D7.2	
Version	1.0
Author	CNR
Dissemination	PU
Date	30-12-2019
Status	Final

D7.2 ElasTest validation methodology and its results v2

Project acronym	ELASTEST
Project title	ElasTest: an elastic platform for testing complex distributed large software systems
Project duration	01-01-2017 to 31-12-2019
Project type	H2020-ICT-2016-1. Software Technologies
Project reference	731535
Project website	http://elastest.eu/
Work package	WP7
WP leader	Antonia Bertolino (CNR)
Deliverable nature	Report
Lead editor(s)	Antonia Bertolino, Eda Marchetti (CNR)
Planned delivery date	31-12-2019
Actual delivery date	30-12-2019
Keywords	Open-source software, cloud computing, software engineering, operating systems, computer languages, software design & development

Funded by the European Union

License

Contributors

Name	Affiliation
Zahoor Ahmed	TUB
Antonia Bertolino	CNR
Antonello Calabrò	CNR
Felicita Di Giandomenico	CNR
Francisco Gortázar	URJC
Francesca Lonetti	CNR
Varun Gowtham	TUB
Eda Marchetti	CNR
Marisol Prieto	ATOS - WLI
Guiomar Tuñón	Naeva Tec
Hemant Zope	Fraunhofer FOKUS

Version history

Version	Date	Author(s)	Description of changes
0.1	05/09/2019	CNR	Toc and preliminary content structure
0.2	30/10/2019	CNR	Updated description of methodology and structure for study reporting
0.3	15/11/2019	CNR	Preliminary plots and reports from QEs and CCS
0.4	22/11/2019	CNR	Data and reports from ES
0.5	22/11/2019	CNR, URJC	QoE experiment description
0.6	November 2019	CNR, ATOS, FOKUS, Naeva Tec	Description of vertical demonstrators
0.7	8/12/2019	TUB	Description of TUB demonstrators
0.8	14/12/2019	All	Several revisions
0.9	16/12/2019	URJC	Internal review
1.0	30/12/2019	CNR	Final revision

Table of contents

1 Executive summary 11

2 ElasTest validation objectives 12
2.1 Metrics for Objective 1 12
2.2 Metrics for Objective 2 12

3 Validation methodology 14
3.1 Quasi Experiment (QE) 14
3.2 Comparative Case Study (CCS) 15
3.3 Empirical Survey (ES) 16

4 Adopted metrics and data sheets 17
4.1 Procedures 17
4.2 Critical revision of ElasTest validation metrics 19
4.2.1 Time to Market 20
4.2.2 Reusability 21
4.2.3 Productivity 22
4.2.4 Maintenance 23
4.2.5 Field Failure 23
4.2.6 Scalability 23
4.2.7 Robustness 24
4.2.8 Security 24
4.2.9 Quality-of-Experience 25

5 Schedule of validation studies 27

6 ElasTest vertical demonstrators 29
6.1 ATOS demonstrators 29
6.2 FOKUS demonstrators 31
6.3 Naeva Tec demonstrators 33
6.3.1 Naeva Tec QE demonstrator 33
6.3.2 Naeva Tec CCS demonstrator 34
6.4 TUB demonstrators 35
6.4.1 TUB QE demonstrator 35
6.4.2 TUB CCS demonstrator 36

7 Results 37
7.1 Summary of VMs and their assignments 37
7.2 Quasi experiments results 37
7.2.1 QE results from ATOS Worldline 37
7.2.2 QE results from FOKUS 42
7.2.3 QE results from Naeva Tec 43
7.2.4 QE results from TUB 45
7.3 Comparative case studies results 47
7.3.1 CCS results from ATOS Worldline 47
7.3.2 CCS results from FOKUS 50
7.3.3 CCS results from Naeva Tec 52
7.3.4 CCS results from TUB 53
7.4 Empirical surveys results 54
7.4.1 Simplicity 55

7.4.2 Satisfaction *56*
7.4.3 Efficacy *57*
7.4.4 Confidence and Lack of Risks *57*
7.4.5 Usefulness *58*
7.4.6 Efficiency *58*

8 Validation conclusions 60
8.1 Testing time 60
8.2 Reusability 61
8.3 Productivity 62
8.4 Maintenance 62
8.5 Scalability, robustness, security and QoE 63
8.6 Subjective feelings 64

9 References 66

Appendix: Tester Survey 67

List of figures

Figure 4.1: Mapping of objective metrics to demonstrators.....20

Figure 5.1: ElasTest revised validation schedule.....27

Figure 6.1: FOKUS Open5GCore toolkit.....31

Figure 7.1: Clock time spent by ATOS WL testers in the QE over Online Supermarket.....38

Figure 7.2: Clock time spent by ATOS WL testers in the QE over Messaging Platform.....39

Figure 7.3: Reusability estimation in QE for Messaging Platform.....39

Figure 7.4: Productivity data from QE for ATOS WL testers on OS (bottom) and MP (top) demonstrators.....40

Figure 7.5: Effort (in minutes) spent by WE and WO testers in QE for fault localization and correction in OS and MP demonstrators. Red bars refer to cumulative effort; blue bar to effort per fault.....41

Figure 7.6: Security data from QE by ATOS WL testers for the WE branch; data for WO branch (in italics) are estimates as no security testing was performed.....41

Figure 7.7: Clock time spent by FOKUS testers in each testing phase and totally in the QE.....42

Figure 7.8: Number of reused tests by FOKUS testers in QE.....43

Figure 7.9: Clock time spent by Naeva Tec testers in the QE.....44

Figure 7.10: Clock time spent by TUB testers in each testing phase and totally in the QE.....46

Figure 7.11: Data collected by TUB to assess scalability KPI in QE.....47

Figure 7.12: Clock time spent by ATOS WL testers in the CCS over Online Supermarket.....48

Figure 7.13: Clock time spent by ATOS WL testers in the CCS over Messaging Platform.....48

Figure 7.14: Productivity data from CCS for ATOS WL testers on OS (bottom) and MP (top) demonstrators.....49

Figure 7.15: Effort (in minutes) spent by ATOS WE and WO testers in CCS for fault localization and correction in OS and MP demonstrators. Red bars refer to cumulative effort; blue bar to effort per fault.....50

Figure 7.16: Security data from CCS by ATOS WL testers for the WE branch; data for WO branch (in italics) are estimates as no security testing was performed.....50

Figure 7.17: Clock time spent by FOKUS testers in each testing phase and totally in the CCS...51

Figure 7.18: Number of reused tests by FOKUS testers in CCS.....52

Figure 7.19: Data from TUB CCS.....54

List of tables

Table 1: Mapping between validation metrics (in columns) and the different types of empirical studies undertaken (in rows): Top table illustrates the mapping planned in the DoA; Bottom table shows the actual mapping applied……18

Table 2: Time-to-Market data sheet……21

Table 3: Reusability data sheet……22

Table 4: Productivity data sheet……23

Table 5: Corrective maintenance data sheet……23

Table 6: Scalability data sheet……24

Table 7: Robustness data sheet……24

Table 8: Security data sheet……25

Table 9: Features of ATOS E-commerce platform……30

Table 10: Features of ATOS messaging platform……30

Table 11: Testers' demographics……55

Table 12: Average scores of responses related to simplicity from WO and WE testers……56

Table 13: Average scores of responses related to satisfaction from WO and WE testers……56

Table 14: Average scores of responses related to efficacy from WO and WE testers……57

Table 15: Average scores of responses related to confidence and lack of risks from WO and WE testers……58

Table 16: Average scores of responses related to usefulness from WO and WE testers……58

Table 17: Average scores and delta for efficiency according to WO and WE testers……59

Table 18: Summary results for testing time……61

Table 19: Summary results for reusability……61

Table 20: Summary results for productivity……62

Table 21: Summary results for maintenance……62

Table 22: Summary results for VM 2.3 aspects……63

Table 23: Summary results for subjective metrics (VM 1.4 and 2.4) ……64

Glossary of acronyms

Acronym	Definition
CCS (Comparative Case Study)	It is an examination undertaken over time useful for a comparison within and across contexts. It involves the analysis and synthesis of the similarities, differences and patterns across two or more cases (in our case two testing processes) that share a common focus or goal
DoA (Description of Action)	This refers to the document that describes the activities planned within the ElasTest project and their organization in Work Packages
EDS (Device Emulator Service)	EDS is an ElasTest component that deploys emulated sensors or actuators on demand
ES (Empirical Survey)	It is a collection of information from a sample of individuals through interviews performed via computer-assisted questionnaires
GDPR (General Data Protection Regulation)	The EU Regulation 2016/679 lays down rules relating to the protection of natural persons with regard to the processing of personal data and rules relating to the free movement of personal data
IOT (Internet of Things)	IOT is the network of physical devices, vehicles, home appliances, and other items embedded with electronics, software, sensors, actuators, and connectivity which enables these things to connect and exchange data
KPI (Key Performance Indicator)	A KPI denotes a measurement established for evaluating the progress and performance of a specified activity under control; in the context of WP7 KPIs refer to success of the ElasTest project objectives.
LOC (Line of Code)	This refers to lines of code considered during testing experimentation
oneM2M (Machine to Machine Communications)	OneM2M is the global standards initiative for Machine to Machine Communications and the Internet of Things
QE (Quasi Experiment)	It is a controlled study used to estimate the causal impact of an intervention on its target population without random assignment to treatment or control

QoS (Quality of Service) and QoE (Quality of Experience)	QoS and QoE refer to non-functional attributes of systems. QoS is related to objective quality metrics such as latency or packet loss. QoE is related to the subjective quality perception of users. In ElasTest, QoS and QoE are very important for application to multimedia systems
REST (Representational State Transfer)	REST is an architectural style that defines a set of constraints and properties based on HTTP
SiL (System in the Large)	A SiL is a large distributed system exposing applications and services involving complex architectures on highly interconnected and heterogeneous environments. SiLs are typically created interconnecting, scaling and orchestrating different SiS. For example, a complex microservice-architected system deployed in a cloud environment and providing a service with elastic scalability is considered a SiL
SiS (System in the Small)	SiS are systems basing on monolithic (i.e. non-distributed) architectures. For us, a SiS can be seen as a component that provides a specific functional capability to a larger system
SUT (Software under Test)	This refers to the software that a test is validating. In this project, SUT typically refers to a SiL that is under validation
T-Job (Testing Job)	We define a T-Job as a monolithic (i.e. single process) program devoted to validating some specific attribute of a system. Current Continuous Integration tools are designed for automating the execution of T-Jobs. T-Jobs may have different flavors such as unit tests, which validate a specific function of a SiS, or integration and system tests, which may validate properties on a SiL as a whole
TSS (Test Support Service)	TSS is a test Support Service (TSS) useful for test execution
TTM (Time To Market)	TTM measures the time elapsed during a new product development process: it should include the whole time from a product idea until its commercialization
VM (Validation Metric)	The metric adopted inside ElasTest to assess whether (and to what extent) the DoA objectives are met
WE (With ElasTest)	It denotes the measures taken by using ElasTest
WO (Without ElasTest)	It denotes the measures taken without using ElasTest

1 Executive summary

This document is the final report from ElasTest Work Package WP7. All along the project lifecycle, WP7 has addressed the crucial objective of conducting the **validation of the developed platform** by assessing its usage within **different domains** and scenarios. The validation work has been driven by the so-called "Outcome objectives" that had been set already in the project DoA along with a corresponding set of KPIs.

Specifically, over the three years the WP has completed the following activities:

a) designed the **ElasTest validation methodology**, instantiating and continuously revising it based on the needs of the different demonstrators, as well as on the platform features and releases;
b) defined a specific **set of metrics** for validating the DoA KPIs, and developed a collection of **data sheets** for their assessment;
c) tailored to the validation purposes several applications over four application scenarios (also referred to as the **vertical demonstrators**): these are described in more detail in the companion deliverables D7.3 (public demonstrators) and D7.4 (private demonstrators);
d) provided **feedbacks and requirements to ElasTest** platform developers;
e) performed in several iterations a collection of rigorous **empirical studies**, from which the validation data have been collected;
f) gathered and **analyzed the validation studies results**.

Part of the above activities has been conducted during the first period and has been reported at M18 in Deliverable D7.1. In this document, we start again describing the validation methodology applied in the second reporting period, not only to make the document self-contained, but also to illustrate some revisions/improvements on the methodology previously described in D7.1 that have been judged necessary or convenient. Then the bulk of the document consists of the results collected from the extensive empirical studies carried out in the second reporting period.

The document is structured as follows: in Chapters 2 and 3 we recall from D7.1 the validation objectives and the validation methodology. In Chapter 4 we then present extensively the procedures followed, the data sheets devised for metric collection, also explaining the revisions we did over the original plans in the DoA. In Chapter 5 we present the overall schedule of WP7 work, which has also been critically revised from the version presented in D7.1. In Chapter 6 we give a brief description of the vertical demonstrators, and in Chapter 7 we present the results of the studies on each of them, covering QE, CCS and ES studies. Finally, in Chapter 8 we draw overall conclusions, by collecting and comparing the results across all the studies. We include in Appendix the full list of questions into the survey questionnaire for the ES study.

2 ElasTest validation objectives

The objectives of ElasTest validation activity have been early fixed in the project DoA [1], and have been previously described in Deliverable D7.1 [2]. We briefly summarize them below for the sake of self-completeness.

ElasTest platform aims at improving the testing of large complex distributed systems that is notoriously difficult and effort-prone. This aim has been developed along the two following outcome objectives:

Objective 1: to improve the efficiency, productivity and code reusability of the testing process of SiL.

Objective 2: to improve the effectiveness of the testing process and, with it, the quality of SiL software.

Then, to be able to assess whether (and to what extent) such objectives are met, the ElasTest DoA associated them with a set of validation metrics, and their relative KPIs, as reported below.

2.1 Metrics for Objective 1

- Validation metric 1.1: To reduce the overall time to market of SiL in an average factor of 20%
- Validation metric 1.2: To increase the reusability of code, tools and architectures devoted to non-functional software testing on SiL in a factor of, at least, 500%
- Validation metric 1.3: To increase the overall tester productivity (measured as lines of code tested per time unit) for integration and system tests in a factor of, at least, 100%
- Validation metric 1.4: To increase the tester subjective feelings of simplicity, satisfaction, efficacy, confidence and usefulness when involved in testing tasks for SiL in a factor of at least 1 per each in a scale of 5

2.2 Metrics for Objective 2

- Validation metric 2.1: To decrease the corrective maintenance effort of SiL in a factor of, at least, 50%
- Validation metric 2.2: To decrease field failure reports of SiL in a factor of, at least, 30%
- Validation metric 2.3: To increase the scalability (measured as the total number of concurrent supported sessions), robustness (measured as down-time after computing failures), security (measured in incidents per time unit) and QoE (Quality of Experience) (see QoE metrics in Task 5.1) of SiL in an average factor of 20%
- Validation metric 2.4: To increase the subjective feelings of end-users in terms of efficiency, overall satisfaction and lack of risks when using SiL-based applications in a factor of at least 1 per each in a scale of 5

WP7 assessed the above listed metrics over the vertical demonstrators made available by the partners contributing to WP7 with four application scenarios, as planned in the DoA. This required defining concrete approaches to measure the above VMs, in most cases described at quite abstract levels. Precisely, not all metrics could be relevant or feasible for each single demonstrator, but overall we aimed at covering all of them across the four domains. In doing this, in some cases we had to revise some of the above definitions, because along the process we better understood some limitations of the above formulations. Our revision of the VM is discussed in the following in Section 4.2.

In the next chapter we provide a detailed and updated report of the validation methodology.

3 Validation methodology

WP7 targets quite ambitious goals, as the established set of validation metrics covers quite disparate properties of a software process and product, and their assessment required to put in place different approaches and studies. In particular, considering the metrics that have to be assessed, the validation methodology adopted inside the ElasTest project included three different types of empirical studies that complemented each other in evaluating ElasTest results:

- **Quasi Experiment (QE)**: i.e., a controlled study used to estimate the causal impact of an intervention on its target population without random assignment to treatment or control.
- **Comparative Case Study (CCS)**: i.e., an examination undertaken over time useful for a comparison within and across contexts. It involves the analysis and synthesis of the similarities, differences and patterns across two or more cases (in our case testing activities) that share a common focus or goal.
- **Empirical Survey (ES)**: i.e., a collection of information from a sample of individuals through interviews performed via computer-assisted questionnaires.

In the following we describe the procedures followed for each type of study.

3.1 Quasi Experiment (QE)

For all the vertical demonstrators, the general procedure for the QE assessment foresees two different teams (testers) performing the testing activity on a same application, so that:

- the Without ElasTest group (WO) conducts the testing following the best practices commonly adopted in the company and using the standard tools and facilities;
- the With ElasTest group (WE) conducts the testing ALSO using the support of ElasTest.

Note that in most studies the WO and WE teams consisted of one person only each, due to limited resource availability.

More specifically, the steps for the QE included:

1. Each testing team (WO and WE) starts analyzing the SUT documentation; this included a set of test directives (i.e., a preliminary test plan)
2. The Development Team develops (finalizes) the first version of the SUT
3. The test directives are translated into:
   - A WO test plan by the WO testers
   - A WE test plan by the WE testers
4. In parallel testers in each team start testing and, if applicable, raising bugs, namely:
   - Testers in WO start testing and raising bugs with label ‘WO’
   - Testers in WE start testing and raising bugs with label ‘WE’

5. Possibly found WO and WE bugs are sent back to developers for fixing them
   - NOTE: bug fixes are not deployed until both teams (WO and WE) have finished the first test iteration.
6. Once both teams (WO and WE) finish their first test iteration, a new version of SUT with all the bugs fixed is deployed.
7. All along the above process, both teams (WO and WE) collect the requested set of metrics and report them in the ad-hoc prepared forms (we describe these in detail in Chapter 4).

Specifically, the forms that we developed for QE data collection are structured in several different sheets, each one associated with a different validation metric among those defined in Chapter 2. The data sheets are presented in the next chapter.

3.2 Comparative Case Study (CCS)

While in the QE we aim at isolating the potential cause of different responses, in the comparative case study, we aim at observing the effect of introducing the treatment (ElasTest) in the actual (testing) environment without using any control. Yin ([3], p. 16) defines a case study as "*an empirical inquiry that investigates a contemporary phenomenon (the case) in-depth and within its real-world context, especially when the boundaries between phenomenon and context may not be clearly evident.*"

Thus, similarities and differences to support or refute hypotheses and to evaluate the impact of the causality (i.e., the extent to which the intervention caused the results) have to be collected. Usually the selection of specific subjects is linked to the metrics to be evaluated, and generally (as in our case) both qualitative and quantitative data are considered.

Considering WP7 data collection, the procedure we adopted for conducting the CCS includes the following steps:

a) Baseline (or historical) data collection stage - labeled WO (WithOut ElasTest):

- Each of the demonstrator partners selects one or more projects comparable to the selected ElasTest demonstrators;
- For each selected project, the partner fills the pre-specified forms with the data relative to the relevant metrics.

b) ElasTest data collection stage – labeled WE (With ElasTest):

- Each of the demonstrator partners selects one or more projects that are developed/tested using the ElasTest platform;
- For each selected project, the partner fills the pre-specified forms with the data relative to the relevant metrics.

c) The data relative to stages WO and WE are analyzed by CNR and the comparative value across all metrics are derived.

Differently from the first period and what we report in D7.1, during the second period we decided to use exactly the same templates for CCS and QE metrics data. Moreover,

with respect to the very detailed data collection forms used previously, we opted in this second period for a simpler format: we learnt in fact from the pilot studies carried out in the first period, that testers did not collect the specific data we requested, and instead they used to only collect coarse more high-level data. We hence decided to distribute simplified data sheets, which are those described in Chapter 4 (whereas previous forms are described in D7.1).

3.3 Empirical Survey (ES)

The empirical survey type of study is adopted to assess subjective metrics. These include VM 1.4 related to tester subjective feelings of simplicity, satisfaction, efficacy, confidence and usefulness; and VM 2.4 related to end users' subjective feelings in terms of efficiency, overall satisfaction and lack of risks when using SiL-based applications. We collected such data in several iterations through web forms. In practice upon terminating any branch of the study (be it a QE or a CCS) we asked each participant to fill the questionnaire.

The empirical surveys conducted during the second period involved both:

1. Qualitative data about the experience of testers and end users in using ElasTest and its comparison with previous used procedures and tools (i.e., without ElasTest). Testers and end users are asked to freely report both positive and negative subjective feelings from their experience with the ElasTest platform.
2. Personal data relative to the testers and end users' technical expertise and professional profile, for example their role in the project, years of experience, technologies mastered, and similar, and gender information.

Due to its entry into force since May 2018, in the second period for the latter data we took proper action to guarantee their treatment in compliance with GDPR. CNR is responsible for treatment of any personal data collected within WP7.

In particular, the participation to the survey is totally voluntary and informed. All testers and end users involved in the experimentation have been informed in respect of the current laws[1]. All personal data have been processed manually and with the support of Excel for elaboration and analysis, but always guaranteeing maximum security and privacy levels according to current state of art technology that are applied in a specific Regulation at CNR-ISTI. The personal data have not been communicated to any subject outside CNR-ISTI. Only persons in charge for the study from CNR-ISTI had access to the personal data, and used them in agreement with current laws, and in line with GDPR and GDPR-compliant CNR-ISTI procedural Regulation.

For more information, a copy of the prepared questionnaire (which is the same already used for D7.1) is attached in Appendix to this deliverable.

[1] At the time of project start the GDPR (General Data Protection Regulation) had not yet come into force, so concerning the data collected in the previous deliverable D7.1 (which were collected and analysed before May 25th, 2018), the Italian law DLgs196/2003 was followed. From M19 we adopted the GDPR norms.

4 Adopted metrics and data sheets

As above anticipated, project objectives are assessed over a set of KPIs, which can be distinguished between objective and subjective ones. In this chapter we present in detail the specific metrics collected for assessing the ElasTest KPIs. We confirm some of the metrics introduced in D7.1, but also introduce some revised definitions, and some novel metrics that we could not yet assess in the first period.

Objective metrics refers to properties related to the efficiency and effectiveness of the software testing and maintenance process, which we aim to improve thanks to adoption of ElasTest. This applies to validation metrics 1.1, 1.2 and 1.3 and 2.1, 2.2 and 2.3. Their assessment requires comparing, on a set of similar products, process quantitative measures taken after putting ElasTest in place for some consistent period against historical measures collected before adoption of ElasTest (CCS). Alternatively, we can conduct an experiment comparing in a controlled environment the measures achieved with and without ElasTest (the treatment under study) on a same system (QE).

4.1 Procedures

We asked testers to collect exactly the same data, be it for CCS or QE. For each VM we defined one devoted data sheet on which to report the measures, except for VM 2.3 that actually includes several sub-metrics, and hence was developed into as many sheets.

Concerning the latter type of metrics, i.e., subjective ones, these refer more specifically to KPIs indicated within validation metrics 1.4 and 2.4. For these we had developed already during ElasTest first reporting period a questionnaire collecting focused groups of questions. We use such questionnaire during an Empirical Survey (ES) study to gather human feedback by the testers themselves, both using and not using ElasTest.

In Table 1 below, the mapping between the validation metrics presented in Chapter 2 and the different types of empirical studies is provided. Precisely, Top Table presents the tentative mapping as defined upstream in the DoA, whereas Bottom Table illustrates the more extensive mapping that we aimed at downstream, after deeper comprehension of the vertical demonstrators, with the aim of collecting as much data as possible.

Technique/Validation Metric	1.1	1.2	1.3	1.4	2.1	2.2	2.3	2.4
Empirical Survey				✔				✔
Comparative Case Study	✔		✔		✔	✔		
Quasi Experiment		✔					✔	

Technique/Validation Metric	1.1	1.2	1.3	1.4	2.1	2.2	2.3	2.4
Empirical Survey				✔				✔
Comparative Case Study	✔	✔	✔		✔	✔	✔	
Quasi Experiment	✔	✔	✔		✔	✔	✔	

Table 1: Mapping between validation metrics (in columns) and the different types of empirical studies undertaken (in rows): Top table illustrates the mapping planned in the DoA; Bottom table shows the actual mapping applied

All these studies include two stages: the **baseline** data collection and the ElasTest data collection, whereby ElasTest is the "**treatment**" being assessed.

The baseline data collection is needed to obtain the reference measures relative to the different demonstrators in each study. The validation metrics defined in the previous chapter need to be compared against these baseline measures, to assess whether we improve in the expected directions.

During the first period (M1-M18), we worked hard with the four partners conducting the validation, to collect for this purpose the “historical data”: with this we refer to measures that are recorded by the partners along their usual development and testing process, before any application of ElasTest components or improvements. We did so as early as possible, to ensure a proper assessment of the starting situation before any influence by ElasTest could bring potential changes to the process.

As we will see in Chapter 6, the four scenarios over which the vertical demonstrators are run are quite different in terms of scope, processes followed, personnel involved. Hence, it was necessary to specialize the procedure for data collection according to the specific processes in use within the four partners. To this purpose, we recall that in the first period we conducted a two-stage survey, including: *i)* a preliminary remote interview to the personnel in charge of managing the demonstrators within the four partners, and then ii) remote interviews with the representative of the four vertical demonstrators.

The outcomes from these two-stage surveys have been used by CNR to prepare a draft version of both: the data collection structure for the CCS, and the already mentioned questionnaire surveys for testers. These draft versions have been both used for a pilot measurement at M9 (see D7.1), with the aims of: ensuring the validity of the artifacts; ensuring that the measurements are feasible (e.g., data collection does not impact partner’s process, or the interview stays within the expected time limits); and better

understanding and refining the data collection framework and the questionnaires. Finally, in the first period, we also launched a pilot QE that involved all four partners and collected a first set of data. Not all metrics have been assessed, due to the preliminary status of ElasTest platform at the time of that study.

Along all data collection steps, the partners have been carefully instructed to collect correct measures without introducing any perturbation or bias so that later we can measure actual improvements obtained with ElasTest.

The data collected from CCS, QE and ES in the first period are extensively reported in Deliverable D7.1 [2]. Along the project life, in the second period (M19-M36), we repeated the collection of baseline measures at different times (see project schedule in next section). We did this for two pragmatic reasons: *i)* to have data redundancy that can better sustain meaningful measures, and *ii)* to take into account potential variations of the testing environment and processes.

In the remaining of this section details about the metrics in each of the prepared sheets are provided. It can be noted that the data sheets we employed during the second period have been simplified with respect to those used for D7.1: we observed from their usage that they were burdensome and unnecessarily detailed over some fields (which in practice the partners left empty).

4.2 Critical revision of ElasTest validation metrics

The KPIs defined for ElasTest validation in the DoA (which we have described above in Chapter 2) had been early conceived at the time of ElasTest project proposal definition. At completion of the first period (M18), after the preliminary studies already carried out on ElasTest application on the four scenarios, it is somewhat natural that we have got a deeper understanding of implications and assumptions behind such metrics, and could revise them in more useful and realistic way.

Having conducted the validation in several iterations, we could in fact discern what were realistic goals, and on the other hand which assumptions, made at stage of proposal writing, were not realistic or not doable for some of the vertical demonstrators.

There are a few metrics that we cannot evaluate within the lifetime of the project: we refer to those metrics related to observing in operation for a period of time the behaviour of products tested with ElasTest and comparing with similar metrics previously observed before using ElasTest. We cannot do this for obvious time-clashing: the platform in its mature configuration is only released at end of the project, and there is no time left to have an extended observation of the products tested with it after they are released; but also due to constraints in the lifecycle of the adopted demonstrators: they are still undergoing development at time of writing. Specifically, the metrics we could not observe include: VM 2.2, according to which we aimed at observing field failure reports; and VM 2.4 which would require to collect subjective feelings of end-users of the product in the field. Concerning the latter, we however

included questions relative to the subjective feelings covered in VM 2.4 in the questionnaire we distributed to testers, so that at least we could receive their intermediate feedback.

In addition we made a careful concertation during the first period to properly assign the collection of metrics among the four demonstration scenarios, so to ensure that, even acknowledging that not all metrics can be taken by each of them, every metric (except 2.2) is covered by at least one of the demonstrators. This distribution concerned the objective metrics: precisely, with reference to metrics 1.1, 1.2, 1.3, 2.1, and 2.3 the result of such concertation is summarized in the matrix shown below in Figure 4.1. The matrix associates the metrics (in rows) to demonstrators (in columns): each cell metric_a/demonstrator_x is labeled with QE and/or CCS, if demonstrator_x could collect metric_a within the QE and/or CCS study. When empty the demonstrator could not provide such measures. This happened because either they did not have in place appropriate measuring procedures for gathering that metric, or the specific VM is not relevant in the context. For example, QoE (VM 2.3.4) was only meaningful for the OpenVidu component used by the FullTeaching application. We see that in the case of ATOS, the partner conducted the studies over two systems: the Online Supermarket (OS), and the Messaging Platform (MP), so in total we performed the validation over 5 demonstrators.

VM id	VM label	ATOS OS	ATOS MP	FOKUS	Naeva Tec	TUB
1.1	Time-to-Market	QE, CCS	QE, CCS	QE2,CCS	QE, CCS	QE, CCS
1.2	Reusability		QE, CCS	QE2, CCS		
1.3	Tester productivity	QE, CCS	QE, CCS	QE2, CCS	CCS	CCS
2.1	Corrective maintenance	QE, CCS	QE, CCS		QE, CCS	CCS
2.3.1	Scalability					QE
2.3.2	Robustness					QE
2.3.3	Security	QE, CCS	QE, CCS		QE	
2.3.4	Quality-of-Experience				QE	

Figure 4.1: Mapping of objective metrics to demonstrators

Concerning instead the ES studies for subjective metrics, every tester that planned to use ElasTest within the lifetime of the project has been asked to answer the survey relatively to his/her subjective feelings about testing without and/or with ElasTest.

4.2.1 Time to Market

Time to Market (TTM), which refers to VM 1.1, aims at assessing the time elapsed during a new product development process: ideally it should include the whole time from a product inception until its commercialization. In practice our observation window within WP7 was limited to the testing process, because our studies covered just the steps listed in Sections 3.1 and 3.2.

Our expectation was that ElasTest can notably decrease the effort and time needed for testing and maintenance, and hence, as a consequence of testing time reduction, TTM can be decreased as well: the KPI estimate in the DoA said on average by 20%.

However, we could measure this reduction not over the whole development cycle, but the testing process.

Time to market				
Specify time unit				
Metrics	**Unitary**	**Integration**	**System**	**End2End**
N Developers				
N Testers				
Clock time total (1)				
Clock time in human thinking and testing preparation				
Clock time in test coding				
Clock time in test execution				
Clock time in result analysis, fault localization, and debugging				

(1) Clock time measures the time employed by the human subject for the testing - start clock when starting, pause clock when interrupting

Table 2: Time-to-Market data sheet

In particular, considering the values for completion of the development process (time from start to end) collected both WO and WE, to achieve the aimed improvement of at least 20% we need to measure:

$$\Delta TTM = (\Sigma \text{TTM}_{WE} - \Sigma \text{TTM}_{WO}) / \Sigma \text{TTM}_{WO} \quad \text{(Eq. 1)}$$

where the suffixes WE and WO denote the measure taken by using ElasTest, or without, respectively. As shown in Table 2 above, in addition to total time, we asked testers to collect also the time spent within the four stages of test planning, test coding, test execution and results analysis. These detailed measures can be useful to interpret the observed results, and to derive other meaningful comparisons.

4.2.2 Reusability

According to the DoA, the reusability KPI has been defined in relation to the percentage of code, tools and architectures reused in testing activity. Correspondingly we defined a data sheet as shown in Table 3, in which we ask testers to report how many tests have been reused, and where available also provide more specific characterization of reuse.

Reusability per time unit				
Specify time unit				
Metrics	**Unitary**	**Integration**	**System**	**End2end**
Tests reused (in total or in part) (1)				
of which, # of Non-functional tests				
LOCs total of such reused tests				
LOCs reused in each test case (estimation if not known)				
N. of basic facilities reused in each test case				
N. of modules reused in each test case				
N. of portings performed				
Total N of new test cases				

(1) Reuse may refer to any of:
- in rounds following the first, reuse of tests coded in previous phase(s)
- code reused between tests
- from other projects

Table 3: Reusability data sheet

The expectation on the long term is that ElasTest can greatly support reuse of tests, especially in the context of non-functional testing. However, within the lifetime of ElasTest project, and specifically within the test activity observed in WP7 it has not been possible to assess properly such metric, because the tests employed within the vertical demonstrators used are generally developed for the first time. So, what we could measure is only the aspect of reuse related to reusing test artifacts across releases. With the possibility to further continue the empirical observation, we might likely measure more and more reuse also among different projects. However at time of writing we do not have such data and hence unfortunately we can only hypothesize this trend.

4.2.3 Productivity

In the ElasTest DoA tester productivity is related to VM 1.3 and has been defined as "lines of code tested per time unit". Accordingly we derived the data sheet reported in Table 4 below, which aims at summarizing “how much” of the SUT has been tested within a same time unit, and with how many test cases.

An important point in measuring tester productivity is to establish the points at which measurements are performed, i.e., the **time unit**, which can vary across verticals. Productivity measures can be daily, monthly, at a project closure, at each release and so on.

Productivity per time unit				
Specify time unit				
Metrics	**Unitary**	**Integration**	**System**	**End2end**
N Lines of code tested				
N modules/functions tested				
N of test cases executed				
Test code size LOCs (average)				

Table 4: Productivity data sheet

In particular to achieve the aimed improvement of at least 100% we need to observe:

$Productivity\Delta = (\Sigma$ #LOC tested per TU_{WE} - Σ#LOC tested per $TU_{WO})/ \Sigma$#LOC tested per TU_{WO} (Eq. 2)

4.2.4 Maintenance

Thanks to the extensive monitoring and reporting features offered, it is expected that ElasTest can reduce the corrective maintenance effort of SiL: see VM 2.1. To measure the latter, we introduced the metrics in the first column of Table 5 below.

Maintenance				
Metrics	**Unitary**	**Integration**	**System**	**End2end**
Total N of defects found during testing				
Clock time spent in fault localization				
Total N of defects closed				

Table 5: Corrective maintenance data sheet

In particular to achieve the aimed decrease of at least 50% we need to observe:

$Maintenance\ \Delta\ =$ (Total person-$hours_{W0}$ - Total person-$hours_{We}$)/ Total person-$hours_{W0}$ (Eq. 3)

4.2.5 Field Failure

VM 2.2 aims at monitoring the number of field failures reported, i.e. when the SUT is delivered to production. To measure it, we would need to consider failures reported by final end-users/clients of similar products tested with and without ElasTest. However, considering the timeline of the project, we were not able to observe this feedback before project termination.

4.2.6 Scalability

Scalability is one of four dimensions covered in VM 2.3. In principle, by increasing scalability, we aim at supporting testing of larger and more complex applications. According to the DoA scalability is measured as the total number of concurrently supported testing sessions. During the first period, we identified that such concern is important mostly for the TUB vertical, as their IOT demonstrator requires to test many

sensors and actuators. The form adopted for data collection is presented in Table 6 below.

Scalability				
Metrics	**Unitary**	**Integration**	**System**	**End2end**
N applications tested in parallel				
N Sensors and actuators				
N of tested devices				
N concurrent supported sessions				
N. of clients				

Table 6: Scalability data sheet

4.2.7 Robustness

Also robustness is covered under VM 2.3. In the DoA a suggested metric was "downtime after a failure", however this definition does not properly capture the notion of robustness: downtime after a failure should actually refer to maintainability, which is related to how the software is designed and not to how it is tested. Hence this metric is another aspect that we considered worth revising. In literature software robustness refers "*the degree to which a system or component can function correctly in the presence of invalid inputs or stressful environmental conditions*" (see IEEE Std 610.12-1990 [3]). To test for robustness a system must be subject to such invalid or stressful conditions. Therefore we considered to measure as a proxy measure for improved robustness thanks to ElasTest support, the effort that is required to conduct robustness testing without and with the platform, as well as how many failures are revealed, as shown in Table 7 below.

Robustness				
Metrics	**Unitary**	**Integration**	**System**	**End2end**
N. of failures/warnings raised doing robustness testing				
Robustness test strategies applied				
Time/Effort devoted to robustness testing				

Table 7: Robustness data sheet

4.2.8 Security

A third aspect that VM 2.3 aims to observe is security: this is a very important concern in most modern connected applications. Among the four demonstrators, security testing has been conducted by ATOS and Naeva Tec. The ElasTest platform does not aim -at time of writing- at including novel security testing strategies, but it greatly facilitates security testing by providing state-of-art security tests well-integrated with the other facilities offered by the platform. Specifically the platform, based on OWASP ZAP, currently supports the following types of security tests:

- Cross-site scripting vulnerabilities (XSS) (for example, enter HTML Javascript tags and execute code within the web);

- SQL injection vulnerabilities (e.g., executing SQL queries within form fields to access private data in the database, or taking advantage of injectable parameters in the URL).
- Cookie checks (i.e., check if cookies are properly protected by the server by using specific cookie attributes)

We expect that having such basic test capabilities integrated within the platform promotes the adoption of systematic security testing, as opposed to ad-hoc basic testing approaches (as it was in Naeva Tec) or even no security testing (as it was in ATOS). What we aim to measure -in this view- is whether doing security testing via ElasTest can reduce the time/effort required. This is shown in the adopted data sheet (Table 8).

Security				
Metrics	**Unitary**	**Integration**	**System**	**End2end**
N. of incidents/vulnerabilities raised doing security testing				
Time/Effort devoted to security testing				

Table 8: Security data sheet

4.2.9 Quality-of-Experience

Finally, VM 2.3 also covered an advanced aspect in modern multimedia applications, that is the Quality-of-Experience (QoE). One important aim of ElasTest platform is in fact to facilitate non-functional testing for QoE evaluation of multimedia applications and specifically WebRTC. Therefore the vertical demonstrator that was the natural candidate for this study is the FullTeaching application provided by Naeva Tec.

In the consortium experience (for example over the OpenVidu platform), testing for QoE is an effort intensive and very complex task. The ElasTest project includes a specific research activity within Task 5.1 in WP5 devoted to devise more effective and automated ways to measure user's experience, see for example project work in [5] [6].

The DoA planned to genuinely measure end-user's QoE for applications tested using ElasTest, but again, as we explained for 2.2, we could not wait until the demonstrators are released and used by final consumers. Therefore what we did has been to devise an alternative study in which we can compare the respective performances of non-functional testing for QoE of the FullTeaching application, by traditional approaches (mostly manual), and by using the support provided by ElasTest.

The experiment that we set up considered the scenario that OpenVidu testers aim at selecting the best AWS virtual machine type for their Video Conferencing system. The scenario for which they want to select the best AWS virtual machine consists of 5 independent videoconference sessions with 7 users each, whereby all sessions are supported by a single OpenVidu instance. The testers from either branch of the study will check which of three virtual machine sizes: mini (t3.medium), medium (t3.large), full (t3.xlarge) fits best for the purpose, taking into account that the bigger the machine, the higher the costs. So they want to select a size fit for purpose, which is small but still provides good video and audio quality.

In the study, the end-users can be emulated using different browsers running in AWS virtual machines. Then, to measure the quality of the video session and decide whether it is good enough, testers can estimate QoE by means of two metrics provided by all browsers: delay and jitter. Specifically, they need to retrieve these two metrics every second, and make an average across all browsers. If at any moment the delay goes above 150 or the jitter goes above 100, the test fails and the machine instance size used for OpenVidu is discarded.

On the one side, testers NOT using ElasTest will have to provide the entire infrastructure AND implement the test and all the necessary tools for executing the test (this is current state of practice). On the other side, testers using ElasTest will have to configure a multi TJob with 3 IP values so that ElasTest will run the three tests one after another and produce graphs with the delay & jitter metrics for the three executions that are plotted together for a better comparison.

5 Schedule of validation studies

At project start, we had set a schedule for the WP7 validation strategy including several iterations for the three different types of empirical studies. The schedule, presented in D7.1 showed the planned activities for QE, CCS and ES along the three years of project lifecycle, and related to the planned platform releases, from R1 until the last R9.

During the first year, when the complete ElasTest platform was not yet available, we have worked to prepare the validation strategy, by launching some pilot studies. Precisely, we completed the first period with the following tasks:

- We have designed the validation strategy, and prepared all forms to collect data and questionnaires
- We have validated and refined the strategy by performing a first round of studies with the project partners

Then the plan was that in the second year, in preparation of first review (i.e., until M18), we collected some preliminary data, while in the second reporting period we progressively collected more consolidated results.

After completion of first period, we carefully revised the schedule, which was mostly confirmed and followed. The resulting final schedule, which is how we worked, is depicted in Figure 5.1 below.

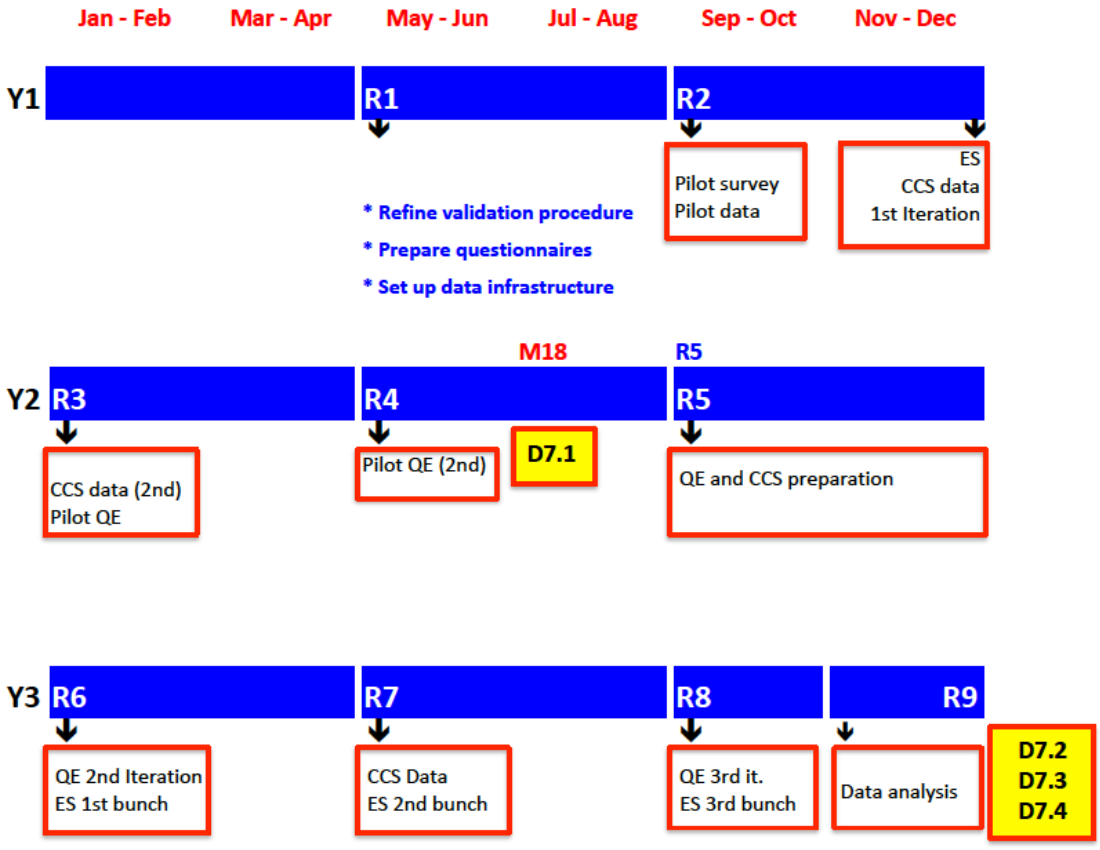


Figure 5.1: ElasTest revised validation schedule

With regard to the original schedule, described in Deliverable D7.1, we realized the following improvements:

- We realized that the foreseen first iteration of QE by 3rd quarter of Y2 would not yet provide meaningful results, because the R5 platform release was still not ready for some of the required features (e.g. security, cross-browser testing).
- We decided to use instead the 2nd half of Y2 to better set the objectives of the empirical studies, also by providing continuous feedback to platform developers about which features to prioritize in the Agile ElasTest development process.
- Concerning the ES, we understood that the questionnaires should be launched in tandem with the studies for objective metrics, and not conceived as self-standing studies. In fact, the questions aim at understanding how testers felt after a testing campaign, so it makes more sense to distribute the questionnaires as a follow-up activity at completion of each study.

6 ElasTest vertical demonstrators

This section will include a description of ALL demonstrators used in the validation, dividing the case of QE and CCS.

The demonstrators used in WP7 span over different application domains, including telecommunication infrastructures and networks (Task 7.2), WWW and mobile applications (Task 7.3), smart environments and Internet of Things (Task 7.4), and multimedia communication (Task 7.5).

As described in Chapter 3, the validation was conducted through both Quasi-experiments and Comparative Case Studies, which were then followed by Survey Questionnaires handled to all participants to assess subjective feelings.

The applications used as subjects for the Quasi-Experiment across the above domains included, respectively, the following applications:

- ATOS Worldline E-commerce platform
- ATOS Worldline Messaging platform
- FOKUS Open5G Core
- Naeva Tec FullTeaching and OpenVidu
- TUB IIOT platform

The demonstrators used in the CCS included the following applications:

- ATOS Worldline E-commerce platform
- ATOS Worldline Messaging platform
- FOKUS Open5G Core
- Naeva Tec Bank Onboarding
- TUB EMBERS

A detailed description of the artifacts is given in the companion deliverables D7.3 and D7.4 to which we refer for details. In the rest of this section, to make the document self-contained, we provide just a summary description of such applications.

As it can be noticed, some partners used the same applications for both types of study, of course relying on different testers and under different scenarios.

6.1 ATOS demonstrators

Two applications have been used to validate the performances of ElasTest, both for the QE study and the CCS one:

- An E-commerce platform for an **Online Supermarket** (OS): through this application a user can make online purchases of food and household products. A characterization of the application is provided in Table 9.

E-COMMERCE PLATFORM: ONLINE SUPERMARKET	
Users	60.000
Daily purchase transactions	>1000
Servers	15
DD.BB. size	67,3GB
Number of pages	30
Home weight	2,5MB

Table 9: Features of ATOS E-commerce platform

The number of products on sale will depend on the selected shop; it can roughly vary between 2,100 and 24,000 products.

- **Messaging platform** (MP): this application allows to manage the sending of PUSH notifications to different apps, also SMS and recently, WhatsApp to different mobile devices. A characterization of this other application is provided in Table 10.

MESSAGING PLATFORM	
Users	>600
SMS sent monthly	>5M
Push sent monthly	>1,5M
Servers	15 (2 web servers, 8 application servers, 2 dd.bb, 1 connectivity, 2 monitoring)
DD.BB. size	50 GB
Lines of code	338K
Home weight	1,2MB

Table 10: Features of ATOS messaging platform

A more extensive description of the two demonstrators can be found in the companion deliverable D7.4.

For the execution of the different rounds of QE, ATOS Worldline had a team of 4 QA Testers, as follows:

Project	#QA Without ElasTest	#QA With ElasTest
e-Commerce Platform	1	1
e-Messaging Platform	1	1

And for the execution of the CCS, the team was formed by 2 QA Testers:

Project	#QA
e-Commerce Platform	1
e-Messaging Platform	1

6.2 FOKUS demonstrators

The Fraunhofer FOKUS Open5GCore toolkit is a practical implementation of the carrier-grade network towards the 5G environment. It mirrors, in a prototypical form, the pre-standard advancements on the core network, radio network integration, distributed management and virtualization.

The Open5GCore aims at providing support and speeding-up research, facilitating know-how transfer from Fraunhofer FOKUS towards partners. It serves as a consistent basis for research projects with meaningful results, enabling fast and targeted innovation, hand-on fast implementation, realistic evaluation and demonstration of novel concepts and technology opportunities.

The overall environment will be a virtual deployment of an 5G network, and an abstract architecture is depicted in Figure 6.1.

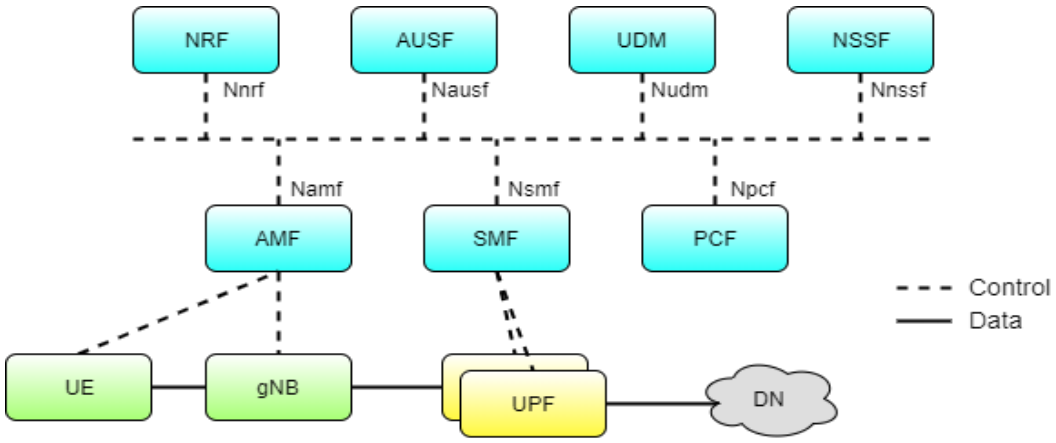


Figure 6.1: FOKUS Open5GCore toolkit

In the second round of data collection the product considered in the experiment was an Open5GCore Rel.4 that includes initial implementation of 5G functionalities tested by a feature-based approach.

In particular, for this QE ElasTest must be run remotely, i.e., outside the main infrastructure to be tested, because vendors and operators are not likely to accept additional deployments within their infrastructure. This forced FOKUS to develop a way for running the tests from ElasTest remotely, and ElasTest on the other side introduced the necessary endpoints so that metrics and logs can be sent from an external infrastructure.

The software core network has been selected as it is the most suited for System in the Large experimentations. Additionally, the main role of 5G Core is to provide a communication platform between services and end-users, being thus very well suited for complex system end-to-end experimentation and testing.

In the QE, the personnel involved in the testing phase were mainly developers with a minor support of testers. The former mainly involved in the first stages of developing process, the latter in the remaining.

The test cases derived (in the tens order) have been produced using a feature-based approach and executed and analyzed by means of proprietary tools. The process is summarized below:

1. Creating a setup for the Open5GCore, i.e., preparing a set of virtual machines and configuring the proper networks.
2. Deploying the selected Open5GCore Components and compile the code.
3. Testing basic functional connectivity between the 5G components (interfaces, APIs).
4. Testing standard procedures using the built-in test component (BT).
5. Testing performance of the SUT using the built-in test component (BT).

The Fraunhofer FOKUS experiments involved the same Open5GCore toolkit already illustrated above, both for the QE and the CCS studies.

The ElasTest platform had the testbed as external SUT and a set of pre-defined functional and performance tests. The setups dependent on the customer's requirements were: KVM, Vmware, Openstack or Kubernetes. For all setups, the ElasTest integrates with the BT to test the 5G Core component. Also for different types of tests, the ElasTest can also communicate directly with the component.

The configuration of the WO branch (kvm/vmware) consisted in:

- Flowmon
- Benchmarking tool
- Zabbix Agents
- Dedicated GUI

The configuration of the WE branch (docker) consisted in:

- ElasTest
- Benchmarking tool

For the quasi-experiment assessment, the experiment was performed with the standard delivery period time. Once the test are written for one release of Open5GCore, then we just have to change the SUT IP and the whole test and be executed on the new SUT.

For the execution of the QE, FOKUS had a team of 2 testers (1 WO + 1 WE), whereas for the execution of the CCS, the team was formed by 3 testers (2 WO + 1 WE).

6.3 Naeva Tec demonstrators

NaevaTec demonstrators focus on Real Time communication based on the WebRTC Technology. WebRTC is a free, open project that provides browsers and mobile applications with Real-Time Communications (RTC) capabilities via simple APIs.

6.3.1 Naeva Tec QE demonstrator

The application chosen as the subject for the QE study is FullTeaching, an e-learning platform that provides teachers and students a full set of collaborative tools under the umbrella of a unique application. This is the same application already used for the first exploratory studies in Deliverable D7.1 [1], but which has since undergone a lot of development and improvements. Notably, the test suite has grown in size and scope: for instance in the QE we also included security testing and QoE testing, which were not performed in first QE round.

FullTeaching is an open-source application to leverage online classes for teachers as well as students. Its main functionalities include:

- Teachers:
  - Multiple course creation.
  - Course attendance management.
  - Session (real time lectures) programming.
  - Resources repository.
- Students:
  - Multiple course assistance and management.
  - Calendar with all the programmed sessions (real time lectures) of all the courses.
- Communications:
  - Forums grouped by course.
  - Sessions (real time lectures) with student intervention if the teacher allows it.
  - Real-time chat during sessions.

FullTeaching provides multiple communication channels between teachers and students (forums, chats, and real time video sessions) and is provided as a WebRTC demonstrator for the ElasTest project by NaevaTec.

Technically, FullTeaching WebRTC functionalities are based on OpenVidu. WebRTC is the ultimate responsible for all media transmission at the very heart of OpenVidu. WebRTC is a modern, cross-platform framework that democratizes media transmission over the Internet. As such, it is promoted by Google, Mozilla, Opera and others. OpenVidu wraps and hides all the low-level operations so it provides a simple, effective and easy-to-use API.

As described in D7.3 FullTeaching demonstrator cannot yet be considered a ready to production app. Its first release has just been released. Such a new project has as main advantage that it provides the demonstrator with much room for improvement, but on the other hand it does not provide a big set of end-users.

"Traditional testing" for this kind of web applications that includes real time WebRTC communications won't grant a satisfying communication. We need to be able to test what the user is experiencing during the communication session, not only how the system and the application behaves under specific circumstances. We call that Quality of Experience (QoE) testing. QoE is defined by the International Telecommunication Union (ITU) as "The overall acceptability of an application or service, as perceived subjectively by the end-user." This is quite an abstract definition but the key is that it is subjective to the user and this is the main challenge for automating QoE testing. The approach we will follow, when testing QoE in WebRTC, is defining some objective parameters such as transmission rate, delay, jitter, bit error rate, packet loss rate, etc. and assign them some thresholds where we consider the communication can be successful. Currently these parameters aren't easy to measure on automatic tests and nearly impossible to use as acceptance metrics. ElasTest is improving the simplicity to take these measures and is expected to allow us to use them as acceptance metrics on QoE tests. This will make a difference and bring to the table a whole new set of possibilities for testing QoE on WebRTC communications, it will also give us the ability to increase the overall quality of this kind of applications while reducing TTM.

For the purpose of QoE testing we have isolated the communication functionalities of FullTeaching provided by OpenVidu and created a test application, easier to launch but that would cover strictly the real time video communication. That has been done in order to reduce the time of test execution.

For the execution of the QE (both on FullTeaching and specifically OpenVidu for the QoE study), Naeva Tec had a team of 2 testers (1 WO + 1 WE).

6.3.2 Naeva Tec CCS demonstrator

For the Comparative Case Study Naeva Tec selected the "Naeva Tec Digital Bank onboarding" Proof of Concept. This is a complete application developed to showcase how OpenVidu can be added to banking platforms to allow digital onboarding following the new European Banking directive that enables the digital onboarding based on video-conferences as a valid method to register, hire or buy banking products. The PoC is based in a one to one recorded video-conference where a bank agent reads and provides the information about the product and the client accepts the conditions read by the agent, also in the same video a valid ID should be shown. The recorded video-conference is proof of the contract.

The application is considered a PoC as it isn't used by banks directly but the technology behind is added and integrated in their own system/platform so different modifications may be needed.

6.4 TUB demonstrators

6.4.1 TUB QE demonstrator

For this vertical demonstrator, TUB has chosen OpenIoTFog (OIF), a set of Industrial Internet of Things (IIoT) applications that involves sensors and actuators commonly found on the industry shopfloor which are deployed on fog/edge nodes. OpenMTC lies at the core of OIF, enabling Machine to Machine (M2M) communication between applications and used it as a middleware. It is important to note that TUB provides a Test Support Service (TSS) called ElasTest Device Emulator Service (EDS) that is involved in deploying emulated sensors or actuators on demand as needed by the application implemented in the demonstrator. During Release 8 of ElasTest, EDS already encompasses 3 applications that were part of OIF. Furthermore, EDS is available as a full fledged application of OpenMTC containing a manifest of 3 different sensors and a simple actuator. The demonstrator is thus able to get data from the sensors, apply logic to the data and flag the actuator based on the logic. One can imagine System in Large, composed of several applications that can be tested using EDS. In an ideal setting, minimal EDS would provide more general variety of possible sensors and actuators and the user implementing the demonstrator application can choose from them and implement an IIoT application.

For the purpose of first round of Quasi experiment, TUB used OpenMTC. OpenMTC is an implementation of the oneM2M standard. OpenMTC is offered as open source software and is currently in beta release. A user can clone the software and write an application and test it using OpenMTC. In the context of ElasTest, a demonstrator application implemented by a user, becomes a System under Test (SUT) and since the OpenMTC is used to implement such an application, OpenMTC now becomes part of SUT. TJobs can now cover tests belonging to specifics of the implemented demonstrator application as well as OpenMTC. Continuing with this argument, TUB proposes to use test specifications of oneM2M to come up with TJobs concerning the development of OpenMTC. The advantage of testing OpenMTC in this regard is that it can cover test cases for a wide range of possible future demonstrator applications. This can provide a good view on the proposed EDS architecture and demonstrator applications to reviewers in terms of metrics provided by quasi experiment.

For the purpose of second round of quasi experiment, TUB used OpenMTC as a middleware to develop and test an IIoT application based on the concept of control loop. The IIoT application comprises of sensor, logic and actuator. In the control loop, a temperature sensing application is run that flags an alarm if temperature goes above 50 degrees centigrade. For this a temperature sensor is needed which feeds data in periodic intervals say 1 second to the logic. The logic decides if an actuation is needed by checking whether the temperature provided by sensor is greater than 50 degrees. If greater than 50 degrees an actuating signal is sent to actuator that may be an alarm. In the above-mentioned example of temperature monitoring, the temperature sensor and the actuator are provided by EDS, while the logic is implemented by the demonstrator, collectively acting as a System under Test (SUT). Furthermore another

Test Support Service (TSS) of ElasTest called ElasTest Monitoring Service (EMS) was used along with OpenMTC as a middleware to construct TJobs.

The QE involved one WE tester and one WO tester of comparable expertise.

6.4.2 TUB CCS demonstrator

For the purpose of CCS, TUB used EMBERS, a Horizon 2020 project funded from the EU. EMBERS is a smart city project, the tests were performed on a smart city platform, called the Mobility Backend as a Service (MBaas). For the purpose of CCS, different kinds of configuration were used for WE branch. These configurations include different software for several tests, i.e. for connection testing between the broker and the clients. The tools include Docker images, shell scripts, and test suites. In case of ElasTest high level manual and automated SDK testing were performed to create TJobs, which is the job to run the SUT and initiate testing.

7 Results

In the following of this chapter, we report and comment the results achieved over the above presented vertical demonstrators. We first illustrate the results from the Quasi Experiment, then those from the Comparative Case Studies and finally from the Empirical Surveys.

7.1 Summary of VMs and their assignments

As above discussed, the demonstrators provide quite different application domains, development processes and technologies. So, when planning the studies we somehow expected the results we would observe differ as well, as it was also evident from our preliminary observations at M18 reported in D7.1.

As described in Section 4.2, we had to face the reality of the project time constraints on the one side, and the effort available and constraints coming from demonstrators on the other. Hence, in the second stage of evaluation we made a careful and incremental revision of metrics for validating the KPIs. As explained, we could not adequately measure those aspects that could only be evaluated after some period of usage in operation of the demonstrators validated by ElasTest. This applies to VM 2.2 and VM 2.4. In practice, for both of them, we should assess the effect for the final users of the systems tested with ElasTest, precisely in terms of reduced field failure reports (VM 2.2) or more positive feelings of end-users (VM 2.4). As this deliverable is prepared in parallel with finalization of the ElasTest platform and its usage within the demonstrators, we could not yet materially collect field usage data to report at time of writing. For the subjective metrics, we have instead taken the data from testers; for VM 2.2 the metrics was not evaluated.

Moreover, we could not evaluate all of the metrics at each of the four partners. This did not come as a surprise, as of course it depends on the specific processes in place within each partner, as well as on the product under test. In fact, in the planning stage of the validation, we made a collaborative planning to try to ensure that as many KPIs as possible could be assessed, by at least one of the partners.

As anticipated in Figure 4.1, we report in a matrix format the mapping of validation metrics to vertical demonstrators and to type of study. In the following we report the measures gathered for each demonstrator individually, and then in Chapter 8 we attempt to draw some overall conclusion across the four of them.

7.2 Quasi experiments results

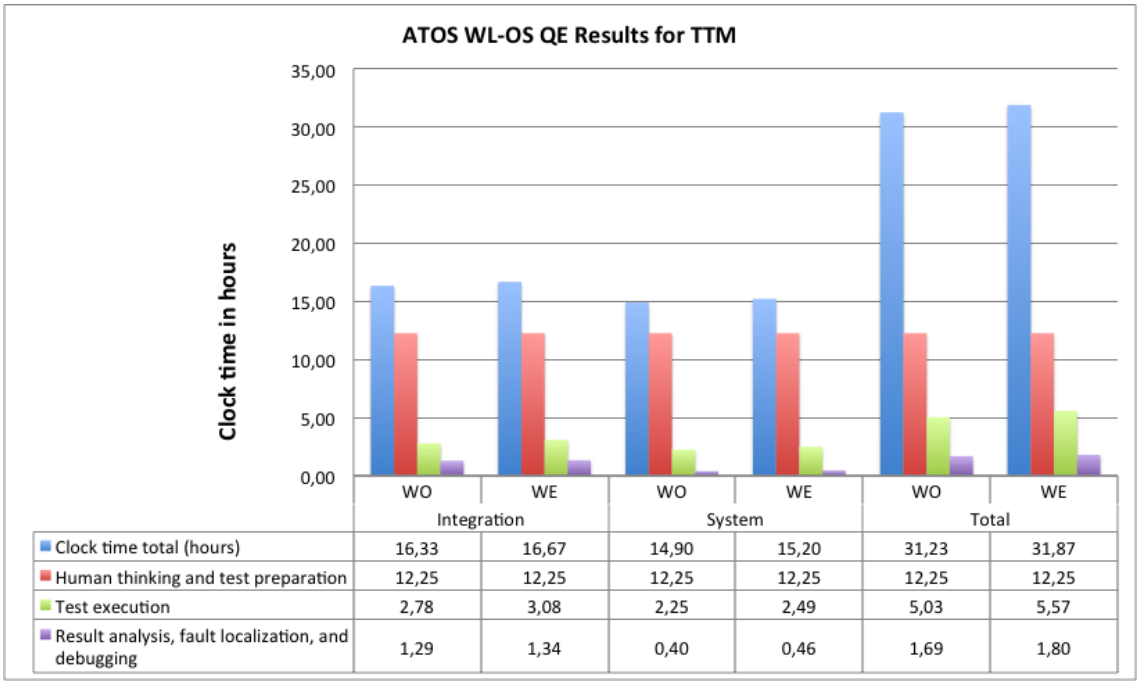
In this section we report the results from the Quasi Experiments for the four partners.

7.2.1 QE results from ATOS Worldline

In the second period, the ATOS partner conducted the QE over two different products, as described in Section 6.1. Before illustrating the results, it is worth explaining that the scenario of ElasTest adoption for ATOS differed significantly from the other partners, because in the two selected projects the testing process is mostly manual.

Manual testing continues to run largely on new, changing and non-stabilized applications, where the testers derive and execute manually the test cases. The tests are managed by means of TestLink, which is a widely used Open Source platform (available from http://testlink.org/) supporting Test Management for web systems. To facilitate adoption of ElasTest by ATOS, as well as by any other company using TestLink, the platform has been extended to include specific features that allow to access TestLink GUI from ElasTest, as well as to bind TestLink annotations with ElasTest logs. A specific tutorial is made available from https://elastest.io/docs/testlink/. While clearly the best results from ElasTest adoption are foreseen for automated test processes, we could also observe some improvements to ATOS process, as illustrated in detail below.

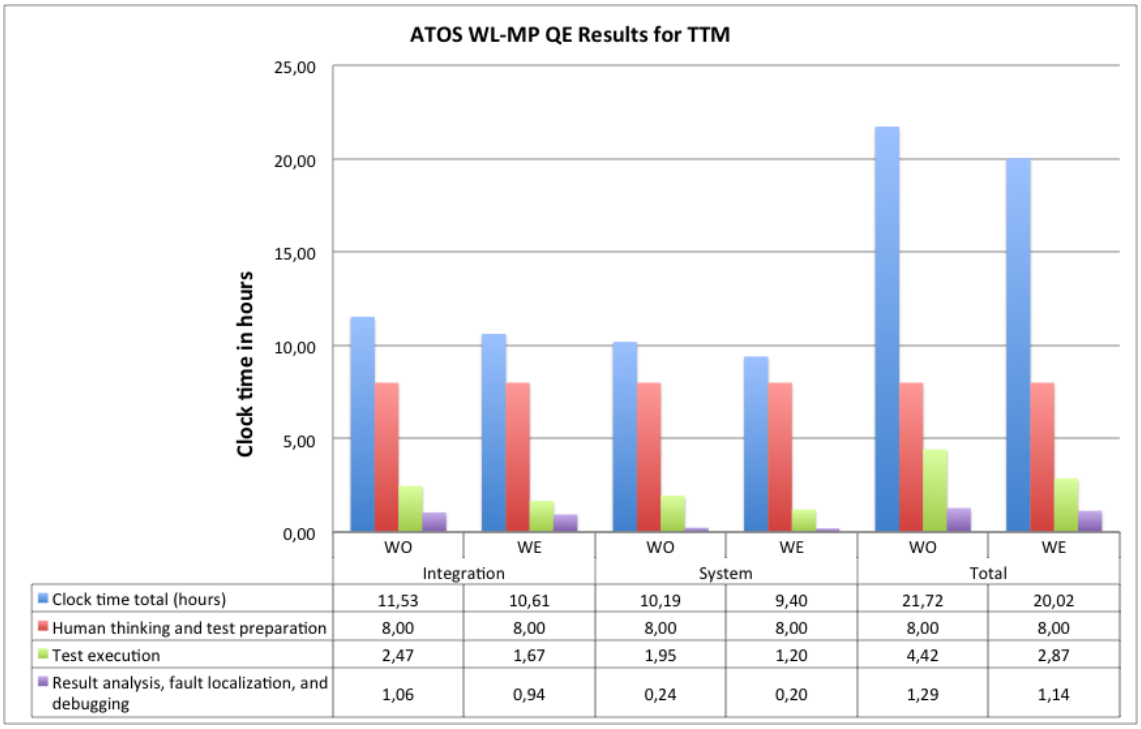
In Figure 7.1 and Figure 7.2 we report the data relative to the times measured by the WO tester and the WE tester, for the Online Supermarket (OS) and Messaging Platform (MP), respectively.



	WO	WE	WO	WE	WO	WE
	Integration		System		Total	
Clock time total (hours)	16,33	16,67	14,90	15,20	31,23	31,87
Human thinking and test preparation	12,25	12,25	12,25	12,25	12,25	12,25
Test execution	2,78	3,08	2,25	2,49	5,03	5,57
Result analysis, fault localization, and debugging	1,29	1,34	0,40	0,46	1,69	1,80

Figure 7.1: Clock time spent by ATOS WL testers in the QE over Online Supermarket

The fact of using ElasTest in the execution of the test cases does not provide an improvement in OS in the **testing time** is due to the slowdown of the use of ElasTest in the execution of the test cases. The more the log and metrics disposition time is reduced, the more the test execution is slowed down. If we add that the use of Cross Browser is not feasible with this application, the overall result is not as satisfactory. In deliverable D7.4 this situation is detailed in more depth (Chapter 2).



	Integration		System		Total	
	WO	WE	WO	WE	WO	WE
Clock time total (hours)	11,53	10,61	10,19	9,40	21,72	20,02
Human thinking and test preparation	8,00	8,00	8,00	8,00	8,00	8,00
Test execution	2,47	1,67	1,95	1,20	4,42	2,87
Result analysis, fault localization, and debugging	1,06	0,94	0,24	0,20	1,29	1,14

Figure 7.2: Clock time spent by ATOS WL testers in the QE over Messaging Platform

Although, the preparation and thinking time does not vary due to the use of ElasTest, since the basic tool used is the same, Testlink, - in the same way as in OS -, it is important to highlight that the **test execution time** decreases considerably (40%) using ElasTest. This decrease in runtime is mainly due to the use of Cross Browser. In the messaging platform, the use of Cross Browser allows to reduce the execution time significantly, since the same test case can be executed synchronously on two browsers at the same time. In deliverable D7.4 this topic is further discussed.

Concerning **reusability** (VM 1.2), the ATOS partner reported that due to their applied manual test process, reusability was low: they could reassign test cases across different test plans after adjustments, but test execution with TestLink was manual. However, in the MP application, thanks to the CROSS BROWSER functionality made available from the ElasTest platform, the testers were able to reproduce with only one execution the test results of executing the SUT within two different browsers. Their assessment is that the effort saved thanks to such reuse was 88% (see Figure 7.3 below).

Reusability	**Release 2.3.4 (WE) vs Release 2.2.2 (WO)**			
MESSAGING PLATFORM	**Integration**		**System**	
	WO	**WE**	**WO**	**WE**
Tests reused (in total or in part)	0%	88%	0%	88%

Figure 7.3: Reusability estimation in QE for Messaging Platform

Concerning tester **productivity** (VM 1.3) we report the results from both OS and MP applications in Figure 7.4 below. The bars report the number of test executed over one release cycle of the demonstrators, in orange color for the WO branch, and in blue color for the WE branch.

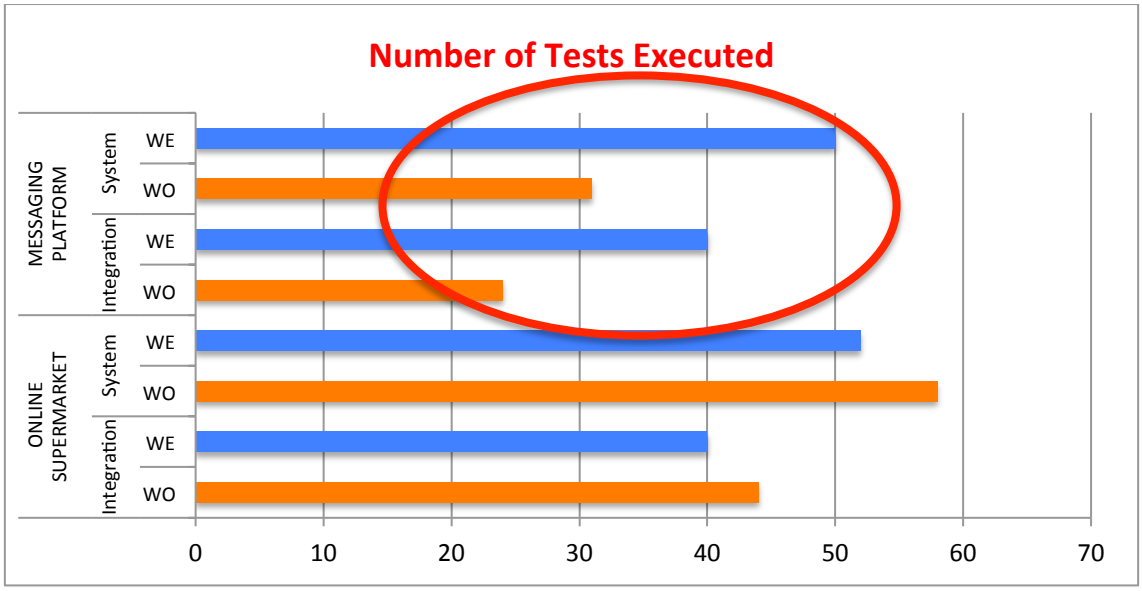
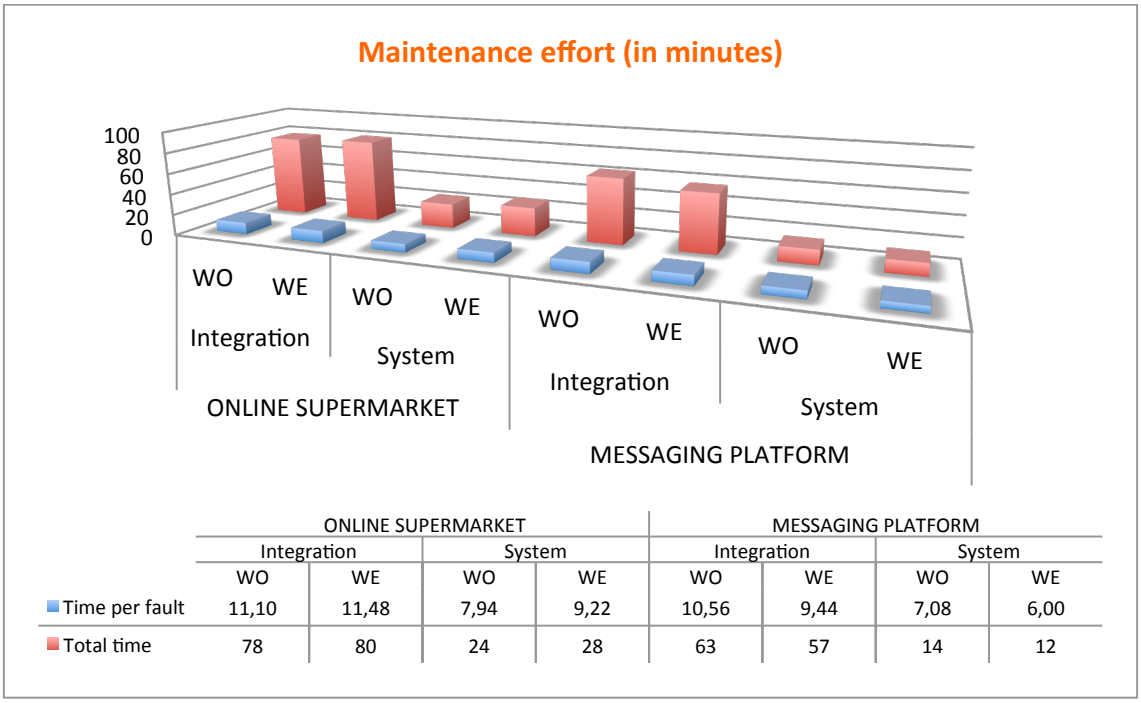


Figure 7.4: Productivity data from QE for ATOS WL testers on OS (bottom) and MP (top) demonstrators

In particular, we notice how using ElasTest impacted negatively on productivity for the OS application: WE testers could execute 40 and 52 test cases in integration and system testing, respectively, against a total of 44 and 58 test cases executed by the WO testers. The partner reported that the degraded performance was due to their lack of access to the OS test server for getting metrics and logs. So for reporting the logs to ElasTest they artificially created an external *cron* function, which slowed down as a result the test execution. Hence the results are not a genuine snapshot of actual performance.

With MP the server could be configured to send metrics and logs to ElasTest. It should be noted that the provision of online logs in ElasTest also slowed down the execution of test cases, although not as drastically as in OS. In fact, in MP it was obtained **an improvement in productivity, between 40-60%,** and this was due to the use of Cross Browser functionality implemented in ElasTest. The use of Cross Browser increases the test cases to be tested in the same period of time.

Concerning **corrective maintenance** (VM 2.1), in Figure 7.5 we show the effort spent on the two demonstrators.



	ONLINE SUPERMARKET				MESSAGING PLATFORM			
	Integration		System		Integration		System	
	WO	WE	WO	WE	WO	WE	WO	WE
Time per fault	11,10	11,48	7,94	9,22	10,56	9,44	7,08	6,00
Total time	78	80	24	28	63	57	14	12

Figure 7.5: Effort (in minutes) spent by WE and WO testers in QE for fault localization and correction in OS and MP demonstrators. Red bars refer to cumulative effort; blue bar to effort per fault

We did not observe large differences: reasoning on the density measures (blue bars in Figure 7.5), for the OS application we observed an **increase of maintenance effort of 3% and 16%** for integration and system testing, respectively. For the MP application, instead, we could observe a **decrease of maintenance effort of 11% and 15%** for integration and system testing, respectively. In both cases such values are quite far from the expected KPI of 50% decrease, confirming that the expected advantages in maintenance effort reduction are not fully achieved in a manual testing process such as the one in ATOS.

On the other hand, ATOS reported quite positive feedbacks concerning **security** testing (VM 2.3.3). In fact, thanks to ElasTest they could launch security tests (as explained in Section 4.2.8) in few minutes: the data for the two demonstrators are reported in Figure 7.6. Please note that they did not execute security testing without ElasTest support (data reported are educated estimates).

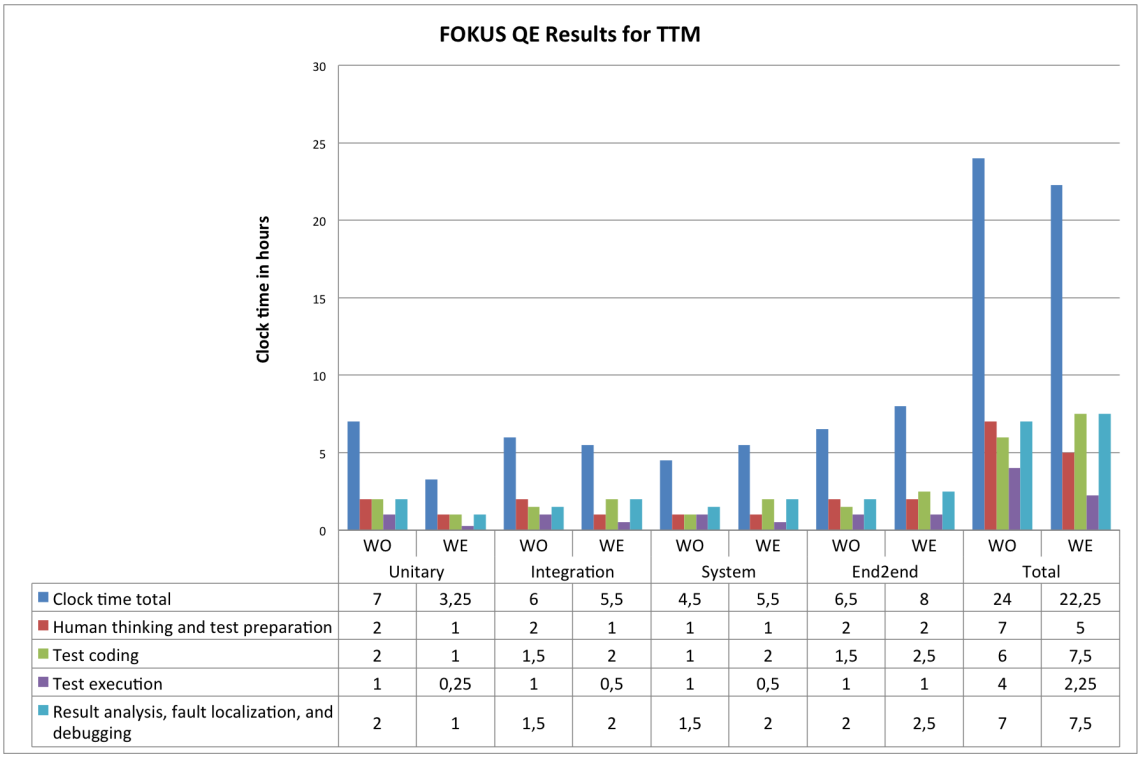
Security	ONLINE SUPERMARKET				MESSAGING PLATFORM			
	Integration		System		Integration		System	
	WO	**WE**	**WO**	**WE**	**WO**	**WE**	**WO**	**WE**
N. of vulnerabilities		0		1		1		1
Time	*16 h (est)*	2 min	*14 h (est)*	2 min	*16 h (est)*	8 min	*14 h (est)*	5 min

Figure 7.6: Security data from QE by ATOS WL testers for the WE branch; data for WO branch (in italics) are estimates as no security testing was performed

7.2.2 QE results from FOKUS

During the QE study in FOKUS, we collected measures of time employed along the testing process, test reusability and tester's productivity.

In Figure 7.7 we illustrate the detail of **time** data measured by the WO tester and the WE tester. We can see a trend of time reduction, although limited, by comparing the total time spent in testing that reduced from 24 hours to 22,25, with overall delta of 7%. However, the reduction is achieved mostly in the unitary stage, whereas the time needed for E2E testing increases in the WE branch.



	WO	WE	WO	WE	WO	WE	WO	WE	WO	WE
	Unitary		Integration		System		End2end		Total	
Clock time total	7	3,25	6	5,5	4,5	5,5	6,5	8	24	22,25
Human thinking and test preparation	2	1	2	1	1	1	2	2	7	5
Test coding	2	1	1,5	2	1	2	1,5	2,5	6	7,5
Test execution	1	0,25	1	0,5	1	0,5	1	1	4	2,25
Result analysis, fault localization, and debugging	2	1	1,5	2	1,5	2	2	2,5	7	7,5

Figure 7.7: Clock time spent by FOKUS testers in each testing phase and totally in the QE.

On the other hand, in the same study we could observe a better reusability of test code for the last testing stages. As we anticipated in Section 4.2, it has been quite difficult to assess reusability improvement thanks to ElasTest in the course of the project. In particular, it has not been easy to do unitary and integration tests with ElasTest due to the type of design of product. In fact, Elastest is used as external tool with this demonstrator, which makes it difficult to do tes cases in which the product does not provide an open interface to interact from outside. In the QE study, however, the FOKUS testers could evaluate whether some of the test cases included code or functionalities reused from other test cases. The results are depicted in Figure 7.8 below.

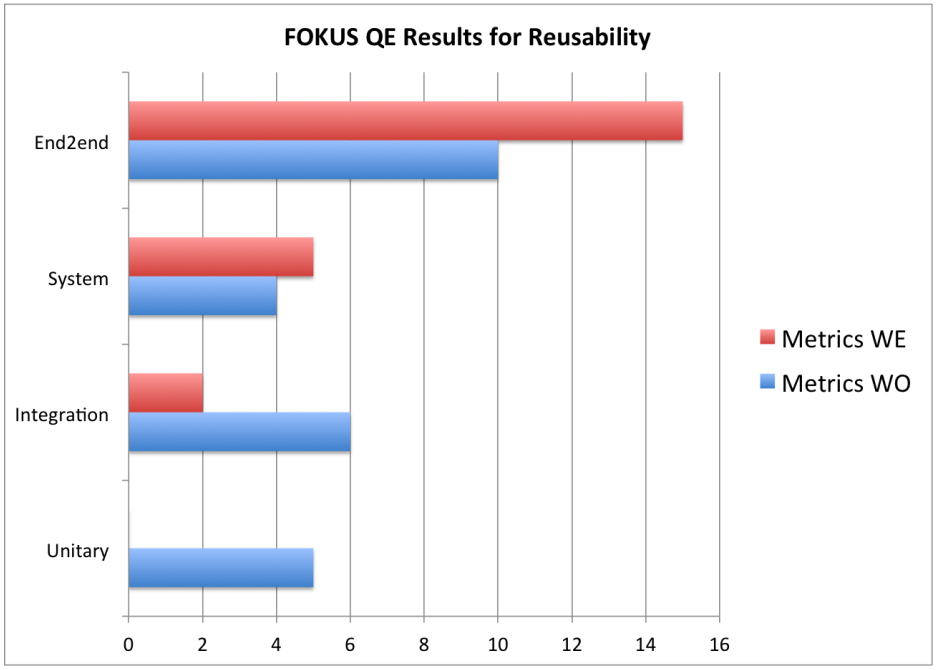


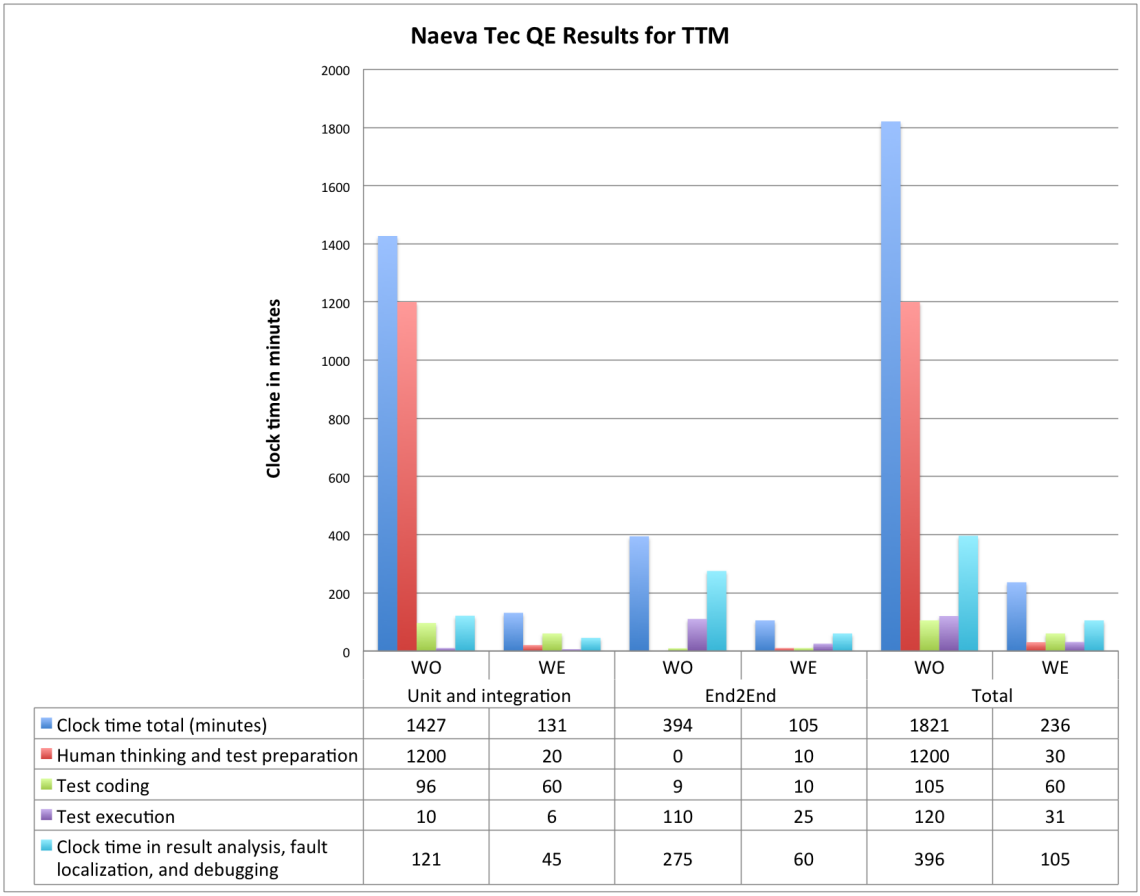
Figure 7.8: Number of reused tests by FOKUS testers in QE

During the study we also collected data concerning **productivity**. Here the results look promising and also helped explaining the results observed for TTM. In the same observation time the WO tester wrote and executed 15 E2E tests against a double number of 30 E2E test cases for WE. Hence we observed **an overall increase in productivity of 100%,** measured over the number of tests. If we also consider the LOCs of test code, we noticed that the ElasTest-supported test cases included on average 30 LOCs against the 20 LOCS of WO testing. This might be due to the fact reported by FOKUS testers that during testing Open5GCore acts as a black box for ElasTest, that needs to find ways to circumvent this for analysis of SUT behavior.

7.2.3 QE results from Naeva Tec

The QE study at the Naeva Tec partner could assess the following KPIs: testing time, maintenance, security and Quality-of-Experience (QoE) on the FullTeaching (FT) demonstrator. In particular, we recall that this was the only demonstrator on which we could evaluate QoE, and this could be done by an ad hoc devised study over the OpenVidu component (see Section 4.2.9).

The results assessed for **time**, which is measured in minutes, are illustrated in Figure 7.9. This partner measured cumulatively the time spent in unit and integration testing, and then the time spent in E2E. Overall the time reduction achieved all along the testing process thanks to ElasTest is impressive, especially for the support provided in the unit and integration stage, which reduces the overall time from 1427 minutes to 131, i.e., with a **reduction of more than 90%**. Specifically, the **reduction for the E2E stage is 73%**, still very high.



	WO	WE	WO	WE	WO	WE
	Unit and integration		End2End		Total	
Clock time total (minutes)	1427	131	394	105	1821	236
Human thinking and test preparation	1200	20	0	10	1200	30
Test coding	96	60	9	10	105	60
Test execution	10	6	110	25	120	31
Clock time in result analysis, fault localization, and debugging	121	45	275	60	396	105

Figure 7.9: Clock time spent by Naeva Tec testers in the QE

Concerning **maintenance**, the Naeva Tec testers reported the numbers of bug found, respectively: 2 bugs during End-to-End testing both in the WO branch and WE branch, and 0 bugs during Unit and integration testing in the WO branch, against 7 bugs detected for the same phase in the WE branch. What is interesting for us is then to measure the time needed for fault localization as we expect that ElasTest can provide useful support for such activity. In fact, for localization of the 2 bugs found by the WO tester, she spent 53 minutes. In comparison, the WE tester spent 35 minutes for localization of the 7 bugs detected in Unit and integration, and only 8 minutes for the 2 E2E bugs. Overall, the time spent on average on each bug without ElasTest support is 26.5 minutes, against a corresponding average of 4.78 minutes using ElasTest. **The overall observed decrease in maintenance effort is of 82%.**

Even more startling are the results observed for **security** testing. The Naeva Tec demonstrator included in fact important security aspects, and during the QE study a security testing activity was also launched. The WO tester had to manually set up and launch the vulnerability detector tool, and could finally detect 2 vulnerabilities employing a total of 1298 minutes. The WE tester instead could just launch the security engine embedded within the ElasTest platform, and he could complete the testing session after 10 minutes finding 1 vulnerability. Considering the **density of time spent for vulnerability, ElasTest decreases it of 98%**.

As said, this demonstrator is the only one that also conducted a study to indirectly measure potential improvement in **Quality-of-Experience.** With reference to the experiment steps described in Section 4.2.9, we collected the feedback from WO and WE testers both performing those same steps, concerning their effort and "how much" they could stress the OpenVidu system for assessing delay and jitter.

As expected the results, in terms of users/sessions/vm-instance, are similar, in fact both branches tested the same system, and if the study was well conducted they should observe same delay and jitter (these should not vary with the testing tool used). What differs is the effort and complexity in conducting the study. Quantitatively, WO testers, who used JUnit, Selenium and Maven (plus manual reviews), employed about 73 hours to perform the study, of which the greatest part (~66 hours) have been employed in test coding + test set up. On the other side, WE testers employed all in all ~4 hours to conclude the same study, of which 2 hours were needed for test coding. An important difference was the coding of the retrieval of values within the test code. The WE testers can use the EUS and relay in ElasTest taking care of the metrics in the browser, and having to develop a system where browsers made accessible the statistics for the test to retrieve and present (a very complex task).

Perhaps more relevant is the qualitative feedback from the study: a conversation post-study between the WO and WE testers led to agree on the following advantages in using ElasTest:

- A lower number of test instances to be executed, thanks to the multiconfiguration browsers;
- A more comfortable way to save and inspect logs of non-functional data;
- The information about jitter/delay was retrieved without any extra coding;
- WE testers had more time for configuring a custom solution, tailored on the desired performance and quality, reducing the costs in instances;
- Faster debug and analysis, because by just looking to few video frames, without need to look at detailed numbers, WE testers can judge the quality. For example, in the configuration that involved the server with lower performance, by just glimpsing at the audio file available in the ElasTest results page it can be immediately noticed that video and audio were out of sync of many seconds; this allowed the WE tester to discard the configuration without wasting further time in analysis.
- WE testers could more easily correlate the video with the data and the graphs that provide hints about what and where was the problem (in the browser, or in OpenVidu for example).

7.2.4 QE results from TUB

In the QE, for the TUB demonstrators we could collect data concerning: Time-to-Market, Scalability and Robustness.

The results collected for Time-to-Market are visualized in Figure 7.10, in which we also report the detailed data.

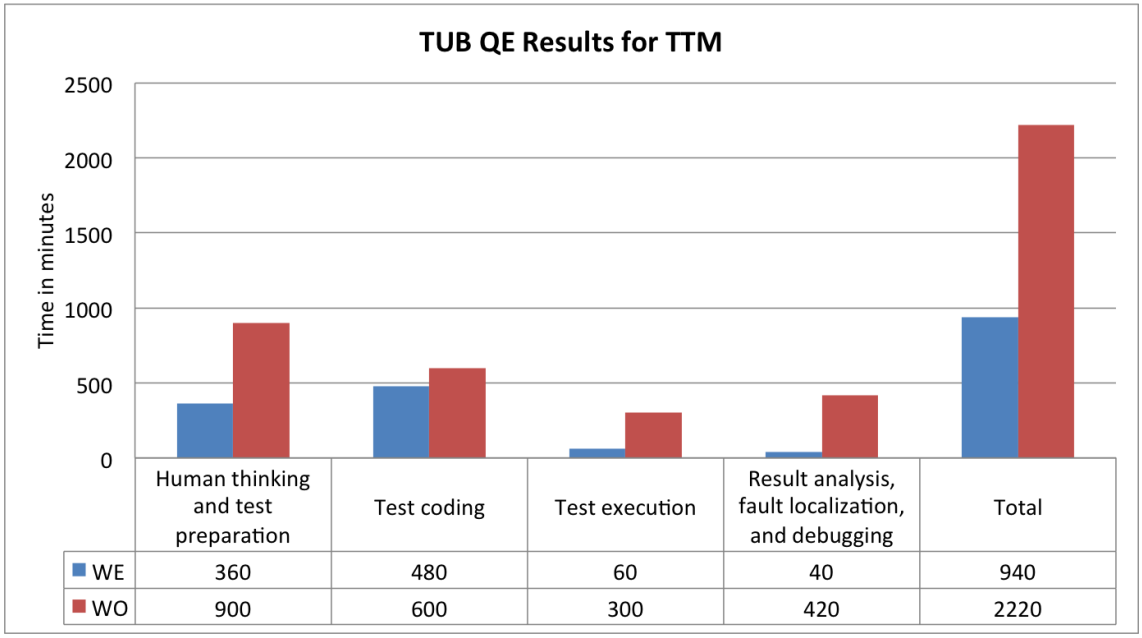


Figure 7.10: Clock time spent by TUB testers in each testing phase and totally in the QE.

As shown the results reveal a large gain in the time spent for the various test activities. Considering the total time spent, the WE tester employed 15 hours and 40 minutes, against a total duration of 37 hours taken by the WO tester: the overall reduction is 58%, and the gain concerned all phases. In the experiment we could only measure the time spent in the testing phase. However, as well known testing can take a substantial part of TTM, even up to 50%; hence, these observations hint at great potential for TTM reduction for the TUB partner.

In Figure 7.11 below we show the data collected for **scalability** KPI. By using ElasTest, the tester executed a lower numbers of parallel applications (3 against 9) and also of concurrent sessions (6 against 10). This might at first sight be interpreted as counter-intuitive, apparently saying that WO can perform "more" parallel test: the TUB testers explained that the data provide opposite feedback: because the ElasTest platform enhances visibility they could achieve same confidence from a lower number of concurrent test executions. The lower number of concurrent executions helps in lower resources consumption and programming complexity. This goes a long way when testing SiL IIoT applications.

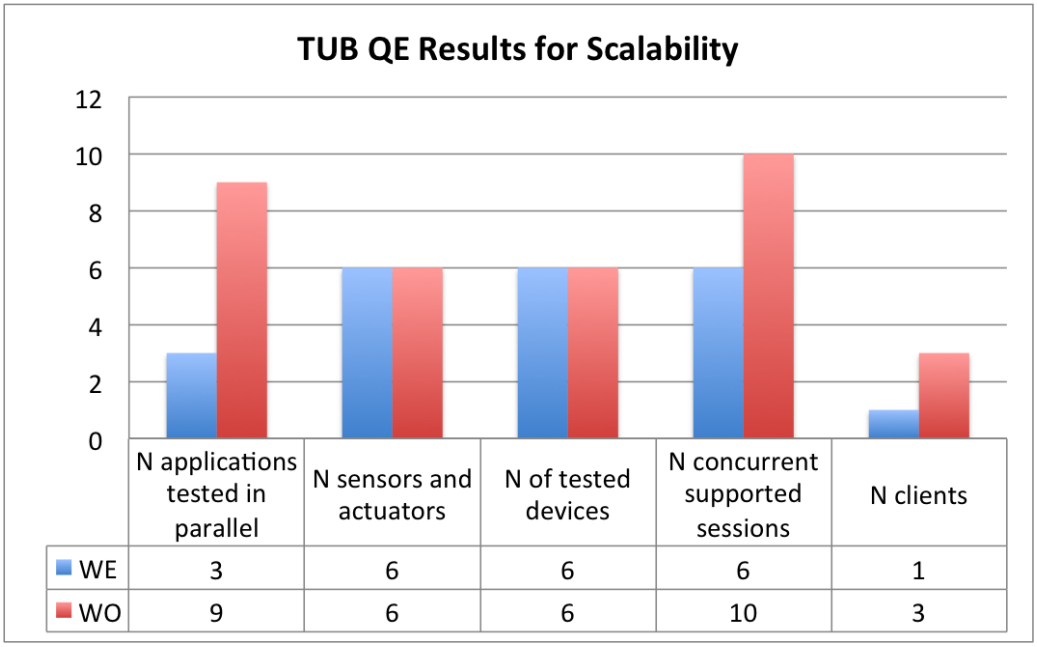


Figure 7.11: Data collected by TUB to assess scalability KPI in QE

In the TUB QE study, we could also collect data concerning a **robustness** testing session. During such testing session, for the purpose of evaluating the fault detection capability, both the WE and the WO testers testers were given the task to detect a couple of unknown random faults injected, one each in TJob and SUT while the test was executing. In both branches they detected the two faults, although they used quite different approaches. The WO tester employed static and dynamic techniques for introducing faults in the device scripts. The whole session took 4 hours. In contrast, the WE tester could rely on the API from EDS engine: he could easily send through such API a request to modify the sensor under test. The whole session for WE took 1 hour. Hence, in terms of **time required for robustness testing** ElasTest for the TUB vertical demonstrator ElasTest allows a 4 times reduction, or **75% decrease**.

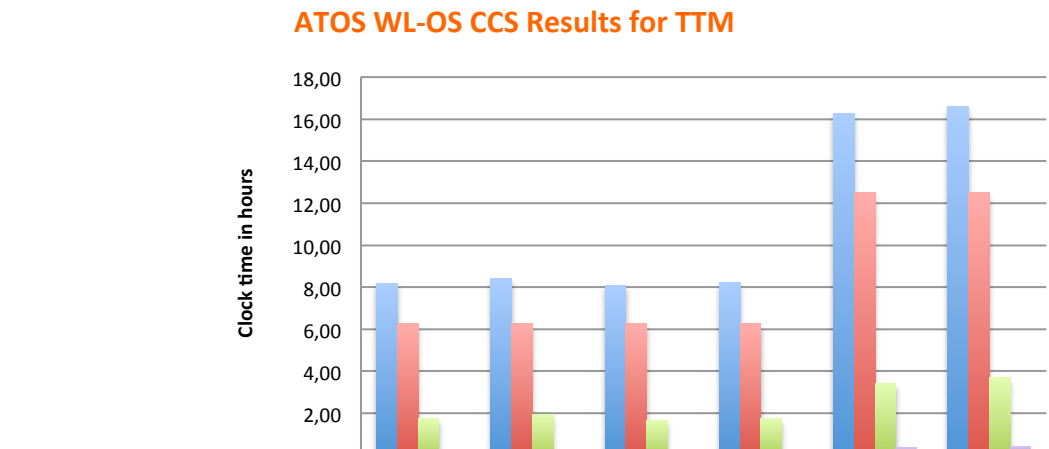
7.3 Comparative case studies results

In this section we now report the results from the Comparative Case Studies for the four partners.

7.3.1 CCS results from ATOS Worldline

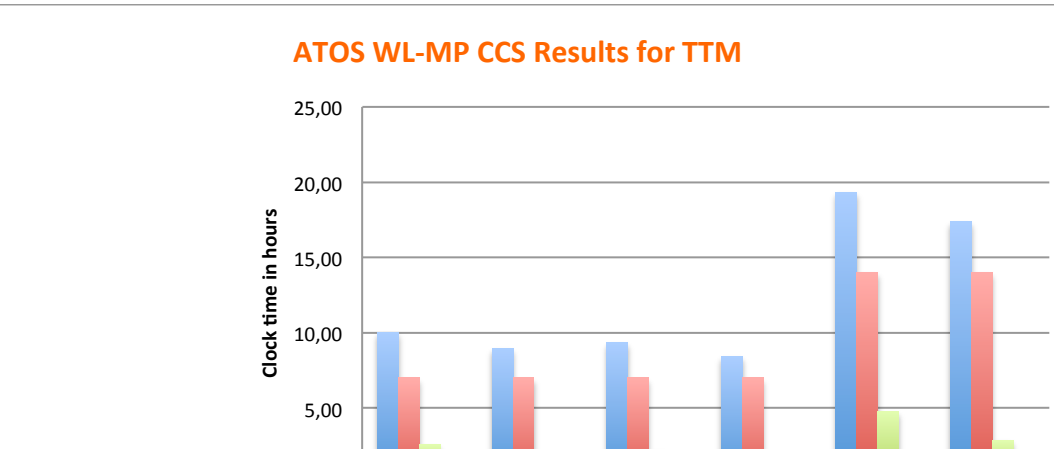
In the CCS, ATOS used the same applications of QE (OS and MP). The testers are other professionals from ATOS WL that are not involved in the project and performed the testing without any control parameter.

We measured the time needed for the different testing activities (preparation, execution and analysis) over the Integration and System testing stages. The results are shown in Figure 7.12 and Figure 7.13.



	Integration WO	Integration WE	System WO	System WE	Total WO	Total WE
Clock time total (hours)	8,18	8,40	8,10	8,21	16,27	16,61
Human thinking and test preparation	6,25	6,25	6,25	6,25	12,50	12,50
Test execution	1,73	1,96	1,67	1,76	3,41	3,71
Result analysis, fault localization, and debugging	0,19	0,19	0,18	0,20	0,37	0,40

Figure 7.12: Clock time spent by ATOS WL testers in the CCS over Online Supermarket



	Integration WO	Integration WE	System WO	System WE	Total WO	Total WE
Clock time total (hours)	10,02	8,95	9,32	8,41	19,34	17,36
Human thinking and test preparation	7,00	7,00	7,00	7,00	14,00	14,00
Test execution	2,60	1,55	2,18	1,33	4,79	2,88
Result analysis, fault localization, and debugging	0,42	0,40	0,14	0,08	0,55	0,48

Figure 7.13: Clock time spent by ATOS WL testers in the CCS over Messaging Platform

The trend is similar to what we observed in the QE: the usage of ElasTest lightly increases (~2%) the time taken to conclude the testing of OS application, while on the contrary it lightly reduces (~10%) the testing time on the MP application.

Concerning **reusability**, the WE tester assessed approx. 80% of effort reduction for the MP application, thanks to Cross Browser feature offered by the platform. This is very similar to the results observed in the QE.

Concerning tester **productivity**, the CCS results are shown in Figure 7.14. The bars report the number of test executed over one release cycle of the demonstrators, in orange color for the WO branch, and in blue color for the WE branch.

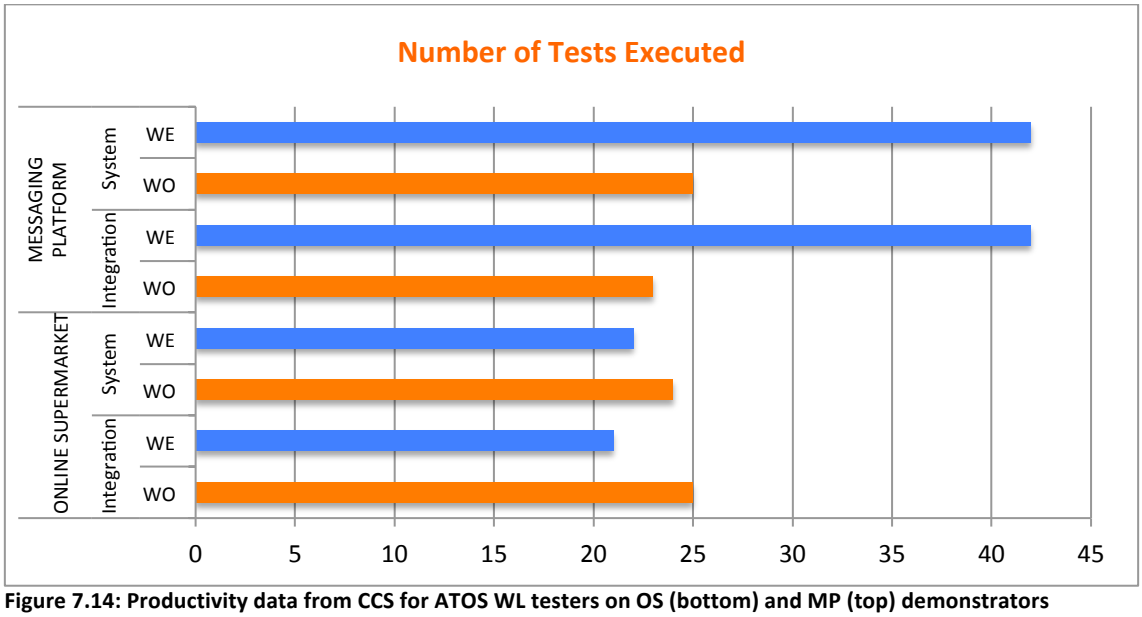
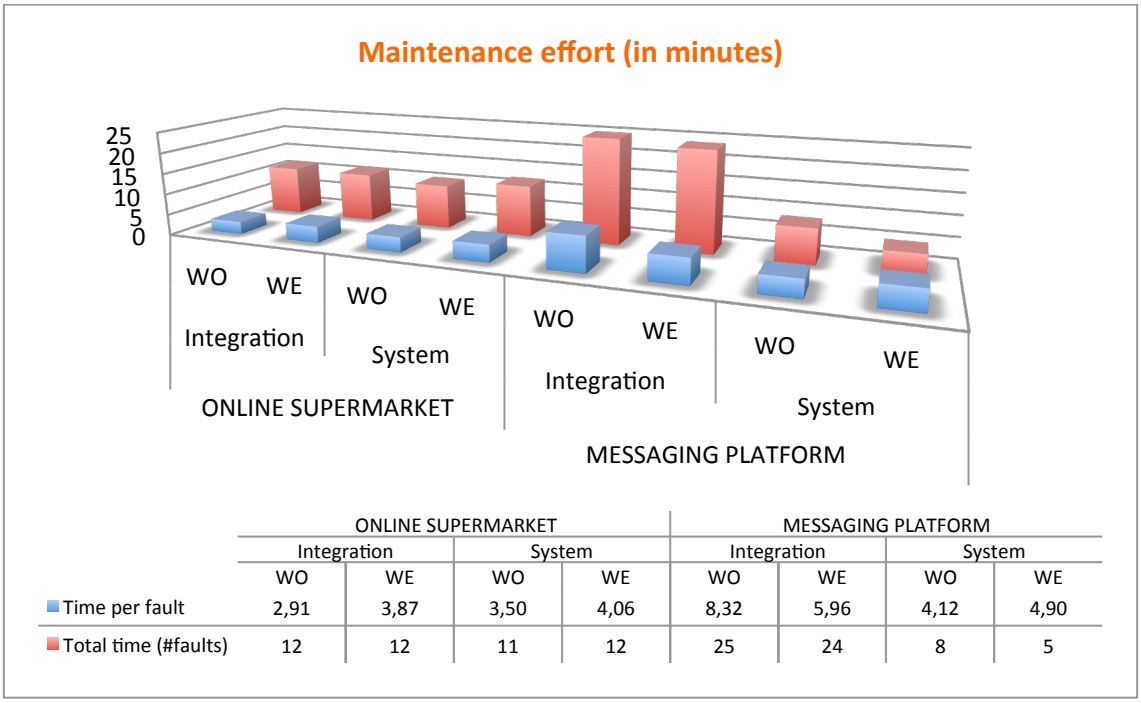


Figure 7.14: Productivity data from CCS for ATOS WL testers on OS (bottom) and MP (top) demonstrators

In particular, for the OS application the WE testers could execute 21 and 22 test cases in integration and system testing, respectively, against a total of 25 and 24 test cases executed by the WO testers. For the MP application, the WE testers executed 42 and 42 test cases in integration and system testing, respectively, against a total of 23 and 25 test cases executed by the WO testers. Hence, still remaining valid the observations made when presenting the QE results for productivity (see Section 7.2.1), the productivity increase for MP is consistent: **~82% and ~68% more test cases could be executed in integration and system testing stages**, respectively.

The CCS collected data relative to maintenance effort are shown in Figure 7.15.



	ONLINE SUPERMARKET				MESSAGING PLATFORM			
	Integration		System		Integration		System	
	WO	WE	WO	WE	WO	WE	WO	WE
Time per fault	2,91	3,87	3,50	4,06	8,32	5,96	4,12	4,90
Total time (#faults)	12	12	11	12	25	24	8	5

Figure 7.15: Effort (in minutes) spent by ATOS WE and WO testers in CCS for fault localization and correction in OS and MP demonstrators. Red bars refer to cumulative effort; blue bar to effort per fault

Considering the density measures (blue bars in Figure 7.15), for the OS application no significant different is observed in practice. For the MP application, there is a very small decrease of effort.

Also for the CCS study, **security** testing was executed only by the WE testers (see Figure 7.16). The estimates for the WO branch are the same as for the QE, while the times for WE branch are actual ones. Security testing was carried out for longer time than the QE study, but it found here a higher number of vulnerabilities.

Security	ONLINE SUPERMARKET				MESSAGING PLATFORM			
	Integration		System		Integration		System	
	WO	WE	WO	WE	WO	WE	WO	WE
N. of vulnerabilities		0		1		6		4
Time	*16 h (est)*	14 m	*14 h (est)*	3 m	*16 h (est)*	62 m	*14 h (est)*	49 m

Figure 7.16: Security data from CCS by ATOS WL testers for the WE branch; data for WO branch (in italics) are estimates as no security testing was performed

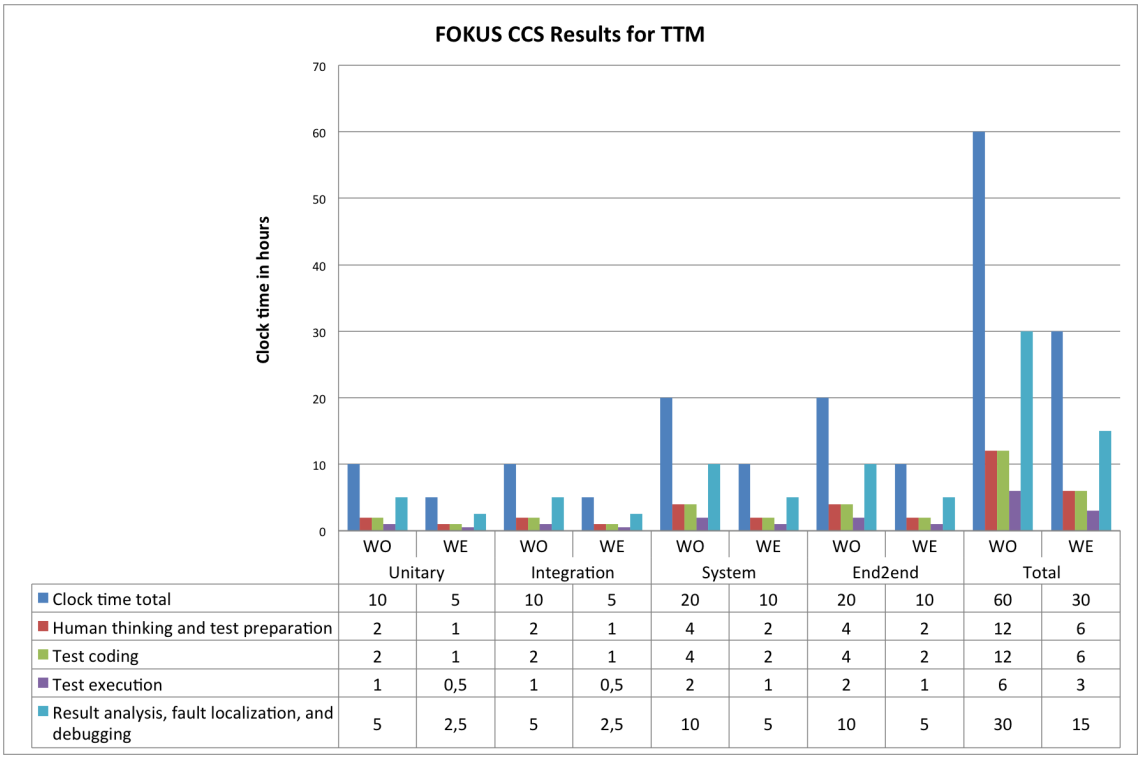
7.3.2 CCS results from FOKUS

To conduct a Comparative Case Study in FOKUS, we collected on the one side measures from clients using the standard platform for testing the 5G networking system. On the other side, we also released the same platform along with ElasTest to

another groups of clients. We collected data relative to testing time, reusability and productivity for test execution,

It is important to note that clients were reluctant to collect and provide data, as this was perceived as an extra-effort within a busy schedule. Hence eventually we could only collect limited data, which however provided encouraging results.

Concerning the time spent in testing, in Figure 7.17 below we report the time measure in hours along the various stages of testing activity. Note that time measures are rounded to the hour.

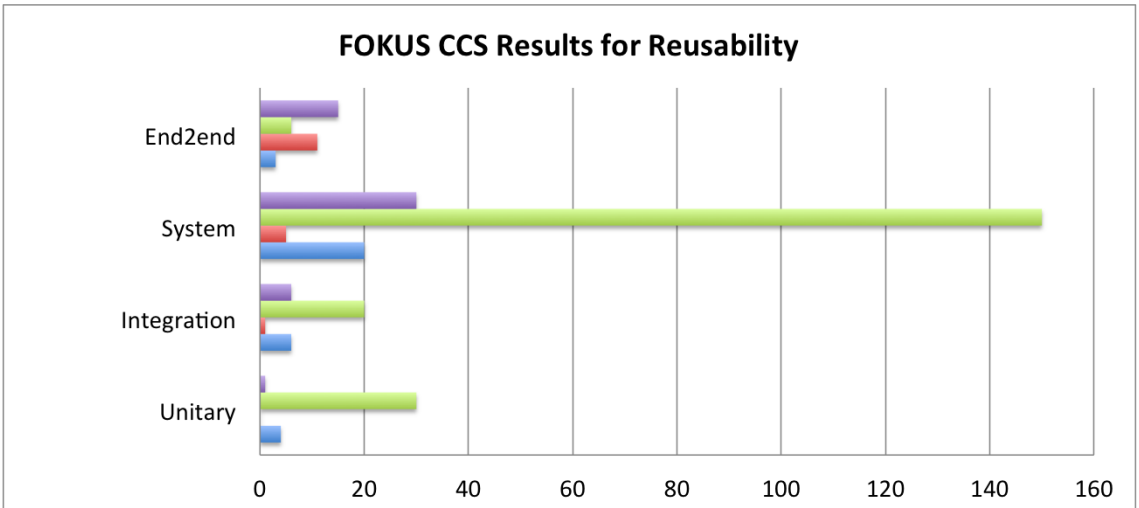


	Unitary		Integration		System		End2end		Total	
	WO	WE	WO	WE	WO	WE	WO	WE	WO	WE
Clock time total	10	5	10	5	20	10	20	10	60	30
Human thinking and test preparation	2	1	2	1	4	2	4	2	12	6
Test coding	2	1	2	1	4	2	4	2	12	6
Test execution	1	0,5	1	0,5	2	1	2	1	6	3
Result analysis, fault localization, and debugging	5	2,5	5	2,5	10	5	10	5	30	15

Figure 7.17: Clock time spent by FOKUS testers in each testing phase and totally in the CCS

We can notice that the time spent by clients using ElasTest amounts consistently to **one half of the time spent in traditional approach**.

In Figure 7.18 we also report the data collected for measuring reusability.



	Unitary	Integration	System	End2end
WE #reused tests	1	6	30	15
WO #reused tests	30	20	150	6
WE non-functional	0	1	5	11
WO non-functional	4	6	20	3

Figure 7.18: Number of reused tests by FOKUS testers in CCS

As foreseen in the project DoA, the real context where reusability is highly improving thanks to ElasTest relates to non-functional tests in the E2E testing stage: 11 test cases were reused by ElasTest-using clients against 3 from WO clients. The **overall increase would amount to 267%**, which is a nice result but still far from the targeted 500% but already in good progress.

With regard to productivity, in the CCS we observed a similar result than during the QE, in particular that the WE clients produced more E2E tests, i.e., 15, than the WO clients, i.e., 10: on average we achieve **an increase in productivity of 50%**.

7.3.3 CCS results from Naeva Tec

The completed ElasTest platform has been released and is now adopted in Proof-Of-Concept production within Naeva Tec for the E2E testing stage. We collected data from one tester concerning time spent, productivity and maintenance. However, it must be noted that the company structure has undergone deep transformation since the time the first CCS data were collected, therefore it is not very meaningful to perform now a comparison of the reported values with those collected in the baseline.

We can report qualitative feedback by the involved testers, that using ElasTest has radically changed their way of testing this type of applications. Before ElasTest the approach followed was manual and ad hoc, i.e., not repeatable nor measurable. And moreover it required the collaboration of a whole team to conduct each test.
Thanks to ElasTest adoption, they have made the process more systematic and objective, so that any potential issue is also more easily detectable and manageable. Also, only one tester could obtain the data that previously required a whole team (all the people in the Naeva Tec company).

7.3.4 CCS results from TUB

In addition to the controlled study over the IIOT system, in TUB data were also collected from the EMBER system. As described in Section 6.4, this is a software system for smart city, which is being developed at TUB using an Agile process.

The product considered in the experiment was a web application involving more than 10 single functionalities tested used a Scrum approach. In particular the application is a middleware that is an implementation of the OneM2M standard for machine-to-machine communication. This middleware can be exploited to realize Industrial IoT applications. During the second round of Quasi experiments, TUB developed custom IIoT applications using OpenMTC, such that, the developed applications were typical in an industrial setting.

The application has been selected because the most representative of the TUB current situation, before the introduction of the features of ElasTest project.

The personnel involved in the testing phase were 12 developers and 6 testers. The former mainly involved in the analysis and development phase, the latter in System Test. In particular the developers and testers constitute personnel having background in computer science, computer programming and industrial machine to machine communication. Most of the developers have worked full time, while some of them are students working part time.

The test cases derived (in the fifty order) have been produced using an agile process executed and analyzed by means of Jenkins, REST, Shell scripts tools applied in combination during the different product development phases.

In a first observation period the system was tested following the approaches in use at the partner. Testing was supported by several state-of-practice tools, including JMeter, TTWorkbench Testsuite, Locust, and ad-hoc scripts (Python & Shell). The partner devoted a total of 8 FTE (Full Time Equivalent) developers and 4 FTE testers to develop and test a first release of the system, which lasted in total 18 months, and covered in total 13 test cases.

After the complete ElasTest was released, the partner continued to test the same EMBER system, but now adopting the platform in production. We notice that in this case the partner charged 2 FTE developers, and 2 FTE testers. At time of writing only the three phases of Analysis and development, Unit and component testing and End-to-end testing have been conducted using ElasTest.

In the figure below, we depict the time-related observed data. We can notice that the time taken for testing is clearly reduced: for the End-to-end testing stage over an average of 4,5 months per test needed before adoption of ElasTest, afterwards they employed 1 month per test, **with a reduction in TTM of 78%.** Note that although the test performed is only one, there are a number of pre-requisites to run the test. For example, certain interactions are only possible using specific protocols; therefore it is necessary to understand the protocol architecture first. In other cases, it might be that the complexity to manage several variables at once could be much higher with manual

testing. In this case ElasTest would take care of most of the orchestration and test configurations, thus relieving the tester from worrying about lower level details.

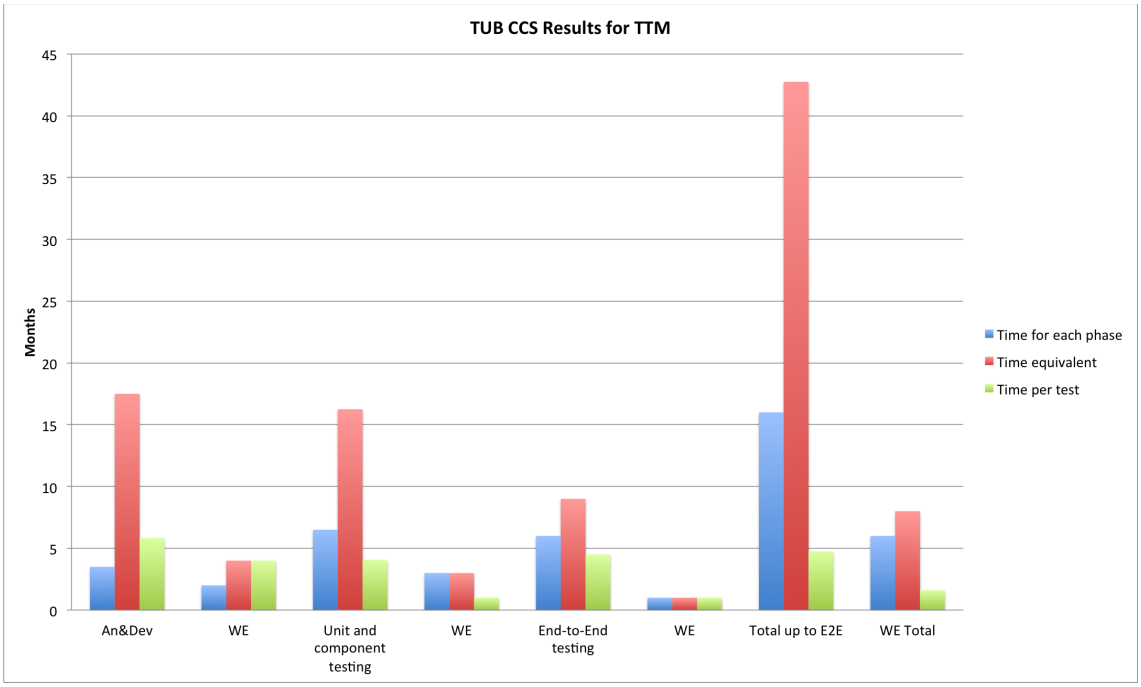


Figure 7.19: TTM data from TUB CCS

We also measured productivity (pertaining to KPI 1.3), tracking over the period of observation the average number of LOCs tested in the Time Unit. Before ElasTest the performance was around 556 LOCs, whereas afterwards it increased to 667, with an **overall gain in productivity of 20%.**

Maintenance of EMBER was previously performed using TTWorkbench Suite and ad-hoc Shell scripting. The testers measured on average that 12 person-hours were required for each KLOC. After ElasTest adoption, the maintenance effort for EMBER notably decreased to an average of 2,5 person-hours per KLOC. The **overall reduction in maintenance effort is around 79%**.

7.4 Empirical surveys results

In this section we report the results collected for the subjective metrics, i.e. VM 1.4 and VM 2.4. As explained in Section 4.2, the former metric targeted the impressions from testers when using ElasTest, whereas the latter metric was originally conceived for end-users when using the systems (tested with ElasTest), but in the impossibility of collecting end-users' feedback within the project life time, we resolved to direct VM 2.4 related questions to testers as well.

The questionnaires have been developed and assessed during the first period within a pilot study. In the second reporting period we used the assessed final questionnaire to collect feedback from all involved testers, both in the QE studies and in the CCS studies.

Overall, we received a total of 34 completed questionnaires, of which 22 by testers from the WO branch, and 12 from the WE branch. The two groups are numerically unbalanced, and this is comprehensible considering that for collecting feedbacks in the WE branch we need testers that actually used ElasTest, whereas for the WO branch we could more easily interview the testing teams of the partners about their feelings of the traditional testing context. We collected statistical information about tester's experience and gender, and the overall data are shown below in Table 11.

	Respondent Stats	
	WO	WE
Testing experience		
From 0 to 3 years	64%	67%
From 3 to 6 years	23%	8%
More than 6 years	14%	25%
No response or other	0%	0%
Gender		
Male	59%	42%
Female	41%	50%
No response or other	0%	8%

Table 11: Testers' demographics

The results are reported in percentages over the two sizes, which as said are different. Overall we can see that the population of testers is well balanced both in terms of expertise and of gender distribution.

In the following subsections, we report in order the results for the experienced characteristics of simplicity, satisfaction, efficacy, confidence and usefulness (VM 1.4), as well as of efficiency and lack of risks (VM 2.4).

7.4.1 Simplicity

The first group of questions targets the subjective feeling about how simple is the testing environment. To evaluate this aspect, during the first period we had identified a set of questions, so that beyond just asking plainly whether the tester thought the test environment was simple, we also asked his/her impression of the simplicity in common test tasks. In Table 12 we show the responses: in particular the first column reports for completeness an excerpt of the question (with the question label in parentheses)[2]; the second and third columns give the average of the collected values in the WO and WE cases, respectively; finally the fourth column shows the delta of WE-WO. We recall that all responses are measured on a 5 Likert scale, with 1) corresponding to very difficult, and 5) to very simple.

As the expected KPI is an average increase of at least 1, we show in green color the metrics where such KPI is reached.

[2] The entire questionnaire is included in Appendix.

Simplicity-related group of questions	WO	WE	Delta
simplicity (q1)	3,23	4,17	0,94
code just enough test code (q2)	3,67	4,80	1,13
feedback for interpreting test outcome (q3)	3,42	4,36	0,94
documentation you need to write (q4)	3,28	4,00	0,72
documentation you need to read (q44)	2,40	3,63	1,23
help in measuring test progress (q5)	2,95	4,10	1,15
focus testing activity on relevant parts (q6)	3,05	4,11	1,07
weighted average	3,10	4,11	1,02

Table 12: Average scores of responses related to simplicity from WO and WE testers

We can see that testers not using ElasTest on average scored their test environment simplicity (first row) as 3,23, whereas ElasTest users scored it 4,17, with an increase of 0,94. By looking at the different specific questions, we see that the highest increase in simplification is 1,23, which is achieved with documentation to read (likely ElasTest usage is self-explanatory). Also, the highest average score of 4,80 is given to the question whether the test environment helps to write just enough test code. In the last row we calculate a weighted average from all answers (we use weighted calculation because not all responders answered all questions), and **we get as an overall evaluation: 3,10 from the WO testers, and 4,11 from the WE testers, with an increase of perceived simplicity overall of 1,02, which nicely hits the targeted KPI**.

7.4.2 Satisfaction

The second group of questions concerns testers' satisfaction. Also in this case we designed a group of questions, aiming at understanding different aspects influencing this aspect. The average responses are depicted in Table 13, where we provide a sketch of questions in the first column (see the full questionnaires in Appendix).

Satisfaction-related group of questions	WO	WE	Delta
satisfaction (q7)	3,33	4,18	0,85
satisfied with performed testing activity (q8)	3,67	4,80	1,13
satisfied with performance of test results (q10)	3,42	4,36	0,94
satisfied with the productivity of your job (q11)	3,28	4,00	0,72
satisfied with the adopted test process (q12)	2,40	3,63	1,23
weighted average	3,14	4,11	0,97

Table 13: Average scores of responses related to satisfaction from WO and WE testers

The last column provides the delta of WE minus WO score: as we can see testers feel more satisfied in each feature: activity, test results, job productivity and process, although the difference varies between 0,72 delta for job productivity up to 1.23 for

adopted process. Making **a weighted average across all features, we observe overall an increase of 0,97, which is below but very close to the target of 1 factor**.

7.4.3 Efficacy

The third group of questions concerns efficacy. Following same structure as above, the results are shown in Table 14.

Efficacy-related group of questions	WO	WE	Delta
efficacy (q13)	3,05	4,17	1,12
environment helps stay within scheduled time? (q14)	2,90	4,08	1,18
automated facilities (q15)	3,29	5,00	1,71
managing unexpected problems (q16)	2,60	3,83	1,23
identifying more relevant quality aspects (q17)	2,90	3,45	0,55
effectiveness (capability to produce desired results) (q51)	3,18	4,17	0,98
environment helps monitor the testing activity (q18)	3,52	4,00	0,48
environment helps measure quality aspects (q19)	3,14	3,82	0,68
covering all prefixed testing features (q20)	3,10	4,25	1,15
collecing test logs (q21)	3,48	4,58	1,11
weighted average	3,12	4,07	0,95

Table 14: Average scores of responses related to efficacy from WO and WE testers

In this case we observe that when directly asked about the efficacy of the testing environment (see first row, question q13), WE testers have been quite positive, scoring an average value of 4,17 and an increase factor or 1,12. However we collected a lower increase, although still positive, concerning the help in identifying more relevant quality aspects and monitoring support. As **an average across all questions, we obtain an increase factor of 0,95, which is again very close to the targeted KPI of 1**.

7.4.4 Confidence and Lack of Risks

For convenience in the questionnaire we grouped together a question concerning the confidence in the testing environment and a set of four questions related to risks felt. The resulting table with questions and average response scores is depicted below (Table 15).

Confidence & Risk-related group of questions	WO	WE	Delta
Confidence (q559)	2,23	4,24	2,01
Lack of risks of:			
no unauthorized/malicious resource use (q555)	2,32	3,75	1,43
no unauthorized modifications (q557)	2,07	4,09	2,02
no distruction (q558)	2,54	4,33	1,79
no unauthorized/malicious disclosure(q560)	2,43	4,00	1,57
weighted average for lack of risks	2,33	4,02	1,69

Table 15: Average scores of responses related to confidence and lack of risks from WO and WE testers

As can be seen, in this case the response are extremely positive: WE testers feel much more confident in the testing environment, with **an average increase factor to the direct question of 2,01** over the WO testers' feeling. WE testers also clearly perceive the SUT as **less subject to potential risks with an average increase across all four questions of 1,69.** Such results are in line with the very positive objective metrics we could register for the security testing studies.

7.4.5 Usefulness

Finally, VM 1.4 also includes a group of questions concerning perceived usefulness, as shown in Table 16.

Usefulness-related group of questions	WO	WE	Delta
usefulness (q22)	3,55	4,58	1,04
use testing environment without expert support (q26)	3,36	4,00	0,64
testing facilities well integrated (q27)	2,82	4,36	1,55
feel environment can drive your testing activity (q28)	3,58	4,27	0,69
environment is intuitive (q29)	3,00	4,27	1,27
weighted average	3,18	4,22	1,04

Table 16: Average scores of responses related to usefulness from WO and WE testers

Also in this case the results testify that using ElasTest provide testers with an increased feeling of usefulness, although in some of the questions this increase is below 1: precisely, the increase regarding capability to use it without expert support is 0,64, and environment potential to drive the testing activity is 0,69. On the other hand, the WE testers feel that the testing facilities are well integrated with an average score of 4,36, against the 2,82 score given by WO testers. Overall **the average across all questions is over the target KPI of 1.**

7.4.6 Efficiency

Finally concerning efficiency, which belongs to VM 2.4, we characterized it by the capability of performing the testing in time, and posed this question to testers in the

same group of efficacy (see Section 7.4.3). We show the results for just this direct question, as follows:

	WO	WE	Delta
Efficiency (q50)	3,05	4,08	1,03

Table 17: Average scores and delta for efficiency according to WO and WE testers

We can see that **WE testers feel more positive about testing efficiency than WO testers, with an increase factor of 1,03**.

8 Validation conclusions

In the previous chapter we have reported in detail the data observed in the QE (Section 7.2), CCS (Section 7.3) and ES (Section 7.4) studies carried out in the period M19-M36. The goal of all these validation activities was to assess the validation metrics (both objective and subjective) against the expected KPIs that had been set in the DoA before the project start. As a summary, the mapping of the empirical studies over the established VMs and the different application scenarios provided by the four partners: ATOS, TUB, FOKUS and Naeva Tec is provided in Table 1.

In this chapter we now collect and analyse the data across the different demonstrators, and comment about the degree of accomplishment of the targeted KPIs.

8.1 Testing time

The first validation metric was the overall time to market, and the project aimed at reducing overall time to market of SiL in an average factor of 20%.

This metric has been validated by all four partners, within both QE and CCS studies. We cannot directly compare the results collected across the different domains, as the studies employed different units of measure (in some cases minutes, in other hours, etc) and involved very different contexts and settings. What we report below is the **relative delta** calculated in each study, so that we can discuss and compare the results in terms of relative increase or decrease of testing times. We recall in fact that, as already discussed in Section 4.2.1, the time measurement within the WP7 studies could only cover testing stage and not the whole TTM since product design (which would go well before and behind the ElasTest lifecycle).

In Table 18 we show the relative delta between WE and WO data. The table is divided in two parts: in the top part we show the data relative to the quasi-experiments, and in the bottom part those relative to the comparative case studies. The KPI is -20%, precisely we expect that the time measured by using ElasTest is decreased on average by a factor of 20%. For ease of inspection, we show the data that reach this KPI on a green background. A cell marked with “na” means that the data were not available in that study. In particular, for the CCS by Naeva Tec we reported in Section 7.3.3 the measures for the WE branch and discussed them qualitatively, but we cannot here report the delta because we lack comparable baseline data.

Delta Time (WE-WO) in QE					
KPI: -20%	ATOS OS	ATOS MP	FOKUS	Naeva Tec	TUB
Delta total testing time	2%	-8%	-7%	-87%	-58%
Delta-Time thinking and preparation	na	na	-29%	-98%	-60%
Delta-Time in test coding	na	na	25%	-43%	-20%
Delta-Time in test execution	11%	-35%	-44%	-74%	-80%
Delta-Time in analysis, fault localization, and debugging	7%	-12%	7%	-73%	-90%
Delta Time (WE-WO) in CCS					
KPI: -20%	ATOS OS	ATOS MP	FOKUS	Naeva Tec	TUB
Delta total testing time	2%	-10%	-50%	na	-81%
Delta-Time thinking and preparation	na	na	-50%	na	-77%
Delta-Time in test coding	na	na	-50%	na	na
Delta-Time in test execution	9%	-40%	-50%	na	-84%
Delta-Time in analysis, fault localization, and debugging	8%	-13%	-50%	na	na

Table 18: Summary results for testing time

Overall, we can conclude that in most of the studies this KPI has been reached and largely overcome: for instance in the quasi experiment the Naeva Tec partner could reduce the overall testing time of a 87% factor, and TUB of a 58% factor.

As already discussed, for the ATOS partner that adopts a manual testing process, ElasTest was not immediately integrated in natural way, and for the OS application forcing the usage of ElasTest according to the experiment settings even resulted in an overall increase of testing time. On the other hand, for the MS application, in which they had more capability to manage to test server, they could nevertheless observe some limited reduction of testing time.

8.2 Reusability

Our second KPI was related to the reusability of code, tools and architectures devoted to non-functional software testing that was set to an average increase factor of, at least, 500%.

As evident, the project has set very ambitious expectations over such KPI: we are confident that routine usage of ElasTest in production will in fact strongly push reuse of artifacts and ease automation of clerical and repetitive tasks. However, as anticipated in Section 4.2.2, it was not possible to assess such expected improvement along the lifecycle of the project, because of course we could only perform a few iterations of studies, in which the reuse across projects was not possible, and only partial reuse across different releases of one same application was observable.

The data observed from such limited empirical observations are reported in Table 19.

Reusability (WE-WO)					
KPI: 500%	ATOS OS	ATOS MP	FOKUS	Naeva Tec	TUB
Overall QE	na	88%	50%	na	na
Overall CCS	na	80%	267%	na	na

Table 19: Summary results for reusability

As shown, the results are quite far from the planned KPI, however in relation to the limited window of observation they are already encouraging.

8.3 Productivity

In the DoA we expected to increase tester's productivity in integration and system tests in a factor of, at least, 100%.

As a matter of fact, it is quite difficult to draw some general conclusions for this metric across the four partners, because they reported quite different types of data. With the warning that in some columns the data are relative to the number of tests that could be performed within a same time unit (ATOS and FOKUS), whereas for TUB the measure of productivity in CCS was related to the amount of code that could be tested, we show only for summary reference the data in Table 20 below.

Productivity (WE-WO)					
KPI: 100%	**ATOS OS**	**ATOS MP**	**FOKUS**	**Naeva Tec**	**TUB**
Overall QE	-10%	64%	100%	na	na
Overall CCS	-12%	75%	50%	na	20%

Table 20: Summary results for productivity

Even though the target KPI of 100% could not be reached but in one case, we can clearly see how productivity tends to improve. The results are very positive for FOKUS and also for ATOS in the case of the MP application.

8.4 Maintenance

Thanks to the support in logging and monitoring the data from testing session, the project expects that the corrective maintenance effort is decreased in a factor of, at least, 50%.

Maintenance (WE-WO)					
KPI: -50%	**ATOS OS**	**ATOS MP**	**FOKUS**	**Naeva Tec**	**TUB**
Overall QE	11%	-17%	na	-82%	na
Overall CCS	33%	-92%	na	na	-79%

Table 21: Summary results for maintenance

As shown, the results are not uniform: we have achieved and even largely overcome such KPI in the Naeva Tec QE study, and in the CCS studies for TUB and ATOS MP, but in the ATOS OS platform the maintenance effort has even be higher than for the WO case. However, knowing that the ATOS OS application was not a good target for the platform, we interpret these results as good promise that in proper contexts, ElasTest can considerably help reducing maintenance effort.

8.5 Scalability, robustness, security and QoE

ElasTest validation metric 2.3 groups together measures of scalability, robustness, security and Quality of Experience, setting for each of them an expected average increase of 20%.

Unfortunately, during the planning for the validation studies, we soon discovered that the partners involved in WP7 validation did not perform in their standard testing processes all of such types of testing. As illustrated in Figure 4.1, we could eventually manage to cover these metrics by at least one partner. In particular, scalability and robustness testing was carried out by TUB, security testing was conducted by ATOS (in both OS and MP) and Naeva Tec, and finally QoE testing was evaluated within an ad-hoc tailored test session carried out by Naeva Tec on OpenVidu.

We show the results cumulatively in the following table. It is worth noticing that the metrics we adopted for assessing some features differ from those initially planned in the DoA, because we could not actually derive any measures from field usage. So they were adapted to metrics suitable for the testing stage.

Deltas for VM 2.3 (WE-WO)				
KPI: 20%	**ATOS OS**	**ATOS MP**	**Naeva Tec**	**TUB**
Scalability	na	na	na	˜66% (see Fig 7.12)
Robustness	na	na	na	˜75% (see Sec 7.2.4)
Security	˜99% (see Fig 7.7 and 7.17)	˜97% (see Fig 7.7 and 7.17)	43% (see Fig 7.12)	na
QoE	na	na	See qualitative results in Sec. 7.2.3	na

Table 22: Summary results for VM 2.3 aspects

The results reported look promising as for the metrics assessed the KPIs seem reachable. However no generalization is currently possible beyond the specific study, due to both the limited observation and the ad hoc nature of the studies.

8.6 Subjective feelings

We report together the results for VM 1.4 and VM 2.4. that have been more extensively presented in Section 7.4. We recall that the surveys here summarized were filled by both the testers performing the QE and those performing the CCS.

	Simplicity	Satisfaction	Efficacy	Confidence	Usefulness	Efficiency	Lack of risks
KPI: >= 1	1,02	0,97	0,95	2,01	1,04	1,03	1,69

Table 23: Summary results for subjective metrics (VM 1.4 and 2.4)

We can see that the expected KPI of at least an increase factor of 1 over a 5-points Likert scale has been fully achieved for Simplicity, Confidence, Usefulness, Efficiency, and Lack-of-risk, and even for Satisfaction and Efficacy in which it was not reached, the results are anyhow very close to 1 (0,97 and 0,95 respectively).

In summary we can draw general conclusions on three aspects:

- **Measuring capability:** some of the validation metrics that had been set at the project start were very ambitious and very difficult, if not impossible, to get within the lifecycle of the project, also in consideration of the fact that the full platform functionality was only achieved at M32. Through a constant and incremental approach to metric revision, and a careful planning of empirical studies we resorted to assess anyhow all measures, in some cases adopting some surrogate ones.
- **Improvements achieved:** concerning the objective KPIs that had been set in the project DoA (after proper revision of the VMs), we could successfully achieve many of them. In general we could observe quite positive evaluation over almost all metrics, although some of them were only assessed by a subset of the four involved partners.
- **Subjective feedbacks:** overall, we are very satisfied with the subjective feedbacks, as we reached the expected improvements in 5 of 7 aspects, and the remaining 2 were very close to the target.

All reported results cannot be generalized beyond the specific studies, due to limited number of observations, as well as the ad hoc measurement adaption within each context that makes it difficult comparing the outcomes across domains. Indeed, in this chapter we have attempted a general overview and comparison across the studies whose detailed results can be analyzed per partner. We acknowledge though that our conclusions are limited to the studies, and do not have statistical value.

In preparing this document, we have put all care to get measures in unbiased and rigorous way, and we report here faithfully the registered values. Nevertheless, we

cannot exclude possible errors or biases introduced within each experimental venue, as the data refer to actual testing experiences within each partner. As with any real-world study, many factors could escape control and influence the results. Due to limited effort and time available, we could not repeat each study several times to mitigate potential random variations, as is usually done in laboratory studies.

In next future, the established measurement settings within each partner could be reused to collect further data and increase our knowledge of ElasTest impact on their testing processes.

9 References

[1] ElasTest project Description of Action (DoA) – part B. Amendment 1. Reference Ares (2017)343382. 23 January 2017.

[2] Deliverable D7.1 ElasTest validation methodology and its results v1 (06/30/18), online at:

https://elastest.eu/resources/deliverables/D7.1_ElasTest_validation_methodology_and_its_results_v1_FINAL.pdf

[3] IEEE Standard Glossary of Software Engineering Terminology," in IEEE Std 610.12-1990 , pp.1-84, 31 Dec. 1990, doi: 10.1109/IEEESTD.1990.101064

[4] Yin, Robert. Case Study Research: design and methods. 5 ed. Thousand Oaks,CA: Sage, 2014.

[5] Boni García, Micael Gallego, Francisco Gortázar, Antonia Bertolino. Understanding and estimating quality of experience in WebRTC applications. Computing 101(11): 1585-1607 (2019)

[6] Boni García, Luis López-Fernández, Francisco Gortázar, Micael Gallego. Practical Evaluation of VMAF Perceptual Video Quality for WebRTC Applications. Electronics (Computer Science & Engineering Section). Volume 8, Issue 8. 2019.

Appendix: Tester Survey

Agreement

Project information

I do consent to the treatment of personal data in line with the above information.

Archiving data

Please enter your name *

Gender *

Please choose **only one** of the following:
Male Female Other No answer

Please specify your position *

Please write your answer here:

Testing experience *

Please choose **only one** of the following:
From 0 to 3 years From 3 to 6 years More than 6 years

Specify the name of your company/university *

Please write your answer here:

Number of employees/researcher in the whole company/university? *

Please write your answer here:

Which kind of product/s are you testing? *

Please write your answer here:

Simplicity

How simple is it to use your testing environment?
Please answer the question based on your experience.
Scale: 1 very difficult to use - 3 neutral - 5 very simple to use *
Please choose **only one** of the following: 1 2 3 4 5

To what extent are you able to code just enough test code to get your test executed?

Please answer the question based on your experience.
If not applicable, leave the scale empty.
Scale: 1 I need a lot of extra coding - 3 neutral - 5 I code just what is necessary
Please choose **only one** of the following: 1 2 3 4 5

How would you rate the feedback you get for interpreting the test outcome?

Please answer the question based on your experience.
If not applicable, leave the scale empty.
Scale: 1 Too much/Too few - 3 neutral - 5 I get just enough feedback
Please choose **only one** of the following: 1 2 3 4 5

How much documentation do you need to write in your test activity?

Please answer the question based on your experience.
If not applicable, leave the scale empty.
Scale: 1 A lot of documentation - 3 neutral - 5 I write just enough documentation
Please choose **only one** of the following: 1 2 3 4 5

How much documentation do you need to read in your test activity?

Please answer the question based on your experience.
If not applicable, leave the scale empty.
Scale: 1 A lot of documentation - 3 neutral - 5 just enough documentation
Please choose **only one** of the following: 1 2 3 4 5

To what degree does the testing environment help you measure your progress in testing?

Please answer the question based on your experience.
If not applicable, leave the scale empty.
Scale: 1 I cannot measure any progress - 3 neutral - 5 It is very supportive
Please choose **only one** of the following: 1 2 3 4 5

To what degree are you able to focus the testing activity on the parts that are relevant to you?

Please answer the question based on your experience.
If not applicable, leave the scale empty.
Scale: 1 It is very difficult - 3 neutral - 5 It is very easy
Please choose **only one** of the following: 1 2 3 4 5

Satisfaction

To what degree are you satisfied with your testing environment?

Please answer the question based on your experience.
Scale: 1 Not satisfied at all - 3 neutral - 5 very satisfied *
Please choose **only one** of the following: 1 2 3 4 5

To what degree are you satisfied with the testing activity you perform?

Please answer the question based on your experience.

If not applicable, leave the scale empty.
Scale: 1 not satisfied at all - 3 neutral - 5 very satisfied
Please choose **only one** of the following: 1 2 3 4 5

To what degree are you satisfied with the performance of the results of the testing activity?

Please answer the question based on your experience.
If not applicable, leave the scale empty.
Scale: 1 not satisfied at all - 3 neutral - 5 very satisfied
Please choose **only one** of the following: 1 2 3 4 5

To what degree are you satisfied with the productivity of your job?

Please answer the question based on your experience.
If not applicable, leave the scale empty.
Scale: 1 not satisfied at all - 3 neutral - 5 very satisfied
Please choose **only one** of the following: 1 2 3 4 5

To what degree are you satisfied with the adopted test process?

Please answer the question based on your experience.
If not applicable, leave the scale empty.
Scale: 1 not satisfied at all - 3 neutral - 5 very satisfied
Please choose **only one** of the following: 1 2 3 4 5

Efficacy

How efficacious is your testing environment?

Efficacy: the ability to achieve the results.
Please answer the question based on your experience.
Scale: 1 not efficacious at all - 3 neutral - 5 very efficacious *
Please choose **only one** of the following: 1 2 3 4 5

How much does the testing environment help you to complete the testing activity within the scheduled time?

Please estimate the question concerning your experience.
If not applicable, leave the scale empty
Scale: 1 not at all - 3 neutral - 5 very much
Please choose **only one** of the following: 1 2 3 4 5

How much does the automated facilities help you?

Please estimate the question concerning your experience.
If not applicable, leave the scale empty.
Scale: 1 not at all - 3 neutral - 5 very much
Please choose **only one** of the following: 1 2 3 4 5

How much does the testing environment help you to manage unexpected problems/failures?

Please estimate the question concerning your experience.
If not applicable, leave the scale empty.
Scale: 1 not at all - 3 neutral - 5 very much
Please choose **only one** of the following: 1 2 3 4 5

How much does the testing environment help you to identify the quality aspects more relevant to you?

(e.g., reliability, availability, usability, performability ...)?

Please estimate the question concerning your experience.
If not applicable, leave the scale empty.
Scale: 1 not at all - 3 neutral - 5 very much
Please choose **only one** of the following: 1 2 3 4 5

How effective is your testing environment?

Effectiveness: the capability of producing a desired result.
Please estimate the question concerning your experience.
Scale: 1 not effective at all - 3 neutral - 5 very effective *
Please choose **only one** of the following: 1 2 3 4 5

How much does the testing environment help you to monitor the testing activity?

Please estimate the question concerning your experience.
If not applicable, leave the scale empty.
Scale: 1 not at all - 3 neutral - 5 very much
Please choose **only one** of the following: 1 2 3 4 5

How much does the testing environment help you to measure the quality aspects more relevant to you?

(e.g., reliability, availability, usability, performability, ...)?

Please estimate the question concerning your experience.
If not applicable, leave the scale empty.
Scale: 1 not at all - 3 neutral - 5 very much
Please choose **only one** of the following: 1 2 3 4 5

How much does the testing environment help you to cover all the prefixed testing features?

Please estimate the question concerning your experience.
If not applicable, leave the scale empty.
Scale: 1 not at all - 3 neutral - 5 very much
Please choose **only one** of the following: 1 2 3 4 5

How much does the testing environment help you to collect logs of the testing activity?

Please estimate the question concerning your experience.
If not applicable, leave the scale empty.
Scale: 1 not at all - 3 neutral - 5 very much
Please choose **only one** of the following: 1 2 3 4 5

Confidence

How much are you confident about the security of your testing environment?

Please estimate the question concerning your experience.
Scale: 1 not at all - 3 neutral - 5 very much
Please choose **only one** of the following: 1 2 3 4 5

How much are you confident the security features can avoid unauthorized/malicious resource use?

Please estimate the question concerning your experience.
Scale: 1 not at all - 3 neutral - 5 very much *
Please choose **only one** of the following: 1 2 3 4 5

How much are you confident the security features can avoid unauthorized/malicious modifications?

Please estimate the question concerning your experience.
Scale: 1 not at all - 3 neutral - 5 very much
Please choose **only one** of the following: 1 2 3 4 5

How much are you confident the security features can avoid unauthorized/malicious destruction?

Please estimate the question concerning your experience.
Scale: 1 not at all - 3 neutral - 5 very much
Please choose **only one** of the following: 1 2 3 4 5

How much are you confident that the security features can avoid unauthorized/malicious disclosure?

Please estimate the question concerning your experience.
Scale: 1 not at all - 3 neutral - 5 very much
Please choose **only one** of the following: 1 2 3 4 5

Usefulness

How useful is your testing environment?

Please estimate the question concerning your experience.

Scale: 1 not at all - 3 neutral - 5 very much *
Please choose **only one** of the following: 1 2 3 4 5

To what degree are you able to use the testing environment without the support of a technical expert?

Please answer the question based on your experience.
If not applicable, leave the scale empty.
Scale: 1 very difficult - 3 neutral - 5 very easy
Please choose **only one** of the following: 1 2 3 4 5

To what degree do you find the various testing facilities well integrated in your testing environment?

Please answer the question based on your experience.
If not applicable, leave the scale empty.
Scale: 1 not at all - 3 neutral - 5 very much
Please choose **only one** of the following: 1 2 3 4 5

To what degree do you think your testing environment can drive your testing activity?

Please answer the question based on your experience.
If not applicable, leave the scale empty.
Scale: 1 not at all - 3 neutral - 5 very much
Please choose **only one** of the following: 1 2 3 4 5

To what degree do you think your testing environment is intuitive?

Please answer the question based on your experience.
If not applicable, leave the scale empty.
Scale: 1 not at all - 3 neutral - 5 very much
Please choose **only one** of the following: 1 2 3 4 5

Efficiency

How efficient is your testing environment?

Efficiency: The ability to perform testing activity within the scheduled time.
Please estimate the question concerning your experience.
Scale: 1 not efficient at all - 3 neutral - 5 very efficient *
Please choose **only one** of the following: 1 2 3 4 5

Thank you for completing this survey.