호환성 테스트의 안정화 및 시간 단축을 위한 자동화 도구 (한컴 오피스)

김준기*, 최윤석**
*상명대학교 컴퓨터과학과
**한글과컴퓨터 선임연구원
e-mail: yajungi@hanmail.net

Automation tools for stabilizing compatibility testing and reducing time (Hancom office)

Jun-Gi Kim*, Yoon-Seok Choi**
*Computer Science, Sang-myung University
**Hancom

요 약

사람이 만드는 소프트웨어는 개발을 시작함과 동시에 오류를 만들어내기 시작한다. 간단히 생각해보면 소프트웨어를 개발하는 사람이 이러한 오류를 잘 알 수 있을 것이라고 생각하지만 개발자 스스로가 자신의 소프트웨어의 오류를 찾아 판단하는 것은 매우 어려운 일이다. 그렇기에 소프트웨어의 테스터가 따로 존재하게 되는데 소프트웨어 테스트의 방법은 크게 화이트박스 테스트와 블랙박스 테스트로 나누어 볼 수 있다. 호환성 도구는 블랙박스 테스트를 기반으로 호환성 테스트를 수행하며 자동화를 결합시켜 사람이 호환성 테스트를 수행할 때의 물리적인 시간의 한계를 극복하는데 목적을 갖고 있다. 목적에 따라 호환성 테스트를 위한 적절한 테스트 샘플을 제작한 후 다양한 테스트 케이스를 통해 호환성 테스트를 수행한 뒤 수행한 결과를 바탕으로 사람이 호환성 테스트를 진행할 때와 비교하여 시간적 효율성과 오차 범위를 줄임으로써 신뢰도를 증가시키고 이를 통해 호환성 도구의 유용함을 밝히고자 한다.

1. 서론

사람이 만드는 소프트웨어는 개발을 시작함과 동시에 오류를 생성하게 된다. 이러한 오류는 예측할 수 없을 정도로 끊임없이 생성되며 오류를 발견하는 일조차 쉬운 일이 아니다. 발생한 오류는 개발자가 쉽게 발견할 것 같지만 개발자는 자신만의 목적을 갖고 목적에 해당하는 테스트를 진행하기 때문에 오류가 발생하지 않을 입력 데이터 범위 내에서 테스트를 수행하게 되고 이는 결국 올바른 테스트를 진행했다고 할 수가 없게 된다.

이러한 이유로 소프트웨어 테스트를 위해 테스터가 별도로 존재해야 하며 이러한 테스터들은 소프트웨어 테스트를 위해 내부 코드를 확인하는 화이트박스 테스트와 기능의 유효성을 확인하는 블랙박스 테스트 크게 두 가지를 활용한다.

두 가지 테스트 방법 중에 호환성 도구는 한컴 Office를 기반으로 한컴 Office의 호환성 기능에 대한 점검을 위한 도구로서 블랙박스 테스트를 진행한다.

테스트를 위해 호환성 샘플을 작성하였으며 한컴 Office의 제품군에 따라 한글, 한셀, 한쇼, 한워드로 분리하여 호환성 테스트를 진행하도록 한다.

호환성 도구의 호환성 테스트는 사람이 호환성 테스트를 진행할 때와는 달리 평균적인 오차 범위를 줄일 수 있다는 장점을 갖고 있으며 최종적으로는 물리적인 시간의 한계를 극복하는데 목적을 두고 있다.

2. 호환성 도구의 자동화

2.1 Autoit

Autoit은 일반 스크립팅을 자동화하기 위해 설계된 프리웨어 BASIC 계열의 스크립팅 언어이다. 이 언어는 모든 표준 Windows 컨트롤과 상호작용할 수 있는 장점이 있다.

2.2 자동화 도구의 활용

현재 소프트웨어 테스터들은 소프트웨어의 테스트에는 수많은 경우의 수가 존재하며 이를 모두 테스트하기에 물리적인 시간의 한계에 부딪히기 때문에 자동화 테스트를 많이 활용하고 있다. 반복적인 업무가 많은 소프트웨어 테스트를 자동화 도구를 통해 수행함으로써 사람이 테스트를 수행할 때와 달리 물리적인 시간과 오차 범위가 크게 감소한다는 결과를 얻을 수 있기에 최근 자동화 도구의 활용에 관한 연구가 활발히 진행되고 있다.

3. 호환성 도구의 구성

3.1 호환성 도구의 설계

호환성 도구를 설계하는데 있어 테스트를 수행하는 도중에 생기는 오류에 대한 처리를 하는 것이 중요하다.

한컴 Office를 실행하는데 있어 각 제품군마다 조금씩 차

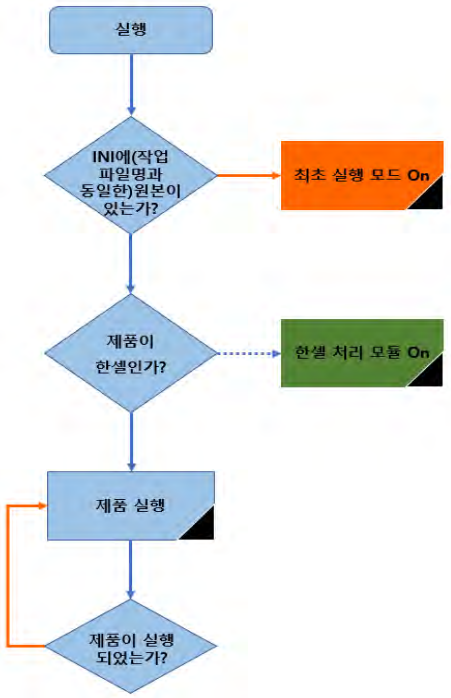
이가 존재하는데 이러한 차이는 테스트를 수행하는데 다른 작동을 유발할 수 있다. 이러한 이유로 호환성 도구는 제품군에 따른 구분된 설계를 진행하였고 이에 따라 호환성 샘플도 제품군에 따라 다르게 생성하여 각각 호환성 테스트를 수행할 수 있도록 하였다.

다음은 한컴 Office 제품군의 특징을 비교한 표이다.

<표1> 한컴 Office 제품군의 특징

제품	특징
한글	.hwp 확장자를 사용하며 cursor가 존재 페이지 구분
한셀	.cell 확장자를 사용하며 cursor가 존재하지 않고 페이지 구분 없음, sheet를 통한 구분
한쇼	.show 확장자를 사용하며 초기 알림 창 생성, cursor가 존재하지 않고 슬라이드 구분
한워드	.hwdt 확장자를 사용하며 한글과 비슷한 특징을 가짐

다음과 같은 특징을 참고하여 호환성 도구가 제품군에 따라 다르게 작동하지만 하나의 호환성 도구로 완성하기 위하여 통합 작업을 진행하였다.



(그림 1) 호환성 도구 제품 선택 (예: 한셀) 및 오류 처리 과정

호환성 도구를 실행하면서 중복되는 정보를 방지하기 위해 기존에 호환성 검사가 진행되었던 것인지 확인하는 절차가 필요하다. 이러한 절차 확인 후에 한컴 Office의 제품군을 구별할 필요성이 있는데 이는 한컴 Office도 제품에 따라 작동하는 기능이 다르고 확인해야 할 호환성 문제도 제각각이기 때문이다.

중복 여부와 제품 선택에 대한 확인이 끝난 후에 제품을 실행하여 제품 실행 시 호환성 도구의 오작동을 방지하기 위하여 오류 처리를 해주는 작업을 진행하였다.

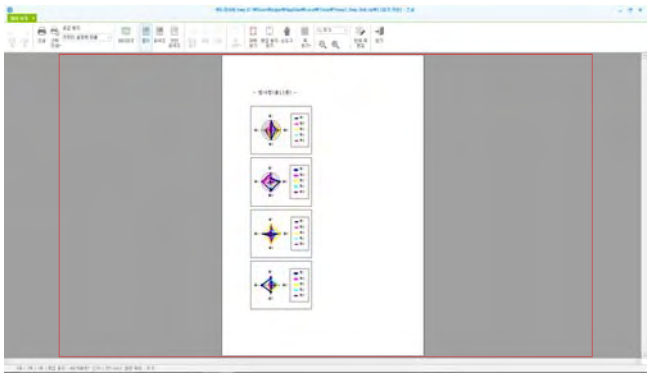
호환성 테스트는 이미지 비교를 통해 진행하도록 하였다. Autoit의 Pixelchecksum 기능을 이용하여 반복적인 테스트를 진행하면서 이미지의 값 변화를 통해 결과를 얻는 방법이다. 이미지 값의 변화를 4개의 제품군 모두 동일한 상태에서 진행할 수 있도록 미리 보기 모드를 활용하였다.

<표2> 한컴 Office 제품군별 문서 진입 상태와 미리 보기 모드의 특징

제품	문서 상태	미리 보기 모드
한글	Cursor 존재	4개의 제품군 모두 동일한 상태
한셀	Cursor 없음, 페이지 분할 없음	
한쇼	프레젠테이션 선택 창 존재, Cursor 없음	
한워드	Cursor 존재	

Pixelchecksum 기능은 cursor의 깜박임이나 미세한 차이에 따라 값이 달라질 수 있어 호환성 검사에 오류를 초래할 수 있다. 그러므로 호환성 검사를 위해 사용하는 이미지는 동일한 조건 내에서 미리 보기 모드에서의 페이지 쪽 수를 기준으로 각 이미지마다 배열에 데이터를 담도록 하였으며 배열의 비교를 통해 제품의 호환성 기능이 제대로 작동하고 있는지 확인하도록 한다.

미리 보기 모드 진입 후 문서 상태를 기록하는 방식은 다음과 같다.



(그림 2) Pixelchecksum 위치 지정

Pixelchecksum에서 빨간색 테두리와 같이 고정된 위치를 지정시킨다. 빨간색 테두리 외의 범위의 이미지 상태는 문서 상태 외에도 문서 상태 표시줄, 문서 제목 등이 기존의 문서 상태와 다른 값을 측정할 수 있으므로 호환성 테스트에 필요한 문서 상태만 기록하기 위해 설정한 것이다.

3.2 호환성 도구의 Data관리

Pixelchecksum을 통해 측정된 값은 ini 파일을 통하여 기록되는데 이는 스크린 샷을 찍어 이미지를 비교하는 방식과는 달리 메모장 형식의 적은 용량으로 기록, 보관할 수 있다는데 용이하다. 이러한 편의성 때문에 이전의 호환

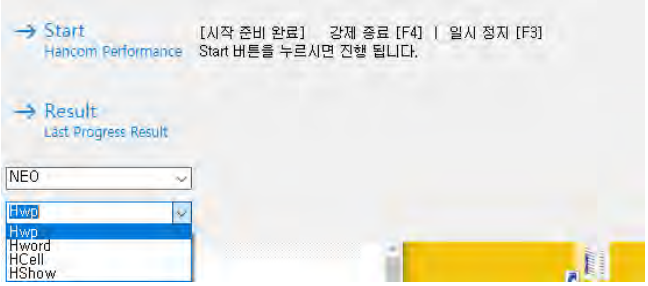
성 검사에 대한 Data를 관리하는데 편리하다는 장점을 갖고 있다.

(그림 3) 호환성 테스트 (표 스타일) 이후 ini파일 자동 기록

3.3 호환성 도구 UI

호환성 도구의 UI는 기본적으로 Start와 이전 기록 결과에 대해 확인할 수 있는 Result 버튼을 추가하였으며 호환성 검사를 하고자 하는 제품을 선택할 수 있도록 설정하였다.

호환성 도구의 UI는 다음과 같다.



(그림 4) 호환성 도구 UI

또한 호환성 도구를 실행하고 난 뒤 이전의 결과를 확인할 수 있는 Result를 통해 호환성 검사 결과를 확인하고 어떠한 문제점이 발생하였는지 (그림 4)와 같이 확인할 수 있도록 하였다.

(그림 5) 호환성 도구 Result

호환성 도구 사용 결과가 예상 결과와는 달리 (그림 4)에서 Fail로 처리되는 경우 추가적인 업데이트를 진행할 수 있도록 하여 올바른 결과를 이끌어 낼 수 있도록 한다. 자동화 테스트에서 발생할 수 있는 검사 결과에 대한 오차 범위를 줄여나갈 수 있다.

4. 호환성 도구를 통한 테스트 결과

4.1 호환성 테스트의 시간 단축

사람이 수작업으로 테스트를 진행하는 경우와 사람이 호환성 도구를 사용하여 테스트를 진행하는 경우 각각 사람이 호환성 테스트에 투입하는 시간을 비교해보았다.

일반적으로 1개의 호환성 샘플(50page)을 갖고 호환성 테스트를 진행하는 경우 사람은 개개인의 능력에 따라 1페이지 당 20초에서 1분을 소요하여 테스트를 완료하는데 약 40분을 소요하고 테스트의 결과를 확인하려면 그 이상의 시간을 소요하게 된다. 반면 호환성 도구를 사용하면 한 개의 샘플을 테스트하는데 약 2분을 소요하고 사람은 호환성 도구가 테스트한 결과만 확인하면 되기 때문에 호환성 샘플 1개의 결과를 확인하는데 약 30초의 시간이면 충분하다. 이를 통해 수작업으로 호환성 테스트를 진행하는 경우와 호환성 도구를 사용하여 호환성 테스트에 사람이 투입하는 평균 시간을 그래프로 나타내면 다음과 같다.

(그림 6) 호환성 도구를 사용하는 경우 사람의 투입시간 비교 그래프

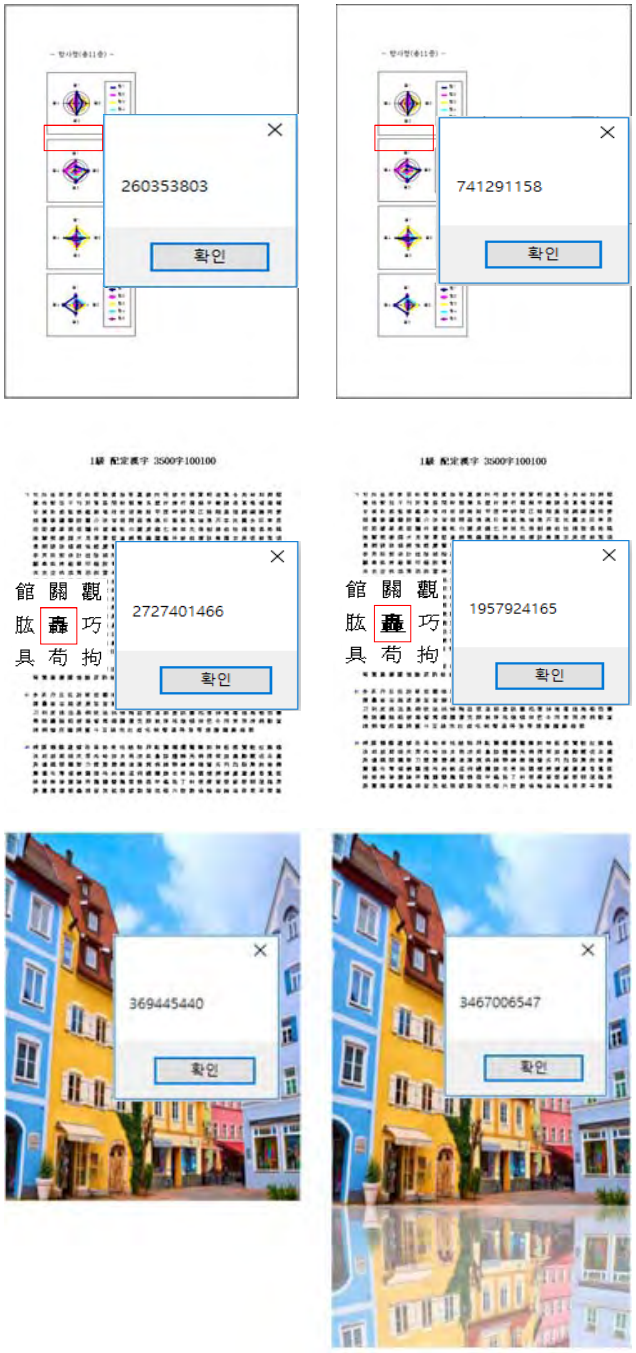
수작업으로 진행하는 호환성 테스트는 샘플 100개를 테스트하는데 걸리는 시간이 약 3일 정도의 시간을 필요로 하는데 호환성 도구를 통해 호환성 테스트를 진행할 경우 실제 사람이 사용하게 되는 시간은 약 1시간 정도의 시간을 소요함으로써 많은 양의 물리적인 시간을 단축할 수 있다.

4.2 호환성 테스트의 오차범위 감소

사람의 수작업으로 진행되는 호환성 테스트의 경우 전반적인 모든 테스트를 사람 개개인의 능력에 따라 검사 결과의 신뢰도가 달라지기도 하며 이에 따른 호환성 테스트의 전체적인 오차 범위는 커질 수밖에 없다.

하지만 호환성 도구를 통한 테스트는 호환성 테스트를 진행할 샘플의 이미지를 값으로 기록하여 개개인의 테스트 결과의 오차 범위를 크게 줄일 수 있었고 신뢰도의 오차 범위가 상당히 줄어드는 것을 확인할 수 있었다.

다음은 호환성 도구의 자동화 테스트의 일부 이미지이다. 이미지의 일부가 약간이라도 수정될 경우 다른 값을 나타내기 때문에 호환성 테스트를 진행하는데 있어서 전체적인 오차 범위를 줄이는데 효과적인 방법이다. 이미지를 비교할 수 있는 값에 대한 정확성은 입증할 수 있었으나 다른 이슈를 불러일으킬 수 있기 때문에 오류 처리를 보완하고 있다.



(그림 7) 호환성 도구의 호환성 테스트 (방사형, 한자, 이미지) 측정 값

5. 결론

호환성 도구를 통한 자동화 테스트를 구현하면서 전반적으로 물리적인 시간의 단축은 큰 효과를 거두었다. 하지만 모든 소프트웨어 테스트가 그렇듯 테스트의 결과가 좋다고 하더라도 다양한 사용 환경에 따른 모든 경우의 수를 테스트할 수 없기 때문에 소프트웨어의 안정성과 품질에 대한 확실한 보증을 할 수는 없다. 이러한 단점을 보완하는 방법으로 호환성 도구의 장점인 적은 시간을 이용하여 다양한 테스트 케이스를 수행할 수 있었으며 이를 통해 소프트웨어의 정확성을 보완하였다.

하지만 추가되는 기능과 여러 요구 사항에 따라 생성되는 매 빌드마다 이러한 호환성 도구를 적용하는 것은 어느 정도 일정 수준을 유지할 수밖에 없고 절대적인 신뢰도를 가질 수는 없다는 한계가 있었다.

자동화 테스트는 현재 전체 범위의 약 5% 내외로 소프트웨어 테스트에 활용되고 있지만 이는 주로 간단한 반복적인 테스트에 이용되고 있는 추세이다.

하지만 Windows 기반으로 활용하고 있는 호환성 도구와 같이 적절한 샘플을 갖고 다양한 테스트 케이스를 진행하게 된다면 앞으로 자동화 테스트에 대한 활용도는 20% 정도로 증가할 것이라고 예상하고 있다.

현재 호환성 도구는 한컴 Office를 기반으로 구현하고 있지만 앞으로 지속적인 연구를 통하여 발전시킨다면 다른 소프트웨어 테스트에도 유용하게 사용할 수 있을 것이라 생각된다.

참고문헌

[1] 채현철 외 1명, 모바일 응용 S/W GUI 자동화 테스트를 위한 방법 및 도구 구현, 한국정보처리학회, 2008

[2] 박종혁, “정확하게 만들기”, 2017

[3] 유경상, “단위 테스트, 그는 적인가 아군인가?”, 2014