ARINC-661 개발 도구의 DO-330 도구 자격 획득을 위한 시험 자동화 에이전트 구현

[1*]김도균, [2]김영곤

Implementation of Test Automation Agent for DO-330 Tool Qualified of ARINC-661 Development Tool

[1*]Do Gyun Kim, [2]Younggon Kim

요 약

DO-330 소프트웨어 도구 자격증명 고려사항은 항공기에 탑재되는 소프트웨어 및 하드웨어를 개발/검증 하기 위해 사용되는 도구 개발 프로세스에 적용하기 위한 지침이다. 이 지침 상의 도구 개발 프로세스 중 검증 프로세스는 DO-330을 준수하기 위해 달성해야 하는 목표 중 많은 비중을 차지하고 있어 상당히 중요하다. 특히, 도구의 안전성 수준이 높은 개발 도구의 시험 목표들은 독립적으로 수행되어야 하기 때문에 많은 시간, 비용, 그리고 인력이 투입되어야 한다. 시험 절차를 잘 수립 해 놓았을 지라도 시험의 복잡도가 높아지면 인적 오류가 발생할 확률이 높아진다. 본 논문에서는 한화시스템에서 개발한 A661UAGEN 도구의 효율적인 DO-330 검증 프로세스를 진행하기 위한 스크립트 기반의 시험 자동화 에이전트 소프트웨어 구조를 제시하고 평가하였다. 그 결과 스크립트 기반의 시험 자동화 에이전트를 통해 자동화 된 시험이 테스트 엔지니어에 의한 수동 시험 보다 시험 수행 시간은 87.5%가 감소되었고, 시험 생산성은 43.75%가 향상되었음을 확인하였다.

Abstract

DO-330 Software Tool Qualification Considerations is a guideline for development of tools used to develop/verify software and hardware installed on aircraft. And among several processes, the verification process is very crucial as it occupies a large proportion for DO-330. Especially, in order to qualify tool with high safety level, test objectives must be performed with independence, accordingly, more time, cost, and manpower are required than other objectives. In addition, even if the test cases or test procedures are well defined, the higher the complexity of the test the higher probability of human error occurs. In this paper, we propose Script-based Test Automation Agent software structure for efficient DO-330 verification process of A661UAGEN tool developed by Hanwha Systems. Compared to the test performed manually by the test engineer, testing time of the Script-based Test Automation Agent is reduced by 87.5% and testing productivity is increased by 43.75%.

Keywords: *DO-330, Verification, Validation, Tool Qualification, Test Automation*

[1*] 교신저자 *한화시스템 항공연구센터 연구원 (dogyun.kim@hanwha.com)*

2 한화시스템 항공연구센터 전문연구원 (younggon.kim@hanwha.com)

Received: Sept. 09, 2020, Revised: Oct. 13, 2020, Accepted: Dec. 29, 2020

I. 서론

항공기에 탑재되는 소프트웨어 및 하드웨어는 운용 중 에러가 발생 할 경우, 최악의 경우 추락의 위험을 초래한다[1]. 항공기가 추락하게 될 경우, 많은 인명피해가 발생할 수 있기 때문에 안전에 매우 민감하다. 또한 항공기에 탑재되는 소프트웨어/하드웨어를 개발 또는 검증할 때 사용 되는 도구도 마찬가지로 개발한 소프트웨어/하드웨어의 결함을 발견하지 못할 가능성 또한 결함을 주입할 가능성이 있다. 따라서 개발/검증 도구 또한 안전에 민감한 소프트웨어로 구분 된다[2].

DO-330 소프트웨어 도구 자격증명 고려사항(Software Tool Qualification Considerations)[2]은 항공기에 탑재되는 소프트웨어 및 하드웨어의 개발 및 검증을 위해 사용되는 도구를 개발하기 위한 프로세스 이며, 해당 프로세스를 준수함으로써 개발/검증을 위한 도구의 신뢰성을 향상 시킬 수 있다. 그림 1[2]에서 확인할 수 있듯이 DO-330 소프트웨어 도구 자격증명 생명주기 프로세스(Software Tool Qualification Life Cycle Process)는 여러 단계의 프로세스로 분류되는데, 그 중 검증 프로세스(Verification Process)는 개발된 도구에 유입될 수 있는 에러를 탐지하고 보고 하기 위해 사용 된다. 해당 프로세스는 DO-330 목표(Objective) 중 많은 비중을 차지하고 있으며, 그 중 시험 목표(Test Objective)는 다른 검증 프로세스를 수행하기 위한 필수 요소 중 하나이다. 또한 도구 자격 증명 수준(Tool Qualification Level, TQL)이 TQL-1 또는 TQL-2 인 경우, 시험 목표는 독립(Independence)적으로 수행되어야 하기 때문에 개발자가 수행을 하는 것이 아니라, 별도의 테스트 엔지니어가 수행을 해야한다. 또한 소프트웨어 수정이 발생하였을 때 마다 시험을 수행해야하기 때문에 다른 프로세스들에 비해 많은 시간, 비용(Costs), 인력이 들어가는 과정이다[4,5]. 따라서 해당 프로세스를 자동화 하는 것은 DO-330 생명주기 프로세스를 진행함에 있어 중요한 부분이다. 또한 시험 자동화를 통해 인적 오류(Human Error)를 감소시켜 효율적으로 DO-330 검증 프로세스를 진행할 수 있다.

본 연구에서는 한화시스템에서 개발한 A661UAGEN 도구의 효율적인 DO-330 생명주기 프로세스 진행을 위한 스크립트 기반의 시험 자동화 에이전트를 제안하며, 이를 통해 시험에 들어가는 많은 비용과 인적 오류를 감소시키는 것을 목표로 하고 있다. 본 논문의 구성과 각각의 내용은 다음과 같이 기술 된다. 2 장에서는 DO-330 생명주기 프로세스에 대한 설명과 시험 자동화 에이전트를 구현하는 방법에 대해 기술한다. 3 장에서는 제안하는 스크립트 기반의 시험 자동화 에이전트에 대해 설명하고, 4 장에서는 제안하는 시험 자동화 에이전트를 사용한 결과에 대해 분석한 결과를 기술할 것이다. 마지막으로 5 장에서는 본 논문의 결론을 도출할 것이다.

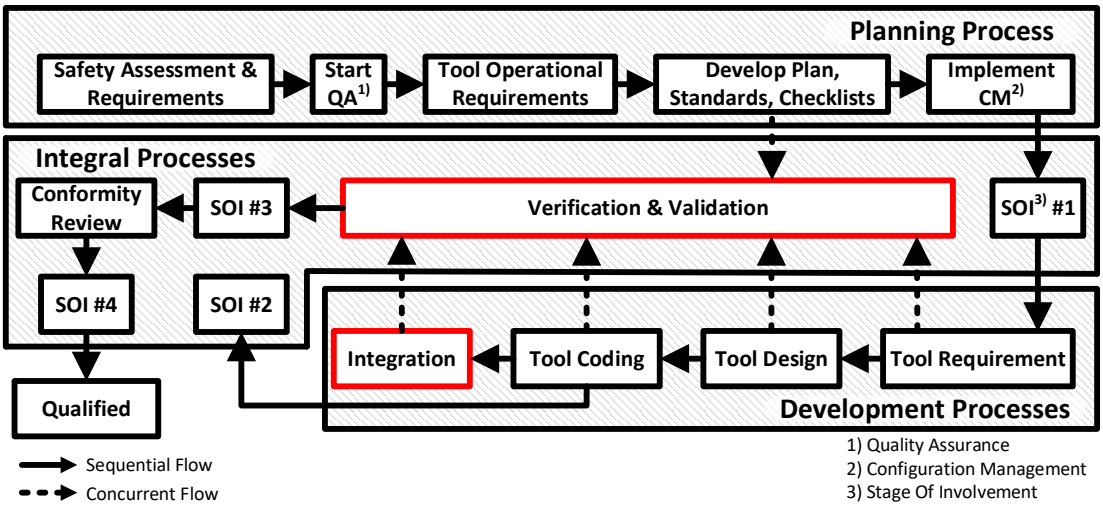


Figure 1. DO-330 Life Cycle Process

II. 배경지식

2.1 DO-330 소프트웨어 도구 자격 생명주기 프로세스

DO-330 소프트웨어 도구 자격증명 고려사항은 항공기에 탑재되는 소프트웨어 및 하드웨어를 개발/검증 하기 위해 사용하는 도구 개발을 위한 지침이다. 그림 1 은 DO-330 지침상의 도구 생명주기 프로세스를 보여준다. 도구 생명주기는 크게 4 단계로 구성된다:

- 계획 단계(Planning Phase)
- 개발 단계(Development Phase)
- 검증 단계(Verification Phase)
- 최종 단계(Final Phase)

미연방항공국 (FAA: Federal Aviation Association)으로부터 도구 자격 승인을 받기 위해서는 각 단계 별로 FAA 에서 임명한 인증 대리인인 DER(Designated Engineering Representative)로부터 감사(Stage Of Involvement, SOI)를 받아야 한다.

2.1.1 계획 프로세스

계획 프로세스(Planning Process)는 도구 생명주기 프로세스들의 활동들을 정의하고 조정하는 단계이다. 계획 프로세스의 목표는 도구 자격 증명의 필요성을 식별하고 도구 자격 증명 수준(Tool Qualification Level, TQL)을 결정 하여 해당 자격 증명 수준에 맞게 계획 및 표준을 정의하는 것 이다.

2.1.2 개발 프로세스

개발 프로세스(Development Processes)는 도구 요구사항 프로세스(Tool Requirements Process), 도구 설계 프로세스(Tool Design Process), 도구 코딩 프로세스(Tool Coding Process) 및 도구 통합 프로세스(Tool Integration Process)로 구성되며, 계획 프로세스에서 정의한 계획과 표준에 따라 도구를 개발하는 프로세스이다.

2.1.3 일체형 프로세스

일체형 프로세스(Integral Processes)는 도구 검증 프로세스, 도구 형상관리 프로세스, 도구 품질보증 프로세스 및 자격증명 연락 프로세스로 구성된다. 이 프로세스들은 다른 프로세스들의 일부로서 동시에 진행되는 프로세스이다. 특히 도구 검증 프로세스는 표 1 에서 확인할 수 있듯이 TQL-1 자격인증을 받기 위한 DO-330 목표 중 약 47.4%(36/76)를 차지하고 있으며 이를 통해 검증 프로세스가 중요한 것을 확인할 수 있다.

Table 1. Number of DO-330 Objectives

Aspect	TQL-1	TQL-2	TQL-3	TQL-4	TQL-5
# of Total Objects	76	74	70	38	15
# of Objects with Independence	35	22	5	2	2
# of Test Objects	4	4	4	2	0
# of Verification Objects	36	34	30	11	0
# of Verification of Test Objects	9	7	6	2	0

2.2 도구 시험

도구 시험(Tool Testing)의 절차는 그림 2[3]에서 볼 수 있듯이, 요구도 기반 시험(Requirements-Based Testing)을 수행하여 도구 요구도 커버리지(Tool Requirements Coverage)와 도구 코드 커버리지(Tool Code Coverage)를 분석한다. 도구 코드 커버리지의 목표는 TQL 에 의해 결정되며,

TQL-3 은 Statement, TQL-2 는 Branch, 그리고 TQL-1 은 MC/DC(Modified Condition/Decision Coverage)를 만족해야 한다. 특히, TQL-1 과 2 에 해당하는 도구의 시험은 독립적으로 수행되어야 한다. 즉, 도구 개발자가 아닌 별도의 테스트 엔지니어가 시험을 수행해야 한다. 도구 시험은 TQL 에 해당하는 시험 목표를 달성할 때까지 지속적이며 반복적으로 진행된다. 높은 신뢰성을 요구하는 항공 소프트웨어 도구에 대한 시험을 수동으로 수행할 경우, 많은 시간과 노력이 필요하며, 인적 오류가 더해질 가능성이 높고, 이로 인해 전체 개발 비용 및 일정을 증가시킨다. 본 논문에서는 이러한 문제점을 해소하기 위해서 스크립트 기반의 시험 자동화 에이전트를 제안한다.

각 시험은 시험 케이스 와 시험 절차로 이루어진다. 각 시험 케이스 들은 요구도를 기반으로 생성되며 입력, 조건, 예상 결과 그리고 합/불 기준을 명시한다. 시험 절차는 시험 케이스로부터 생성 된다. 요구도에 대한 시험은 정상 범위 시험 (Normal range test)과 강건성 시험 (Robustness test)으로 구분된다. 정상 범위 시험은 유효 범위 내의 입력 데이터를 사용하여 평가하는 시험이다. 강건성 시험은 비정상적인 동작을 수행하거나, 유효 범위 밖의 입력 데이터 등 모든 실패할 수 있는 경우에 대해 평가하는 시험이다.

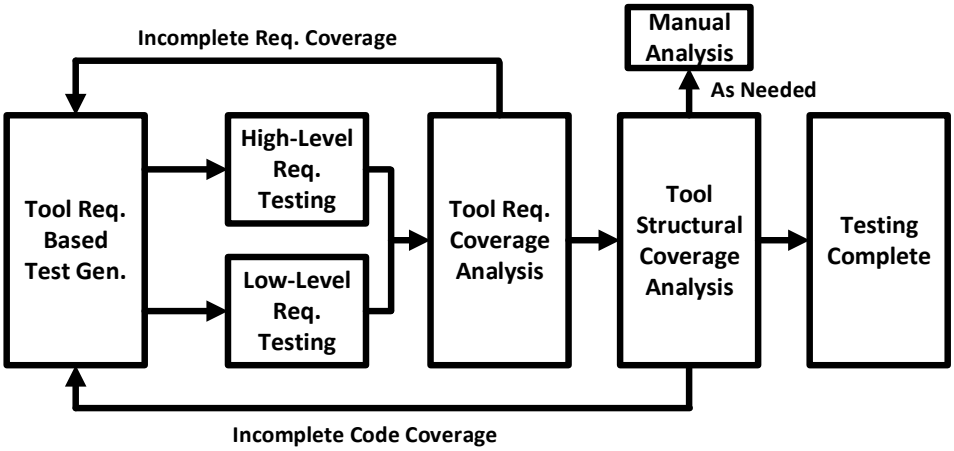


Figure 2. Tool Testing Process

2.2 테스트 자동화 에이전트 구현 방법

본 절에서는 시험 자동화 에이전트 구현에 대한 방법인 좌표 기반 자동화(Coordinate-based Automation), 이미지 기반 자동화(Image-based Automation), 메시지 후킹 기반 자동화(Message Hooking-based Automation)에 대한 설명과 각 방법들에 대한 장단점에 대해 기술할 것이다.

2.2.1 좌표 기반 자동화

좌표 기반 자동화(Coordinate-based Automation)는 그림 3 과 같이 컴퓨터 화면상의 좌표를 기반으로 하여 자동화 하는 방식으로 해당 도구의 버튼, 메뉴 등의 화면상의 좌표로 마우스를 이동하여 클릭하는 방법이다. 해당 방법의 경우 간단하고 빠른 실행 시간을 보장하지만 도구의 화면상 위치가 이동 되면 좌표가 바뀌게 됨으로써 정상 동작을 하지 않는다는 단점이 있다.

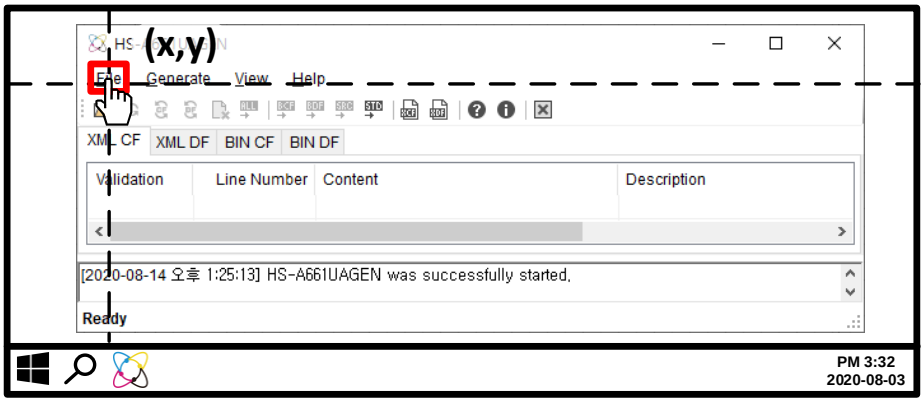


Figure 3. Coordinate-based Automation

2.1.2 이미지 기반 자동화

이미지 기반 자동화(Image-based Automation)는 그림 4 와 같이 특정 버튼이나 메뉴 각각을 이미지화 시켜 화면에 있는 픽셀(Pixel)을 맞추는 방식이다. 해당 방식의 경우 도구의 위치에 의존적이지 않고 도구의 화면상 위치가 바뀌어도 정상 동작을 하지만, 전체 화면을 다 찾아야 된다는 단점이 있다. 따라서 빠른 실행 시간을 보장하지 않는다.

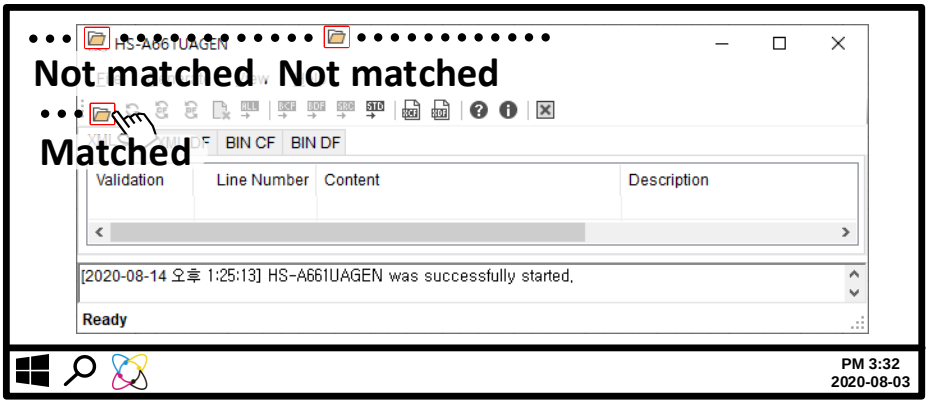


Figure 4. Image-based Automation

2.1.3 메시지 후킹 기반 자동화

메시지 후킹 기반 자동화(Message Hooking-based Automation)는 그림 5 와 같이 해당 버튼이나 메뉴에 할당된 타입 또는 이름을 가지는 객체를 찾아 자동화하는 방식이다. 해당 방법은 좌표 기반 자동화의 단점인 유연성(Flexibility)를 가지며, 도구가 어떤 위치에 있던 해당 객체가 있는 위치를 찾아내기 때문에 유연성을 가질 수 있다. 또한 이미지 기반 자동화처럼 전체 화면을 찾지 않고, 현재 실행되는 프로그램 중 하나의 프로그램에 연결하여 객체를 찾으므로 빠른 실행 시간 또한 보장할 수 있다. 따라서 본 논문에서는 메시지 후킹 기반 자동화 방식을 통해 DO-330 시험 자동화 에이전트를 설계하였다.

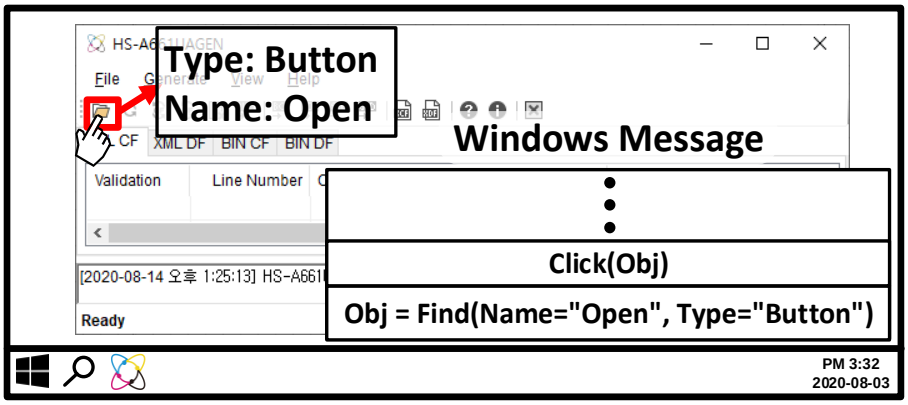


Figure 5. Hooking-based Automation

III. 스크립트 기반 테스트 자동화 에이전트

메시지 후킹 기반 자동화 방식을 통해 설계 된 스크립트 기반 자동화 에이전트는 그림 6 과 같은 구조를 가지고 있다. 제안하는 스크립트 기반 자동화 에이전트는 사용자 서비스(User Service)와 자동화 에이전트(Automation Agent)로 구성되어 있다. 본 논문에서 제안하는 테스트 자동화 에이전트는 ARINC-661 UA(User Application)[6]를 생성하는 A661UAGEN 도구 시험을 위해 구성되었으며, 다음 절에서 상세히 기술한다.

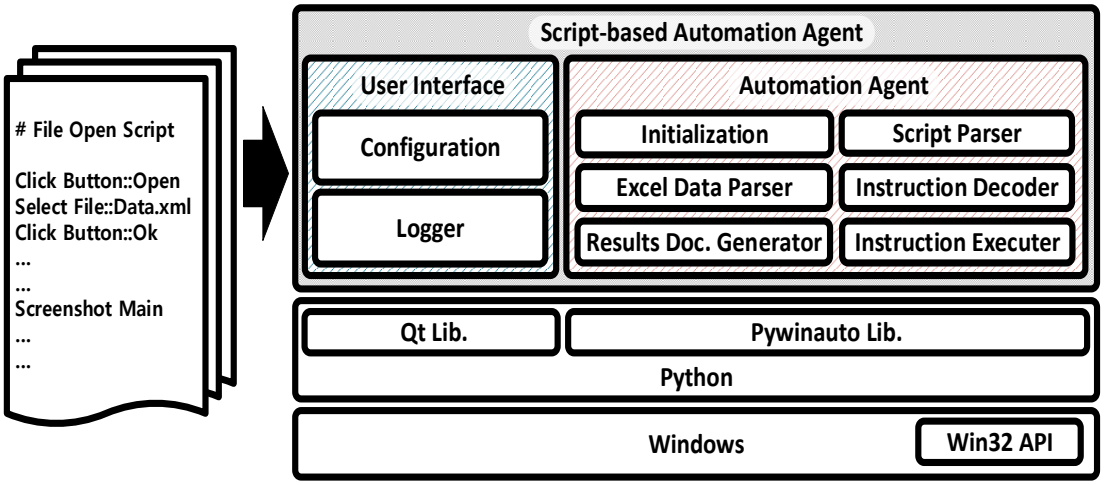


Figure 6. Architecture of Proposed Test Automation Agent

3.1 테스트 자동화 에이전트 구조

테스트 자동화 에이전트는 사용자 인터페이스와 자동화 에이전트로 구성되어 있으며, 사용자 인터페이스는 사용자의 편리함을 위하여 GUI(Graphic User Interface)를 통해 설정하는 부분과 자동화 에이전트의 동작을 기록하여 확인하는 Logger 부분으로 이루어져 있다. 자동화 에이전트 부분은 스크립트를 해석해 테스트 자동화를 진행하는 부분과 자동화 결과를 문서화 하는 문서 생성 부분으로 이루어져 있다.

3.1.1 사용자 인터페이스

사용자 인터페이스(User Interface)는 Qt 프레임워크를 기반으로 구현되었다[7]. 사용자가 자동화 에이전트를 편하게 사용하기 위한 설정 부분 및 로깅 부분으로 이루어져있으며, 설정 부분을 통해 테스트 스크립트가 위치해있는 디렉토리 설정, 테스트 결과를 저장할 디렉토리 설정, 자동화할 프로그램의 위치 등을 설정하는 부분으로 이루어져있다. 로깅 부분은 자동화 도구 실행 시, 어떤 테스트까지 진행이 되었는지 또는 오류의 발생 여부에 대해서 로그를 남기고있다.

3.1.2 자동화 에이전트

자동화 에이전트(Automation Agent) 부분은 두가지 프레임워크를 사용하였는데, 테스트 자동화를 위해서는 pywin32 기반의 GUI 자동화 프레임워크인 pywinauto 를 사용하였다[8]. 또한 문서 자동화를 위해서는 win32com 프레임워크를 사용하였다[9].

테스트 자동화 부분은 유연성 및 수정이 용이하도록 명령어(Instruction)으로 이루어져 있으며, 해당 명령어는 표 2 에서 확인할 수 있다. 또한 명령어의 구조는 그림 7 과 같으며, 총 4 가지 부분으로 이루어진다. 어떠한 명령어인지 판별하는 Command Type 부분과 어떠한 아이템인지를 판별하는 Item Type 부분, 해당 아이템이 어떤 이름을 가지는지에 대한 Item Name 부분, 마지막으로 어떤 윈도우에서 선택을 할 것 인지 선택하기 위한 Window Name 부분으로 이루어진다.

본 논문에서 제안하는 테스트 자동화 에이전트는 해당 명령어 집합으로 이루어진 스크립트를 기반으로 동작하며, 스크립트 분리기(Script Parser)를 통해 각각의 명령어를 분리한다. 그리고 명령어 해석기(Instruction Decoder)를 통해 명령어를 해석하고 명령어 실행기(Instruction Executer)를 통해 GUI 도구를 자동화 한다. 명령어 집합은 A661UAGEN 시험 절차를 자동화 하기 위한 몇가지 특별한 명령어(e.g. Replacement, Chmod, …) 및 일반적인 GUI 어플리케이션 테스트를 위한 명령어(e.g. Select, Click, Keyboard,…) 로 구성하였으며, 상세한 동작 절차는 3.2.1 절에 기술되어 있다.

결과 문서 자동 생성 부분은 문서를 자동으로 생성하기 위하여, win32com 프레임워크를 통해 Microsoft Word 와 VBA(Visual Basic for Applicaion) 명령을 통해 자동적으로 결과 표를 생성하는 방식으로 구성되었다. 결과 문서를 자동 생성하기 위해 예상 결과가 정리되어 있는 엑셀 데이터를 읽어와 분리하는 기능(Excel Data Parser)과 VBA 명령을 통해 Word 에 결과 표를 생성해주는 기능(Results Document Generator)으로 이루어져있으며, 상세한 B3D9 작 절차는 3.2.2 절에 기술되어 있다.

Table 2 Instruction Set of Test Automation

Name	Function
Select	Select the menu
Click	Click the button
Screenshot	Capture the tool screen
Open	Open the some programs(e.g. window explorer, notepad)
Close	Close the open programs with the open command
Replacement	Change the content of the notepad
Rename	Rename the file
Keyboard	Input the keys of keyboard
Chmod	Change the permission of file or folder
#	It is a not Instruction, just comment for script readability

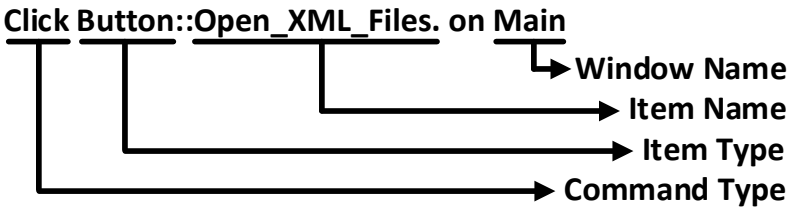


Figure 7. Structure of Instruction

3.2 테스트 자동화 에이전트 동작 흐름

테스트 자동화 에이전트는 테스트 자동화와 문서 자동생성 부분이 각각 다른 흐름을 가지고 수행되며 테스트 자동화는 스크립트를 입력으로 받아 스크립트 분리기, 명령어 해독기 그리고 명령어 실행기를 거쳐 A661UAGEN 도구 시험을 자동으로 진행한다. 문서 자동생성은 테스트 데이터(목표, 예상결과 등)을 입력으로 하여 엑셀 데이터 분리기를 거쳐 결과 문서 생성기를 통해 결과문서를 생성한다. 각 부분의 자세한 동작 흐름은 다음 절에서 기술 한다.

3.2.1 테스트 자동화 동작 흐름

테스트 자동화 동작 흐름은 그림 8 과 같으며, 자동화 스크립트를 입력으로 받아 스크립트 분리기에서 스크립트를 명령어 별로 분리하여 버퍼에 넣는다. 명령어 해독기는 명령어 버퍼에 있는 명령어를 하나씩 가지고 와, 어떠한 커맨드 인지 어떠한 아이템을 수행하는 것인지 해석하여, 명령어 실행기로 보내 해당 명령어를 수행하도록 한다.

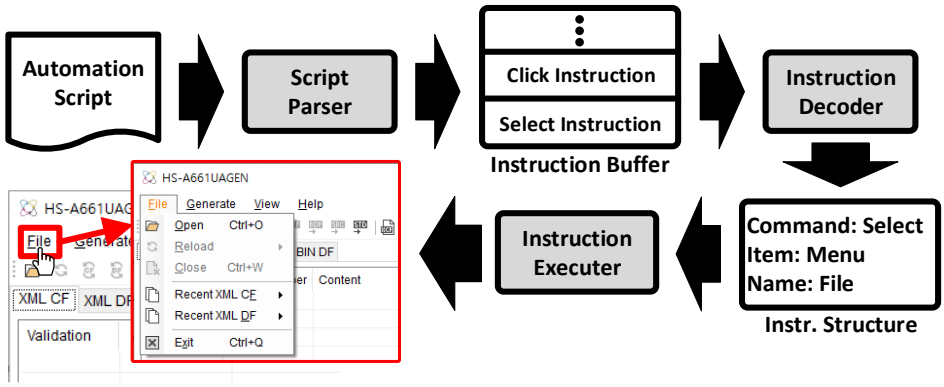


Figure 8. Operation Flow of Test Automation

3.2.2 문서 자동생성 동작 흐름

문서 자동생성 동작 흐름은 그림 9 와 같으며, 테스트 데이터를 입력으로 받아 각 테스트 별 데이터를 분리하여 버퍼에 넣는다. 그리고 결과 데이터와 테스트 케이스별 데이터를 조합하여 결과 문서를 자동으로 생성한다.

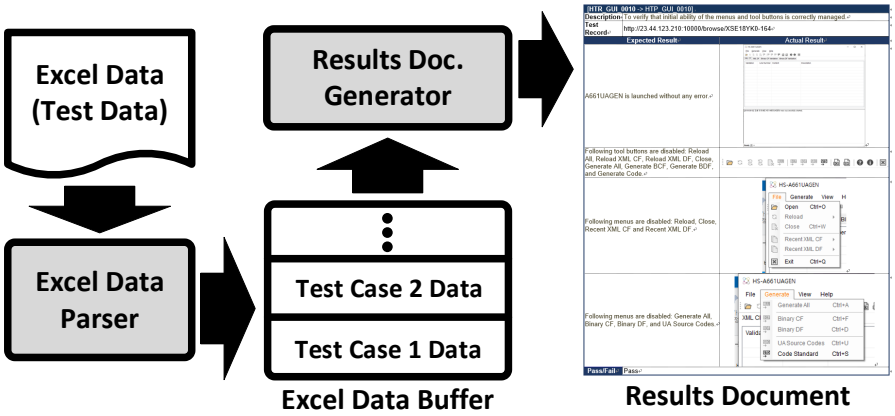


Figure 9. Operation Flow of Document Generation

IV. 결과 및 분석

본 장에서는 제안하는 스크립트 기반의 시험 자동화 에이전트의 실행 결과를 토대로 기존대비 얼마나 향상되었는지 분석할 것이다. 본 논문에서 제안하는 스크립트 기반 시험 자동화 에이전트의 시험 환경은 그림 10 과 같으며, 시험을 진행한 PC 의 사양은 표 3 과 같다. A661UAGEN 도구를 시험하기 위하여 작성된 각 테스트 별 스크립트를 입력으로 사용하여 시험 자동화 에이전트를 동작 시켜 결과들을 뽑아낸 이후, 테스트 데이터와 결과 데이터를 입력으로 사용하여 결과 문서를 생성하였다. MC/DC 100%를 만족하기 위한 A661UAGEN 도구의 전체 시험 개수는 87 개이며 세부 내역은 표 4 와 같다. 각 시험의 결과는 JIRA 의 애드온인 Zephyr[10]를 사용하여 관리하였다.

본 논문에서 제안하는 테스트 자동화 에이전트의 실행 화면은 그림 5 (a), (b) 에서 확인할 수 있으며, 생성된 결과 문서는 그림 5(c)에서 확인할 수 있다.

Table 3. Specification of Test Environment

List	Spec/Version
CPU	Intel i5 @ 1.6GHz
RAM	8GB
OS	Windows 10 Enterprise
Python	3.7
pywinauto	0.6.8
PyQt	5

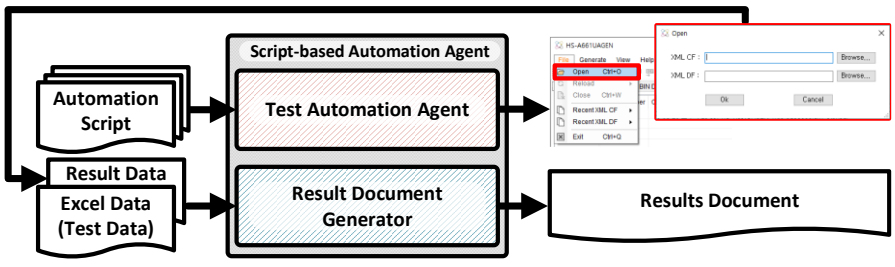
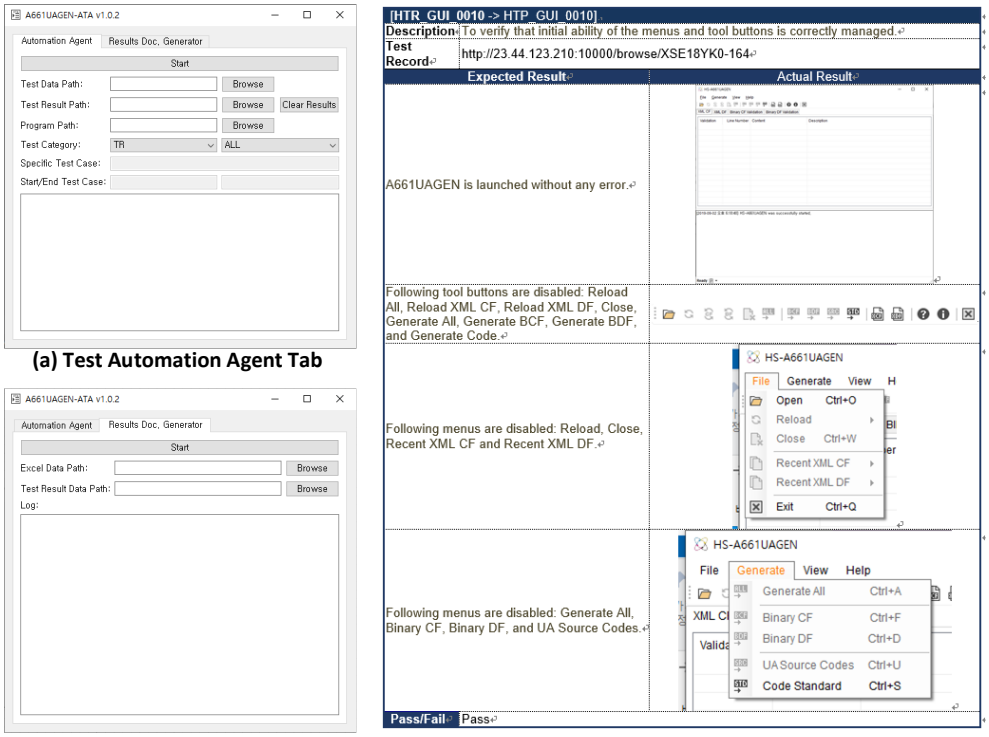


Figure 10. Test Process of Script-based Automation Agent

Table 4. Number of A661UAGEN Test

Test	High-Level	Low-Level	Total
Normal	35	9	44
Abnormal	18	23	41
Robustness	2	0	2
Total	55	32	87



(a) Test Automation Agent Tab

(b) Results Document Generator Tab

(c) Generated Test Verification Results Table

Figure 11. Tool Execution Screen and Result Document

본 논문에서 제안하는 스크립트 기반의 시험 자동화 에이전트를 사용한 결과는 표 5 와 같다. 시험 시간(Testing Time)의 경우, 스크립트 기반의 테스트 자동화 에이전트 사용 이전에는 시험의 복잡도가 높고 반복성 작업이 많아 약 하루 정도가 소요 되었으나, 자동화 에이전트 사용 후에는 한시간 정도로 시험 시간이 감소하였으며, 87.5% 정도 개선 되었다. 또한 시험 이후 코드 커버리지 분석을 하여 코드를 수정하는 시간을 포함한 생산성의 경우, 2 M/D 에서 1.125 M/D 로 43.75% 정도 생산성이 향상 되었다.

Table 5. Result of use to Script-based Test Automation Agent

Testing Indicator	Without	With
Testing Time	1 day	0.125 day
Code Revision Time For MC/DC Coverage	1 day	1 day
Testing Productivity (Testing Man/Day)	2 M/D	1.125 M/D

* 1 day: 8 hours
* M/D: Man/Day

V. 결론

본 논문에서 제안하는 스크립트 기반 시험 자동화 에이전트는 메시지 후킹 기반으로 제작되어 좌표 기반, 이미지 기반의 자동화 도구들에 보다 높은 유연성을 가지며, 스크립트를 기반으로 하여 사용자가 시험 케이스 및 시험 절차 구현 및 수정이 용이하다. 이를 통해 높은 시험 복잡도와 반복성을 가진 시험들을 수행하는데 테스트 엔지니어가 직접 수행하는 것 보다 87.5%의 시험시간이 단축되었으며, 시험 생산성의 경우 43.75% 정도 개선되었다. 또한 스크립트 기반의 시험 자동화 에이전트를 이용해 테스트 엔지니어가 시험을 직접 수행하는 것 보다 인적오류가 줄어들었으며, 스크립트 기반 시험 자동화 에이전트를 사용하여 A661UAGEN 도구 시험을 진행해 FAA 로부터 TQL-1 등급을 자격 인증 받았다. 향후 계획으로는 스크립트를 테스트 엔지니어가 작성하는 것이 아니라, 녹화(recoding) 기능을 추가하여 사람이 스크립트를 작성할 때 생기는 인적 오류 또한 줄일 수 있도록 자동화 에이전트를 고도화 할 예정이다.

VI. 참고문헌

[1] J-.G. Heo et al, “The Study on Airworthiness Certification Process on Military Airborne Safety Critical based on DO-178”, in Journal of Aerospace System Engineering, Vol. 13, No. 1, pp. 62-68, January 2019.
[2] Y. Kim, "Consideration of DO-330 Software Tool Qualification Life Cycle Data," in KSAS 2019 Fall Conference, November 20-23 2019.
[3] "DO-330: Software Tool Qualification Consideration," in RTCA, December 13 2011.
[4] H-.K. Kim, “A Study for the Reduction of the SW Reliability Test Time and Human Errors using the SW Reliability Test Automation,” in Journal of The Korea Society Computer and Information, Vol.20, No. 10, pp. 45-51, October 2015
[5] Y. Choi et al, “A Study on the Code Coverage Goal Setting for Improving Reliability of Military UAV Software,” in KIISE 2014 Winter Conference, December 18-20 2014.
[6] "ARINC Specification 661 Supplement 6, Cockpit Display System Interface to User Systems," in SAE ITC, September 1 2016.
[7] Qt Company. Qt 5. 2020 [Online]. Available: https://www.qt.io
[8] M. M. Mahon. pyWinAuto. 2018 [Online]. Available: https://pywinauto.readthedocs.io
[9] M. Hammond. pywin32. 2020 [Online]. Available: https://github.com/mhammond/pywin32
[10] Zephyr Project. Zephyr. 2020[Online]. Available: https://zephyrproject.org

저자 소개

김도균(*Do Gyun Kim*)

2017 년 2 월 충남대학교 전자공학 학사
2019 년 6 월 서울과학기술대학교 전기정보공학 석사
2019 년 7 월 ~ 현재 한화시스템 항공연구센터 연구원

관심분야 : 임베디드 시스템, 인공지능, 사무자동화

김영곤 (*Younggon Kim*)

2002 년 2 월 경일대학교 컴퓨터공학 학사
2002 년 3 월 ~ 2016 년 10 월 ㈜ 휴원 국방항공사업부 부장
2009 년 2 월 포항공과대학교 정보통신공학 석사
2016 년 11 월 ~ 현재 한화시스템 항공연구센터 수석연구원

관심분야 : 항공전자, 그래픽스, 디스플레이 시스템