 소프트웨어는 컨텐츠들이 더욱더 GUI화 되리라 보며 앞으로 더욱더 발전하는 GUI시장에 대비하여 시장성을 확보하기위해서 안정성이 확보된 GUI 테스트가 필요하리라 생각된다.

앞으로 본 논문에서는 정적방식의 테스트뿐만 아니라 동적 방식의 테스트할 필요가 있다. 또한 GUI 테스트 지원기의 UI를 사용자 중심으로 바꾸어 초보테스터라도 쉽고 간편하게 테스트를 할 수 있도록 해야 할 것이다.

마지막으로 현재는 에뮬레이터만 사용하고 있지만 실제 모바일 디바이스에도 적용해 연구해야겠다.

참고문헌

[1] Pressman, R., Software Engineering: A Practitioner's Approach, McGraw-Hill, 2003.

[2] Patton, R., Software Testing, Sams, 2000.

[3] NIST, "The Economic Impacts of Inadequate Infra-structure for Software Testing", 2002.5

[4] 한국정보통신기술협회, 소프트웨어테스트 전문기술 응용분야, 한국정보통신기술협회, 2005.

[5] SQA G. S/W Center, "Manual vs. Automated Test 에 대한 사례 연구 소개", 2003

[6] 권원일. "모바일 소프트웨어 테스팅 현황과 표준적인 테스트 케이스".

[7] 이정규. "Record-Playback 기술 기반의 GUI 테스트 케이스 자동생성, 한국컴퓨터종합학술대회 논문집, 2007

[8] Binder, R.V., "Testing Object-Oriented Systems: A Status Report," American Programmer, vol.7, no.4, April 1994.

[9] NIST, "The Economic Impacts of Inadequate Infra-structure for Software Testing", 2002.5