IVVES

Industrial-grade Verification and Validation of Evolving Systems

Labeled in ITEA3, a EUREKA cluster, Call 5

ITEA3 Project Number 18022

D4.4 – Data-driven engineering methods and techniques: final version

Due date of deliverable: Mar 30, 2022
Actual date of submission: Mar 30, 2022

Start date of project: 1 October 2019		**Duration:** 39 months
Organisation name of lead contractor for this deliverable:		RISE

Author(s):	Mehrdad Saadatmand (RISE, SWE), Niclas Ericsson (RISE, SWE), Yaping Luo (ING, NLD), Tim Soethout (ING, NLD), Juan Leandro (Aunia, ESP), and WP4 partners
Status:	Final
Version number:	V1.0
Submission Date:	
Doc reference:	IVVES_Deliverable_D4.4_V1.0 - Data-driven engineering methods and techniques final version.docx
Work Pack./ Task:	WP4
Description: ***(max 5 lines)***	This deliverable reports on the final results and activities of WP4 on applying data-driven engineering solutions in IVVES.

Nature:	☑ **R**=Report, □ **P**=Prototype, □ **D**=Demonstrator, □ **O**=Other		
Dissemination Level:	**PU**	Public	**X**
	PP	Restricted to other programme participants	
	RE	Restricted to a group specified by the consortium	
	CO	Confidential, only for members of the consortium	

DOCUMENT HISTORY

Release	Date	Reason of change	Status	Distribution
V0.1	24/02/2022	First draft based on D4.2 V1.0	Draft	All
V0.2	21/03/2022	Applied partners confidentially checks. Removed sections about IVVES data sharing. Changed T4.1 Plans to Addressed/Solution.	Draft	All
V0.3	25/03/2022	Applied partners text updates and removed resolved comments	Concept	All
V1.0	30/03/2022	Applied the final set of review comments	Final	All

Table of Contents

Glossary

Abbreviation / acronym	Description
AI	Artificial Intelligence
API	Application Programming Interface
CI	Continuous Integration
CD	Continuous Deployment
FPP	Full Project Proposal
GUI	Graphical User Interface
ML	Machine Learning
GDPR	General Data Protection Regulation
PLC	Programmable Logic Controller
RCA	Root Cause Analysis
SLA	Service Level Agreement
SUT	System Under Test
WP	Work Package

1 Executive Summary

WP4 in IVVES, titled ‘Data-Driven Engineering’, focuses on implementing solutions for identifying data correlations and behavioral patterns throughout the entire product life cycle with respect to component failures, and with the ultimate goal of enabling predictive maintenance and anomaly detection. In particular, application of machine learning techniques is considered to achieve this goal and *making sense* of the collected data. From this perspective, the scope of WP4 covers both the development phases as well as system operation.

This deliverable is the final and public version of the WP4 series of deliverables on data-driven engineering methods and techniques in IVVES. It reports on the baseline for i) existing solutions and techniques provided by technology providers including added extensions and achieved improvement during the IVVES project, and ii) initial state of industrial use-cases with respect to the objectives of WP4 at the beginning of the project, their challenges and achievements in addressing the identified challenges. The content of the deliverable is structured as specific sections based on the scope of Task 4.1 ‘Data Collection Techniques, Instrumentation, and Smart Probes’, T4.2 ‘Pattern Recognition for Predictive Maintenance and Fault Analysis’, and T4.3 ‘Data Analytics in Engineering and Operation’.

2 Introduction – Data Driven Engineering

Shortening the feedback cycles, particularly with respect to customer inputs, has always been one of the general challenges in engineering of software systems. Approaches such as continuous integration (CI) and continuous deployment (CD) are examples of solutions to perform more frequent tests and deployments in order to shorten such feedback cycles. Moreover, they can enable collection of diagnostic, performance and operational data in each iteration to quickly learn the implications of changes and new features and react accordingly [1]. In such a context, systematic collection and use of data is not only an effective way to replace opinion-based decision making with data-driven decision making about system performance and quality characteristics [1], but also can enable the automation of such a decision making process with the help of techniques such as machine learning.

Data-driven development is defined as “the ability of a company to acquire, process, and leverage data in order to improve efficiency, iterate and develop new products, and navigate the competitive landscape” [1], [2]. In IVVES and particularly in the scope of WP4, we define the term data-driven engineering to refer to **an engineering process based on systematic collection and processing of data and automation of data-driven decision making with the purpose of improving both the overall quality characteristics of a system and its development process**. For this to happen, various data sources and data artifacts can be considered from not only the development phases of a system, but also its operation and during its runtime (e.g., runtime logs and monitoring information); hence there is a focus on DevOps in this WP.

In connection to this, the work will especially include the application of AI and ML to find data correlations and behavioral patterns throughout the entire product life cycle, for instance with respect to component failures, and to avoid error prone manual labor to resolve problems earlier (left-shifting) and increase automation during development and in operations for specific parts of the project use cases.

To achieve the above-mentioned goals of WP4, the following topics (described in detail in D4.1) are of particular interest for the work and solutions that are developed in this WP:

- Data handling, management, and privacy: dealing with aspects of privacy and security, anonymization and sharing of sensitive industrial data among different stakeholders
- Data sources and virtual sensors: discussing various sources of data that can be used in a data-driven engineering process
- Data quality: discussing quality of input data and its important role in performance, accuracy and output of machine learning algorithms and for data-driven decision making
- xOps: engineering processes suitable for rapid development and delivery of evolving systems while improving their quality at the same time
- Predictive maintenance: processing of data to predict future failures and plan maintenance activities accordingly
- Model synthesis, extraction and construction: using data to construct models of a system
- Fault prediction: using different software metrics, properties, and fault data to predict faulty modules typically before dynamic testing
- Change impact and root cause analysis: processing of data to identify and analyze the effects of a change and also the root cause of a problem
- Data collection in exploratory testing: use of data to guide and automate exploratory testing

In this deliverable, we look more closely into data-driven engineering methods and techniques focusing on: data sources and data collection, data handling and management, data modelling, and data understanding and processing.

3 Industrial Problems Overview

3.1 Use-Case Challenges of ABB

The ABB use-case aims to use ML/AI to improve and automate analysis of resource utilization when performing system regression testing of ABB robot control software.

Furthermore, the ABB use-case has an optional goal to make it possible to decide if a resource utilization problem in a customer program is caused by the customer program or by the ABB robot control software.

The main challenge for the ABB use-case is to identify deviations in resource utilization in the robot control software, since it has an open and highly customizable nature.

By utilizing ML/AI and automating testing and analysis of resource usage in system regression tests, the coverage can be drastically improved. This is expected to improve quality regarding system resource utilization in the ABB robot control software.

3.2 Use-Case Challenges of Bombardier

The use case challenges for Bombardier are mainly split in two different scenarios.

- Current existing systems on trains already deployed around the world
  - The health of the system is monitored using existing diagnostic data, collected using the ADDTRAC and TMS systems from BT where Addiva provides the operation services and continued development. This data is then used as basis for predicting anomalies using AI/ML The challenges here is to identify what model and data tagging methodologies could improve the precision of the predictions. Also identify if additional data, not already collected, would be needed for improved prediction. That would likely require and update of the diagnostic applications on the trains and is a big challenge.
- Research topics in newer systems
  - Challenge here is to identify indicators to predict failure and performance degradation. Focus will be on detecting and classifying anomalies in process data on the train network as well as on the device. Anomalies that are of interest are e.g.:
    - Configuration or systematic errors
    - Resource usage degradation
    - Cyber Security intrusions
  - Infrastructure and lab for this research

3.3 Use-Case Challenges of F-Secure

In order to effectively detect novel and existing attacks against different targets including operational technology (OT) networks using ML/AI a substantial number of different scenarios need to be generated to introduce enough training data for the algorithms as:

- there is a vast number of underlying phenomena leading to the data; the operating system activities, the user behavior, various background programs etc;
- and OT data is extremely hard to obtain due to many times confidential production processes or sensitive production information.

The data needs to be simulated in as close to real environment as possible. The simulation is also important as due to the sensitive nature of the OT equipment and possible disruptions of real production facilities and production equipment can rarely be used. Using real production facilities also present an important challenge that is, to obtain large enough data set to effectively train ML/AI algorithms hundreds of different facilities would need to be visited.

In order to effectively simulate large amounts of different configurations, networks, facilities and attack scenarios a flexible test bench and simulation platform needs to be built. A flexible test bench allows:

- for simulation of various known cyber security attacks including attacks against OT equipment and networks;
- and test the ML/AI algorithms against novel attacks and malware.

The objective is to develop methodology and tools to generate artificial data that can be used for testing and simulation and that shares the same characteristics as real one by simulating:

- vast number of underlying phenomena leading to the data: the operating system activities, the user behavior, various background programs etc;
- large amounts of different configurations, networks, facilities and attack scenarios.

3.4 Use-Case Challenges of ING

The agile test automation pyramid proposed by Mike Cohn sets a guideline for making the most benefits of test automation. The automated GUI tests stand on the top of the pyramid. The reason for running as few of these GUI tests (just the most critical test cases) as possible is that they're the most costly, time-consuming and brittle. Moreover, these tests are most likely to provide false positives and negatives. However, to reach the ideal situation of the test automation, we need to investigate how to automate these GUI tests and reduce false positives which are introduced by the existing tools.

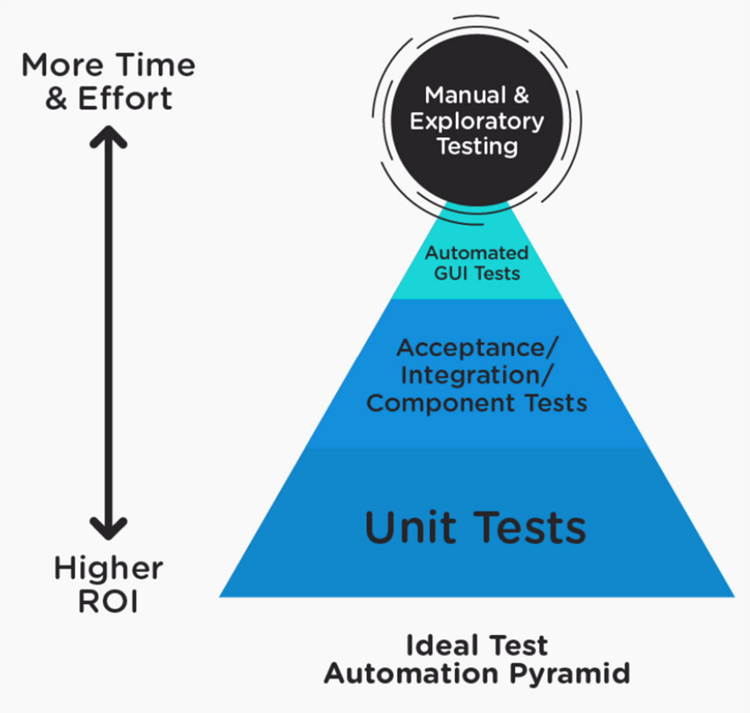


Figure 1. The Test Automation Pyramid by Mike Cohn [3]

The GUI testing was initially, and still is for many companies, a human executed process where the tester is executing a sequence of actions manually [4]. Typical errors in GUI testing are functional errors, technical errors, availability problems, response time problems (which makes an application unusable), and data quality issues. A semi-automated approach is the scripted GUI testing. An example of scripted GUI testing is to record a test sequence which can then be executed automatically. However, this still requires that a human has created the initial execution. As it is still human controlled, it keeps relying on the testers to create test sequences. Testers need to anticipate the unpredictable behavior of users or spend a lot of resources to cover every part of the GUI and maintain those tests. In order to lower the expense of testing, as well as improve the testing process, the automation of tests through scriptless testing has been developed. Scriptless testing is a completely automated testing process in which the tests are both generated and executed automatically.

One of the current issues of scriptless testing is that it usually generates the test sequences randomly. This in turn rarely satisfies the testing criteria, as covering all the GUI test elements through random actions usually requires a lot of execution time. Covering all those elements is required, as the users tend to have unpredictable behavior while interacting with GUIs. To move forward, some kind of intelligence needs to be added to automated testing. With the increasing power of the hardware and maturity of Artificial Intelligence (AI) algorithms, it has become a popular approach for improving automated testing. Last et al. [5] argues that the software quality problems are not too different than the other tasks that AI successfully solves and discusses the applications in software testing. As software testing tends to use a major part of these sources in software development for most companies, as confirmed by ING, it is crucial that these resources are fully utilized to achieve customer satisfaction and requirements fulfillment. For the ING use case, we would like to investigate how to make use of AI techniques to reduce the effort spent on GUI testing of ING's customer facing applications.

3.5 Use-Case Challenges of Philips Netherlands

The Philips Netherlands use case (PHN-2) aims to enhance the verification workflow by using ML/AI. Due to the regulatory requirements to be able to deliver products with every release, we need to prove that all requirements have been implemented successfully. Having this in place, sometimes defects slip into the field. To improve test coverage and to drive a 'shift left' culture we want to combine data from different sources to derive operational profiles for (typical) customer use as input for test cases and use data from test, defect and configuration management systems, to improve and automate the impact analysis and test case selection process.

Challenges are with the amount of information to be processed, connecting the different data sources together, validation of the data and the algorithms used for automating the impact analysis.

3.6 Use-Case Challenges of RHEA Technologies Lab

RHEA Technologies' tool, rapidPHIRE, is a threat hunting platform receiving thousands of alerts per day notifying users about potential attacks on their network. The issue with the platform is that the majority of attacks (>90%) are false positives. RHEA's analysts spend too much of their time sorting these alerts, i.e. correlating and grouping alerts related to the same event and eliminating false positives (the low added value work of a level 1 analyst), and do not have enough time to analyze the real threats in depth (level 2 and 3 analysts). By optimizing the work of RHEA's analysts (automation of the low added value sorting task), the same analyst will be able to monitor the network of more customers in parallel. This will allow RHEA to improve the performance of their threat detection and optimize their costs, thus providing better ROI to their clients.

The use-case challenges are:

- Management of several data tables, nested features and time series data.
- Choosing the right data annotation scheme and format.
- Possibly having to deal with imbalanced data.
- Selecting the right ML algorithms based on RHEA's use case.
- Implementing a solution that will be easily integrated in rapidPHIRE.

3.7 Use-Case Challenges of SII Concatel/Netcheck

AI for ESG investment area is focusing in alternative sources of data. Given the vast amount of nonfinancial data and information available, the analysis can only be done with AI-based systems. As fast as modern algorithms have begun to search and collate new sources for non-financial information, companies have improved the communication with external audiences, being savvier with their wording. Hence, sophisticated artificial intelligence is needed to learn and infer context. ML-enabled models must be able to continuously adapting to face a changing environment, where the quality of massive unstructured data sources, rewording of corporate reports, sentiment analysis and fake news must be taken into account. At the same time, ESG investment is linked to auditing processes and reporting requirements that demand transparency and explainability, for a better human understanding of AI decisions.

The main objective is to provide an effective tool to assess and validate performance and reliability of ML-enabled systems for ESG scoring based on new AI-based V&V techniques; increasing the adoption of ML-based services from Home Offices, Security Agencies and Fintech companies.

3.8 Use-Case Challenges of Keyland

KEYLAND has focused in this task on the analysis of requirements, implementation of environments and evaluation of tools for the Industrial Automation use case.

For the definition of the use case, information from different stakeholders and customers has been taken into account, and a model that will integrate different solutions has been proposed. The objective is to perform the integration of real systems, currently deployed, and to analyze how these can be coordinated in a common environment, together with new ML-based SAAs systems.

KEYLAND has carried out an exhaustive analysis of different solutions, technologies and products that can be deployed in the same environment (such as logistics AGVs, collaborative robotics systems, indoor positioning solutions, sensor networks and existing machinery (i.e. elevators...).

The main objective is to establish, through our work in IVVES, a technological base that allows us to evolve our current services. It is a priority and essential for KEYLAND to establish a framework to design, deploy and monitor AI-based services in an industrial environment.

4 Data Collection Techniques, Instrumentation, and Smart Probes (Task 4.1)

The main goal in this task is to implement data collection mechanisms to identify and extract required (raw) data and critical system properties to enable predictive maintenance, fault analysis and anomaly detection (using the techniques developed in T4.2). This task starts with the investigating the data sources that can be used from the development process and tools, as well as from the operations of the developed systems. Typically, development data can be collected from software and hardware tools, including simulators, while operations data can be collected though IoT channels. The needs of the industrial partners and characteristics of the use-cases and demonstrators from different domains, have been taken into account for tailoring and customization of the solutions that are produced in this task.

4.1 Use case providers

4.1.1 ABB: Customer program resource utilization in production

Currently is manual fault prediction conducted. For each new build of the robot control software, we collect over 4000 different metrics from three different reference programs. Parts of the metrics are manually analyzed to find anomalies. It is mostly relative CPU utilization that are examined. The system size is roughly 3.2 million lines of code, 7000 files, 65000 functions. Architecture evolved over the last 30 years.

Use case situation at the start of IVVES

Data related to system resource usage is collected from three references scenarios from almost each daily build of the robot control software. The data collected are e.g. task/thread executing properties such as CPU utilization, execution times, priorities, periodicity, stack usage, events, memory usage, message queue usage. Some of the data can be used directly. The data regarding task/thread execution properties are pre-processed using the tool Tracealyzer (from Percepio) and collected by instrumentation in place with negligible overhead on the target system. The data is stored in raw format as files on a company server. The data is also stored in compiled/processed format for manual analysis in an Excel-sheet.

Operational data from deployed systems cannot be directly accessed, but we can access operational data from failing systems, when bugs are reported to ABB.

Addressed challenges in IVVES

During the executing of the IVVES project, the following improvements have been done:

- More data is collected from each scenario, e.g., binary sizes, live CPU load measurements, histograms of generated events, startup timing metrics, etc.
- More scenarios have been added, including more scenarios close to customer use-cases including additional applications.
- More hardware platforms have been included. In the beginning we only collected on one type of the current generation of our platform. Now we are collecting data from three variants of the hardware platform and on a simulated version of the system. We are now also collecting data from the previous generation of the hardware and software, which are still evolving for existing customers.

4.1.2 Bombardier: Data collection and analytics in Engineering and Operations

No predictive maintenance is currently included in the use cases, but the intention is to find indicators to predict failure and performance degradation. The systems consist of more than 1 million lines of code (to be detailed later). Focus will be on data collection needed to find solutions for detecting and classifying anomalies in process data on the train network as well as on the device. Anomalies that are of interest are e.g.:

- Configuration or systematic errors
- Resource usage degradation
- Cyber Security intrusions

Use case situation at the start of IVVES

Data collection from development and operations needs to be improved over the whole product life cycle.

The software development process used is a traditional V-model that needs to move towards a more agile CI/CD model, in order to meet the expected delivery rate. For this Bombardier will continuously evaluate potential tools and solutions to integrate into their development pipelines, e.g., to automate verification and validation activities.

Addiva are partnering with Bombardier in collecting live data from common customers with systems currently in operation. This collaboration forms a base of knowledge for identifying improvements on e.g., what to collect and data quality for system updates and upgrades.

Addressed challenges in IVVES

We have improved our strategy for data collection in operations for e.g., anomaly detection. A new virtual environment for products has been developed. This will aid in development of new products before actual hardware is available and produce simulated data for a complete system of systems for both functional development as well as for anomaly detection. We have addressed and developed low intrusion data collection probes adapted to this virtual environment.

In our new DevOps environment, we collaborated with Sogeti on data collection in our CI/CD pipeline for analysis and anomaly detection with “DevAssist”.

4.1.3 F-Secure: Simulating OT networks

F-Secure do not do any kind of predictive maintenance but with the F-Secure built OT simulator it is possible to simulate also faults and test predictive maintenance algorithms. That is, given that the physical fault constraints can be accurately simulated.

The simulation system / rig consists of:

- 3 Siemens S7-1200 PLCs
- 2 Industrial PCs
- 2 HMI displays
- 2 L3 switches
- 1 Firewall
- Number of I/O devices, sensors and actuators
- ~ 500 lines of STL (Structured Text) code for physical constraints simulation
- ~ 6 networks of 30 rungs of LAD (Ladder logic) per PLC for control logic

Use case situation at the start of IVVES

Network traffic from OT devices from routers are collected, with the aim to accurately and safely simulate OT attacks for wide range of scenarios. Data can be collected either for one of the industrial PCs in the simulator or for any external devices using port mirrors for LAN or WiFi connections, storing the data in raw PCAP format. The data is collected for local computer and from there it can be transferred to long term storage and for other researcher’s use within F-Secure.

No operational data are collected.

Addressed challenges in IVVES

Data collected at customer premises cannot usually be taken out of the customer premises due to the sensitivity of the data. Hence it cannot be used outside of the specific customer project scope to develop new tooling and methods. It is important to train the consultants, first responders and F-Secure software to accurately detect and respond to OT attacks. As OT domain is very specific in nature the reliability and correct functionality are absolutely critical. The only way to accurately generate data outside of the customer premises is to build a simulation rig utilizing real OT equipment.

Only OT data, no other kind of data are already available, nor is any data from development considered relevant.

4.1.4 ING: GUI testing in the acceptance phase

Use case situation at the start of IVVES

The following data is collected for GUI testing:

- (Functional) test scripts
- User journeys: How is a customer expected to interact with a system?

The motivation for data collection in general is that ING would like to improve its systems over time, which means that ING wants to try to improve the process of creating the artifacts. When the process is improved, ING assumes that the quality of the output of the process has also improved. ING believes that there are opportunities to improve the processes with the data we now collect, however ING does not know how. ING would like its partners to help with this. The collected data needs to be processed and made presentable. For example: user journeys are the result of an in-house application of an industry-standard modelling toolkit and test cases and/or traces are present in various formats, but always in a very low level and tied to technology platforms. Several techniques are used for the collection such as:

- Harvesting log files
- Acquiring source code (that is accompanied by tests)
- Analyzing process models
- Analyzing GUI test scripts

The formats vary from e.g., text, XML, SQL Database, proprietary tools (i.e. behind a reporting frontend but without easily accessible raw data/APIs), git. Everything is stored in an internal infrastructure hosted by / inside ING, as well as (re)using parts of a large cloud provider.

The overhead for collection is not applicable, since our scope is the development and/or testing process, not the production stack directly.

ING owns all systems and data of all customers, there is no data and/or systems on customer sites.

Addressed challenges in IVVES

Automatic generated test sequences from Testar are captured. Bug/Fail runs are reproducible this way.

We analyzed the current (manual/unit) test scripts of Mobile, based on customer journeys, in order to compare the scripless with the script-based testing. The results are captured in chapter 4 of the MSc thesis titled “mTESTAR for scriptless GUI testing on Android and iOS application”.

Comparison is done on code coverage metrics with different algorithms used in scriptless testing (Q-learning, Random Actions, Unvisited Actions First, with different configuration). The result is that both testing strategies are complementary. Data collection is needed to measure both.

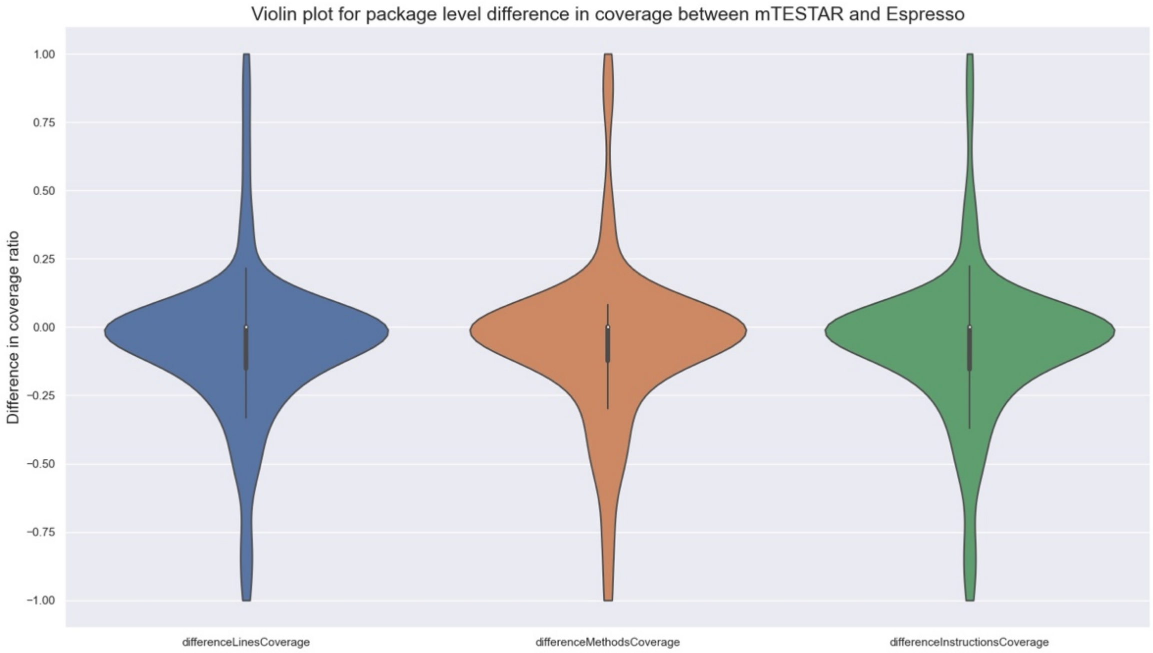


Figure 2. Test coverage of both scriptless (>0) and scripted (<0) tests.

Above violin plot (reproduces from said MSc thesis) shows test coverage of both scriptless (>0) and scripted (<0) tests. Result density near 0.0 shows that both testing approaches cover the same files and lines. The positive and negative peaks shows coverage of only one of the approaches. This shows that both testing strategies cover the mostly the same code, but are also complementary in their outliers.

This addresses the coverage KPI of IVVES.

4.1.5 Philips NLD: Using AI to enhance the verification flow

Currently impact analysis of changes is a manual activity. Information from different sources and data available in different tools is analyzed as input for our verification and validation plans. Execution is mainly a manual activity where test cases are based on the requirement specifications. Predictions for our defect inflow are available during development and we run reliability tests to predict reliability in the field.

The size of the system is confidential information.

Use case situation at the start of IVVES

Data currently collected are e.g.:

- Logfiles from our Installed Base systems
- We have a test management system with test result from running and previous programs
- Defect records for running and previous programs are managed with a defect management tool
- SW version control information on e.g. code changes
- Code coverage and code quality data is recorded for in-house test configurations

The intent of use case PHN2 is to bring our test coverage closer to real customer use of our systems, where we also want to improve our test coverage by smarter test case selection, based on changes in the SW and learnings from previous programs. For now, data needs to be processed manually, intent is to have this processing automated by using ML/AI algorithms. A remote service network allows us to collect logfiles from the field. Furthermore, we have different tools to support the development processes. Data is currently available in different formats, e.g. CSV files, Excel files, SQL database. Intent is to have all data stored in an internal infrastructure proprietary to Philips.

Most of our Install Base systems are connected to a remote service network. As such operational data can be accessed and can be used for the purpose of quality and reliability improvements.

Addressed challenges in IVVES

Scan protocol data retrieved from the installed base is not always complete and/or contains wrong information on the anatomic region being scanned. Validation of the scan protocol data and update/enrich data using machine learning techniques has been the first challenge that has been addressed. To validate the algorithm a golden dataset is required. For now, based on the large amount of protocol data from the installed base (> 150M), we assume wrong information is not leading to false outcomes.

Process mining techniques have been used to implement scripts that automate the generation of the set of 'typical' scan protocols as used in the installed base for the different anatomic regions.

The automatically generated scan protocols are used as input for:

- Basic regression test suite which is part of the CI pipeline tests.
- E2E verification workflows for the different anatomies and clinical domains as part of formal verification
- Reliability test suites.

Daily update of scan protocols is now possible based on changed use as seen in the installed base.

4.1.6 RHEA: Detecting abnormal network behavior

No predictive maintenance or fault prediction is performed.

While no actual line count has been performed, we are estimating there are about 50,000 lines of code across 20-30 components.

Use case situation at the start of IVVES

The data used by rapidPHIRE is derived from dissecting IP network packets. The packets are parsed, and their metadata is recorded and sent to the data processor. The data is text-based and consists of two "types" of data. The first is what I call consistent data, which is comprised of the IP addresses, ports used, timestamps, direction of travel, payload volumes, and duration of connection. The second is what I call dynamic data, which is based on the various bits of information determined by the nature of the communication. The "various bits of information" will be different depending on the application protocol being used. For example, a web browser will communicate using HTTP or HTTPS protocols, both of which contain elements only seen in HTTP/HTTPS packets. This is what I consider to be dynamic data. Both consistent and dynamic data contain elements that can be matched to known-bad values for easy identification of potentially malicious communications. The number of data points in any given connection will vary from as few as about 5 to as many as a couple dozen.

The data elements noted above are collected and monitored with the goal of identifying anomalies and inconsistencies that vary from the expected protocol (IP and application) behaviours. Behaviours that deviate from the expected norms are investigated. The data elements collected are also used to identify connections to IP addresses or URLs that match known-bad entities – the "low hanging fruit."

The data is used in its raw format. The additional processing that is performed by rapidPHIRE is mainly centered around data augmentation – rapidPHIRE will try to gather additional information about the data points it has received, such as geo-location, Autonomous System Number (ASN), etc. Essentially, rapidPHIRE is a collect, augment, store, and present, type of system that just doesn't do enough to provide better value from the data being collected.

rapidPHIRE uses a physical or virtual "sensor" to passively collect data directly from the customer's network switch. Multiple sensors can be deployed for the same customer. A sensor is a processing machine that runs the rapidPHIRE sensor software to process copies of network packets that are sent to it by the customer's network switch. The packet parsing and initial match against known-bad elements is performed on the sensor, with the sensor sending the metadata to the analytic cluster for augmentation and storage.

The data is stored in a non-SQL Cassandra database in its raw format, that is stored locally and backed up to local and remote (non-cloud) locations. The amount of overhead on a system for data collection and monitoring is dependent on the amount of traffic being received by the sensor. Typically, for most of the sensors we have deployed, a CPU load of between 0.5 and 2.0 can be expected as normal with ethe majority of the sensors running closer to 0.5.

There is no data collection from the development process.

We have access to the sensor's system parameters only. We do not have access to any other data at the location where a sensor is deployed.

Addressed challenges in IVVES

There are not currently any other kinds of data available, nor are there data collection from the development process. In IVVES we built data management solutions to address these challenges.

4.1.7 SII CONCATEL/NETCHECK: V&V for ESG investment system

Currently there is no predictive maintenance available in the end user environment. This is preventing the system to scale, since a manual analysis is still required to periodically validate the outcomes of the ESG-scoring system. Hence, besides credibility analysis and explainability, it is mandatory to establish a proper DevOps pipeline.

Use case situation at the start of IVVES

Initially, there has been no predictive maintenance deployed in the end user environment. This is preventing the system to scale. Data collection can be done as soon as the proper DevOps pipelines is integrated and validated.

Addressed challenges in IVVES

Data collection should enable seamless monitoring of the overall system, and should provide alerts and notifications to experts. The CCTL/NC will work on integrating tools for change impact analysis and fault identification for ML components and data analysis for ML-based engineering and operations. The developed components will be based on Fintech assets and ESG news classification in such a way to be GDPR compliant at all times and enable proper investment in secure assets using various artificial intelligence and machine learning techniques.

4.1.8 Keyland: Predictive Analysis in Industrial Environment

Although industry is already planning the deployment of ML components and ML-based systems, there is no industry-wide verification and validation (V&V) approach for full quality control of SAAs.

At present, there is no production system that can quickly adapt to market demands for new production models with short life cycles. This prevents the system from being scalable. Using V&V techniques for ML-based systems will prevent the technology from stalling growth.

Use case situation at the start of IVVES

KEYLAND has focused in this task on requirements analysis, environment implementation and tool evaluation for the Industrial Automation use case.

For the definition of the use case, information from different actors and customers has been taken into account, and a model that will integrate different solutions has been proposed. The objective is to perform the integration of real, currently deployed systems and to analyse how these can be coordinated in a common environment, together with new ML-based SAAs systems.

KEYLAND has carried out an exhaustive analysis of different solutions, technologies and products that can be deployed in the same environment (such as logistics AGVs, collaborative robotics systems, indoor positioning solutions -IPS-, sensor networks and existing machinery (i.e. elevators...).

The main objective is to establish, through our work in IVVES, a technological base that will allow us to evolve our current services. It is a priority and essential for KEYLAND to establish a framework that allows us to design, deploy and monitor AI-based services in an industrial environment.

Addressed challenges of IVVES

Adaptive manufacturing relies on complex systems, with AI-based components that must be safe and reliable and must be validated and certified against standards before they can be put into production. The lack of V&V techniques for ML-based systems is limiting the adoption of the technology, especially in those areas of ML where hidden layers work as a black box (Deep Learning). Another challenge is that the optimal configuration of the ML workflow will vary over time, possibly in a matter of hours or days. The result is that models will become obsolete very quickly, as the training data differs from the actual data in the operating environment. In IVVES we built solutions to alleviate these challenges.

4.2 Tool and Solution providers

4.2.1 Ekkono: Fault analysis and anomaly detection

The input to our tool is sensor data and the output is the results derived from ML models. The file format is Arff or CSV files and any file format is supported by the python package pandas. The tool has no specific data requirements. Data is collected directly from industrial systems execution in operation or from processed data. No tool is provided for the anonymization process itself, however, the tool can work with anonymized data.

Solution situation at the start of IVVES

Ekkono's solution enables predictive maintenance at the edge. It is an online tool that does not include probes or instrumentation. Nor does it require any specific infrastructure or store any data.

The solution can make 1000s predictions per seconds on a Cortex M0+.

While the tool does not provide a function to anonymize the data, it can work with already anonymized data.

Solution extension in IVVES

For Task 4.1, Ekkono has extended their solution with the following:

- Fast Fourier Transform (FFT) and Wavelets for real-time feature engineering at the edge. While traditional approaches need a complete batch of data to perform FFT or wavelets transformations, Ekkono's team have developed new algorithms that can perform these transformations incrementally without the need to save the data for offline processing.
- Hoeffding Trees: Ekkono has developed a prototype for Hoeffding trees that is able to perform regression predictive modeling on streaming data.

4.2.2 OU NL: GUI testing with TESTAR

Input for TESTAR tool is a way for the tool to connect to the GUI application under testing (for example path to the executable or URL), and optional system specific configurations in form of settings file and possibly a Java file. The output includes test results and the generated test sequences, optionally inferred state models in OrientDB graph database, and logs.

The state model inference is based on observed behavior of the GUI during automated exploration / scriptless GUI testing.

Solution situation at the start of IVVES

At the start of IVVES project, TESTAR already provided support for overwriting any configuration through command line startup command and that has been used in integration with CI systems. TESTAR produced some reports in HTML format, and the generated test sequences are saved in an internal binary format that allows re-executing them with TESTAR. The inferred state models are stored into OrientDB graph database that provides a documented API to query the results.

The collaboration in ING use case has focused on two directions: improving TESTAR to test ING web applications and implementing support for testing ING mobile applications (Android and iOS).

At the start of IVVES project, TESTAR did not support all the web elements used in ING web applications. The support has been increased when issues have been reported. Also, TESTAR action selection struggled with filling long web forms with valid data that allows submission. TESTAR did not support testing mobile apps before IVVES project.

Solution extension in IVVES

Regarding Task 4.1: Data Collection Techniques, Instrumentation, and Smart Probes, TESTAR has been extended with:

- better support for testing web applications (using WebDriver),
- support for Docker containers for parallel/distributed testing of web apps, improving the speed of GUI exploration (data collection) for model inference purposes,
- support for testing mobile applications (integration with Appium), implementing a way to collect data from mobile apps,
- more data for building new ML algorithms for action selection, for example measuring the change impacted on GUI by executed actions,
- automated form filling capabilities based on generated template input files, allowing TESTAR to access more states during data collection.

4.2.3 Praegus B.V.: Test duration optimization in the testing phase

Input to our tool Orangebeard (a.k.a. augmented testing) uses SCM-input (i.e. Git), build output and test reports as input.

Outputs are insight in test performance (duration, failure rates, etc.) and charts, in our Orangebeard Control Room Next to that want to optimize the duration of a testset run depending of e.g. SCM-input, in our Orangebeard Auto Test Pilot.

The data formats are standardized JSON input (e.g. GitHub graphQL API output) for Git/SCM input sources. Next to that we provide custom REST-API's (JSON-based as well) in order to receive data about test suite runs (JUnit, FitNesse acceptance tests) and build pipelines (Jenkins/CircleCI/etc.)

Regarding data from use cases. We apply source code metadata (e.g. commit messages and changed filenames) and test reports. We focus mainly on the Use Cases that are similar to the ones provided by ING and F-Secure. Focusing mainly on the testing phase. In addition are the tool are able to anonymize data.

Solution situation at the start of IVVES

We develop a suite of listeners that hook into the different test tools (e.g. JUnit, NUnit, Ranorex, FitNesse) that collect test reporting information. Depending on the tool we apply probes that hook into the test execution. The collected data is sent to our Cloud Native Backend application for processing. The test execution events are stored in our Event Store, at the moment events are stored as XML - but this may very likely change in the future, we are looking at JSON and BSON (Binary JSON) at the moment.

Currently the tool does not anonymize the data but depending on the new data source we may want to tap into this may change.

There is no relevant overhead for data collection as we only run our listeners inside the DevOps pipeline and not in the production code.

Solution extension in IVVES

In IVVES Orangebeard have extended the module "Auto Test Pilot", to be able to perform basic subsetting and prioritization of testcases against the relevant context. At this point we are able to collect relevant testoutput and change metadata from different sources. The collected relevant test output and processed meta data have been combined into a machine learning model.

4.2.4 RISE: Performance testing with SaFReL/RELOAD

SaFReL/RELOAD. It is a reinforcement Learning-assisted performance testing framework consisting of two tools. The proposed framework learns the optimal policy to accomplish the intended test objective without access to system model or source code of SUT. Once it learns, it is able to reuse the learned policy in further testing cases. The proposed framework consists of two performance testing tools: SaFReL and RELOAD.

The tools have no need for a set (batch) of training data. It works based on a continuous interaction between a smart agent and the system under test (SUT). It implies that the agent executes several test cases on the SUT and learns how to accomplish the intended test objective. Output after the learning convergence is the effective/efficient test cases to meet the test objective.

Currently, the supported format is the format of performance test cases. For example, SaFReL is a self-adaptive fuzzy reinforcement learning test agent that generates platform-based test cases, indicating the amount of resources' capacities that are going to be granted to the SUT during the performance test. While RELOAD, as a RL-driven load testing agent, generates the workload-based test cases, i.e., the workload that is submitted to the SUT during the performance test. The workload is defined in terms of the (HTTP) requests submitted to SUT, to perform a set of transactions.

It needs access to the SUT to be able to execute the test cases on the SUT. After executing each test case, it collects the status of SUT through some metrics such as response time, throughput, and resource utilization which can be collected either directly from the system (i.e. depending on the use case) or as processed data through another tool.

SaFReL can perform efficiently and adaptively on different software programs., i.e., CPU-intensive, memory-intensive and disk-intensive SUTs. It accomplishes the intended test objective, i.e., finding performance breaking point, more efficiently in comparison to a typical stress testing and can lead to test efficiency improvement. RELOAD learns the optimal policy to generate a cost-efficient workload to meet the test objective during an initial learning, then it is able to reuse the learned policy in later tests within the continuous testing context, e.g., for meeting further similar test objectives. It generates a more accurate and efficient workload to accomplish the test objective compared with baseline and random load testing techniques and once it learns, it is able to reuse the learned policy in further situations, i.e., testing w.r.t different objectives on the SUT and still keep the improved efficiency over later test episodes.

These smart test agents would fit the testing phase of DevOps. They learn how to accomplish the performance test objective for evolving releases of SUT during the DevOps practice and are beneficial for regression performance testing.

Solution situation at the start of IVVES

SaFReL was at the stage of "proof of concept" at the beginning of IVVES and RELOAD was developed in the first year of IVVES project. We have used Apache JMeter as an actuator and a monitoring system in connection with RELOAD. RELOAD is mainly intended for performance testing web applications. Apache JMeter is an open-source load generator/executor which also collects some performance metrics based on the received responses from the SUT.

Solution extension in IVVES

Some extensions in the learning part of RELOAD were done. RELOAD was developed and evaluated based on both Q-learning and DQN. More extensive empirical evaluation using different SUTs were carried out for RELOAD. An evolutionary search-based addition to RELOAD was developed in a master thesis project in IVVES.

4.2.5 RISE: Performance test analyzer

PerfTestAnalyzer was developed by RISE in collaboration with ABB. It is a performance test analyzer that utilizes various statistical and ML techniques to detect the suspect software builds leading to emergence of performance anomalies over the CI/CD process. The input to the tool is a test result file that contain the performance metrics measured and collected during the regression performance tests executed on ABB robot controller software. A set of certain performance test procedures are executed after merging the changes into the system--basically after every build of the system. PerfTestAnalyzer in its current version utilizes different statistical methods, principle component analysis (PCA), Unsupervised anomaly detection techniques, such as clustering algorithms, e.g., hierarchical and K-means w.r.t different distance and quality metrics and ISOForest together with visualization options to detect the software builds that potentially could lead to software performance anomalies (I.e., performance degradation).

It also benefits from various pre-processing techniques and state-of-the-art statistical methods, e.g., canonical correlation analysis, for dimension reduction, and finding highly correlated performance metrics in the performance test results.

Solution situation in IVVES

The solution was developed in IVVES project for ABB use case. A separate investigation and work on “Data-driven Software Performance Anomaly Detection Through Analysis of Regression Test Repository“ was also done within the body of the research and development phases of this tool.

4.2.6 Sogeti NL: Software quality in the development phase

Sogeti plans to develop a quality assurance solution in a form of a smart monitoring dashboard that assesses quality throughout the development phase. This solution will collect, analyze and visualize integrated data sources throughout the DevOps pipeline. The aim is to provide traceability amongst sources (Code changes, Test Cases, Defects); monitor key KPIs and machine learning results that drive the smart selection and automation of test cases; and to provide visibility over the entire software development cycle. This will result in compliant development, shorter release times, optimized allocation of resources and insightful reporting.

Some key inputs to the solution will be logfiles (change logs etc.) and other structured tabular data or an API structured output from other assessment tools like static analysis tools, code risk prediction tools and data quality tool metrics. The data that is fetched from logs or other tools provided by Bombardier, will be processed into a structured format. The output is a smart dashboard with visualizations and metrics that monitors and evaluates software quality.

The monitoring solution will also include the output from WP3, the code risk prediction tool & static analysis to determine code quality and thereafter can be used to prioritize test cases. The solution will primarily be used during QA testing and operation (pre-testing) phases of the software development lifecycle.

Solution situation at the start of IVVES

The purpose is to collect data from testing and results of ML models and visualize it offline and before execution, without developing probes or any other instrumentation. Infrastructure for transfer/communication of collected data depends on the use case. The data format is log files and tabular. The tool does not incur any overhead, nor does it provide anonymization.

Solution extension in IVVES

This is a new solution that has been developed within IVVES. Sogeti has worked with Bombardier to understand requirements and built a DevOps monitoring tool that can be used on-premises and in the cloud. The Infrastructure has been developed to be generally applicable, so that the tool can be used for additional use cases, working with e.g,, other programming languages and CI/CD pipelines.

4.3 Summary

The state-of-practice for the different use case providers differs, with the commonality that all have been focusing on data collection and mechanisms that can capitalize on recent AI and ML developments. While some of the partners were already advanced DevOps practitioners, others explored the possibilities of adapting DevOps in their domain while still fulfilling legislation, regulations and certifications that are required for their systems. The challenges and possibilities related to data varies as the systems comes from different domains, and thereby mixing IT and OT use cases with data collection from development and operations. Several use cases focus on applying AI/ML techniques to aid in improving the workflow during development with an emphasis on V&V activities, such as, test selection and test yields based on log-files, software versions and code coverage. During operations are anomaly detection a reoccurring topic that the use case providers state and interest in, for example, performance degradation and abnormal network and device behavior.

The tool and solution providers have been addressing various challenges from both development and systems in operation, by using AI and ML techniques. Sogeti, for example, has focused more on software quality assurance in the development phase with a smart monitoring dashboard that provides traceability and KPI monitoring, to drive the selection and automation of test cases. OU contributed with work on automated GUI testing, Praegus has been focusing on test optimization based on instrumenting the CI/CD pipeline, and RISE has worked on a regression and performance testing framework. Ekkono has been working on instrumentation and ML on embedded systems in operations at the edge.

To conclude, there has been extensive collaboration between the use case providers and the tool and solution providers in IVVES. Notable is that several interests overlap and in some cases partners even have multiple roles, where e.g., F-Secure and RHEA can be both use case providers as well as tool and solution providers.

5 Pattern Recognition (Task 4.2)

In T4.2, the main goal is to develop solutions for predictive maintenance, fault analysis and early anomaly detection. This will be done by using machine learning-based and statistical techniques for analysis of collected data, detection and construction of patterns and model inference for system behaviors (e.g., normal and abnormal operations) and common issues and failures in a system or domain. The development of solutions in T4.2 have been done in a way that can be coupled with the data collection techniques in T4.1 to be able to use the collected data as inputs in this task.

5.1 Use case providers

5.1.1 ABB: Customer program resource utilization in production

ABB is confronted with non-working customer programs due to using system resources over their limits.

Typical problems are:

- The customer program that is working on a previous version of the robot control software, is not working on a newer version, due to increased system resource utilization in the robot control software. This is applicable to customer programs that are utilizing a lot of the available system resources. The production will stop, causing production losses.
- It is important that ABB can measure and verify the utilization of system resource in-between releases of the robot control software and guarantee (as far as possible) that working existing customer programs will still work on newer releases of the robot control software (in terms of resource utilization).
- When a customer program is utilizing the system resources over the limits, it is hard for the customer to identify the cause. Thus, a bug is reported to ABB, while the problem could eventually be fixed by the customer instead. This is time consuming for customers and ABB.

Use case situation at the start of IVVES

ABB does semi-manual fault prediction. For each new build of the robot control software, ABB collects over 4000 different metrics from three different reference programs. Parts of the metrics are manually analyzed to find anomalies. It is mostly relative CPU utilization that is examined.

The knowledge of the behavior of the resource utilization under various high load cases is limited.

The system contains about 3.2 million lines of code, 7000 files, 65000 functions. Its architecture has evolved over the last 30 years.

Challenges to be addressed in IVVES

ABB can only access data from customer systems with the permission from the customer.

Achievements and improvements in IVVES

RiSE has developed an AI/ML analyzer tool to help find deviations in metrics data between software builds. The current implementation will make it easier to detect builds that contains changes that suspiciously affects resource utilization. We are still assessing the results of the analyzer tool and investigating how it can help beein a data-driven decision support tool for then engineer of our system.

Manual analysis of various high load cases has also been performed to better understand the limits of the software and the various hardware platforms, including platforms under development. The manual analysis has been important sources for some of our bigger development initiatives (both hardware and software) the latest years, but it has also given us knowledge to use when assessing the analyzer tool.

Furthermore, continuous manual analysis and data collection of the continuous builds has been done, which also helps in the assessing of the analyzer tool.

5.1.2 Bombardier: Data collection and analytics in Engineering and Operations

Bombardier wants to improve its decision making and support. There is still a need to discover typical quality issues in the systems. As reliability and availability is key it is important to continue research, in order to as early as possible detect any quality issues with availability, performance and security.

Use case situation at the start of IVVES

The system contains about 1M lines of code. The health of the system is measured with rule-based monitoring based on domain knowledge using the diagnostic systems ADDTRAC and TMS from BT, that Addiva currently runs, maintains and offer as a predictive maintenance service to train operators in e.g., Scandinavia, India and Singapore.

BT would like to cooperate more with Addiva on defining data needs for future service offerings and improve the precision of the current predictions.

Challenges to be addressed in IVVES

There are many challenges but mainly around data collection and sharing. Live data from common customers is available but must be treated as confidential. Addiva, as partner in this work, handles the data and decides on anonymization and data sharing within IVVES.

Achievements and improvements in IVVES

Together with RISE we have defined key characteristics on performance degradation and identified generic application independent indicators for anomaly detection of this.

5.1.3 ING: GUI testing in the acceptance phase

Use case situation at the start of IVVES

In the initial situation the GUI testing is a manual and time consuming effort. Its system consists of hundreds of interacting systems that are engineered by thousands of people. The ING applications are used by millions of daily individuals.

ING believes that the effort spent on GUI testing can be reduced because there is lots of data available on how tests are done in detail. ING is currently unable to employ that data in any other way than using the data as it is. ING believes there must be a way to extract some sort of generic patterns or features of this data and apply it in such a way that its testing processes require much less manual effort.

The current way of working is that engineers build tests consisting of detailed execution plans that are then automatically executed. Failures mean an unhealthy system, which is easy to interpret. However, the absence of failures has limited it to no value.

Challenges that are addressed in IVVES

The absence of a failure does not indicate the high quality of a system. A user typically does not interact with a system in a random way, making automated testing of a GUI application very hard. Next to that, the assumptions a user has are also important. For example, functional characteristics (i.e., how to interpret the data shown on a screen) can be very important for deciding whether or not a fault occurred, but it is very hard to interpret automatically by a system.

Achievements and improvements in IVVES

To address these aforementioned challenges, ING sees opportunities in moving to a script-less based testing approach instead of the current script-based approach where each test step to be executed is encoded into a test script. ING selected the Testar scriptless testing tool provided by OU. The figure below shows the generic architecture on how to the IVVES solution applies to the ING use case.

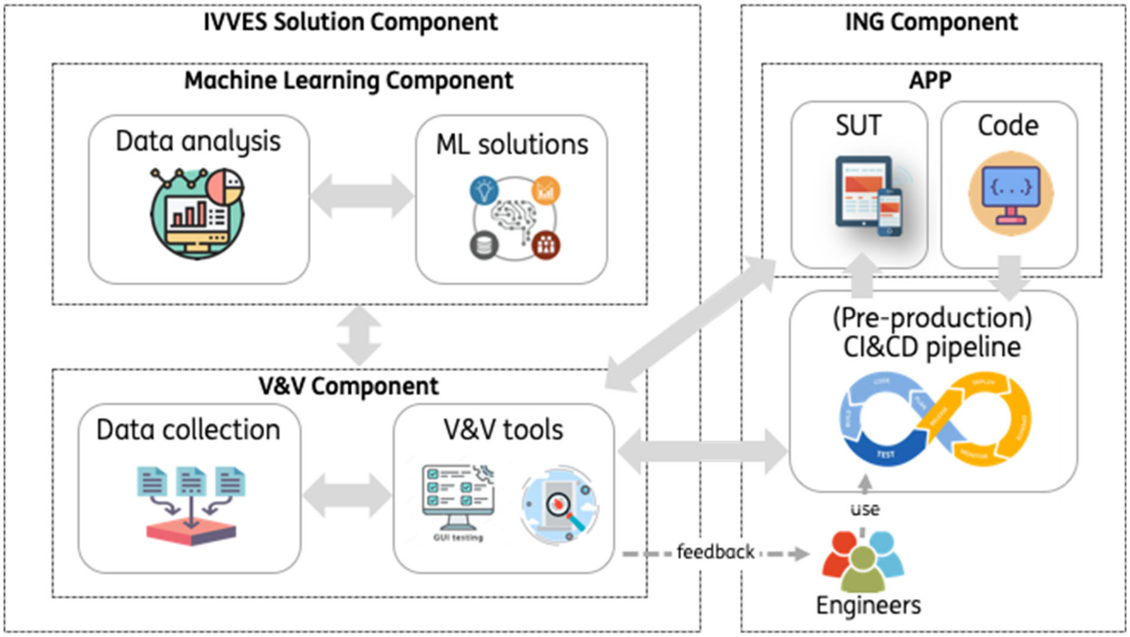


Figure 3. Generic architecture for the ING use case in IVVES; IVVES Solution Component used it TESTAR (provided by partner OU) with a proprietary ING extension.

In this context four pilots were done within ING (Mobile and Web). Software quality issues were found.

One pilot on the Mobile banking app with main question: Can test coverage be improved with scriptless GUI testing? This turns out to be the case and matches with IVVES KPIs. A MSc thesis “mTESTAR for scriptless GUI testing on Android and iOS application” and paper “Scriptless GUI testing on Android and iOS applications” (under submission to ISSTA) detail the results. A mobile extension is created for TESTAR to connect this experiment/pilot.

Three pilots were done for Web with 1 Belgian and 2 Dutch ING teams to find software quality issues in their web software product using TESTAR with ING extension. This addresses the test effectiveness KPI for IVVES and ING. Software Quality issues were found. Using the TESTAR tooling has low maintenance costs, which is one of the cost/economic drivers of ING and test effectiveness KPI of IVVES.

A challenge is ease of (technical) integration. This is a big success factor for adoption within ING.

5.1.4 Philips NLD: Using AI to enhance the verification flow

Use case situation at the start of IVVES

Currently the impact analysis of changes as input for the test plans and the creation and execution of the planned test is mostly a manual process. Although 100% requirements coverage is achieved with every release, issues can still be reported from the field.

Philips NLD wants to improve its test coverage by bringing it closer to real customer use and as such improving the quality of its products and reduce defect slippage. Next to this, Philips NLD want to improve its verification and validation efficiency, by e.g. reducing the number of verification and validation cycles, optimized/automated test case selection and automation test case creation and execution.

Typical quality issues reported are functional errors, but also non-functional errors like performance and reliability are reported. Philips NLD already uses predictions for their defect inflow during development and runs reliability tests to predict reliability in the field.

Philips NLD is using test-(progress) status, defect statistics, performance and reliability measurements for our in-house test configurations. Tools are developed in house. The size of the system is confidential information.

Addressed challenges in IVVES

Challenges that have been addressed were:

- Bringing verification coverage closer to real customer use while still ensuring 100% requirements coverage to a.o. adhere to regulatory requirements.
- Reducing the manual workload by automating impact analysis of changes, test planning and test execution activities.
- Reducing the defect handling lead-time by automatically assigning defects to the appropriate development teams.

Achievements and improvements in IVVES

We have established a framework for continuous test coverage improvements supported by automated tooling:

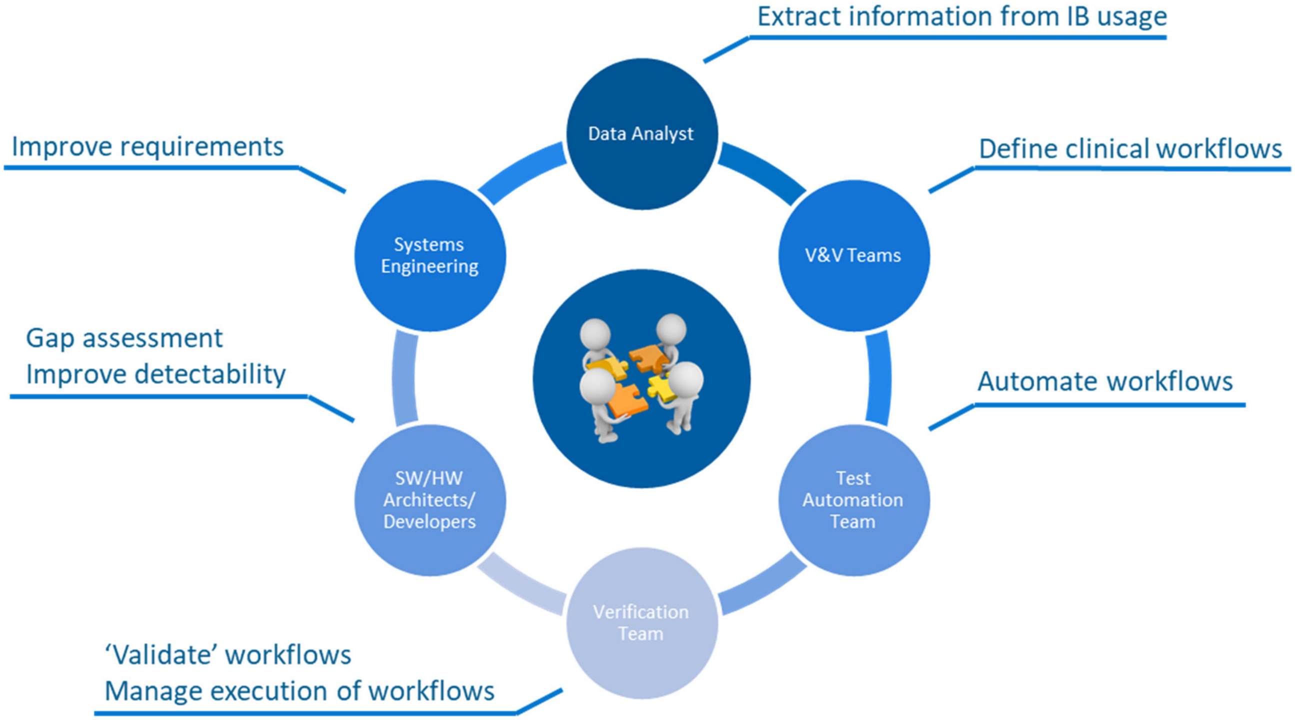


Figure 4. Continuous Improvement Framework.

The framework has following benefits:

- Automatic creation of typical scan protocols as part of the clinical workflows used in our installed base, for all different anatomic regions
- The scan protocols are used in the automated regressions suites for basic regression and reliability as part of the daily CI pipeline
- The scan protocols are used in the formal verification specifications for the different clinical workflows

We have implemented a BERT (Bi-directional Encoder Representations from Transformers) neural network as an intermediate step to automate the defect classification and allocation. The model has been trained on defect headlines with both a maximum vocabulary of 50 words and 256 words. The first pilots for the current test set resulted in total accuracy close to 60% when trained with a vocabulary of 50 words. Total accuracy achieved when trained with a vocabulary of 256 words was slightly higher.

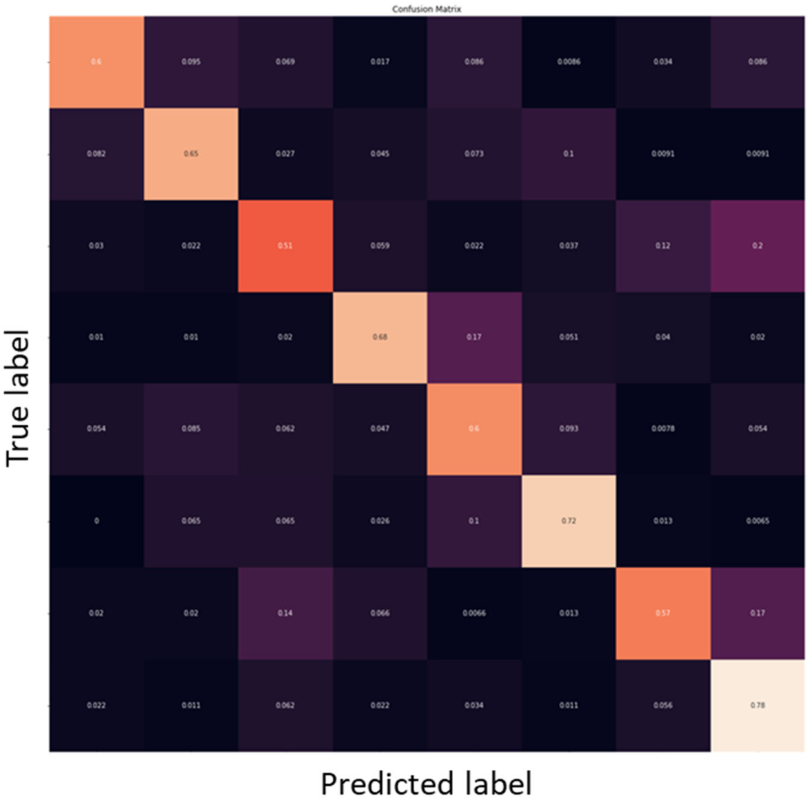


Figure 5. Confusion matrix for training vocabulary with 256 words.

5.2 Tool providers

5.2.1 Ekkono: Fault analysis and anomaly detection

Ekkono’s tool provides fault analysis and anomaly detection based on their own Change detection algorithm. The analysis is not automated but needs to be tailored for the problem at hand.

Tool situation at the start of IVVES

The tool enables predictive maintenance by training models on labeled or unlabeled data. Most often it does predictive maintenance by monitoring deviations from a normal state using our Change detection. Any input format is accepted, both in the form of multi-dimensional signals or a one-dimensional signal. The tool only delivers predictions and use existing communication stack to communicate these predictions.

The tool provides several ML algorithms: linear regression, random forest, multilayered perceptron, conformal prediction, Auto-ML.

Solution extensions in IVVES

With regards to Task 4.2, Ekkono has extended their tool with the following:

- Improvements to the change detection algorithm to work both as change detector on a raw signal, and as a concept drift evaluator on top of any predictive modeling algorithm, monitoring the error of the model in real time.
- New anomaly detector for detecting abrupt anomalies in real time.
- Conformal prediction framework, presented in WP3, to evaluate that the predictions made by any model is done with some degree of confidence.

5.2.2 OU NL: GUI testing with TESTAR

TESTAR is a GUI testing tool aimed at the testing phase but can possibly be used for run-time monitoring in production as well. It provides a measure of quality by doing robustness testing.

Input for TESTAR tool is a way for the tool to connect to the GUI application under testing (for example path to the executable or URL), and optional system specific configurations in form of settings file and possibly a Java file. The output includes test results and the generated test sequences, optionally inferred state models in OrientDB graph database, and logs.

Tool situation at start of IVVES

At the start of IVVES project, TESTAR already provided support for inferring state models stored into OrientDB graph database that provides a documented API to query the results.

TESTAR does not support anonymized data and does not include any data cleaning procedure.

TESTAR’s state model inference and action selection are already using some ML techniques.

Solution extensions in IVVES

- Regarding Task 4.2: Pattern Recognition, TESTAR has been extended with: configuration allowing custom abstraction level for both states and actions for the model inference,
- predicting transitions of GUI actions without executing them, based on similar actions from other states of the inferred model.

5.2.3 Praegus: Test duration optimization in the testing phase

Praegus provides a tool named Orangebeard which aims at giving insight in test performance and optimize testduration based on SCM inputs, build output and test reports.

By helping a DevOps team to focus on the important tests with the highest business impact, Orangebeard indirectly helps to improve the quality of the System under Test (SuT)

Tool situation at the start of IVVES

Orangebeard performs fault analysis of failed automated tests during the build pipeline (CI/CD pipeline). It performs RCA based on correlations with previous failures, that were subsequently categorized by a test oracle (i.e. one of the testers of the team).

OrangeBeard accepts JSON input via HTTP Rest API’s and provides dashboards that indicate what tests have low added value in the build pipelines, in order to improve maintenance costs of testsets.

The tool can work with anonymized data and supports both quantative and qualitive analysis. It also proactively filters sensitive data like passwords from build output.

Solution extensions in IVVES

In IVVES Praegus extended Orangebeard to provide the module "Auto Test Pilot", to be able to perform basic subsetting and prioritization of testcases against the relevant context. At this point we are able to collect relevant test output and change metadata from different sources. Next we worked to combine this in our machine learning model. Praegus is still experimenting with different ML algorithms. The illustration below shows where we want to go.

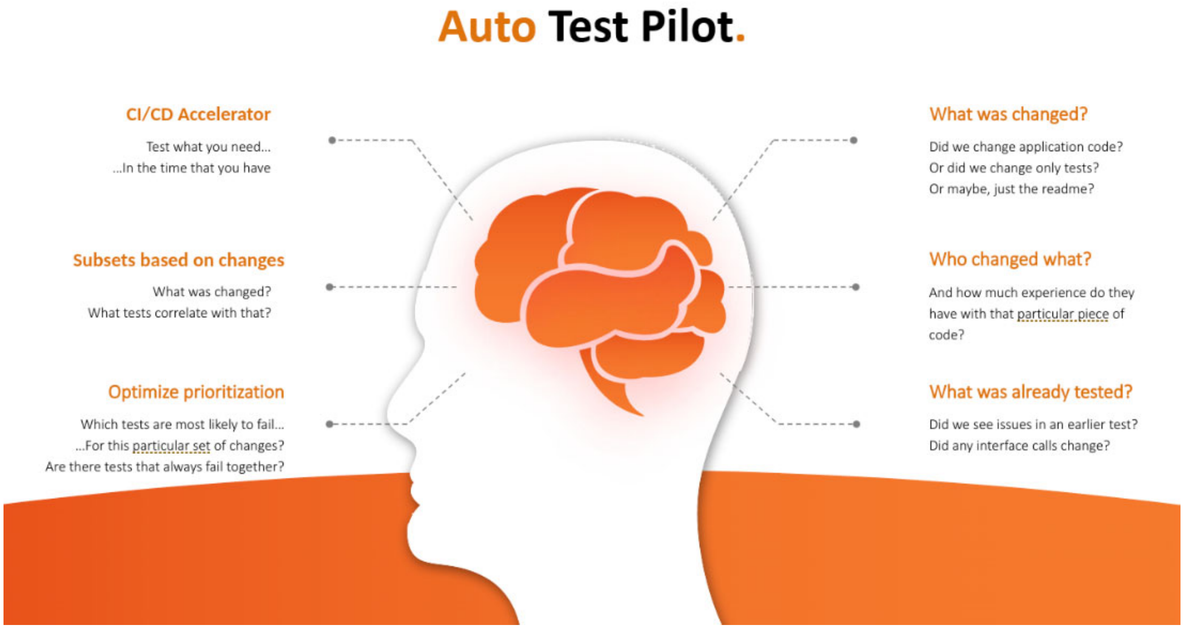

Auto Test Pilot.
CI/CD Accelerator
Test what you need...
...In the time that you have
Subsets based on changes
What was changed?
What tests correlate with that?
Optimize prioritization
Which tests are most likely to fail...
...For this particular set of changes?
Are there tests that always fail together?
What was changed?
Did we change application code?
Or did we change only tests?
Or maybe, just the readme?
Who changed what?
And how much experience do they have with that particular piece of code?
What was already tested?
Did we see issues in an earlier test?
Did any interface calls change?

Figure 6. Auto Test Pilot overview.

5.2.4 RHEA: Detecting abnormal network behaviour

RHEA wants to reduce the false-positive rate in cybersecurity threat alerts and bring greater confidence to the detections of abnormal network behaviour.

Typical quality issues are the false positives in cybersecurity threat alerts. Other issues are mainly performance: It takes a lot of manpower to handle streams of alerts with high percentage of false positives.

However, lowering false positives would also decrease true positive detection rate and lower security.

Tool situation at the start of IVVES

rapidPHIRE (RHEA's Threat Hunting technology) only checks one element from a relatively small subset of possible data points within a connection to determine if an IOC (Indicator of Compromise) exists. When there is a match for a single data point the likelihood of a false-positive is increased.

The overall system health, not related to any detections, is handled by in-house scripts that report back to a data collector.

Missing metadata is usually the most noticeable indicator that there is a problem. Failures with internal processes that manipulate the metadata can also be detected.

While no actual line count has been performed, RHEA estimates there are about 50,000 lines of code across 20-30 components

Solution extensions in IVVES

rapidPHIRE extended to be able to make better determinations even if only a single data point is checked (i.e. we has been working to improve detection performance based on multiple data elements that are not tied to single-data-point threat lists).

5.2.5 RISE: Performance testing with SaFReL/RELOAD

SaFReL/RELOAD is a reinforcement learning-driven performance testing framework consisting of two smart test agents: SaFReL and RELOAD. The test agents are able to effectively and efficiently generate critical performance test scenarios leading to the emergence of performance breaking points. They learn the optimal policy to accomplish the test objective without relying on the system model or source code of SUT. These test agents assume two phases of learning: initial and transfer learning. During the initial learning, they learn an optimal policy to meet the test objective. Once they learn, they are able to reuse the learned policy in further situations, e.g., testing other similar test subjects (SUTs) or to meet other test objectives on the SUT. SaFReL generates platform-based test scenarios, while RELOAD generates a cost-efficient test workload that results in the emergence of the performance breaking point [6,7,8].

These test agents do not need a set (batch) of training data. They work based on continuous interaction with the system under test (SUT). It implies that the agent executes several test scenarios on the SUT and learns how to accomplish the intended test objective. These test agents realize the test automation without relying on models or source code. Furthermore, knowledge formation and the capability of reusing learned policy leads to test efficiency improvement.

Tool situation at start of IVVES

The supported format is the format of performance test cases. For example, SaFReL generates platform-based test cases, indicating the amount of resources' capacities that are going to be granted to the SUT during the performance test. RELOAD generates the test workload that is submitted to the SUT during the performance test. The workload is defined in terms of the (HTTP) requests submitted to SUT, to perform a set of transactions.

The framework needs access to the SUT to be able to execute the test cases on the SUT. After executing each test case, it collects the status of SUT through some metrics such as response time, throughput, and resource utilization which can be collected either directly from the system (i.e. depending on the use case) or as processed data through another tool.

The framework fits the testing phase of DevOps. It learns how to accomplish the performance test objective for evolving releases of SUT during the DevOps practice. It is beneficial for regression performance testing.

Solution extensions in IVVES

Building upon the same learning paradigm, we extended the proposed RL-assisted performance test framework by developing a customized smart test agent for ABB robotic use case. The proposed test agent can generate effective performance test scenarios for performance testing of a robot control software.

5.2.6 RISE: DeepAD: Deep Learning-based Anomaly Detection

DeepAD is an LSTM-Autoencoder based approach that detects anomalies in multivariate time-series data, generates domain constraints, and reports subsequences that violate the constraints as anomalies. The loops in the LSTM structure allow information to persist and make the network learn sequential dependencies among data records. This approach can model non-linear long-term sequential dependencies among the data records in univariate/multivariate time series, which makes them more practical for real-world applications.

Tool situation at the start of IVVES

This tool was initially in early development phase. RISE has been working on LSTM-Autoencoder architecture optimization for the sequence data provided by Bombardier in time-series format.

Solution extensions in IVVES

As part of the development of this tool, RISE has collaborated with Bombardier to extract the abnormal conditions in their system which can lead to performance degradation and service failures. The model developed in this tool helps in extracting the patterns in historical data and uses the extracted patterns to predict future patterns.

5.2.7 Sogeti NL: Software quality in the development phase

Sogeti aims to develop a QA solution (smart monitoring dashboard) that collects, analyses, visualizes, monitors and evaluates software quality by combining logfiles and other structured tabular data or an API structured output from other assessment tools. The aim is to provide traceability amongst sources (Code changes, Test Cases, Defects); monitor key KPIs and machine learning results that drive the smart selection and automation of test cases; and to provide visibility over the entire software development cycle. This will result in compliant development, shorter release times, optimized allocation of resources and insightful reporting.

Tool situation at the start of IVVES

This tool is currently in the PoC development phase. Sogeti is developing the cloud architecture while Bombardier is providing the data.

Solution extensions in IVVES

As part of the development of this tool, Sogeti has collaborated with Bombardier and utilized the Sogeti built tools in other Work Packages:

- WP3: Code risk prediction and static analysis tool metrics to monitor code quality and prioritize test cases efficiently and effectively.
- WP4: integrate data sources and visualize key KPIs, metrics and aforementioned data quality, model quality and code quality results. Provide a holistic software evaluation report.

5.2.8 SII CONCATEL/NETCHECK: V&V for ESG investment system

With respect to data collection techniques, there are a set of tools and components developed by SII CONCATEL for DevOps pipelines. These components are analyzing logs, tickets and source code to provide insights and improve SLA management. The components are interacting with a knowledge graph, focusing in easily providing the context related to any issue.

Tool situation at the start of IVVES

SII CONCATEL and NETCHECK have specialized technical teams with a high experience in the verification and analysis of software to ensure its maximum efficiency and quality. SII CONCATEL offer Testing & QA Factory with up to 140 testers, with offshore and nearshore approaches, based on different SLAs and using different tools in the market.

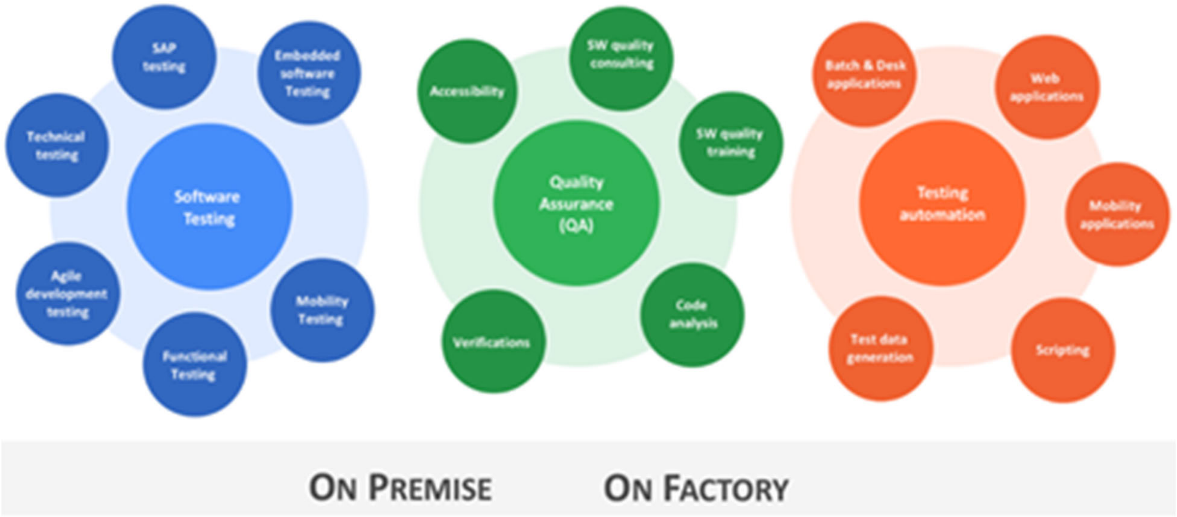


Figure 7. Implemented lifecycle integration services.

Within the SW factory, there are different lifecycle integration services implemented. These services are covering key processes within DevOps lifecycle:

- Test cases design and load.
- Verification and execution of test cases.
- Execution Verification & Validation.
- QA.
- Business Process Test.
- Test Automation.

However, there is not an MLOps pipeline available, and it is mandatory to generate the proper toolchain for ESG environment.

Current Tool situation

The tool is composed of several components which are organized as shown in the following figure:

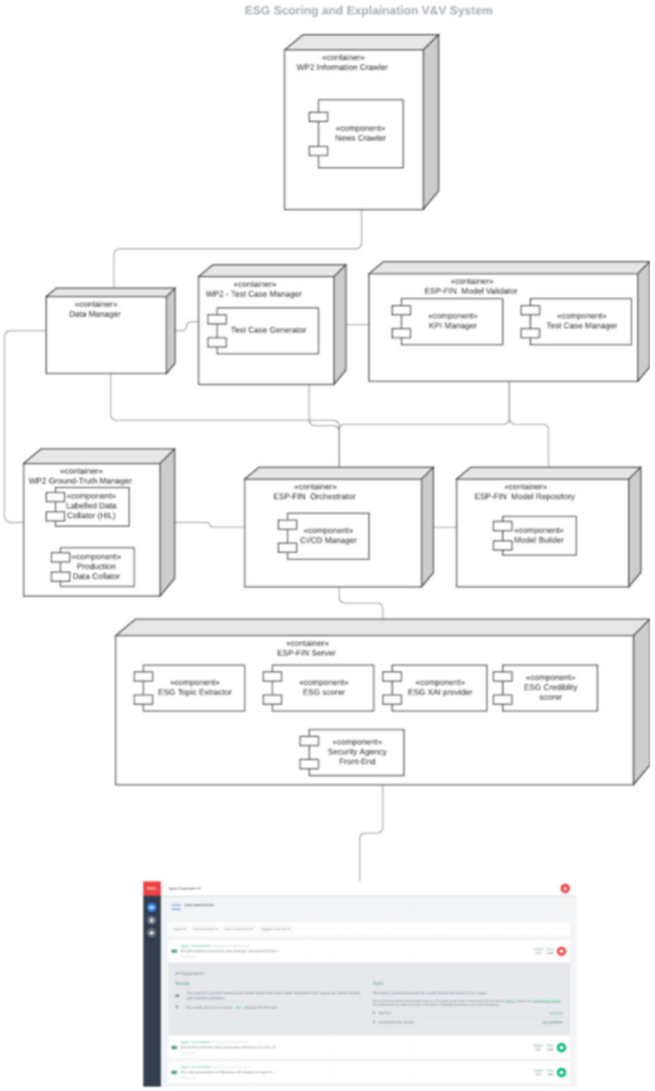


Figure 8. ESG Scoring and Explanation V&V System

Main components:

- Information Crawler

This component is in charge of automatically crawling information related to ESG. The crawler will get information from online sources. The information will be stored in the Data Manager.

Current status:

The information crawler component work in two different phases. In the first phase the crawler search for news urls. For this phase two components have been developed,

- News searcher. It fetches urls using a search term and a time span. It returns basic information such, news title, source, url and publish date.
- GDelt crawler. (https://www.gdeltproject.org/) The second component downloads the gdelt event database in a daily basis and return the same basic information.

The second phase consists in augmenting the basic information. It uses the url provided and crawls the page looking for additional information such article author, title, and full text. Then a NLP pipeline augments the information again providing, summary and keywords.

A final version of the services has been implemented

- Ground truth manager

  This component is responsible for ingesting information related to other sources (e.g. Reuters ESG data) and provides a HIL interface to confirm (and/or improve data labeling). The information will be stored in the Data Manager.

  Current status:

  Several components have been developed.

  - Frontend Component: Provides a user interface to retrieve news from information tracking services and persists them as nodes in a graph database, communicating with the Data Manager. It then provides the necessary tools to annotate the data.
  - Backend Component: It handles all communication between the user interface, the information tracker and the graph database.

  A final version of the services is ready to be used and right now a team is using it to annotate ground truth data. A manual ESG news validation system has been added to improve the training of the algorithm.

- Data manager

  This component manages all data repositories including the graph database server.

  Current status:

  The current version of DM is already managing all items tracked by the information tracking component and ground truth manager annotations.

- Test Case Manager

  This component generates test cases to monitor and validate model performance.

  Current status:

  Development has been centropipelines of text generation with autoregressive models and obtaining first results.

- ESP-FIN Model Validator

  This component monitors the models, generating the most efficient test case for each model.

  Current status:

  The basic components have been developed and are ready to handle the models to be validated and tested. It relies heavily on the Test Case Manager.

- ESP-FIN Model Repository

  This component manages the different models that are being generated and updated. The model will be validated and monitored by the "ESP-FIN Model Validator". Once a version of a model meets the defined KPIs, it is set as ready for deployment on the ESP-FIN Server.

  Current status:

  A final version has been generated.

- ESP-FIN Orchestrator

  This component is responsible for managing the CI/CD pipeline and coordinating communication between the other components.

  Current status:

  A final version of the component has been released.

  - ESP-FIN Server

  This component is responsible for hosting the models that will receive the front-end requests.

  Current status:

  This component is ready to serve the models to be implemented.

Solution extension in IVVES

The main objective has been to integrate and orchestrate all the ML-based components from the SUT, together with the components and tools to be developed for credibility assessment, explainability and test generation and prioritization within a cohesive MLOps workflow, providing logs analysis to identify anomalies and performance degradation. In order to integrate all the components as part of the Testing and QA Factory, the solution was extended to consolidate the results as part of solid, well-established solutions. In this sense, MLflow has also been considered as reference platform.

5.3 Summary

Task T4.2 focuses on predictive maintenance, fault analysis and early anomaly detection. For the use case providers who got involved in this task, ABB and Bombardier will address their challenges on how to analyse the collected data for predictive maintenance and fault analysis; ING and Philips NLD will mainly focus on using AI/ML techniques for early anomaly detection. The status of different partners in IVVES also differs: ABB and ING will collaborate with other tool providers to work on new solutions for their challenges. Bombardier and Philips NLD will improve their existing solutions using AI/ML techniques.

For the tool providers, Ekkono, RHEA, Pragues, and Sogiti have been working to improve their existing tools/solutions on fault analysis or anomaly detection using ML techniques. On the other hand, OU NL, RISE, and SII Concatel/Netcheck focused on testing area: AI-based GUI testing (OU NL), performance testing (RISE), and quality assessment (SII Concatel/Netcheck).

Tool developments for Fault analysis/anomoly detection:

Ekkono has been extending its tool for fault analysis and early anomaly detection with FFT, Hoeffding trees, Lasso, federated learning. RHEA has been improving its tool for cybersecurity thread alerts. Pragues is improving the test duration of its tool and extending its fault analysis by ML experiments, mainly with random forest and support vector machines. Sogeti developed a QA solution (smart monitoring dashboard) that collects, analyses, visualizes, monitors and evaluates software quality by combining logfiles and other structured tabular data or an API structured output from other assessment tools.

Tool developments for Testing:

OU NL has been improving the effectives of its GUI testing tool by trying out several ML algorithms and also extending it for usage with web and mobile applications. RISE extended its performance testing tool with a smart test agent. SII Concatel/Netcheck improved its tool with an MLOps pipeline considering MLflow as a reference platform.

6 Data Analytics in Engineering and Operation (Task 4.3)

This task as a whole, deals with AI-based development of evolving systems. In this task, overall methods and techniques, guidelines, and core services to enable data-driven engineering have been be developed. This has been achieved by exploiting and developing data analytics techniques to detect correlations, and identify and establish (semantic) relationships among data artefacts with the purpose of provide recommendations for developing and engineering of evolving systems. For this activity, T4.3 also benefited from the pattern recognition solutions that are developed in T4.2. The main focus has been to support development of systems with fewer faults and unwanted behaviors while improving their quality attributes e.g., in terms of availability and reliability. This task also addressed operational requirements through the development of strategies and guidelines for service deployment and orchestration in Mobile Edge Computing (MEC), Fog and Cloud computing resources depending on performance, reliability and scalability requirements. The summary of the challenges and expected outcomes from use case providers and tool providers are shown below as follows.

6.1 Use case providers

6.1.1 ABB: Customer program resource utilization in production

Use case situation at the start of IVVES

Regarding data analytics and feedback from development and operation, ABB is currently performing a manual inspection of the measured data. Some single values are verified automatically, by checking against threshold values or standard deviation. There is no semantic analysis of the relationships between data artifacts, neither feedback generated for future iterations in design or engineering. Traceability of new features is done through the overall lifecycle from demand to test: demand, implementation, specification and testing. However, mostly all legacy code lacks of traceability.

Tools and Toolchains already available

ABB has been working on Tracelyzer from Percepio for some preprocessing of the raw data. Also, an in-house tool is used to compile the raw data into a database and do some basic analysis.

Challenges identified and improvements from IVVES

ABB still lacks good means to get resource utilization live data from customers. This is something that will not be addressed within the scope of IVVES.

During IVVES we have improved the way we store and prepare RAW-data to increase scalability. The changes have been enough for the testcases and hardware platforms that we have added during IVVES but is not enough for future needs. We plan to further improve this in future development.

Furthermore, there is an ongoing work to improve traceability between legacy requirements, including non-functional requirements, and test cases. As part of this work, we identify gaps that should be tested for resource utilization as an addition to all test cases we already have that should include resource utilization tests.

The goal of the work is to automate the whole chain, also making resource utilization testing fully automated. Hopefully, the AI/ML analyzer tool will be an important component in the automation.

6.1.2 Bombardier: Data Collection and analytics in Engineering and Operations

Use case situation at the start of IVVES

BT has started evolving from a traditional V-model to a more agile approach. Even though some features are available for traceability, they are typically tool-specific (e.g. Doors, RTC, SVN).

Challenges identified and improvements from IVVES

We have made significant investments in new DevOps environment and tools to move to a hybrid environment with both virtual and physical devices. Here we can simulate system of systems and provide a much more automated development, verification and validation process with shorter lead time and improved software quality. This is the basis for a more agile process and still adhere to traceability as required by safety and security standards for V&V.

Here working with e.g., Sogeti on CI/CD pipeline analysis and anomaly detection with “DevAssist”. Together with RISE we have developed test case optimization tools.

6.1.3 F-Secure: Simulating OT networks

Use case situation at the start of IVVES

F-Secure is currently using proprietary data analysis techniques. These techniques have been developed within F-Secure to map the OT attacks and their steps. Semantic analysis is based on graph analysis of the attack paths. This is being used for analyzing relationships between the equipment and data. The learnings got, are fed back to the F-Secure AV development teams and consultants for providing better results in the future.

Tools and Toolchains already available

F-Secure is working with its own, proprietary OT attack analysis tooling.

Challenges identified and improvements from IVVES

The collected customer data is strictly confidential and cannot be taken out of the customer premises for analysis, hence simulated data needs to be used. F-Secure has been working to configure simulation rig to one or more use cases from industry partners in order to 1) simulate their facility networks and attacks targeted those 2) improve and build new simulation techniques and methodologies. Configuring the simulator rig for specific use cases from other partners would improve existing simulation techniques and methodologies.

6.1.4 ING: GUI testing in the acceptance phase

Use case situation at the start of IVVES

Regarding analytics and feedback from DevOps, currently ING is not using automated data analytics techniques for GUI testing. For this use case, ING is currently working with manual techniques or scripted based testing. A fault on GUI is detected because a test execution yields an unexpected state. Then the engineer who created the test looks at the results, and decides whether it indeed is a failure or not. The worst case of such a fault is detected by our real customers. Therefore, feedback is sent to engineers only in case of failure.

Challenges identified and improvements from IVVES

ING is willing to reduce the amount of required interactions and make the knowledge explicit. One of the main goals is to improve GUI testing, generating intelligent scriptless testing that does not employ monkey testing, but instead takes real user characteristics into account.

One of the challenges identified is that nightly test runs need developer interaction to detect false positives. The tool generates a test result report, which reviewers review. False positive rate needs to be low in order to increase adoption by developers.

Improvements are future work outside of scope of the IVVES project. Possible improvements concern improving the reporting of the tool and define a way of working for handling false-negatives and the feedback of the team, in order to establish a feedback loop with the development team.

6.1.5 Problem area of use case provider Philips Netherlands

Use case situation at the start of IVVES

Philips Netherlands is currently performing process mining. Even though some semantic analysis is done, more sources could be integrated. The feedback provided to engineering and design phases is based on operational profiles, defects and reliability metrics.

Traceability between requirements, test cases, test results and defects is managed, but traceability to code is not in place yet.

Tools and Toolchains already available

Philips Netherlands is currently working on developing scripts, created in Python and Fluxicon Disco mining tool.

Challenges identified and improvements from IVVES

The main challenge identified ensuring detectability of required information in the logfiles. By driving logging improvements, combined with data analytics to label and enrich current data we are able to automatically create and generate scan protocols for all anatomic regions. The automated processing pipeline allows us to update these protocols on a daily basis based on changed customer use. To further enhance the workflows with e.g. clinical post processing flows we're planning further enhancements in our logging.

As we have automated execution of the clinical workflows, we're now able to measure code coverage. Running full coverage measurements at system level is having too much impact on performance and behavior. Therefore we plan to run scripts in cycles where we measure coverage for different parts of the software. Area to further work on is to aggregate results in one overview, so we can compare coverage achieved with changes made in the archive to plan next steps how to address gaps.

6.1.6 Problem area of use case provider RHEA

Use case situation at the start of IVVES

RHEA's use case has not been providing analytics and feedback from DevOps.

Tools and Toolchains already available

There are no tools related to these processes in the current situation.

Challenges identified and improvements from IVVES

The main challenge has been that Non-SQL database makes it more difficult to perform meaningful queries against the data. Thanks to IVVES, a better exploitation of data that improves threat detection and performance of analysts has been defined.

6.2 Tool and Solution providers

6.2.1 Ekkono: Fault analysis and anomaly detection

Tool situation at the start of IVVES

Ekkono is providing a general ML library for edge devices, focused on increasing the success rate of ML-based projects. Ekkono provides a solution for edge machine learning deployment.

Solution extension in IVVES

Regarding Task 4.3, Ekkono has extended the CRISP-DM framework into a refined version for edge ML applications, with specific steps focused on the deployment phase of ML models (MLOps). Ekkono has also verified it on real applications through different customers.

6.2.2 Praegus: Test duration optimization in the testing phase

Tool situation at the start of IVVES

Praegus is applying elastic-search based pattern matching that guides the RCA-process.

Solution extension in IVVES

In IVVES Praegus has developed the so called 'Auto Analyzer' in Orangebeard, that recognizes defects that have previously occurred by comparing testrun output with historical data. We use OpenSearch (AWS) to create and match indexes.

6.2.3 Sogeti NL: Software quality in the development phase

Tool situation at the start of IVVES

Sogeti aims to provide AI-driven quality assurance techniques for the testing of AI systems and non-AI systems. This involves both traditional and non-traditional approaches to testing & development. The objective is to improve the reliability of the system under test by automating quality checks. Sogeti will also provide a quality evaluation framework to assess software before production. In turn, AI-driven engineering will accelerate the validation and verification of evolving systems.

Solution extension in IVVES

The final version of software includes:

- Prediction of features in software development phase that have high probability of defect.
- Monitoring and evaluating DevOps integrated pipeline for smart test selection and accelerated QA.

Architecture of solution

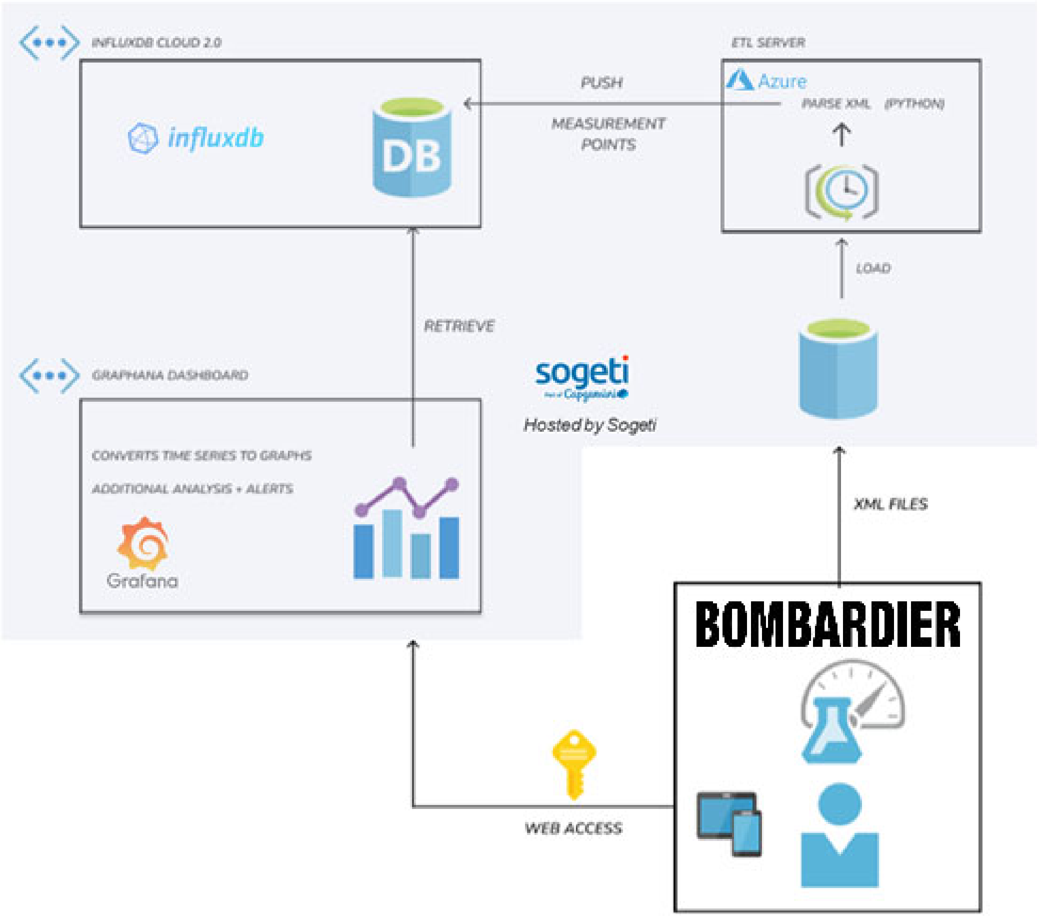


Figure 9. Solution architecture.

6.3 Summary

The increasing amount of available data from operations is providing new opportunities for AI-based development of evolving systems. Data acquisition, processing, and AI-powered interpretation allows us to make better design decisions and build better systems. The findings of the services and components developed during this task, will eventually feed the design and development phases. Initially, almost all the partners started from an early stage regarding the data-driven engineering, though some of them are have been working on the semantic analysis of data. In all the cases, the integration of new sources have been planned, as a step to eventually provide data-driven engineering capabilities.

7 Data-Driven Engineering: Reflections and Adoption

In this section, we provide a summary and highlights from the experiences of the IVVES project in adopting a data-driven engineering approach to enable a smarter and more automated decision-making process in development of software-intensive systems. Along with the achieved benefits and improvements, we also report on the challenges that partners faced in their journey towards adopting a data-driven engineering approach for their industrial use-cases.

7.1 Data-Driven improvements in IVVES

To understand and summarize how IVVES solutions and tools in WP4 have improved the challenges of project partners in moving towards a (more) data-driven engineering solution, a mapping was designed and performed. The mapping was done based on a reference model extending DevOps with Data Management and ML Modeling processes as described in more detail in the following section. We then asked use-case owners to mark which of the phases with respect to the provided reference model, IVVES solutions have provided improvements in the context of their use-cases and in particular which additional phases they can now cover thanks to IVVES tools and technologies. Similarly, tool and solution providers were also asked to mark the capabilities of their solutions with respect to the additional phases of the provided reference models that can now be covered by their tools thanks to the extensions and additional features they developed during the IVVES project.

7.1.1 Extended DevOps reference model

Developing ML components increases the complexity of the development process, as a central part of ML is training and iterations to find the best prediction model. Software development processes like DevOps also are iterative, describing how development and operation activities are related. These two processes need to be combined in order to efficiently deploy ML based applications in real systems.

Figure 10 shows a model of an integrated ML workflow and DevOps process by Lwakatare, I. Crnkovic and J. Bosch [9]. The model consists of:

- DM – Data Management focus on data collection, selection, labelling and cleaning features, involving e.g., extraction in specific formats, access control to data, and dataset building code
- Mod – ML Modelling starts once new datasets are available for training, and evaluations, this includes for example, comparisons with historical baselines from production, specifying data dependencies, and verifying the ML component in isolation
- Dev – Development takes over as the ML model is deployed, followed by code verified to work and integrate with the rest of the system, and finally passing normal integration and system test
- Ops – Operations of the ML model and the rest of the system in the production environment include monitoring the trained ML model and collect live information that can trigger retraining the model

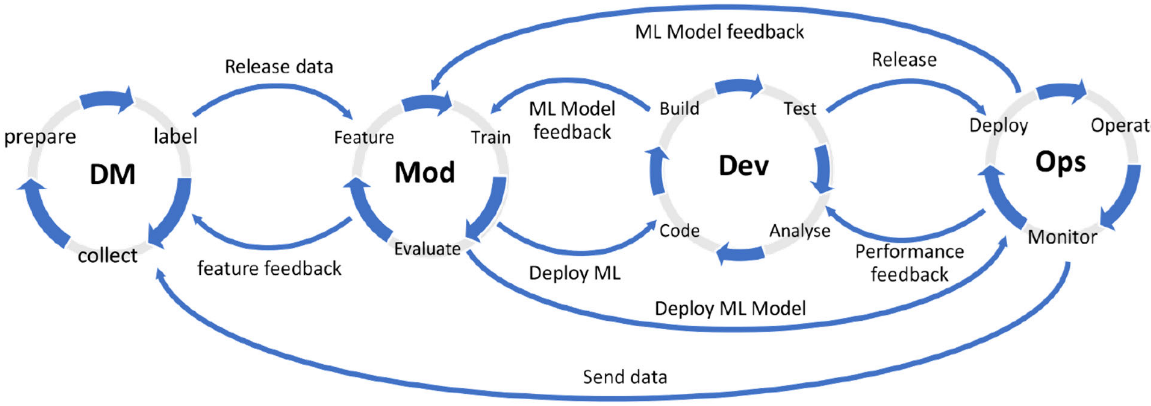
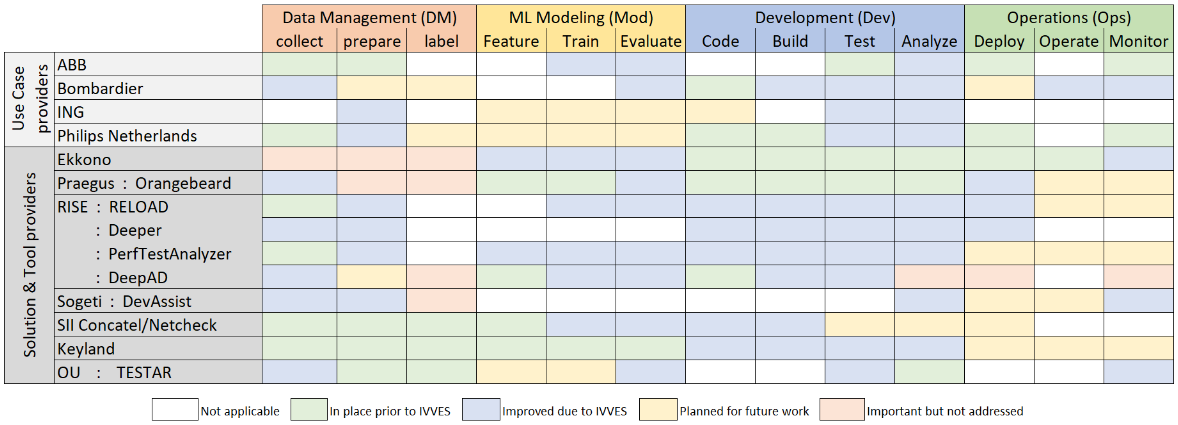


Figure 10. ML workflow and DevOps process integration (figure source [9]).

This process model was used as a base in a survey (presented in section 7.1.2) to identify the use case, tool and solutions providers activities and status before, within and after IVVES.

7.1.2 Mapping analysis results

The following figure illustrates the status of the WP4 use-cases and solutions at the start of the IVVES projects along with their improvements and status at the end of the project with respect to the coverage of the phases of the references model described in the previous section.



		Data Management (DM)			ML Modeling (Mod)			Development (Dev)				Operations (Ops)		
		collect	prepare	label	Feature	Train	Evaluate	Code	Build	Test	Analyze	Deploy	Operate	Monitor
Use Case providers	ABB	🟩	🟩	⬜	⬜	🟦	🟦	⬜	⬜	🟩	🟦	🟩	⬜	🟩
	Bombardier	🟦	🟨	🟨	⬜	⬜	🟦	🟩	🟦	🟦	🟦	🟨	🟦	🟦
	ING	⬜	🟦	⬜	🟨	🟨	🟨	🟨	⬜	🟦	🟦	⬜	⬜	⬜
	Philips Netherlands	🟩	🟦	🟨	🟨	🟨	🟨	🟩	🟩	🟦	🟦	🟩	⬜	🟩
Solution & Tool providers	Ekkono	🟧	🟧	🟧	🟦	🟦	🟦	🟩	🟩	🟩	🟩	🟩	🟩	🟦
	Praegus : Orangebeard	🟦	🟧	🟧	🟩	🟩	🟦	🟩	🟩	🟩	🟩	🟦	🟨	🟨
	RISE : RELOAD	🟩	🟦	⬜	⬜	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟨	🟨
	: Deeper	🟦	🟦	⬜	⬜	⬜	⬜	🟦	🟦	🟦	🟦	🟦	⬜	⬜
	: PerfTestAnalyzer	🟩	🟦	⬜	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟨	🟨	🟨
	: DeepAD	🟦	🟨	🟧	🟩	🟦	🟦	🟩	🟦	🟦	🟧	🟧	⬜	🟧
	Sogeti : DevAssist	🟦	🟦	🟧	⬜	⬜	⬜	⬜	⬜	⬜	🟦	🟨	🟨	🟦
	SII Concatel/Netcheck	🟩	🟩	🟩	🟩	🟦	🟦	🟦	🟦	🟨	🟨	🟨	⬜	⬜
	Keyland	🟩	🟩	🟩	🟩	🟩	🟩	🟦	🟦	🟦	🟦	🟨	🟨	🟨
	OU : TESTAR	🟦	🟩	🟩	🟨	🟨	🟦	⬜	⬜	🟦	🟩	⬜	⬜	🟦

⬜ Not applicable 🟩 In place prior to IVVES 🟦 Improved due to IVVES 🟨 Planned for future work 🟧 Important but not addressed

Figure 11. Mapping of the WP4 use-cases and solutions to the reference model and covered phases

7.2 Reflections and open challenges

To evaluate and capture the experiences of partners in adopting a data-driven engineering solution to solve the challenges of their use-cases using IVVES technologies and tools, we also performed a short survey with a set of industrial partners and problem owners in the project whose use-cases were relevant for WP4. The following subsections highlight some main feedbacks and reflections collected from the survey.

7.2.1 Question 1: Have you made better use of data in the context of your use-case through IVVES?

- **ABB**: “Yes, we have improved in what data to collect and increased the number of test cases that we collect data from”
- **Bombardier**: “Yes. The use case implementation work has given us new insights on the need for data and the right type of data”
- **ING**: “Yes, analyzing Code as ML data is a good idea for improving code quality. The data is already available, and could be leveraged more.”
- **Philips (NLD)**: “Yes, we gained insights in how to better use data e.g. coming from the installed base and also gained insights on data we're missing that can even enhance learnings and use to optimize test coverage and efficiency”
- **SII Concatel / Netcheck**: “Yes, through the functionalities implemented through the use case the use of data has been greatly optimized.”
- **Keyland**: “Yes, through the functionalities implemented through the use case the use of data has been greatly optimized.”
- **Ekkono**: “Yes, by improving on approaches to collect and transform data (e.g., through FFTs). “

7.2.2 Question 2: Has the data-driven approach in IVVES helped you make better decisions? What decisions?

- **ABB:** “Yes, since it has given us improved knowledge of the characteristics of our existing system and our systems in development. One important decision is that we need to redesign our system and add more/better measure points to make it easier to validate parts that requires high determinism.”
- **Bombardier:** “Yes, in our case, the data-driven approach has help us better understand our system's behavior and characteristics to improve identification of anomalies and optimize our testing”
- **ING:** “Steer towards lower maintenance, so tools used, should reduce maintenance costs instead of increase it. This is important in decision making. Testar does not require much maintenance and can just run on your application without prior knowledge.”
- **Philips (NLD):** “Yes, by gaining more insight in how our customers are using our products we have been able to change/increase our test coverage and select protocols and workflows closer to real customer use”
- **SII Concatel / Netcheck:** “Yes, in our case, the data-driven approach has helped us to make an optimal ESG news classification to improve investments within each asset and always preserve GDPR.”
- **Keyland:** “Yes, in our case, the data-driven approach has helped us to conduct a comprehensive environmental study based on industrial environments and has helped us to determine failures in industrial machines, both external and internal.”
- **Ekkono: “**Ekkono has been data-driven from the beginning, which has helped at making more informed decisions on pushing the right features for our software library (e.g., anomaly detector) and making more data-driven decisions with our customers”.

7.2.3 Question 3: Has a data-driven approach in IVVES helped to improve the quality of your products, services, or processes?

- **ABB:** “It has helped us realize some important aspects that can help us improve quality when we redesign parts of the system architecture. With future adaptations of the IVVES work we also hope that the quality of our products will increase even more.”
- **Bombardier:** “Yes, in our case, our tools for anomaly detection in operations, test case selection optimization and CI/CD pipeline data analysis has achieved added value thanks to the techniques researched within IVVES.”
- **ING:** “Quality of products: better code coverage; better code quality by detecting opportunities for code refactoring (code smells).”
- **Philips (NLD):** “Yes, we discovered new defects by using scan protocols created based on the IVVES technologies”
- **SII Concatel / Netcheck:** “Yes, in our case, our Fintech tool has achieved a very high quality thanks to the research conducted at IVVES.”
- **Keyland:** “Yes, in our case, our tool for monitoring and studying industrial environments has achieved added value thanks to the techniques researched within IVVES.”
- **Ekkono:** “Since we are a ML company, for us a data-driven approach is key in moving forward. IVVES has helped in that aspect since the many different approaches at improving new or existing methods (e.g., conformal prediction framework, auto ML)”.

7.2.4 Question 4: What aspects/parts of your products, services, or processes have been improved during IVVES?

- **ABB**: "Mainly how to measure and control system resource utilization in evolving systems."
- **Bombardier:** "In our case, insights on how to change our development process and DevOps has significantly improved"
- **ING**: "Improvement is that we learned that code coverage is a useful metric for comparing different testing methodologies."
- **Philips (NLD)**: "We've been able to improve test coverage and efficiency by focusing on real customer use and as such we also have been able to improve quality and reliability of our products"
- **SII Concatel / Netcheck:** "In our use case, data analysis methods through machine learning have been key to improve our Fintech products."
- **Keyland:** "In our use case, data analysis methods through machine learning and artificial intelligence have been of particular importance when it comes to analyzing failures in industrial environments thanks to AI"
- **Ekkono: "**Algorithms and frameworks such as the Change Detector, Conformal Prediction framework, Anomaly detector and many others have been improved during IVVES"

7.2.5 Question 5: What has been the pros and cons of adopting a (more) data-driven approach?

- **ABB:** "Pros: It has given us insights regarding the characteristics of the systems under test and the trend of the data while the system evolves. The data has also been valuable when designing new systems based on older systems."
- **Bombardier:** "Pros: Mindset change to focus on and identify data that is relevant for a specifc problem. Insigths on Data science. Cons: Actually identifying and collect the relevant data can be very difficult in existing systems. Needs to be designed in early"
- **ING:** "Pros: Leverage existing data (source code) to improve code quality. Cons: No explanation on why code is marked as code smell (black box, more research needed)."
- **Philips (NLD):** "Pros: better insight in the use of our products and able to focus and select relevant test protocols. Cons: we need to be able to trust the quality of the data we use and validating the data is not always easy/possible. Also selecting relevant data from the huge amount of available data point can be very time consuming. Next to that the data used must be under design control so communication of any planned changes is guaranteed"
- **SII Concatel / Netcheck & Keyland:** "The pros have been that by focusing our efforts on data, applications gain a lot of value by having that pre-processing, processing and analysis done to get quality data. The cons have been that getting quality data takes extra effort and can sometimes be tremendously difficult."
- **Ekkono:** "Pros: making informed decisions based on data, being able to detect anomalies by giving the customers the tool to create personalized models for their industrial application. Cons: customers need to have the data collection process in place to be able to obtain relevant data"

7.2.6 Question 6: Have you experienced any challenges in adopting a data driven engineering approach?

- **ABB:** "It is hard to create a scalable solution for collecting, storing and analyzing data. It is not just an engineering problem but also an infrastructure problem. It is hard to get priority for implementing and improving data-driven approaches. You need to be able to show the business value in it and prioritize that against more direct business values."
- **Bombardier:** "There are several challenges. Changing development processes takes time in big organizations. Migration to new tools and infrastructure as well. So research in IVVES is critical to understand this"

- **ING:** “Before implementing it for the whole of ING, first proper metrics and frameworks need to be developed and deployed cost effectively. After that a bigger challenge is to get is applied and supported in the larger organization and it has to fit in long term strategy, instead of ad hoc in a single place.”
- **Philips (NLD):** “Ensuring enough time and focus for the transformation towards a data driven engineering approach given the operational pressure to deliver”
- **SII Concatel / Netcheck:** “In our Fintech use case, the main challenge has been the analysis of quality news and designing a system capable of analyzing, verifying and validating the whole process with the help of machine learning and artificial intelligence has been a challenge at the software engineering level.”
- **Keyland:** “In our use case on industrial environments, the main challenge has been the analysis of quality news and designing a system capable of analyzing, verifying and validating the whole process with the help of machine learning and artificial intelligence has been a software engineering challenge in addition to the added mechanical engineering challenge of having to synchronize the industrial machine with the whole software architecture developed.”
- **Ekkono: “**We haven’t experienced too many challenges in this aspect since the company was created with a data-driven approach in mind by using ML. However, it requires more expertise on the related background to be able to design relevant solutions, algorithms, and extract significant conclusions”.

8 Conclusions

This deliverable, as the final report from WP4, provided a highlight and summary of the work done in the IVVES project on development and adoption of data-driven engineering solutions, in particular, to solve the challenges of the industrial use-cases of the project. After a brief overview of the relevant use-cases in Section 3, Section 4 discussed the technical results of the WP4 and solved challenges of use-cases with respect to data collection and instrumentation as outcomes of Task 4.1. Section 5 reported on the outcomes of task 4.2 regarding pattern recognition results and achievements of the WP4. The outcomes of task 4.3 on data analytics in engineering and operation were then described in Section 6.

Finally in Section 7, we reported on the experiences and reflections of the partners in adopting a data-driven engineering approach, its benefits, and challenges, while highlighting the achieved improvements thanks to the IVVES solutions and technologies. These results and reflections can also help other companies and organizations in their journey towards adopting data-driven engineering solutions with the ultimate goal to enable systematic collection and processing of data and automation of data-driven decisions in the development of software-intensive systems.

9 References

[1] H. Holmström Olsson and J. Bosch, “Data Driven Development: Challenges in Online, Embedded and On-Premise Software,” in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2019, vol. 11915 LNCS, pp. 515–527.

[2]“Building data science teams - O’Reilly Radar.” [Online]. Available: http://radar.oreilly.com/2011/09/building-data-science-teams.html. [Accessed: 26-Jun-2020].

[3] https://abstracta.us/blog/test-automation/best-testing-practices-agile-teams-automation-pyramid/

[4] Kolstromc P Alegroth E, Feldt R. Maintenance of automated test suites in industry: An empirical study on Visual GUI Testing. Master's thesis, 2016.

[5] H. Bunke, M. Last, and A. Kandel. Artificial Intelligence Methods in Software Testing. EBSCO ebook academic collection. World Scientific, 2004.

[6] M. H. Moghadam, M. Saadatmand, M. Borg, M. Bohlin, and B. Lisper, “Performance testing driven by reinforcement learning,” in 2020 IEEE 13th International Conference on Software Testing, Validation and Verification (ICST), 2020, pp. 402–405

[7] M. H. Moghadam, M. Saadatmand, M. Borg, M. Bohlin, and B. Lisper, “An autonomous performance testing framework using self-adaptive fuzzy reinforcement learning,” Software Quality Journal, pp. 1–33, 2021

[8] M. H. Moghadam, G. Hamidi, M. Borg, M. Saadatmand, M. Bohlin, B. Lisper, and P. Potena ,“Performance testing using a smart reinforcement learning-driven test agent,” in 2021 IEEE Congress on Evolutionary Computation (CEC), 2021, pp. 2385–2394

[9] L. E. Lwakatare, I. Crnkovic and J. Bosch, DevOps for AI – Challenges in Development of AI-enabled Applications, International Conference on Software, Telecommunications and Computer Networks (SoftCOM), 2020.