https://doi.org/10.33778/kcsa.2022.22.1.003

통합 개발환경에서 데브옵스 기반 테스트 자동화 모듈 개발에 대한 연구

정 광 락*, 김 선 주**

요 약

개발 프로젝트의 소프트웨어 개발에 있어서 통합 개발환경의 역할은 매우 중요하다. 다수의 개발자가 각기 다른 모듈을 개발하면 컴파일과 디버깅, 그리고 통합과 테스트, 배포 등의 과정을 거치면서 제품이 완성된다. 하지만, 개발 과정에서의 버그나 각종 이슈 등은 소프트웨어 제품의 품질저하 및 요구사항 불만족 등의 문제를 발생시키게 되었고, 해당 문제를 회피하고 품질 향상을 위해 자동화 테스트의 필요성이 증가되었다. 본 연구에서는 통합 개발환경에서의 테스트 자동화를 통해 개발 프로세스 전 과정에서의 품질향상을 위한 4가지 관점의 테스트 자동화 모듈을 제안한다. 각 자동화 모듈은 데브옵스(DEVOPS) 방법으로 구현된 통합빌드프레임워크의 툴 체인 형태로 연결되어 구동된다.

A Study of the DEVOPS Test Automation Module for Integrated Development Environment

Jung Kwang Lak*, Kim Sun Joo**

ABSTRACT

The role of the integrated development environment is very important in software development of a development project. After many developers develop different modules, software product is completed through compile, debugging, integration, testing, and distribution. However, bugs and various issues in the development process cause problems such as quality deterioration of software product and dissatisfaction with requirements. So the need for automated testing to avoid these problems and improve quality has increased. In this study, we propose test automation modules of four perspectives to improve quality throughout the test automation in an integrated development environment. Each automation module operates through the tool chain of an integrated build framework implemented on the devops.

Key words : Test automation, DEVOPS, Unit Test, Performance Test, Acceptance Test, Interoperability Test

접수일(2021년 12월 02일), 수정일(1차: 2021년 12월 21일), (2차: 2021년 12월 29일), 게재확정일(2022년 3월 11일)

* 책임연구원/소프트웨어품질인증단/한국정보통신기술협회
** 센터장/소프트웨어품질인증단/한국정보통신기술협회

1. 서 론

현재 국내 대부분의 소프트웨어 개발 기업에서는 독립된 도구를 사용하여 수동으로 테스트를 수행하고 있어 많은 인력과 시간을 소모하고 있는 상황이다. 그럼에도 불구하고 테스트 인력과 도구의 확보 또한 쉽지 않아 품질관리에 어려움을 겪고 있다.

최근에는 4차 산업혁명으로 전 산업이 소프트웨어와 융.복합화되고 있는 상황에서 소프트웨어의 품질과 중요성이 더욱 강조되고 있으며, 그에 따른 개발과 운영이 통합된 데브옵스(Devops) 개발 방법론이 부각되고 있다.

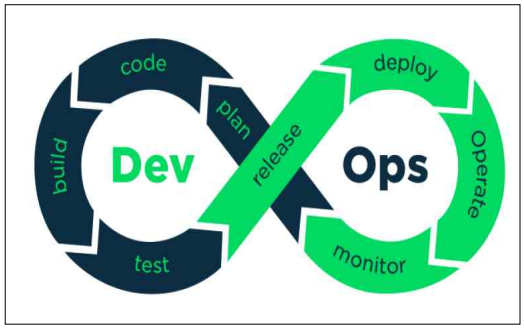
통합 개발환경에서의 소프트웨어 품질 향상을 위해서 다양한 방법이 존재하지만, 본 연구에서는 테스트 자동화 모듈을 개발하여 Devops 기반의 통합 개발환경에 연결.구동하여 품질을 향상시키는 방법을 제안한다. 이는 소프트웨어개발 시작부터 완료까지의 개발 전 주기(소프트웨어 생명주기) 동안 자동화된 테스트를 지속적으로 수행하도록 도울 수 있다. 이를 통해 불필요하게 소모되는 시간을 줄이고 유연하고 기밀한 테스트 수행을 통해 결과물 품질을 목표 수준까지 일관되게 유지시킬 수 있다.

테스트 자동화 모듈은 통합빌드프레임워크에서 데브옵스 기법이 적용된 툴 체인을 통해 구동되어 지속적인 테스트를 수행하고, 테스트 진행현황 및 결과를 로그 및 대시보드 형태로 모니터링하도록 제공한다.

2. 관련연구

2.1 데브옵스(Devops)

데브옵스(Devops) 기법은 소프트웨어개발과 운영이 통합된 의미로[1], 지속적 통합(CI) 와 지속적 배포(CD) 방식에 기반한 데브옵스 툴 체인[2] 형태로 사용된다. 데브옵스 툴 체인은 소프트웨어개발 전체 과정(설계-개발-빌드-테스트-배포-모니터링)을 하나의 체인 형태로 묶어 통합 개발환경의 전 과정에 적용될 수 있다.



(그림 1) 데브옵스 툴 체인

데브옵스는 빠르게 소프트웨어를 개발 및 배포하여 높은 생산성을 가지면서도 목표 품질을 보장하고자 하는 소프트웨어 개발 기업에서 개발/테스트 및 운영을 위한 프레임워크로 활용될 수 있다.

기존의 애자일(Agile) 방법론이 기밀하고 민첩한 대응을 통해 고객요구사항에 대한 빠른 대응이 가능하였다면 데브옵스는 개발/테스트에 운영단계까지 확장되어 개발 이후단계까지 포괄적인 관리가 가능한 장점이 있다.

2.2 소프트웨어 테스팅 동향

최근들어 4차산업혁명 기술의 발전으로 인해 전 산업 분야에서의 소프트웨어 테스팅의 중요성이 지속적으로 증대되고 있다. 특히 자동차, 항공, 철도, 국방 등의 Mission Critical 산업은 물론이고 인터넷 기반의 모바일, 클라우드, 5G, IoT 등에서의 신뢰성 확보를 위한 보안분야 테스팅의 필요성은 더더욱 증가되고 있다.[3],[4],[5] 보안 테스팅 요소에는 기본적인 코드 보안 뿐만 아니라 네트워크, End Point, 웹 취약점, 데이터베이스 등의 전 요소가 포함된다.

특히 데브옵스에서 확장된 데브섹옵스(DevSecOps) 기법을 적용하는 경우, 데브옵스 전 과정에 보안 개발/테스팅이 추가되어 CI 자동화 기반의 보안이 고려된 소프트웨어의 개발 및 테스트가 가능하다.

3. 제안 방법

3.1 테스트 자동화 모듈 개발 고려사항

테스트 자동화 모듈 개발을 위해 고려한 사항은 아래와 같다.

3.1.1 소프트웨어 개발단계 별 테스트 레벨 적용

개발 단계별로 수행되는 테스트(단위테스트, 통합테스트, 인수테스트 등)의 레벨에 따라 사용 가능한 자동화 모듈을 차등 적용하여 개발단계별로 효율적인 테스트가 이루어질 수 있도록 4가지 테스트 모듈을 개발하였다.

3.1.2 소프트웨어 개발 요구사항 수용

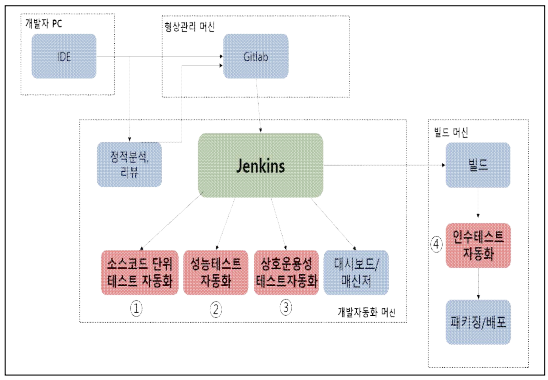
개발 수행 전 수요대상 기업에 대한 현장방문과 인터뷰(개발자,테스터)를 통해 개발 요구사항을 수렴하여 적용하였다. 이후 해당 기업에 대한 시범적용을 통해 개발 결과물에 대한 수요자 검증을 수행하였다.

3.1.3 지속적인 테스트 수행 가능성

테스트 자동화 모듈은 사전 제작된 Testcase 기반으로 구동되도록 설계되었다. Testcase의 경우 개발 전 단계에서 지속적인 추가가 가능하여 상황에 따른 유연한 테스트가 되도록 지원한다. 또한, 제품 출시 이후의 고객 클레임에 의한 수정, 제품 고도화 작업 등에도 상황에 맞는 Testcase를 추가할 수 있어 유연한 확장성을 가질 수 있다.

3.2 테스트 자동화 모듈 개발

테스트 자동화 모듈과 데브옵스 기반 통합빌드프레임워크와의 연동을 위한 전체 아키텍처 구성도는 (그림 2)와 같다.



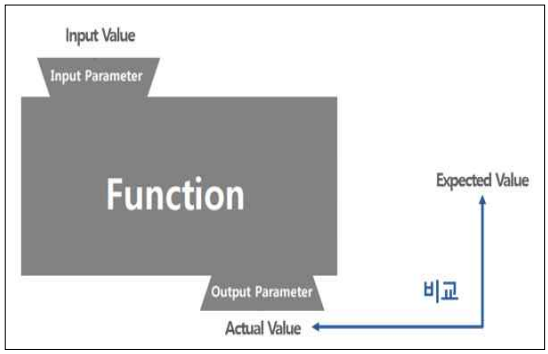
(그림 2) 아키텍처 구성도

테스트 자동화 모듈은 CI 도구인 Jenkins를 중심으로 구성된 통합빌드프레임워크[6]와 연동되어 구동된다. 형상관리 머신은 소스코드의 안정적인 저장을 위해 AS로 운영된다. 빌드머신은 병렬처리를 고려한 단독 장비로 운영되며, 인수테스트를 제외한 테스트 자동화를 위한 도구는 개발자동화 머신에서 운영된다.

테스트 자동화 모듈은 총 4종으로 개발되었으며 상세 개발 내용은 아래와 같다.

3.2.1 소스코드 단위테스트 자동화 모듈

개발 과정에서 단위 모듈이 명세서대로 구현되었는지를 확인하기 위해서는 소스코드 단위테스트가 필요하다. 이에 정상/비정상 입력 파라미터를 입력 시 반환되는 출력값을 구분하여 요구사항을 작성하였다.



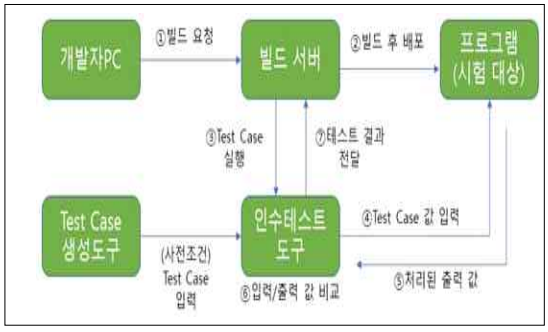
(그림 3) 소스코드 단위테스트 구성도

테스트케이스를 기반하여 클래스 함수를 입력값(Input Parameter)을 포함해 호출하고 반환된 반환값(Output Parameter)이 예상 값과 일치하는지 확인하였다. Java 기반의 Junit 도구를 사용하여 함수 호출이 가능한 테스트 코드를 작성하여 실행하였고 결과는 Run(Pass) /Failure로 나누어 표시하였다.

다음으로, 코드 커버리지 측정을 위해 JaCoCo도구를 통해 빌드 후 코드 커버리지를 측정하여 보고서를 생성하도록 개발하였다. 커버리지 측정결과는 JaCoCo 측정보고서, Jenkins 연동화면에서 확인이 가능하다.

3.2.2 인수테스트 자동화 모듈

사용자(개발자, 고객 등)이 개발 완료된 소프트웨어의 인수여부를 결정하기 위해 인수테스트가 필요하다. 이를 확인하기 위한 검증 요구사항으로 소프트웨어에 입력된 특정 값에 대해서 출력되는 실제 결과 값과 예상되는 기대출력 결과 값을 비교하여 기준을 만족하는지를 확인할 수 있도록 개발되었다.

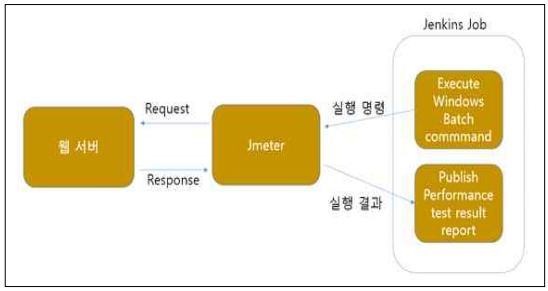


(그림 4) 인수테스트 구성도

사용자는 Testcase를 수동으로 입력하고, 개발자에 의해 빌드가 요청되면 소스코드는 빌드 후 배포된다. 이때 인수테스트 도구(Selenium)는 배포가 완료된 프로그램을 대상으로 입력된 Testcase를 기준으로 검사를 수행하여 입.출력값이 동일할 경우 정상으로 판정하고, 상이한 경우 비정상으로 판정한다. 인수테스트 결과는 Jenkins 관리 화면에서 확인이 가능하다.

3.2.3 성능테스트 자동화 모듈

소프트웨어 개발 결과물에 대한 다수의 사용자 접속과 목표부하를 감당할 수 있는지 측정하기 위해 성능테스트가 필요하다. 성능테스트 자동화 모듈은 Jmeter를 이용하여 웹 서버의 요청에 대한 응답에 소요되는 시간을 측정하도록 개발되었다.

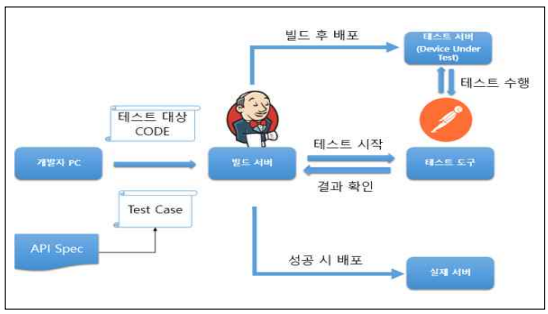


(그림 5) 성능테스트 구성도

Jenkins의 Job 흐름에 따라 Batch Command에서 실행 명령을 수신하고, 웹 서버에 대해 Request를 전송하고 Response를 수신하여 실행결과를 Result Report로 전달하게 된다. 테스터는 목표 가상 사용자 수를 설정하고, 성능 테스트를 통해 그에 대한 에러율과 목표 응답시간 달성여부 측정이 가능하다.

3.2.4 상호운용성 테스트 자동화 모듈

상호운영성 검증은 서버와 클라이언트 개발자가 합의한 프로토콜대로 서버 구현체가 동작하는지에 대한 테스트로, 본 연구에서는 REST API 기반 포맷으로 데이터 교환 가능여부를 판단하는 시스템 동작 적합성 테스트를 위한 검증 모듈을 개발하였다.



(그림 6) 상호운용성 테스트 구성도

테스트 구성을 위해 DUT라는 테스트 대상코드를 이용해 빌드한 서버 구현체의 요청 처리모듈(REST API)을 설정하였다. 테스트 도구는 Newman을 사용하였고, HTTP응답코드, URL, 요청, 응답으로 구성된 Testcase기반으로 테스트하고, JSON 형식의 테스트 결과를 Jenkins에 전달하여 결과를 확인하도록 설계되었다.

4. 제안 검증

본 연구에서 제안한 4가지 테스트 자동화 모듈이 적용된 테스트베드를 Jenkins와 연동하여 구축하였으며, 수요자 차원의 추가 검증을 위해 외부 수요기업을 선정하여 시범적용을 실시하였다.

4.1 테스트 자동화 모듈 구동 결과

테스트 모듈의 구동을 위해 CI도구(Jenkins)에 대상 소스코드, 테스트케이스(TC), 테스트 자동화 모듈을 연계하였다. 테스트는 소스코드 변경 발생이나 설정된 스케줄에 따라 자동으로 시작되었으며, 산출물은 별도의 저장소(Gitlab)에 저장되었고, 사용자는 보고서 다운로드 기능을 통해 테스트 결과 확인이 가능하였다..

(그림 7) 대시보드를 통한 테스트 결과 모니터링

소프트웨어 테스트에 있어서 테스트 결과의 적시 확인은 매우 중요하므로, 오픈소스(Slack)를 활용한 대시보드를 추가로 구축하여 테스트 결과(현황, 커버리지, 결함 등)를 실시간으로 확인 가능하도록 지원하였다. 시범적용 시에도 대시보드는 개발자와 테스터들에게 긍정적인 반응을 얻었다.

4.2 기존 방법과의 비교 분석

테스트 자동화 모듈 개발 전 소프트웨어 개발기업 사전 인터뷰를 통해 개발 및 테스트 작업에서 겪는 불편, 요구사항 등을 수집, 분석하였다. 본 논문에서 제시한 4가지 모듈에서 기존 테스트 방법 대비 개선한 사항은 아래 <표 1>과 같다.

<표 1> 기존방법 대비 개선사항

테스트 모듈	보완점(기존방법)	개선사항
단위테스트	· 매 테스트마다 코드 입력 필요 · 실시간 결과 확인의 번거로움	· CI도구 연동을 통한 자동화된 테스트 수행 가능 · 대시보드를 통한 실시간 결과 확인
인수테스트	· 테스터가 직접 테스트 결과 확인 및 검토 필요	· 테스트 결과의 서버 자동 저장 · 보고서 출력을 통한 현황 파악 용이
성능테스트	· 도구에서 제공하는 결과 외 추가 내용 확인이 어려움	· 에러율 검출 가능 · 목표 응답시간 달성 여부 실시간 확인
상호운용성 테스트	· 각 제품 분야에 따른 표준 분석 필요	· 범용화된 모듈 사용 · 오픈소스 사용을 통한 사용 편리성

기존 소프트웨어 개발기업에서 겪고 있었던 큰 문제점은 Delivery Time 지연 발생(커밋여부/오류 확인 등의 수동 실행, 작업 중지시간 발생, 사람의 실수 등)으로, 본 연구에서 제안한 테스트 자동화 모듈을 통해 유휴시간을 단축시킴으로써 문제를 보완하였다.

추가적으로 외부 환경 변화에 따른 빠른 요구사항 반영이 필요하다는 의견에 따라 커스터마이징이 편리한 테스트케이스(TC) 기반의 자동화 모듈을 설계하여 각종 변경사항에 대한 민첩한 대응이 가능하였다.

5. 결 론

본 연구에서는 데브옵스 기반 프레임워크와 연동되어 구동하는 테스트 자동화 모듈 4종을 개발하였다. 시스템 확장 용이성과 사용자 이용 편리성을 고려하여 오픈소스 기반의 도구로 구성되었으며, CI기반 Jenkins와 연동하여 테스트 자동화 모듈을 통합하였다.

논문에서 제안한 모델에 대한 검증과 요구사항 수렴을 위해 수요기업에 6개월간 시범적용(대전소재 항공SW 개발기업)을 수행하였고, 피드백 의견을 통해 프로세스 최적화, 보안관련 점검[7],[8],[9]를 위한 모듈 추가가 필요하다는 의견을 수렴하였다.

해당 시범적용 검증의 피드백 의견에 대응하여 향후 연구에서는 현재 프레임워크에 누락되었던 소스코드 보안점검 도구와 웹 취약점 진단 도구를 Jenkins에 추가 연동하여 보안 테스팅이 강화된 자동화 프레임워크를 구축할 계획이다. 이를 위해 협회에서 수행하였던 보안시험 서비스, 컨설팅(정적분석, 웹 취약성 점검) 등의 레퍼런스 자료를 적극 활용할 계획이다.

추가적으로, 본 논문에서 제안한 4가지 모듈에 대한 추가적인 신뢰성 있는 결과 도출을 위해 아래와 같은 관련 연구를 통해 비교 분석도 함께 수행할 계획이다.

- 다양한 언어 기반(C언어- Cppcheck 등) 단위테스트 모듈 비교를 통한 고도화 수행
- 다른 UI 테스트 상용도구(Testcomplete) 적용 및 개선점 분석, 알파/베타 인수테스트에 적용
- 다양한 성능 부하모델 사례 분석 및 타 성능도구(LoadRunner, Silkperformer) 비교를 통한 개선점 도출
- 기존에 수행했던 자동차 분야(ISO 26262) 상호운용성 검증과의 비교를 통한 고도화 수행

참고문헌

[1] C. Ebert, G. Gallardo, J. Hernantes and N. Serrano, "DevOps," in IEEE Software, vol. 33, no. 3, pp. 94-100, May-June 2016.

[2] https://ko.wikipedia.org/wiki/데브옵스_툴체인, 위키백과.

[3] 안병구, 유하랑, 장항배, 조직의 능동적 보안문화 형성을 위한 활성화 요인에 관한 연구, 융합보안 논문지 2020, vol.20, no.2, pp. 3-14.

[4] 김민준, 김귀남, 정보보안 거버넌스 프레임워크에 관한 연구, 융합보안 논문지, 2010, vol.10, no.4, pp. 13-199.

[5] 김점구, 이태은, 정보보호 시스템 보안성 자동 분석 방법 연구, 융합보안 논문지, 2008, vol.8, no.1, pp. 117-127.

[6] 김효승, 정지은, 이서정, 데브옵스 환경의 소프트웨어 테스트의 특징과 도구, 한국정보기술학회, Proceedings of KIIT Conference, 2018.11, pp. 478-479.

[7] 김점구, 네트워크 보안시스템 보안성 평가 연구, 융합보안 논문지, 2009, vol.9, no.2, pp. 33-39.

[8] 이동휘, 하옥현, 웹 서비스 보안 성능 평가 테스트 방법론 연구, 융합보안 논문지 2010, vol.10, no.4, pp. 31-37.

[9] 최경호, 이동휘, 외주 개발 웹 어플리케이션 테스팅의 보안성 강화 방안, 융합보안 논문지, 2015, vol.15, no.4, pp. 3-9.

〔저자소개〕

정 광 락 (Kwanglak Jung)
2005년 2월 중앙대학교
전자전기공학부 공학사
2007년 8월 광주과학기술원
정보통신공학과 석사
2005년 3월 현대자동차 남양연구소
연구원
2007년 9월 STX엔진 전임연구원
2008년 2월~ 현재 한국정보통신기술협회
상암SW시험센터 책임연구원

email : iflashy7@tta.or.kr

김선주 (SunJoo Kim)
1999년 2월 배재대학교
전자계산학과 공학사
2001년 2월 배재대학교
컴퓨터공학과 석사
2013년 2월 배재대학교
컴퓨터공학과 네트워크
보안전공 박사
2001년 1월 ㈜케이사인
선임연구원
2003년 9월~ 현재 한국정보통신기술협회
상암SW시험센터 센터장

email : sunjoo@tta.or.kr