Executive Summary Report

INFAST: Intelligent automated Functional And Security Testing

Prepared by abenitez@gmv.com;yalkhazraji@gmv.com

Document Type	Technical Note
Reference	ESR_Executive_Summary_Report
Issue/Revision	1.0
Date of Issue	07/03/2023
Status	First Version

APPROVAL

Title	Executive Summary Report		
Issue Number	1	Revision Number	0
Author	abenitez@gmv.com;yalkhazraji@gmv.com	Date	07/03/2023
Approved By		Date of Approval	

CHANGE LOG

	Issue Nr	Revision Number	Date
First version of the document	1	0	07/03/2023

CHANGE RECORD

Issue Number 1	Revision Number 0		
Reason for change	Date	Pages	Paragraph(s)

DISTRIBUTION

Name/Organisational Unit

Table of Contents

1. INTRODUCTION

1.1. Purpose

This document is the Executive Summary Report for the INFAST project: Intelligent automated Functional and Security Testing. It provides an overall summary of the project targeting non-experts in the field.

1.2. Scope

This document is applicable to the INFAST project and is intended for all stakeholders who need to review the overall project independently from any other documents.

1.3. Document Overview

The document is organized in the following sections:

- **Introduction** describes the purpose and scope of the present document.
- **Motivation of the Study** describes the reasons why this project has started.
- **Objectives and Scope** describes the main goals of this project and the use cases that have been considered.
- **Study Results** contains the most important outcomes of the project.
- **Benefits and Conclusions** summarizes the achievements of the project.

2. MOTIVATION OF THE STUDY

At the European Space Operations Centre (ESOC), complex systems are developed, maintained, and used on a daily basis to support the operation of spacecraft on various missions. These systems make up the ground segment and provide end-users with a range of functionalities that are becoming increasingly complex. This increasing complexity leads to a proliferation of test cases to ensure test coverage and increased effort to support test definition, execution, and analysis of results. Under time and cost pressures, incomplete test coverage compromises the functional and safety aspects of the mission. Therefore, testing plays a fundamental role in the development of projects to: verify compliance with system requirements, prevent future errors and detect errors at an early stage to improve and ensure

the correct functioning of all components, facilitate integration after changes, reduce the cost of project maintenance, and detect security issues or vulnerabilities that can be exploited. Test automation is currently a very interesting area of research. Therefore, projects like INFAST are designed to explore this topic and to find solutions that make the testers' tasks easier.

ESOC and the software industry put a lot of effort into validating the functional and non-functional requirements of their critical platforms. The introduction of the next generation of Mission Control System (MCS) software solutions based on the European Ground Systems-Common Core (EGS-CC) framework is a good opportunity to evaluate new technologies in the testing process. Given that a large part of the testing effort is spent on performing repetitive tasks, it seems reasonable that recent advances in Artificial Intelligence (AI), e.g., machine learning, could help to optimise such testing processes, possibly leading to a higher level of automation in tasks such as test definition, test execution planning, result analysis, etc.

3. OBJECTIVES AND SCOPE

As described in the previous section, the focus of the project is **to investigate the feasibility of incorporating artificial intelligence into the day-to-day tasks of test operators** in the context of mission data systems. To this end, several lines of research are open, and there is an extensive literature on solving different problems using different approaches. Thus, the first objective of the project was therefore to carry out a **study of the state of the art** in artificial intelligence for test automation. In addition, **discussions with various stakeholders served to identify two use cases for the proof of concept**: one related to functional testing and the other to security testing.

On the one hand, the functional testing use case focuses on the results analysis phase, using a system to assist the tester in defect classification and correlation. A system consists of several components, which may include different projects. During the execution of the tests, failures may occur, and the operator must analyse them in order to solve the problem. In this use case, the goal is **to group failures and obtain relationships with previous events to identify possible root causes for a problem that may be common to different projects**. To this end, several available data sources have been analysed for solving the use cases.

On the other hand, the second use case focuses on cybersecurity testing, specifically penetration testing. This type of testing aims to identify vulnerabilities in computer systems to detect possible security breaches and prevent malicious attacks that could compromise the system, or the data stored in it. To do this, Pentesting teams need to define attack plans or decide what actions to take based on the system information available to them. However, choosing the most promising action to achieve the goal can be complex and requires a high level of expertise. Therefore, this use case aims to **automate the process of selecting the actions to be performed in a penetration test**. In this way, it will not be necessary to create an attack plan, but the actions will be selected automatically with the information obtained after the execution of previous actions.

Once the state of the art is known and the use cases have been identified, the aim is to **design and implement a functional AI-based prototype that can be integrated with existing systems and tools at ESA-ESOC**.

Finally, AI-based solutions are becoming increasingly complex, which has led to a debate about the effective control of algorithms and whether they are applied in an ethically correct way. In order to ensure transparency, reliability and causality, explainability is defined as the ability to interpret, understand and explain the technical processes of an AI system and the associated human decisions, in this case by technicians and testers. It is also recommended that explanations are adapted to the level of knowledge of different stakeholders (experts, users, developers, managers, and regulators). In this context, **the last goal of the project is to explore such techniques for the explainability of the designed solution.**

4. STUDY RESULTS

4.1. Automated Functional Testing

In order to search for relationships between test failures, various data sources were analysed. Among them, the use of **test reports and Software Problem Reports (SPRs)** were identified as the most promising. Test reports detail the behaviour of each test step (what each step tested, whether it succeeded or failed, etc.). SPRs, on the other hand, are a collection of failures. These failures can be solved (and therefore closed) or waiting to be solved (open). In addition, they contain a lot of information filled in by the users who have identified and/or fixed the failure. Therefore, the main objective is to analyse these data sources, to group failed test steps and identify relationships with previous events: Similar past failures, existing SPRs (open or closed), etc. In fact, in case no relationships with previous events are found, the new group of similar failures is identified as a potential new SPR. In both data sources it was identified that the most relevant fields are texts. This allowed the use of **Natural Language Processing techniques**, a field of Artificial Intelligence that permits, among other things, to perform tasks such as text classification or automatic question answering. Specifically, techniques that look for **similarity between texts were used to identify related data sources**.

In a first phase of the project, several experiments were carried out to test the suitability of the selected techniques. In this phase **it was possible to identify relationships between test failures from different projects and to discover possible root causes by groups of failures**. However, the search for the SPRs associated with the failures falls short of initial expectations, as very few relationships between them are obtained. One of the reasons identified is the difference in language between the two data sources. The test report texts are generated automatically by the test system, whereas the SPR texts are written manually by the operators. Language written by a human will inevitably contain jargon that is difficult to relate to the automatic texts.

Once the experimental phase was completed, the prototype was developed. **The implemented tool allows the operator to load data and analysis for a given test failure**, providing related data sources.

4.2. Automated Security Testing

ESOC has already developed a proof of concept for penetration testing automation: Penbox. This is an application that contains several Pentesting tools, with the aim of executing them according to a designed attack scenario. This requires not only that these tools are executed in the correct order, but also that each tool is able to retrieve input parameters from the results of the tools that precede it in the execution chain. In this context, the **INFAST project aims to automate the selection of the action to be executed by Penbox** based on the information (observation space) obtained after each execution. In other words, it will no longer be necessary to create an attack scenario, but the attack plan will be created during its execution. A solution based on **reinforcement learning** was chosen for it. This is one of the machine learning techniques that is closest to the way humans learn. It is a paradigm that aims to train machines to maximise a reward given to them for performing certain actions that influence their environment towards a specific goal. In this way**, the use of reinforcement learning has made it possible to train an agent to select the next action based on the available information and network configurations**. This information is provided by the knowledge of the system under test, the success of the previous action and the observations made during its execution. Thus, the agent is trained to maximise reward by selecting the most promising actions to achieve the system compromise and Pentesting objectives. In addition, **a network/topology simulation environment is used for training** to avoid the denial of service of a real network environment during the process, and to obtain a trained solution with sufficient diversity for generalisation.

Again, as in the previous use case, a long period of experimentation was required. Firstly, to create the simulated environments, as the knowledge of a cybersecurity expert was required to create realistic environments. And secondly, to design and test different algorithms. On this last point, the use of **Graph Neural Networks –a novel neural network architecture– for the design of the agent** is noteworthy. This novel design proved to have great potential, **improving the performance of other traditional architectures**.
Finally, work has been done to **adapt the Penbox tool to consume the agent** trained by Reinforcement Learning. At the same time, **a prototype tool was developed to allow operators to train new agents** with new information and to be invoked by Penbox.

5. BENEFITS AND CONCLUSIONS

During the experimental phase, various techniques were used to select the most promising ones. In this type of project, it is often the case that the final results are not sufficient to solve a problem. However, they do show whether the chosen solution has potential and whether the design is on the right track. To this end, **the validation phase has played a key role in determining the benefits of the solution**.

For the functional use case, **the use of similarity-based natural language processing techniques has lived up to expectations**. These have made it possible to find relationships between different test failures, providing related sources that may indicate a possible root cause. This can certainly help the operator analysing a failure by providing past events that are related to a new failure.
However, **possible future steps to improve the solution were also identified**. These improvement aspects include the use of additional data sources to provide more information about the system under test. In addition, the exploration of other approaches such as automatic question-answering or techniques based on statistical theory to identify root causes is suggested. Finally, it is worth mentioning the possibility of creating a text generation standard to help the system recognise similarities between different records.

For the second use case, **a simulated network topology generator was developed for training AI-based agents**. These topologies aim to be as realistic as possible in order to adapt the trained agent's behaviour to what it will encounter in a real environment. The advantages of simulation include: the ability to model relevant system aspects at a higher level of abstraction, such as application-level network communication rather than packet-level simulation, and the ability to ignore low-level details if they are not necessary. In addition, simulation allows easy definition of new machine properties, restricts the action space to a manageable and relevant subset, efficiently captures the global state for easier debugging and diagnostics, and has a lightweight runtime footprint that can be run on a single machine or process. On the other hand, **reinforcement learning techniques seem to be perfectly suited to the needs of the problem and the results obtained are promising.**

However, this is **only the first step in this line of research and there are several options to be explored in the future**, including: improving the transfer of knowledge from the simulated environment to the real one; improving the simulation environment by including new actions and protocols; or exploring other subfields and techniques of reinforcement learning. In addition, the use case focuses on prioritising actions in Pentesting tasks, where the goal is to identify and exploit all existing vulnerabilities. However, another approach could be the Red Team, where the prioritisation of actions is focused on finding a vulnerable machine and making it their own with maximum privileges.

Finally, it is worth mentioning the **use of explainability techniques during the project**. This has made it possible to explain the decision-making process of the models used, avoiding the use of so-called black box models, with the aim of adapting the solution to the current context, which favours the use of trustworthy Artificial Intelligence.