D7.1	
Version	1.0
Author	CNR
Dissemination	PU
Date	30-06-2018
Status	FINAL

D7.1 ElasTest validation methodology and its results v1

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
Deliverable nature	Report
Lead editor	Antonia Bertolino, Eda Marchetti
Planned delivery date	30-06-2018
Actual delivery date	29-06-2018
Keywords	Open source software, cloud computing, software engineering, operating systems, computer languages, software design & development

Funded by the European Union

License

This is a public deliverable that is provided to the community under a **Creative Commons Attribution-ShareAlike 4.0 International** License:

http://creativecommons.org/licenses/by-sa/4.0/

You are free to:

Share — copy and redistribute the material in any medium or format.

Adapt — remix, transform, and build upon the material for any purpose, even commercially.

The licensor cannot revoke these freedoms as long as you follow the license terms.

Under the following terms:

Attribution — You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.

ShareAlike — If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.

No additional restrictions — You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

Notices:

You do not have to comply with the license for elements of the material in the public domain or where your use is permitted by an applicable exception or limitation.

No warranties are given. The license may not give you all of the permissions necessary for your intended use. For example, other rights such as publicity, privacy, or moral rights may limit how you use the material.

For a full description of the license legal terms, please refer to:

http://creativecommons.org/licenses/by-sa/4.0/legalcode

Contributors

Name	Affiliation
Antonia Bertolino	CNR
Antonello Calabrò	CNR
Anton Cervantes	Atos - WLI
Felicita Di Giandomenico	CNR
Ilie-Daniel Gheorghe-Pop	FOKUS
Varun Gowtham	TUB
Eda Marchetti	CNR
Guiomar Tuñón	NaevaTec

Version history

Version	Date	Author(s)	Description of changes
0.1	05/09/2017	CNR	Toc
0.2	1/12/2017	ALL	Description of applications used in the quasi experiments
0.3	10/1/2018	CNR	Description of procedure for historical data collection
0.4	20/1/2018	CNR	Description of procedure for QEs and report on historical data collected
0.5	1/2/2018	CNR	Description of the surveys
0.6	30/4/2018	All	Revising the demos description adopted inside the QE
0.7	9/5/2018	CNR	Report on data collected during the QE
0.8	29/5/2018	CNR	Survey data reporting
0.9	31/5/2018	FOKUS	Data input for section 6
0.10	18/6/2018	CNR	Finalized complete version
1.0	28/06/2018	CNR	Final version after internal reviews

Table of contents

List of figures

List of tables

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
EDS (Device Emulator Service)	EDS deploys emulated sensors or actuators on demand
EPC (Evolved Packet Core)	EPC is a framework for providing converged voice and data on a Long-Term Evolution (LTE) network
ES (Empirical Survey)	It is a collection of information from a sample of individuals through interviews performed via computer-assisted questionnaires
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
LOC (Line of Code)	This refers to lines of code considered during testing experimentation
LTE (Long-Term Evolution)	LTE is a standard for high-speed wireless communication for mobile devices and data terminals
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 particularly important for the characterization of multimedia systems and applications through custom metrics
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
TO (Test Orchestration)	The term orchestration typically refers to test orchestration understood as a technique for executing tests in coordination. This should not be confused with cloud orchestration, which is a completely different concept related to the orchestration of systems in a cloud environment
TORM (Test Orchestration and Recommendation Manager)	Is an ElasTest functional component that abstracts and exposes to testers the capabilities of the ElasTest orchestration and recommendation engines
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
TiL (Test in the Large)	A TiL refers to a set of tests that execute in coordination and that are suitable for validating complex functional and/or non-functional properties of a SiL on realistic operational conditions. We understand that a TiL can be created by orchestrating the execution of several T-Job
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
TU (Time Unit)	The unit of time established for each experimentation
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

ElasTest ultimate goal is to improve the efficiency and effectiveness of the testing process of large software systems. This is achieved thanks to the development and integration of an extensive set of technologies and tools as well as to several focused research advancements in strategic areas. These tasks are carried out within work packages WP2 … WP6 that in synergy contribute to build the ElasTest platform. WP7 has the objective of validating the developed platform by demonstrating its usage in different domains. Specifically, WP7 will:

- develop the four ElasTest demonstration scenarios (also referred to as vertical demonstrators);
- define a common validation methodology and instantiate it on each of the four demonstrators;
- during project lifecycle, in few iterations, perform a set of rigorous empirical studies;
- collect and analyze results, identify strengths and weaknesses, and provide feedbacks to the partners for improvement.

This deliverable describes the defined validation methodology and reports the results and feedbacks collected from the four demonstrators in the first reporting period (M18). Accordingly, the document has two main parts.

The first part, which includes Chapters 2, 3, and 4, provides a detailed description of the validation objectives, metrics, and schedule. We discuss how they have been developed, starting from the high-level "Outcome objectives" for the project established in the DoA. We also present the three types of empirical studies (namely Comparative Case Study, Quasi Experiment and Empirical Survey) through which the achievement of the objectives will be assessed. In Chapter 4, we propose the schedule for the strategy, aligned with project cycles (according to outcomes from next releases the schedule might be adapted). Modulo some fine-grained revision, this first part is thus the finalized ElasTest validation methodology that is used until the project final milestone MS6.

The second part, starting from Chapter 5, focuses on the first reporting period, and presents the results and feedbacks collected so far. As the validation studies performed in this period refer to a preliminary and obviously incomplete version of the platform (specifically to Release 3 at M12), in the context of the devised methodology they cannot be considered as an actual validation, but rather as a set of pilot studies. The objective of these pilots is twofold: on the one side, providing early feedbacks to the consortium about identified strengths and weaknesses that can be very useful to drive next development cycles, and on the other side validating and tuning the validation methodology itself. In particular, in Chapters from 5 to 8 we present the results from each of the four individual demonstrators. Concerning the Empirical Surveys, we report the results altogether in Chapter 9.

Finally, we draw conclusions and hint at empirical studies to be performed in the second period in Chapter 0.

2 ElasTest validation objectives

The ElasTest platform improves the testing of large complex distributed systems that are typically created by interconnecting and orchestrating smaller component systems. Such systems are referred to as SiLs (Systems in the Large). As the testing of SiLs is difficult and effort-prone, ElasTest targets the following two 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.

To be able to assess whether (and to what extent) such objectives are met, the ElasTest DoA document [4] associates to them a set of ambitious validation metrics that are 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

To assess the above listed metrics is the goal of WP7 that is structured around four vertical demonstrators. For each demonstrator we will evaluate the above metrics: we acknowledge that not all metrics may be relevant or feasible for each demonstrator, however considering the four of them together we aim at covering all of them.

The validation metrics cover quite disparate properties of a software process and product, and they will hence require different approaches for assessment. In particular, some of the metrics refer to improving global properties related to the efficiency and effectiveness of the software testing and maintenance process thanks to adoption of ElasTest. Hence, they require that we compare, on a set of comparable products, process quantitative measures taken after putting ElasTest in place for some consistent period against historical measures collected before adoption of ElasTest. Alternatively, we can conduct a rigorous experiment comparing in a controlled environment the measures achieved with and without ElasTest on a same system. This applies to metrics 1.1, 1.2 and 1.3 and 2.1, 2.2 and 2.3. As the experiment takes a double effort for developing a same system, and resources available are limited, we will restrict the experimentation to a subset of the metrics. We will collect all historical data available for all demonstrators.

Finally, we also have some subjective metrics, specifically 1.4 and 2.4: for these we will need to collect testers and users feedback, with and without ElasTest, through focused questionnaires.

3 Validation methodology

Considering the metrics that have to be assessed, the validation methodology adopted inside the ElasTest project includes three different types of empirical studies that complement each other in evaluating ElasTest results:

- **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 two testing processes) 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.
- **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.

In Table 1 below a mapping between the validation metrics presented in Chapter 2 and the different types of empirical studies is provided. This is a tentative mapping that of course needs to be verified and adjusted to the specific realities of the four demonstrators.

Comparative Case Study	Empirical Survey	Quasi Experiment
Validation metric 1.1: To reduce the overall time to market of SiL in an average factor of 20%	**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	**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 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	**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) of SiL in an average factor of 20%
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%		

Table 1. Mapping between validation metrics and the different types of empirical studies

All these studies include two stages: the (historical) baseline data collection and the ElasTest data collection.

The **baseline data collection** is needed to obtain the reference measures relative to the four demonstrators in each study. The validation metrics defined in the previous chapter need to be compared against these baseline measures. By “historical data” 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.

Precisely, in case the partners were already collecting the necessary measures, historical data would be retrieved by their internal database. If not, during the initial stages of the process each partner will have to put in place mechanisms for starting to record such measures.

Along the project life, such baseline measures will be collected at different times. We do this for two reasons: *i)* to have data redundancy that can better sustain statistical measures, and *ii)* to control potential variations. In simple terms, such measurements represent a "snapshot" of the starting status before any influence of ElasTest.

As is the case, the four vertical demonstrators are quite different in terms of scope, processes followed, personnel involved. Hence, it was necessary to specialize the procedure for the historical data collection according to the specific processes in use within the four partners. To this purpose, we conducted a preliminary remote interview to the personnel in charge of managing the demonstrators within the four partners. Afterwards, remote interviews with the representative of the four vertical demonstrators have been also conducted.

These two stages have been used by CNR to prepare a draft version of both: the data collection structure for the CCS, and the questionnaire surveys for testers and end-users. These draft versions have been both used for a pilot measurement, 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.

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.

3.1 Comparative Case Study (CCS)

Yin ([5], 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. In other words, you would want to do case study research because you want to understand a real world case and assume that such an understanding is likely to involve important contextual conditions pertinent to your case.*"

Thus, similarities and differences to support or refute propositions and to evaluate the impact of the causality (i.e., the extent to which the intervention caused the results, particularly outcomes and impacts) have to be collected. Usually the selection of specific cases is directly linked to the metrics to be evaluated, and generally both qualitative and quantitative data are considered.

With reference to the validation metrics (VMs) presented in Chapter 2, CCS would aim at assessing VM 1.1, VM 1.3, VM 2.1, and VM 2.2.

Considering the data collection during the ElasTest project, the procedure adopted for conducting the CCS includes the following stages:

a) Historical data collection stage - labeled WO (Without Elastest):

- Each of the demonstrator partners selects the projects (concluded/ongoing/new) comparable to the selected ElasTest demonstrators.
- For each of the selected projects, the partner fills a specific form (CCS data excel file) with data relative to the 4 CCS 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 features.
- For each of the selected projects each partner fills a specific form (CCS data excel file) with data relative to the 4 CCS metrics.

c) The data relative to stages a) WO and b) WE are analyzed by CNR and the comparative value for each of the 4 metrics derived.

The form that we developed for CCS data collection is structured in 4 different sheets, each one associated with a different validation metric among those defined in Chapter 2. The degree of detail in each sheet varies from more general (at the top level) to more specific. Some of the data are mandatory and strictly necessary for the validation metrics. Others are requested, as they could be useful for tuning and refining the metric itself. Mandatory data are labeled with an "*" superscript.

In the remaining of this section details about the metrics in each of the four sheets are provided.

3.1.1 Time to Market

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. TTM refers to VM 1.1. Our expectation is that ElasTest will notably decrease the testing and maintenance phases, and hence TTM will be decreased: the estimate in the DoA says on average by 20%.

We will collect mandatory measures related to the whole development process, as well as optionally same measures but related to some sub-phases (namely analysis, unit, integration and system test). Moreover, we also provide opportunity to collect measures specific to a single functionality or component beyond those relative to the whole system. As it should be clear, providing such a detailed and customizable form allow us to consider different development processes. The specific measures are reported in Table 2 below.

Global values	Values per functionality/component
N. Developers*	precondition (dependency)
N. Testers*	postcondition (dependency)
Type of product*	N. Developers
Type of development process*	N. Testers
Type of test strategy*	Type of test strategy
Testing tool(s) adopted*	Testing tool(s) adopted
N. of test cases*	N. of test cases
N. of tested devices*	
Time from start to end *	
%Time from start to end	
Time of post processing *	

Table 2. Time-to-Market collected measures

In particular, considering the values for completion of the development process (time from start to end) collected both during the stages a) and b), 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. The other more detailed measures will be very useful to interpret the observed result, and to derive other meaningful comparisons.

3.1.2 Productivity

In the ElasTest DoA tester productivity is related to VM 1.3 and is defined as "lines of code tested per time unit". However, while preparing the validation procedure, we considered that tester productivity could be also related with other evidences related to the target of test. Hence other possible aspects to investigate during validation could cover the following questions: Is it possible to make testing more effective? Is it possible to make testing more efficient? Is it possible to shorten the test cycle? Is it possible to spend less effort on testing? Is it possible to close more (or to discover fewer) defects in each system release? Here below the list of the additional evidences considered in the ElasTest validation procedure.

- Number of test cases run
- Number of defects found
- Number of defects closed
- Number of defects reopened
- Number of defects replicated

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

During the validation activity, considering the descriptions provided by each single partner and the above considerations, the fields included in the form adopted for the productivity data collection are summarized in Table 3 below:

Global values	Values per functionality/component
N. Developers*	N. of lines of code
N. Testers*	Total N. of test cases
Type of product*	Total N. of defects found
N. Lines of code*	Total N. of defects closed
Type of development process*	Total N. of defects reopened
N. of test cases*	Total N. of defects replicated
N. of defects found*	Minimum defect severity
N. of defects closed*	Maximum defect severity
N. of defects reopened*	N. of developers
N. of defects replicated*	N. of testers
Minimum defect severity*	Testing tool(s) adopted
Maximum defect severity*	Time Unit (Hour, Day, Delivery,...)
Testing tool (s) adopted*	Average N. lines of code per Time Unit (TU)
Time Unit (Hour, Day, Delivery,...)*	Average N. test cases executed per Time Unit (TU)
Average N. Lines of code per Time Unit (TU)*	N. of lines of code tested per TU
Average N. test cases executed per TU*	N. of test cases executed per TU
Environment conditions	Total N. of defects found per TU
	Total N. of defects closed per TU
	Total N. of defects reopened per TU
	Total N. of defects replicated per TU
	Minimum defect severity per TU
	Maximum defect severity per TU

Table 3. Productivity collected measures

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)

whereas the other more detailed measures will be very useful to interpret the observed result, and to derive other meaningful comparisons.

3.1.3 Maintenance

In ElasTest a key factor for improving quality attributes such as compatibility, maintainability, portability and so on, is the level of corrective maintenance effort of SiL. This is targeted as VM 2.1. To measure the latter, we introduced the metrics in the left column of Table 4 below. In addition, the metrics considered for each time unit are shown in the right column of the same table.

Global values	Values per functionality/component
Total N. Developers*	Total N. Developers
Total N. Testers*	Total N. Testers
Type of product*	Total N. lines of code
Total N. lines of code*	N. of test cases ri executed
Type of development process*	N. of new test cases
Total N. of test cases ri-executed *	N. of defects found
Total N. of new test cases *	N. of defects closed
Total N. of defects found*	Person-hours devoted to maintenance
Total N. of defects closed*	Testing tools adopted
Total person-hours devoted to maintenance*	Minimum defect severity
Testing tools adopted*	Maximum defect severity
Minimum defect severity*	Time Unit (Hour, Day, Delivery,...)
Maximum defect severity*	average N. of test cases ri-executed per TU
	average N. of new test cases per TU
	average N. of defects found per TU
	average N. of defects closed per TU
	average person-hours devoted to maintenance per TU
	Minimum defect severity per TU
	Maximum defect severity per TU

Table 4. Maintenance collected measures

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

$$Maintenance\ \Delta = (\text{Total person-hours}_{W0} - \text{Total person-hours}_{We}) / \text{Total person-hours}_{W0} \quad \text{(Eq. 3)}$$

Additional metrics derived by the combination of some of the above ones are:

- percentage of number of defects closed / number of defects found
- percentage of number of defects found / number of test cases executed

3.1.4 Field Failure

In ElasTest a key factor for improving quality attributes such as effectiveness, efficiency, satisfaction, functional suitability, performance, reliability, security and so on, is the value of field failures reported. This is targeted as VM 2.2. To measure the latter, we need to consider failures reported by final end-users/clients of the demonstrator with and without ElasTest. However, considering the timeline of the project, we might not be able to observe this feedback before M36 (project termination) for the four partners involved. More specifically, we introduced the metrics in the left column of Table 5 below. In addition, for each time unit, the metrics considered are shown in the right column of the same table.

Global values	Values per functionality/component
Type of product*	N. lines of code
N. lines of code*	N. Developers
N. Developers*	N. Testers
N. Testers*	Type of development process
Type of product*	Total N. of failures reported
Type of development process*	Total N. of failures solved
Total N. of failures reported*	Total person-hours devoted to failure analysis
Total N. of failure solved *	Total person-hours devoted to failure solving
Total person-hours devoted to failure analysis*	Minimum defect severity
Total person-hours devoted to failure solving*	Maximum defect severity
Minimum defect severity*	Time Unit (Hour, Day, Delivery,...)
Maximum defect severity*	Total N. of failures reported per TU
	Total N. of failures solved per TU
	Total person-hours devoted to failure analysis

	Total person-hours devoted to failure solving
	Minimum defect severity per TU
	Maximum defect severity per TU

Table 5. Field Failure collected measures

In particular to achieve the aimed decrease of at least 30% we need to observe:

$FFR\ \Delta\ =$ (Total # reported failures $_{W0}$ - Total # reported failures$_{We}$)/ Total # reported failures$_{W0}$ (Eq. 4)

Additional metrics derived by the combination of some of the above ones are:

- percentage of number of failures solved/ number of failures reported
- number of failures reported / number of lines of code
- number of failures reported / person-hours devoted to failure analysis
- number of failures solved / person-hours devoted to failure solving

3.2 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 plan to collect such data through web forms that have been prepared by CNR or through personal interviews. A copy of the prepared questionnaire is attached in appendix to this document.

The empirical survey involves different kinds of data:

1. Quantitative data relative to project testing and maintenance activity. Testers and end users are asked to fill a questionnaire about specific project data, such as time taken for test selection or test execution, failure reports received in kind and number, artifacts used to carry out development activities, etc.
2. 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 experiencing the ElasTest platform.
3. Personal data relative 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. Such data have been collected so that in addition to global analyses, they can be useful for drawing conclusions about project improvements specialized for different technical expertise. Besides, the data will be also categorized according to the gender of respondents, with the aim to dig out any potential differences in the data.

Responsible for treatment of data relative to item 3) is CNR.

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 are processed manually and with the support of automated tools 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 CNR-ISTI procedural Regulation.

At time of writing no paper has yet been presented about the validation process and results. In future, we cannot exclude that validation data may be disseminated for scientific publication of research (in line with the goals of ElasTest project) but only after cleaning all information relative to personal data, and in aggregated form.

3.3 Quasi Experiment (QE)

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

- the Without ElasTest group (WO): the team testing following the best practices commonly adopted in the company and using the standard tools and facilities;
- the With ElasTest group (WE): the team performing the testing activities with the support of ElasTest.

During the testing activity both groups have to collect a set of metrics relative to the Validation Metrics VM 1.2 and 2.3 (see Chapter 2), as established in the DoA. Moreover, we also ask for more refined metrics, which can help in data analyses.

Specifically, the steps for the QE are:

1. The testing team (WO and WE) starts analyzing the documentation
2. The Development Team develops (finalizes) the first version of the SUT
3. The Quality Analyst (QA) prepares a common set of test directives
4. The test directives are translated into:
   - A WO test plan by the WO testers
   - A WE test plan by the WE testers
5. In parallel testers in each team starts testing and 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’
6. Dev team collects both the WO and WE bugs and fixes them

[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 this deliverable, which were collected and analysed before May 25th, 2018, the Italian law DLgs196/2003 was followed. From M19 we will adapt to GDPR norms.

- NOTE: bug fixes are not deployed until both teams (WO and WE) have finished the first test iteration.

7. Once both teams (WO and WE) finish their first test iteration, Dev Team deploys a new version of SUT with all the bugs fixed.
8. The QA starts a new iteration of testing.

4 Schedule

We have set a schedule for the validation strategy that is aligned with the ElasTest Release plan. The schedule is depicted in Figure 1 below.

We report the flow structured into the three years of project lifecyle. We have also reported the planned releases, from R1 until the last R9.

During the first year, when the ElasTest platform was not yet available, we have worked to prepare the validation strategy, with the help of some pilot studies.

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

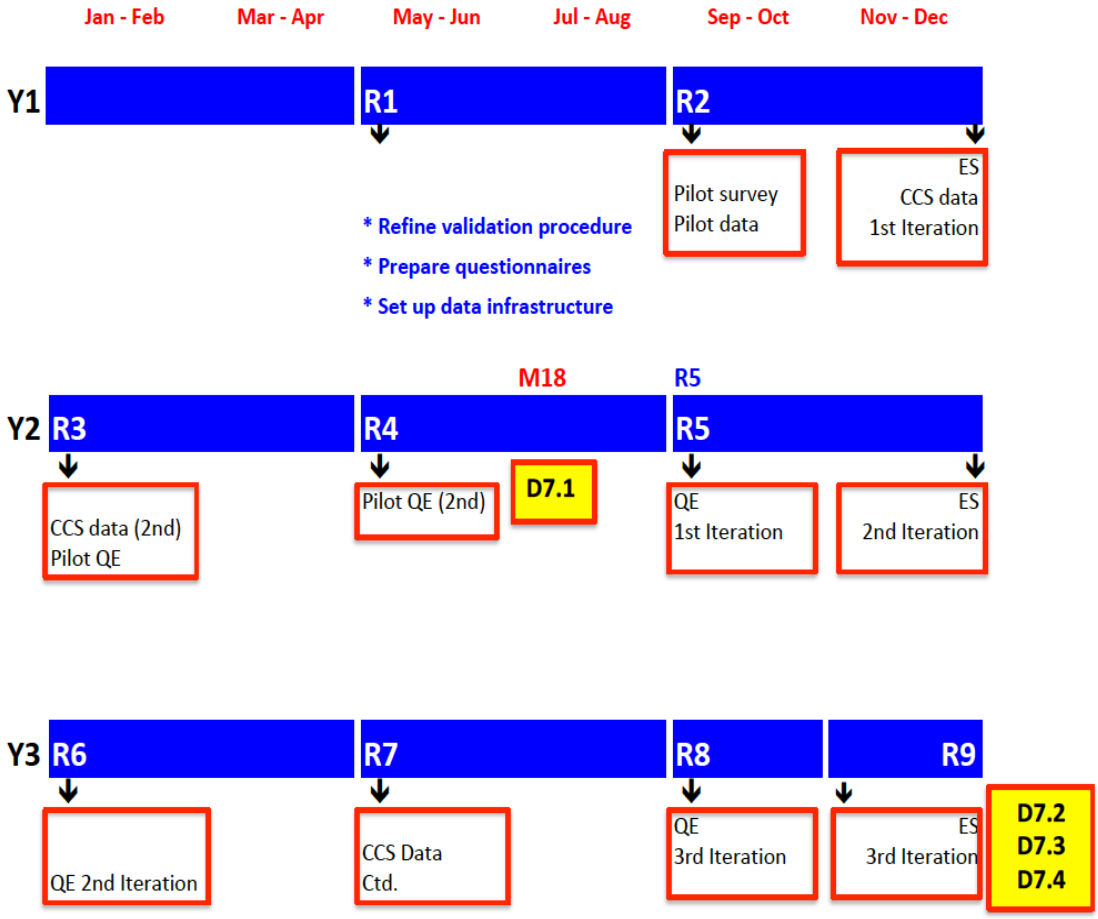

Figure 1. ElasTest validation schedule

With respect to the above schedule, the project is in good shape, although some adjustments have been performed. Precisely, we have currently performed 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

Concerning the ES, we plan to collect questionnaires surveys in 3 iterations: 1st iteration, at M12, is of course relative only to the WO case. After ElasTest Release 5 we will collect more surveys (2nd iteration), and finally we count to get the bulk of questionnaires after final release (M36).

Concerning the CCS, we planned to collect historical data after R3. This was partially accomplished. As we explain in Chapters 5…8 not all demonstrators had in place capabilities to collect the data, so we collected the same metrics during the QE. We refer to Chapters 5…8 for detailed reports.

Concerning the QE, we planned to carry out them along three iterations, starting from R5. However, we have also planned a pilot set of QEs based on R3. As anticipated in the Executive Summary, the objective of these pilots is twofold: on the one side, providing early feedbacks to the consortium about identified strengths and weaknesses, and on the other side validating and tuning the validation methodology itself.

In line with the above, in the following chapters, we report current data from the four demonstrators related to: CCS historical data, QE pilot study, and ES first iteration. Once again, we recall that such data refer to ElasTest Release 3.

5 Atos vertical demonstrator

Atos is responsible for the e-commerce demonstrator. E-commerce applications deal with payment gateway and are one of the most critical kind of websites. Among other aspects, testing such applications provides the challenge of validating their behavior under many different software configurations. The application has been selected because it is the most representative web application in Atos, before the introduction of the features of ElasTest project. The product considered in the experiment is a web application involving more than 10 single functionalities tested using a black box approach.

One of the most important aspects for an e-commerce client is to be sure the platform is working perfectly from all kind of browsers and versions of those browsers. With ElasTest it is possible to do this quite simply and without any additional installation.

At the same time ElasTest allow the QA to test the application manually or automatically leveraging on Selenium the most widely spread technology for automated testing.

Once ElasTest has been installed and configured, visualizing different kind of metrics of the server like memory consumption or CPU usage is really straightforward for the QA.

Saving logs from all executions, having all the execution information correctly saved for further analysis for the near future is done automatically what makes different executions comparison really easy.

5.1 Description of Atos CCS

In this section, considering the metrics relative to the Comparative Case Study, we report the data collected during the step *historical data collection* of the procedure presented in Section 3.1. In particular for each of the four metrics, we provide below the overall evaluation in the respect of the confidentiality agreement established with the partner. More detailed information can be provided on specific demand through private agreement.

To have comparable data, the applications selected for the CCS belong to ecommerce category.

The personnel involved in the testing phase were developers and testers. The former mainly involved in the analysis and development phase, the latter in the Component, Integration and System Test. The strategy adopted was mainly based on black box testing and all the test cases (58 in total) have been executed and analyzed manually with the help of the TestLink tool.

The test phase followed and incremental strategy, new functionalities were delivered to the test team progressively. At the end of the functional test a regression test was performed.

5.1.1 Atos Time to Market

Analyzing the Atos initial situation differently from the other vertical demonstrators, for this partner it has been necessary to collapse the Unit and Component stage with the Integration one. Indeed, in Atos Component and integration testing is the same or done at the same time because components are never tested in isolated environment. Components are always integrated in one environment. Atos TTM data collection refers to the entire development time (from start to delivery date) and it involves 15 developers and 19 testers.

In detail the testing time spent on Analysis and Development was 329.25h, on Component and Integration test 20h, and on System test 3.16h, for a total amount of 352.41h. Figure 2 below reports the percentage of time devoted to the completion of each single development stage.

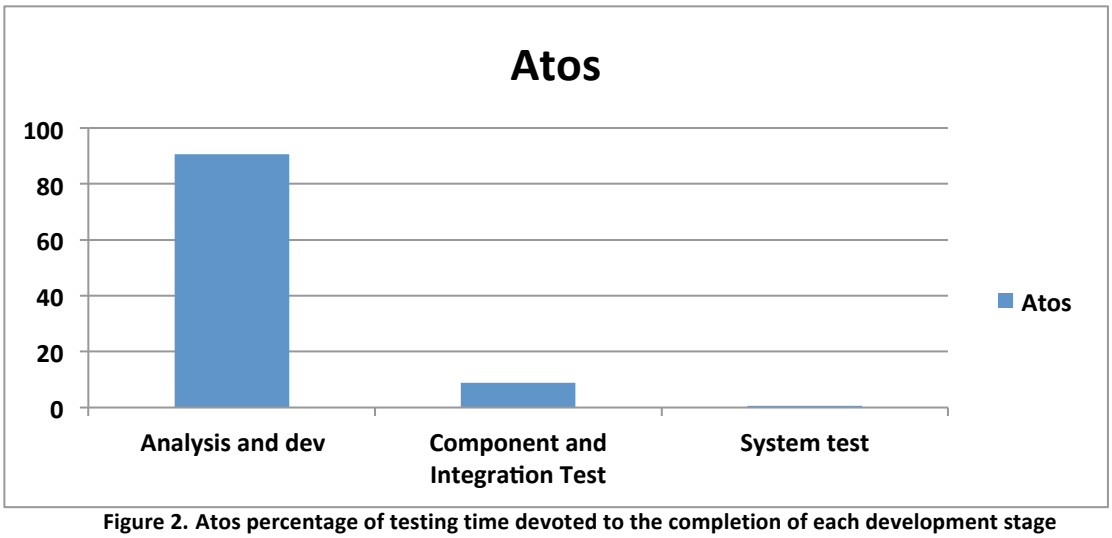

Figure 2. Atos percentage of testing time devoted to the completion of each development stage

Considering the current situation, the most influencing phase on the overall time to market is the testing time devoted to Analysis and development one. The same picture can be derived analyzing the single functionalities of application considered. From these data it is clear the ElasTest adoption could have an impact on the reduction of the testing time devoted to the analysis and development stage.

5.1.2 Atos productivity

Considering the productivity, for the specific type of testing activity performed by the partner, data about the lines of code of the SIL were not available. The number of functionalities under test has been considered instead. Moreover, for having a better picture of the Atos productivity level, different evidences have been considered such as:

- Number of test cases run
- Number of defects found
- Number of defects closed
- Number of defects reopened
- Number of defects replicated

In all these cases the data collection refers to the entire development time and the number of testers involved was 19.

Table 6 below shows the initial situation. Note that the data have been normalized to the number of personnel involved.

	Integration	System	Total
Productivity per Time Unit (TU)	1	0.9	0.9
N. of test cases	2.8	3.4	3.1
N. of defects found	0.7	0.5	0.6
N. of defects closed	0.7	0.3	0.5
N. of defects reopened	0,5	0	0.3
N. of defects replicated	0	0	0

Table 6. Atos Productivity metrics

5.1.3 Atos corrective maintenance

Differently from other vertical demonstrators, Atos testing team is not in charge of maintenance activity. If production bugs are reported, test team validates everything before delivering the hotfix to production, but no special activities are done to validate the platform is working properly day by day, and therefore this parameter has not been collected.

5.1.4 Atos field failure

Analyzing the Atos initial situation, as for the productivity metrics, data about the lines of code of the SIL were not available thus number of functionalities under test has been considered instead. Moreover, the data available did not distinguish the person-hours devoted to failure analysis from those devoted to failure solving. Therefore, a cumulative number has been reported. In all these cases the data collection refers to the entire development time and the number of testers involved was 37.

Table 7 below shows the initial situation (historical data) for the CCS. Note that the data have been normalized to the number of testers involved.

Field Failure metrics	Value
Number of failures reported	1
Number of failures solved	0.1
Total person-hours devoted to failure analysis and solving	0.6

Table 7. Atos field failure metrics

Considering the additional metrics derived by the combination of some of the above ones, Table 8 reports the computed values (note that the data have been normalized to the number of testers involved):

Field Failure additional metrics	Value
% N. of failures solved/N. of failures reported	8.1
N. of failures reported/N. of lines of code	2.1
N. of failures reported/hours for failure analysis and solving	1.6
N. of failures solved/hours for failure analysis and solving	0.1

Table 8. Atos additional field failure metrics

5.2 Description of Atos QE

At the starting date of the QE a very varied set of web applications were available such as:

- Web Portals
- E-commerce
- Blogs
- e-mail managers
- Download web applications
- Banking web applications

For the QE experimentation the final selection was Worldline, which is a web application implementing e-commerce facilities.

In the specific case of the Pilot QE both for the WO and WE took the same overall amount of time measured in hours and involves a tester in each stage. Unfortunately, among Atos personnel we could not select testers among those available at this time that had comparable expertise: one of the testers had much longer experience in software testing. However, the one yielding lower expertise had more time to get acquainted with the subject under test, and we hope that this could mitigate a little the threat. Precisely the two selected testers can be described as follows:

- QA WO - A QA with 6 months of experience in software testing, and 4 months of experience testing this web application.
- QA WE - A QA with 10 years of experience in software testing, and 1 month of experience testing this web application.

5.2.1 QE metrics for Atos

In the first round of QE the main activities for the WE include:

1. Creating a project
2. Configuring new systems under test, i.e., the Wordline application (SuT)

3. Introducing test plans into the system using ElasTest using the TestLink service integrated in ElasTest[2]
4. Executing test plans using ElasTest and the embedded browser
5. Analyzing browser and SuT logs during the test execution time
6. Analyzing browser and SuT logs using log analyzer when execution is already finished
7. Use functionalities provided by ElasTest.

Considering the specific QE metrics in case of Atos only reusability has been considered that evidenced no relevant difference between WO and WE results. This was mainly due to the similarity between the tools provided by ElasTest and those used in the WO scenario at this stage of the project. Once the test orchestration approach currently under development, will be integrated in the TORM, we expect some increase in reusability metrics.

5.2.2 CCS metrics validation through Atos QE

Once the testers were involved into the QE and had to collect data, we considered that it could be useful to look at the forms designed for the CCS and ask testers to also report as many data as feasible during the QE. In this section, considering such metrics relative to the Comparative Case Study, we report the data collected during the QE both for the WO and WE branches. The purpose was not to collect statistical data for CCS metrics validation but to validate the proposed validation methodology itself. The data was therefore a pilot experiment useful for a better tuning of the validation methodology and for collecting challenges and hints useful for the future project development and improvement.

In particular, considering the metrics reported in Sections 3.1.1, 3.1.2, 3.1.3, and 3.1.4, we focused on the metric explicitly related to the testing activity, for which we identify 3 different stages: Integration, System and End2End phase respectively. In the specific case of Atos Unit test is not applicable.

5.2.2.1 Time to Market through Atos QE

Considering the TTM, Figure 3 below reports the time devoted to the completion of the integration, System and End2End stages in the WE and WO branches. As reported in the figure, the most influencing phase in both cases is the Integration one that almost doubles the completion time of the other stages.

[2] It is worth noting that manual testing was not originally considered in the proposal. The possibility of running manual tests allowing a user to interact with the browsers provided by ElasTest was added on purpose and timely to enable the described QE with Atos.

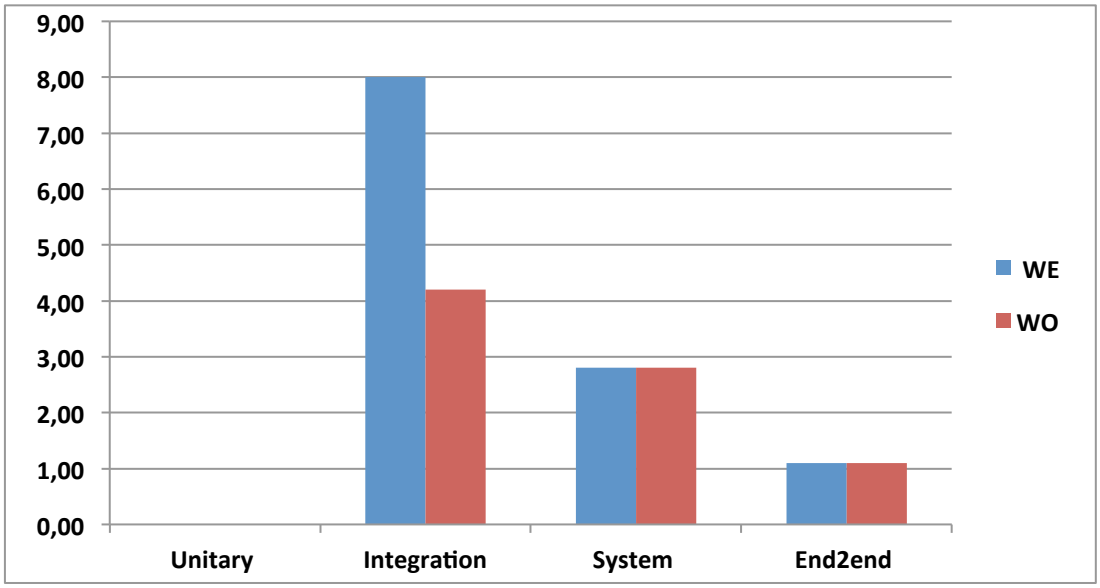

Figure 3. Atos TTM collected during the QE

The figure evidences that in case of integration test the total amount of time necessary for the completion of the WE branch is greater than for the WO. The figure below shows the same analysis in term of percentage of testing time, i.e., the time of each stage has been normalized to the total amount of time dedicated to the entire test activity.

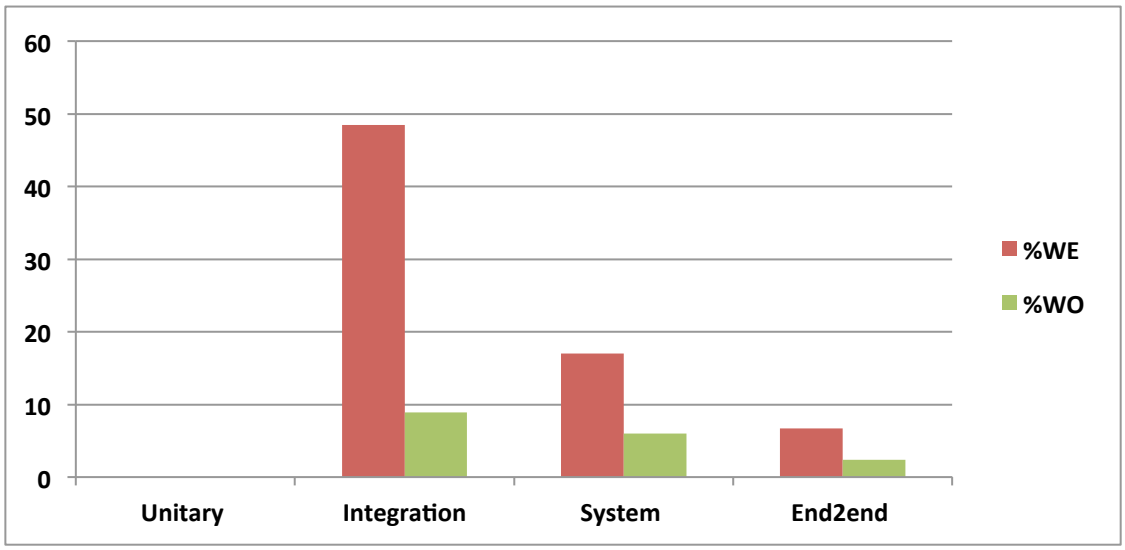

Figure 4. Atos percentage of TTM collected during the QE

For a deeper analysis we performed a fine-tuned analysis, by asking the testers to collect the following information for each testing stage:

- Clock time in human thinking and testing preparation
- Clock time in test coding
- Clock time in test execution
- Clock time in result analysis and debugging

Figure below summarizes the data collected for the WE and WO stages. In the specific case of Atos, the clock time in human thinking, testing preparation and coding was almost negligible in all the testing stage because test suite was already available for both sides of the experiment. Consequently, no differences can be experienced between WE and WO cases. Additionally, the test execution time has not been recorded because included into the result analysis and debugging slot. As shown in Figure 5, the last is the most influencing factor during all the three stages both for the WO and WE sides of the experiment. A general expectation is that the facilities that will be provided by ElasTest in the next releases might reduce this time in all the three stages.

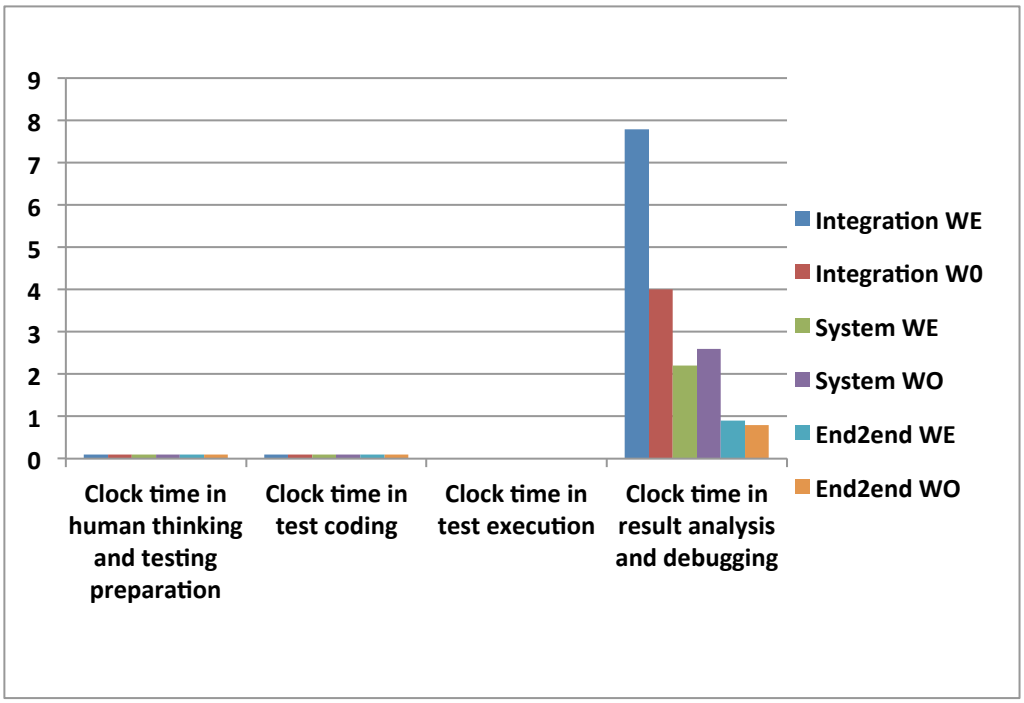

Figure 5. Atos clock time in: human thinking and testing preparation, test coding, test execution, result analysis and debugging

5.2.2.2 Productivity through Atos QE

During the testing activity in both the WE and WO branches, the same number of functionalities has been tested using the same test plan by the same number of testers, therefore no differences have been evidenced in productivity. However, comparing the two settings in number of defects found the WE revealed one more fault than in WO during the integration test. The tester reported that the bug discovered using ElasTest was easily detected because the real time log display helped the user to visualize what was happening in the server side. In this case, some error logs were detected because of real time log functionalities. Table below summarizes the results obtained.

	Integr. WE	System WE	End2end WE	Integr. W0	System WO	End2end WO	Total WE	Total W0
Productivity _TU	8	8	9	8	8	9	25	25
N. of test cases	34	34	9	34	34	9	77	77
N. of defects found	5	0	0	4	0	0	5	4
N. of defects closed	2	0	0	1	0	0	2	1
N. of defects reopened	0	0	0	0	0	0	0	0
N. of defects replicated	0	0	0	0	0	0	0	0

Table 9. Atos productivity metrics collected during the QE

5.2.3 Atos lessons learnt

The execution of the Pilot experiment in Atos for the WE stage has been performed into two rounds. Indeed during the first round the execution of WE revealed some issues of the ElasTest platform hinting at the need of a better management of the containers so to improve the overall performance, at a smarter visualization and management of the browsers during testing execution, and at better management and visualization in case of multiple test cases executions. In order to collect realistic and correct data we forced a preliminary stop of the experiment to let the development team of the project to promptly fix them.

However this was exactly the target of this Pilot Execution: validate the proposed metrics and collect useful information and feedbacks to improve the ElasTest platform.

The data reported in the previous section are therefore relative to the second round of the WE experiment. As reported by the data of the previous section the testing of the web application with ElasTest needs some additional features to become a convenient solution in the Atos situation. In particular most feedbacks and suggestions collected during the second round of the WE were relative to the visualization and management of test cases. For instance, to include real-time log alarms during test case execution; include counters of the number of occurrences of each search pattern; line coloring for lines that meet search patterns; Run test cases in a synchronized way in different browsers.

In the WE branch of the experiment the most encouraging result was the possibility of detecting certain anomalies / bugs which in the traditional way of testing within Atos would have remained undetected.

6 FOKUS vertical demonstrator

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 LTE network, and an abstract architecture is depicted in Figure 6.

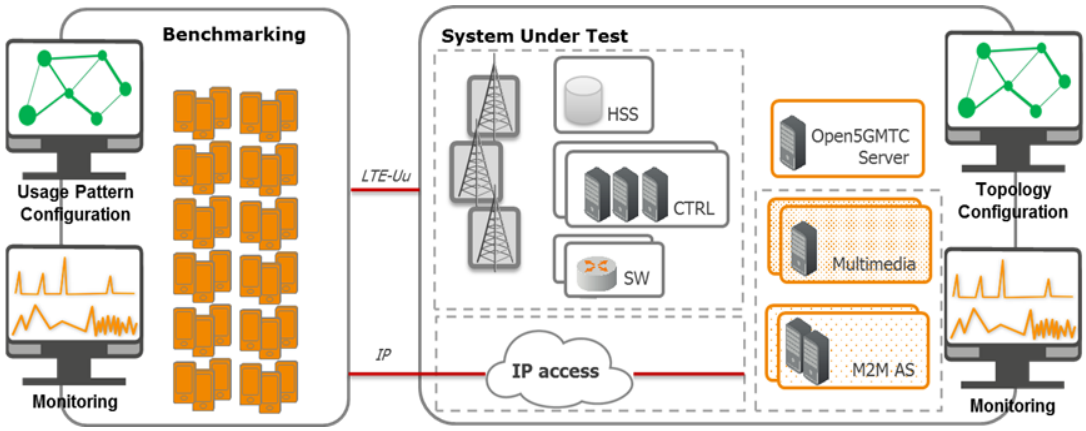

Figure 6. FOKUS Open5GCore toolkit

6.1 Description of FOKUS CCS

In this section, considering the metrics relative to the Comparative Case Study, we reported the data collected during two rounds of historical data collection according to the procedure presented in Section 3.1.

In particular for each of the four metrics we provide below the overall evaluation in the respect of the confidentiality agreement established with the partner. Detailed information can be provided on specific demand through private agreement.

In the first round of data collection the product considered in the experiment was an Evolved Packet Core Network (EPC) that includes standard LTE functionalities tested by a feature-based approach. In the second round, an updated version of the Open5GCore software was used, where specific testing features have been included.

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 an EPC 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 both the rounds 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.

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

6.1.1 FOKUS Time to Market

The data relative to the TTM in the two rounds of historical data collection are reported in the remaining of this section.

6.1.1.1 FOKUS TTM first round

Analyzing the FOKUS initial situation the most influencing phase on the overall time to market are the Unit and Component one. All the remaining phases are almost at the same level with slightly difference for the integration one. From these data it is clear the ElasTest adoption could have a deep impact for the reduction of the testing time devoted to the Unit and Component stage.

TTM data collection refers to the entire development time (from start to delivery date) and it involves 12.5 developers and 3.5 testers. Figure 7 below reports the percentage of time devoted to the completion of each development stage.

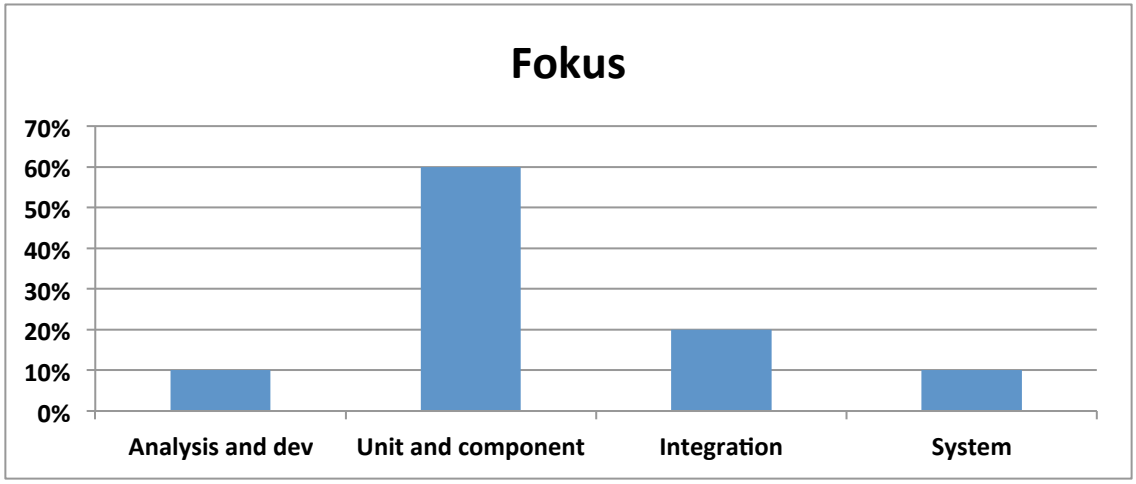

Figure 7. FOKUS percentage of testing time devoted to the completion of each development stage

6.1.1.2 FOKUS TTM Second round

TTM data collection for the second round focuses explicitly on the testing activity from unitary to End to End testing and involves 14 developers and 6 testers.[3]

The second round of data collection confirms the initial situation: the most influencing phase on the overall time to market are the Unitary test. However, the inclusion of the new features moves some effort on the System test. The remaining phases are almost at the same level with slightly difference for the integration one.

From these data and those collected in the previous round it is clear the ElasTest adoption could have a deep impact for the reduction of the testing time devoted to the Unitary testing stage. Figure below reports the percentage of time devoted to the completion of each single development stage.

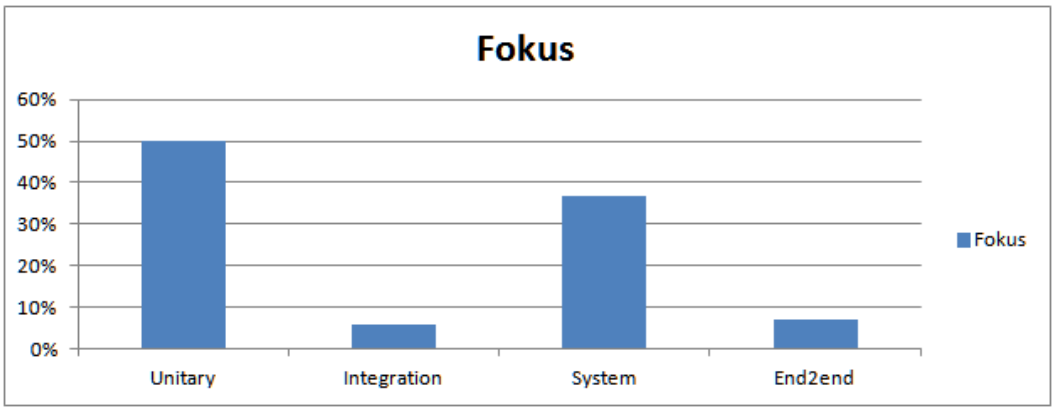

Figure 8. FOKUS percentage of testing time devoted to the completion of each testing stage

[3] Note: in this second-round analysis and development phases were not included and End to End testing added

6.1.2 FOKUS productivity

Analyzing the FOKUS initial situation during the testing activity both developers and testers have been involved. Because a specific role distinction was not possible we grouped them together. Moreover, for having a better picture of the FOKUS productivity level, different evidences have been considered such as:

- Number of test cases run
- Number of defects found
- Number of defects closed
- Number of defects reopened
- Number of defects replicated

In all these cases the data collection covers all the development period. In the remaining we report the data of the two rounds of historical data collection into two separate sections.

6.1.2.1 FOKUS productivity first round

In the first round the overall number of testers and developers involved was 16.5. For the additional metrics from the historical data available it was not possible to recover single values for Integration and System stages. Table 10 below shows the initial situation. Note that the data have been normalized to the number of personnel involved.

	Integration	System	Total
Productivity per Time Unit (TU)	41379.3	300000	36363.6
N. of test cases	-	-	3.0
N. of defects found	-	-	0.6
N. of defects closed	-	-	0.5
N. of defects reopened	-	-	0.1
N. of defects replicated	-	-	0.2

Table 10. FOKUS Productivity metrics first round

6.1.2.2 FOKUS productivity second round

The second focuses explicitly on the testing activity from unitary to End to End testing and it involves 14 developers and 6 testers. In this case from the data collected was not possible to recover the line of codes of the SUT, but only the number of modules involved. Therefore, productivity has been derived in terms of modules tested. Table 11 below summarizes the metrics collected. Note that the data have been normalized to the number of personnel involved.

	Unitary	Integration	System	End2end	Total
Productivity per module	3.33	0.67	5	1,	3
N. of test cases	4.44	1.67	6	1,67	4
N. of defects found	1.33	2.33	0.80	0.33	1.20
N. of defects closed	0.89	1.67	0,60	0	0.80
N. of defects reopened	0.44	0	0	0	0.20
N. of defects replicated	1.11	0	0	0	0.50

Table 11. FOKUS Productivity metrics second round

6.1.3 FOKUS corrective maintenance

Considering the FOKUS initial situation the corrective maintenance effort has been evaluated through:

- Number of lines of code analyzed
- Number of test cases re-executed
- Number of new test cases
- Number of defects found
- Number of defects closed
- Total number person-hours devote to maintenance

In the remaining we report the data collected during only the first round of historical data collection because no data were available for the second one. Thus, the data of this section refers to the entire development time and the number of testers involved was 16.5.

Table 12 below shows the initial situation. Note that the data have been normalized to the number of personnel involved.

Corrective maintenance metrics	Value
N.of LOCs	57142.9
N. of test cases re-executed	4.8
N. of new test cases	0.2
N. of defects found	1
N. of defects closed	0.9
N. person-hours	11.4

Table 12. FOKUS Corrective maintenance metrics

Table 13 below schematizes the initial situation considering the following indirect measures

- percentage of number of defects closed/number of defects found
- percentage of number of defects found /number of test cases executed

Note that the data have been normalized to the number of personnel involved.

Corrective maintenance additional metrics	Value
N. of defects closed/N. of defects found	90
N. of defects found/N. test cases executed	19

Table 13. FOKUS Corrective maintenance additional metrics

6.1.4 FOKUS field failure

As for the other metrics also for the field failure we report the data collected in the two rounds of historical data collection into two separate sections.

6.1.4.1 FOKUS field failure first round

Because a specific role distinction was not possible for the first round of historical data collection we grouped them together. The evidences considered have been summarized in the table below. In all these cases data collection spans the entire development period and the number of testers and developers involved were 4. Note that the data have been normalized to the number of personnel involved.

Field Failure metrics	Value
Number of failures reported	1.3
Number of failures solved	1
Total person-hours devoted to failure analysis	15
Total person-hours devoted to failure solving	5

Table 14. FOKUS field failure metrics first round

Considering the additional metrics derived by the combination of some of the above ones, the Table below reports the computed values. Note that the data have been normalized to the number of personnel involved.

Field Failure additional metrics	Value
% N. of failures solved/N. of failures reported	80
N. of failures reported/N. of lines of code	8.3E-06
N. of failures reported/hours for failure analysis and solving	0.1
N. of failures solved/hours for failure analysis and solving	0.2

Table 15. FOKUS additional field failure metrics first round

6.1.4.2 FOKUS field failure second round

The second round focuses on the testing activity from unitary to End to End testing and involves 14 developers and 6 testers. As for the productivity metric the metrics are provided in terms of number of modules involved. Moreover, from the data collected it was not possible to recover the total person-hours devoted to failure analysis and total person-hours devoted to failure solving. Therefore, some of the metrics have not been calculated.

The evidences considered have been summarized in the table below. Note that the data have been normalized to the number of personnel involved.

Field Failure metrics	Value
Number of failures reported	0.6
Number of failures solved	0.4

Table 16. FOKUS field failure metrics second round

Considering the additional metrics derived by the combination of some of the above ones, Table 17 below reports the computed values. Note that the data have been normalized to the number of personnel involved.

Field Failure additional metrics	Value
% N. of failures solved/N. of failures reported	66.67
N. of failures reported/N. of modules	0.20

Table 17. FOKUS additional field failure metrics second round

6.2 Description of FOKUS QE

The Fraunhofer FOKUS experiment involves the same Open5GCore toolkit already illustrated in Section 6 (see also Figure 6).

For the QE four customers have been selected, two of whom got ElasTest added to their software deliveries (WE), whereas the other two customers used the earlier test component (WO). All customers got to test the software within a pre-established period.

In particular the ElasTest setup consisted of the Core Network base components: hss, mme, switch, PDN-gw, benchmarking-tool deployed via docker-compose. The ElasTest platform had the testbed as external SUT and a set of pre-defined functional and performance tests. Other setups dependent on the customer requirements were: KVM, Vmware, Openstack or plain docker virtual machine image sets.

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 R3 (note that the monitoring engine was not yet integrated)
- Benchmarking tool
- Zabbix
- Dedicated GUI

For the quasi-experiment assessment, the partner QAs will test the Open5GCore at the same time with and without ElasTest. All four customers submitted the results of the testing process after a four weeks evaluation period. The data were afterwards processed in order to assess any improvements resulted from using ElasTest as a testing platform.

6.2.1 QE metrics for FOKUS

In the first round of QE the target metrics have been selected according to the components and features developed in the Release R3 of ElasTest. In case of FOKUS, the target metrics were Reusability, Corrective maintenance, Failure reports and Robustness.

6.2.2 CCS metrics validation through FOKUS QE

In this section, considering the metrics relative to the Comparative Case Study, we reported the average data collected during the QE both for the WO and WE branches. The purpose was not to collect statistical data for CCS metrics validation but for validate the proposed validation methodology itself. The data was therefore a pilot experiment useful for a better tuning the validation methodology and collecting challenge and hints useful for the future project development and improvement.

In particular we focused the metric explicitly on the testing activity thus we identity 3 different stages: Unitary, Integration, System phase respectively. In the specific case of FOKUS End2End test is not applicable.

6.2.2.1 Time to Market through FOKUS QE

In the entire QE the TTM did not evidence differences in the WO or WE. In particular the percentage of the time dedicated to Unitary, Integration and System test for both WO and WE were 0.27%, 0.51 % and 0.22% respectively. Indeed, the facilities provided by the ElasTest did not have an impact on the overall TTM. However, the experiment evidenced the most influencing phase in both the cases is the Integration one that almost doubled the completion time of the other stages.

6.2.2.2 Productivity through FOKUS QE

During the testing activity in both the WE and WO the average line of code was 450K. The same test plan has been developed producing an average of 26.67 test cases both for the WE WO for an overall duration of 28 testing days. The data collected involved only Integration and System stages and no differences have been evidenced in productivity.

6.2.2.3 Maintenance through FOKUS QE

During the testing activity in both the WE and WO the average line of code was 450K. The same test plan has been developed producing an average of 26.67 test cases both for the WE and WO for an overall duration of 28 testing days and involved an average total of 8 testers in each experiment. The data collected involved only Integration and System stages. Both the WE and WO stages have been evaluated through:

- number of lines of code analyzed
- number of test cases re-executed
- number of new test cases
- number of defects found
- number of defects closed
- total number person-hours devote to maintenance

In the remaining we report the average data collected during both the WE and WO data collection are provided. Note that that the data have been normalized to the number of personnel involved.

Corrective maintenance metrics	WE	WO
N.of LOCs	56250	56250
N. of test cases re-executed	2.9	4.6
N. of new test cases	0.9	0,3
N. of defects found	0.8	1
N. of defects closed	0.9	1.1
N. person-hours	10.8	15

Table 18. FOKUS corrective maintenance metrics collected during the QE

Based on the above table, the number of test executed as well as the number of defect found and closed is in favor of the WO. However, during the WE a greater number of test cases have been executed and the overall number of person hours is less than the WO.

Specifically, an API of the Benchmarking Tool was opened towards the Elastest Platform. Thus, tests can be called against an EPC docker deployment by a Tjob instance. The tests consist of standard LTE procedure calls for User Equipment (UE) attachments, detachments and handovers. The number, frequency and duration of the tests are customizable, allowing the testers to perform both functional and performance tests.

Table 19 below schematizes the initial situation considering the indirect measures

- percentage of number of defects closed/ number of defects found
- percentage of number of defects found / number of test cases executed

Note that that the da\ta have been normalized to the number of personnel involved.

Corrective maintenance additional metrics	WE	WO
N. of defects closed/N. of defects found	110	90
N. of defects found/N. test cases executed	22	19

Table 19. FOKUS additional corrective maintenance metrics collected during the QE

In this case the WE evidenced how ElasTest could be a valid help for analyzing the testing data and closing the possible defects found even if the number of defects found per test execution is less than the WO.

6.2.3 FOKUS field failure

We report the average data collected for the Field failure in both the WE and WO stages. We recall that the number of testers and developers involved were 4. Note that that the data have been normalized to the number of personnel involved.

Field Failure metrics	WE	WO
Number of failures reported	1	2.4
Number of failures solved	0.9	2.2
Total person-hours devoted to failure analysis	15	15
Total person-hours devoted to failure solving	5	5

Table 20. FOKUS field failure metrics collected during the QE

As we can see from the above table, the data collected at this stage are in favor of the WO.

Considering the additional metrics derived by the combination of some of the above ones, the Table below reports the computed values. Note that that the data have been normalized to the number of personnel involved.

Field Failure additional metrics	WE	WO
% N. of failures solved/N. of failures reported	92	90
N. of failures reported/N. of lines of code	8,9E-06	2,1E-05
N. of failures reported/hours for failure analysis and solving	0,1	0,2
N. of failures solved/hours for failure analysis and solving	0,2	0,4

Table 21. FOKUS additional field failure metrics collected during the QE

6.2.4 FOKUS lesson learnt

During the QE one of the major issues was the long time necessary for customers to adapt to the software evaluation/testing process. In particular the main difficulty for

customers has been to adapt their testing procedure to the docker environment: a lot of convincing work was needed. However, customers reported that the testing tool (BT) was easier to use with ElasTest.

In particular they appreciated: the test Instrumentation, the test definition and execution, the test composition features, the test templating/ reusability/ test repository – BT now supports Jason-based inputs for tests, the deployment of the platform with multiple virtualization solutions (kvm, vmware, openstack) and finally the automatic deployment and execution of all validation tests as a showcase with test orchestration, as part of the deployment process.

7 NaevaTec vertical demonstrator

NaevaTec demonstrator is focused 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.

The application chosen as a demonstrator is FullTeaching, an e-learning platform that provides teachers and students a full set of collaborative tools under the umbrella of a unique application. 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.
  - RealTime 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. 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.

7.1 Description of NaevaTec CCS

In this section, considering the metrics relative to the Comparative Case Study, we report the data collected during the step a (historical data collection) of the procedure presented in Section 3.1. In particular for each of the four metrics we provide below the overall evaluation in the respect of the confidentiality agreement established with the partner. Detailed information can be provided on specific demand through private agreement.

The FullTeaching product considered in the experiment was a Web Application with WebRTC, which includes 3 main functionalities tested through an Agile approach. The application has been selected because the most representative of the NaevaTec current situation, before the introduction of the features of ElasTest project. Indeed, it was just starting as the ElasTest Project begun so it could be easily instrumented to collect the CCS metrics required. Forcing our company to collect for the first time this kind of information was an important side effect, because we were able to have an overall picture of the quality of our products and to improve the process itself.

The personnel involved in the testing phase were mainly two developers and a tester. The former mainly involved in the analysis and development phase, the latter in the Component, Integration and System Test.

In particular the developers involved have a junior profile (over 1 year experience) and senior (7+ years experience) one. Both have been developing their career with Java (programming language), web services, and Multimedia related projects, more specifically with WebRTC and Kurento. The tester has about 5 years experience (with 3 mainly devoted to QA) involved is specialized in e2e testing, but with wide experience in unitary and integration tests.

The test cases derived (in the hundred order) have been produced using a feature based approach and executed and analyzed by means of JUnit, Pitest, Selenium, JMeter tools applied in combination during the different product development phases

As the project was quite new at time of data collection, we were in the stage of improving the tests development augmenting the coverage of the tests and implementing automatic execution of the developed tests. We follow the logic of maximizing coverage for each feature at a time, beginning with the most critical part login.

7.1.1 NaevaTec Time to Market

In the ElasTest DoA tester productivity is defined as line of code tested per time unit. However, analyzing the NaevaTec initial situation differently from the other vertical demonstrators, for this partner it has been necessary to add an End to End phase. This is because the communications between users are fundamental features of the WebRTC. End to End testing assures that user (or user impersonated) actions and the media streams are generated over the full application deployed and running. In our case integration tests have been considered less important than End2End as the component (outside the application) that manages the Media streams is open source

and it is being (integrated) tested by many other applications that use the same capacities that we use in our application.

Considering the current situation, the most influencing phase on the overall time to market is the Analysis and development one. All the remaining phases are almost at the same level with slightly difference for the End to End one. Figure 9 below illustrates NaevaTec situation. From these data it is expected the ElasTest adoption could have a deep impact for the reduction of the testing time devoted to the Analysis and Development stage.

TTM data collection refers to the entire development time (from start to delivery date) and it involves two developers and one tester.

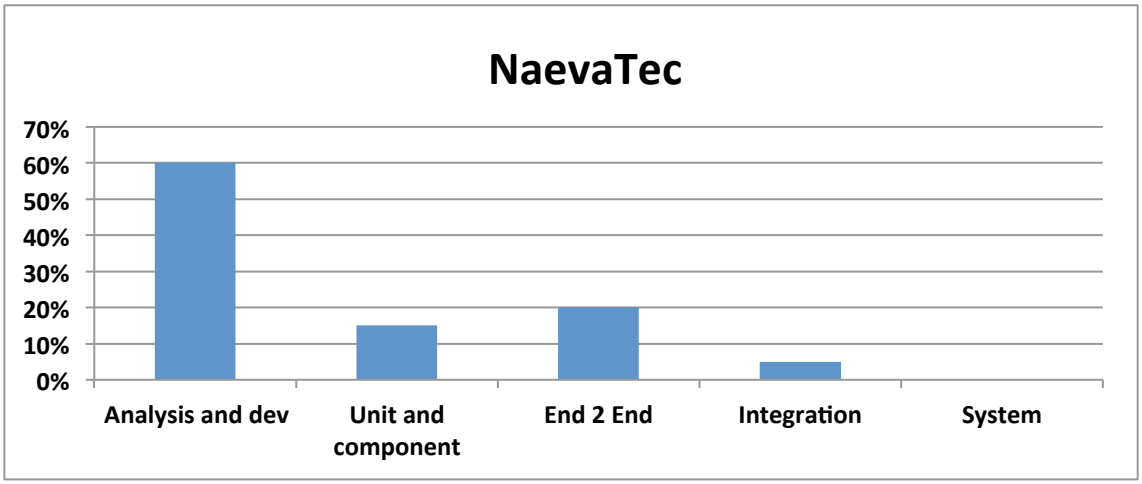

Figure 9. NaevaTec percentage of testing time devoted to the completion of each development stage

7.1.2 NaevaTec productivity

While analyzing the NaevaTec initial situation during the testing activity both developers and testers have been involved. As a specific role distinction was not possible, the results have been grouped together. Moreover, for having a better picture of the NaevaTec productivity level, different evidences have been considered such as:

- Number of test cases run
- Number of defects found
- Number of defects closed
- Number of defects reopened
- Number of defects replicated

In all these cases the NaevaTec performed data collection on a monthly basis and the overall number of testers and developers involved was 2. For the additional metrics from the historical data available it was not possible to recover single value for Integration and System stages.

Table 22 below schematizes the initial situation. Note that the data have been normalized to the number of personnel involved.

	Integration	System	Total
Productivity per Time Unit (TU)	-	-	4833
N. of test cases	16	40.5	56.5
N. of defects found	-	-	0.5
N. of defects closed	-	-	0.5
N. of defects reopened	-	-	-
N. of defects replicated	-	-	-

Table 22. NaevaTec Productivity metrics

7.1.3 NaevaTec corrective maintenance

Differently from other vertical demonstrators, NaevaTec is not in charge of maintenance activity therefore this parameter has not been collected. As explained before is quite a new project and at the moment of data collection we weren't in production. We are finishing the development, tests and the maintenance procedure before launching it into production.

7.1.4 NaevaTec field failures

Based on the fact that the NaevaTec demonstrator is still under development, we report here the failures found during development, and not those reported from the field. During this activity both developers and testers have been involved. Because a specific role distinction was not possible we grouped them together.

In all these cases the Time Unit adopted by NaevaTec was the monthly report and the overall number of testers and developers involved was 2. However, from the historical data available it was not possible to recover values for the total person-hours devoted to failure analysis and solving.

The evidences considered have been summarized in the table below. In all these cases the NaevaTec adopted a monthly data collection period and the overall number of testers and developers involved was 2. Note that the data have been normalized to the number of personnel involved.

Field Failure metrics	Value
Number of failures reported	0.3
Number of failures solved	0.3
Total person-hours devoted to failure analysis	-
Total person-hours devoted to failure solving	-

Table 23. NaevaTec field failure metrics

Considering the additional metrics derived by the combination of some of the above ones, the table below reports the computed values. Note that the data have been normalized to the number of personnel involved.

Field Failure additional metrics	Value
% N. of failures solved/N. of failures reported	100
N. of failures reported/N. of lines of code	1.0E-04
N. of failures reported/hours for failure analysis and solving	-
N. of failures solved/hours for failure analysis and solving	-

Table 24. NaevaTec additional field failure metrics

7.2 Description of NaevaTec QE

The FullTeaching demonstrator has been already described. 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.

7.2.1 QE metrics for NaevaTec

The QoE metrics that can be expected to be gathered by NaevaTec QE have been considered. NaevaTec is working specially on the WO branch to set a specific test suite for these metrics. Work has already been done, but it isn't ready to be compared with the metrics that can be already obtained on the WE branch. The main reason is that whether in the ElasTest those metrics are provided by a generic component (EUS) for the WO test the whole suite and framework should be tailor made specifically for this

application (Fullteaching) and the possibility of reusing functionalities from other similar projects is minimal. This implies that much effort is needed in this task.

7.2.2 CCS metrics validation through NaevaTec QE

In this section, considering the metrics relative to the Comparative Case Study, we reported the data collected during the QE both for the WO e WE branches. The purpose was not to collect statistical data for CCS metrics validation but for validate the proposed validation methodology itself. The data was therefore a pilot experiment useful for a better tuning the validation methodology and collecting challenge and hints useful for the future project development and improvement.

In particular we focused the metric explicitly on the testing activity thus we identity 4 different stages: Unitary, Integration, System and End to End phase respectively.

7.2.2.1 Time to Market through NaevaTec QE

Considering the TTM Figure 10 below reports the time devoted to the completion of the Unitary and End2End phases in the WE and WO stages. Indeed, for the specific case of NaevaTec Unitary, Integration and System tests cannot be fully distinguished, so we include all of them into the Unitary one.

As reported in the figure the most influencing phase for the TTM in case of WO was the Unitary one. However also the End2End testing was in favor of the WE even if with less impact. The difference was many due to the coding activity required in case of WO for developing the required test suite.

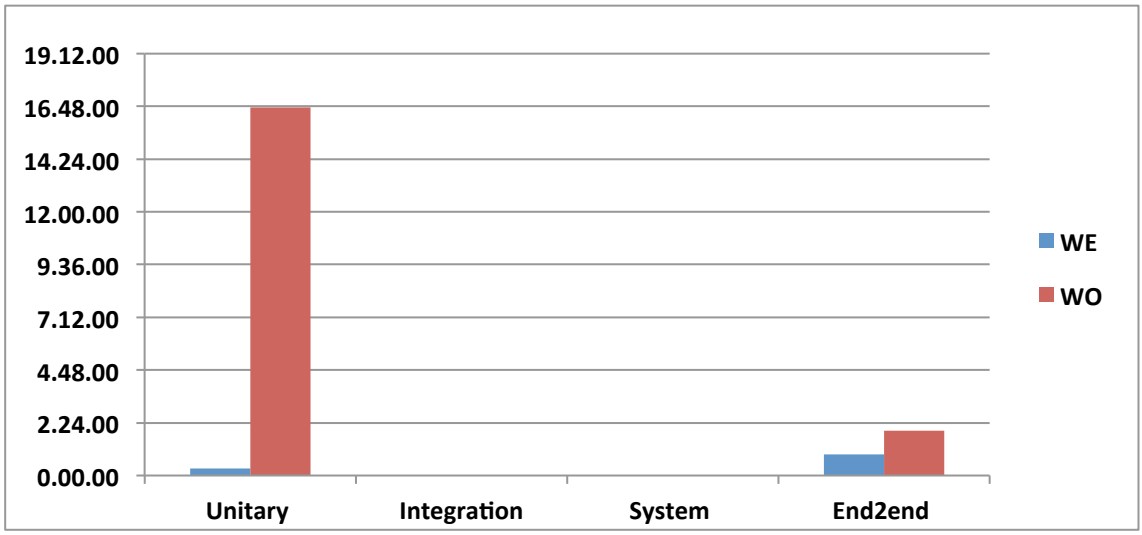

Figure 10. NaevaTec TTM collected during the QE

Figure 11 below shows the same analysis in term of percentage of testing time, i.e. the time of each stage has been normalized to the total amount of time dedicated to the entire test activity.

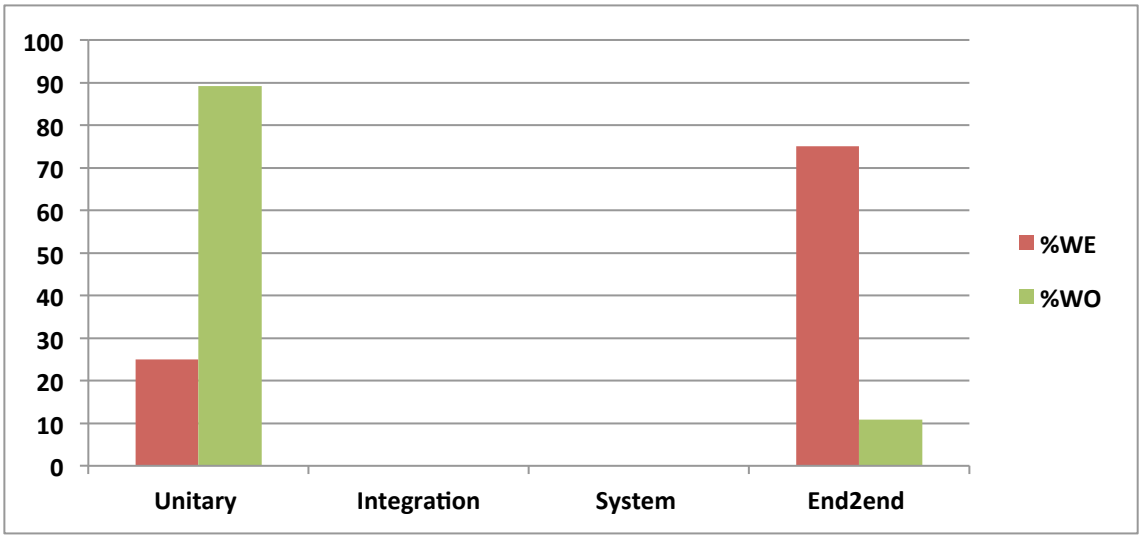

Figure 11. NaevaTec percentage of TTM collected during the QE

As we can see in the picture during the WE experimentation most of the time the tester and developer were involved in the End2End stage while the Unitary test had a small impact on the overall testing activity.

A confirmation of these evidences came from a fine-tuned analysis of the testing data. In this case each testing stage has been divided into conceptually separated steps:

- Clock time in human thinking and testing preparation
- Clock time in test coding
- Clock time in test execution
- Clock time in result analysis and debugging

Figure 12 below summarizes the data collected in the four steps for the WE and WO stages. The clock time dedicated to test coding is extremely high for the Unitary stage of WO. In the remaining steps the required clock time seems to be slightly against the WO.

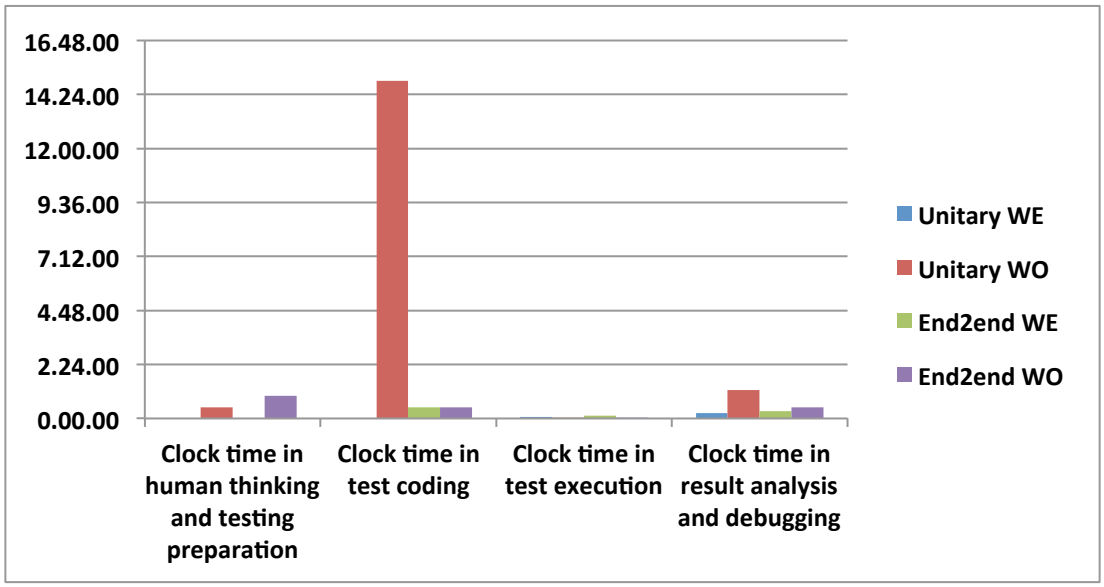

Figure 12. NaevaTec clock time in: human thinking and testing preparation, test coding, test execution, result analysis and debugging

In Figure 13 we report the total amount of time dedicated to each of the four steps for the WO and WE cases.

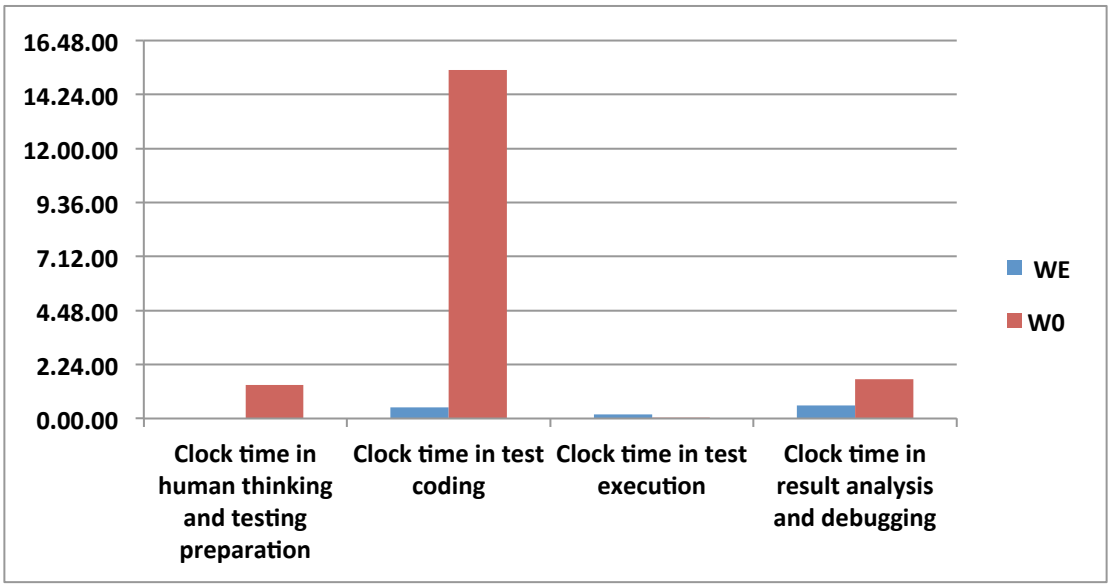

Figure 13. NaevaTec percentage of TTM collected during the QE

7.2.2.2 Productivity through NaevaTec QE

During the testing activity in both the WE and WO the same lines of code have been tested and the same test suite has been developed, therefore no differences has been evidenced in productivity. Comparing the two settings in number of defects found the WO side revealed one more defect than WE side during the unitary test while the same number of defects has been discovered during the End2End testing. However, it is important to note that in the WO branch the defect has been detected only after multiple executions in debug mode while in the WE it was immediately highlighted. Table 25 below summarizes the results obtained.

	Unit. WE	Integ. WE	Syst. WE	e2e WE	Unit. WO	Integ. W0	Syst. WO	e2e WO	Total WE	Total W0
Productivity_TU	1378				1378				1378	1378
N. of test cases	78	30		18	78	30		18	63	126
N. of defects found				1	1			1	1	2
N. of defects closed								1	0	1
N. of defects reopened										
N. of defects replicated										

Table 25. NaevaTec productivity metrics collected during the QE

7.2.3 NaevaTec lessons learnt

The experience in executing testing and improving FullTeaching application within ElasTest was very interesting from the point of view of possible improvements of the overall ElasTest testing process. Initially some issues have been raised during installation due to: i) the complexity of the application itself which: runs on a webserver, has a database and requires a framework to run; ii) the ability of ElasTest to manage huge docker containers. These issues have been overcome and fixed in the version of ElasTest that has been used for the QE.

As an overall feedback the tester in charge of the WE experimentation evaluated the ElasTest version as user friendly and "very helpful" to setup and run the test cases. In particular the docker technologies made it easier to manage the different components and to include and run the docker containers. Moreover, the log facilities were very useful for bug analysis because they also let the tracking of resources usage during the execution of the tests

Improvements are however required to speed up the test execution performance, e.g. by distributing the test execution over multiple machines.

On the WO part of the experiment, we can just asses the common issues when testing this kind of application. Relative to the QoE metrics we were too ambitious and we could not finish the necessary test development in order to get those metrics at this phase, as even with our long-term experience it is long, tedious, and difficult. We are progressing in order to provide those metrics for the next QE iteration.

8 TUB vertical 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 is 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 2 of ElasTest, EDS already encompasses 3 applications that were part of OIF. Furthermore, in the Release 3, a minimal EDS can already deploy sensor and actuator docker containers as required by the demonstrator. 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.

A typical IIoT application comprises of sensor, logic and actuator. For example, in a temperature sensing application, we would like to flag an alarm if temperature goes above 50 degrees centigrade. For this a temperature sensor needed which feeds data in periodic intervals say 1 second to the logic. The logic decides if an actuation is needed by checking the temperature provided by sensor is greater than 50 degrees. If greater than 50 degrees an actuating signal is sent to actuator which 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.

8.1 Description of TUB CCS

In this section, considering the metrics relative to the Comparative Case Study, we reported the data collected during the step (historical data collection) of the procedure presented in Section 3.1.

In particular for each of the 4 metrics we provide below the overall evaluation in the respect of the confidentiality agreement established with the partner. Detailed information can be provided on specific demand through private agreement.

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.

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.

8.1.1 TUB Time to Market

From the data collected by this partner, the most influencing phase on the overall time to market is the Analysis and development one. The same picture can be derived analyzing the single functionalities of application considered. The Figure below visualizes the analysis TUB situation and reports the percentage of time devoted to the completion of each single development stage.

From these data it is clear the ElasTest adoption could have a deep impact for the reduction of the testing time devoted to the Unit and component stage.

TUB TTM data collection refers to the entire development time (from start to delivery date) and it involves 12 developers and 6 testers.

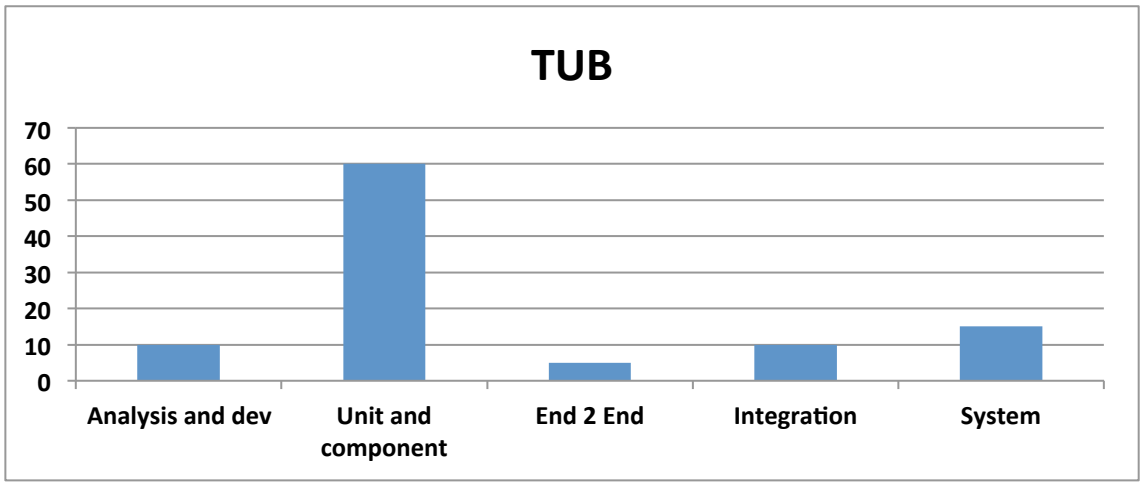

Figure 14. Percentage of testing time devoted to the completion of each development stage

8.1.2 TUB productivity

Analyzing the TUB initial situation, during the testing activity 8 developers and 3 testers have been involved. Because a specific role distinction was not possible the results have been grouped together. Moreover, for having a better picture of the TUB productivity level, different evidences have been considered such as:

- Number of test cases run
- Number of defects found
- Number of defects closed
- Number of defects reopened
- Number of defects replicated

TUB In all these cases the data collection refers to the entire development time. For the additional metrics from the historical data available it was not possible to recover single value for Integration and System stages.

The Table below schematizes the initial situation. Note that the data have been normalized to the number of personnel involved.

	Integration	System	Total
Productivity per Time Unit (TU)			109091
N. of test cases			5.6
N. of defects found			1
N. of defects closed			0.5
N. of defects reopened			0
N. of defects replicated			0.3

Table 26. TUB Productivity metrics

8.1.3 TUB corrective maintenance

Considering the TUB initial situation, the corrective maintenance effort has been evaluated through:

- Number of lines of code analyzed
- Number of test cases re-executed
- Number of new test cases
- Number of defects found
- Number of defects closed
- Total number person-hours devote to maintenance

In all this cases the data collection refers to the entire development time and the number of developers and testers involved was 8 and 3 respectively.

The Table below schematizes the initial situation. Note that the data have been normalized to the number of personnel involved.

Corrective maintenance metrics	Value
N. of LOCs	109091
N. of test cases re-executed	5.6
N. of new test cases	0.5
N. of defects found	1
N. of defects closed	0.5
N. of person-hours	5.5

Table 27. TUB Corrective maintenance metrics

Considering the following indirect measures

- percentage of number of defects closed/number of defects found
- percentage of number of defects found /number of test cases executed

The Table below schematizes the initial situation. Note that the data have been normalized to the number of personnel involved.

Corrective maintenance additional metrics	Value
N. of defects closed/N. of defects found	55
N. of defects found/N. test cases executed	16

Table 28. TUB additional corrective maintenance metrics

8.1.4 TUB field failure

TUB application has been recently released and made open source therefore at the current stage these data are not available.

8.2 Description of TUB QE

For the purpose of Quasi experiment, TUB used OpenMTC. OpenMTC is used as a middleware by OIF and 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.

8.2.1 QE metrics for TUB

In the first round of QE the target metrics have been selected according to the components and features developed in the Release R3 of ElasTest. In case of TUB they were: reusability and scalability. In the following section details of each of them are provided.

The Pilot QE both for the WO and WE took the overall amount of three week each and involves a tester in each experiment.

8.2.1.1 Reusability through TUB QE

According to the DOA the reusability has been defined to the percentage of code, tools and architectures reused in the different testing activity.

During the Pilot QE in both the WE and WO stages the same test plan has been developed producing 18 test cases for the WE and 19 for the WO. In terms of reusability the overall percentage of test reused was greater in the WO but the same percentage of reused facilities has been experienced. This was because the WO tester writes a specific application for launching the different round of test case emulating in some sense the role of ElasTest. This was evident only after a post analysis of the data collected and it was not possible to recover data about the time and effort necessary for writing this kind of application.

However, the focus of the QE Pilot experiments was to validate and refine the validation procedure adopted and to highlight project improvements and required modifications.

Reusability_TU	Integr. WE	System WE	Integr. W0	System WO	Total WE	Total W0
N. of test cases	16	2	15	4	18	19
%tests reused	18.75	50	26.67	100	22.22	42.11
LOCs total of such reused tests	30	40	96	92	70	188
LOCs reused in each test case	10	40	96	92	50	188
N. of basic facilities reused in each test case	9	1	5	5	10	10
% of basic facilities reused in each test case	33.33	100	40	40	40	40
N. of modules reused in each test case	2	3	1	2	5	3

Table 29. TUB reusability metrics

8.2.1.2 Scalability through TUB QE

According to the DOA the scalability is measured as the total number of concurrent supported sessions. During the Pilot QE in both the WE and WO stages the target was to replicate an application execution at least 10 times. As results the same values have been collected in terms of scalability in both the experiments and no difference evidenced.

However, QE Pilot experiments positively validate the procedure adopted for data collection.

Comparing the two settings in number of defects found the WE revealed a fault in EDS while WO demanded a feature from OpenMTC which by default OpenMTC was not designed to handle.

8.2.2 CCS metrics validation through TUB QE

In this section, considering the metrics relative to the Comparative Case Study, we reported the data collected during the QE both for the WO and WE branches. The purpose was not to collect statistical data for CCS metrics validation but for validate the proposed validation methodology itself. The data was therefore a pilot experiment useful for a better tuning the validation methodology and collecting challenge and hints useful for the future project development and improvement.

In particular we focused the metric explicitly on the testing activity thus we identity 4 different stages: Unitary, Integration, System and End to End phase respectively.

8.2.2.1 Time to Market through TUB QE

Considering the TTM figure below reports the time devoted to the completion of the Integration and System phases in the WE and WO stages. Indeed, for the specific case of Unitary, End2End test cannot be applied to this kind of product.

As reported in Figure 15 below the most influencing phase for the TTM in case of WO was the System one. The difference was many due to timed dedicated in result analysis and debugging during the WO. This enforces the idea that ElasTest features aimed at result analysis are indeed useful.

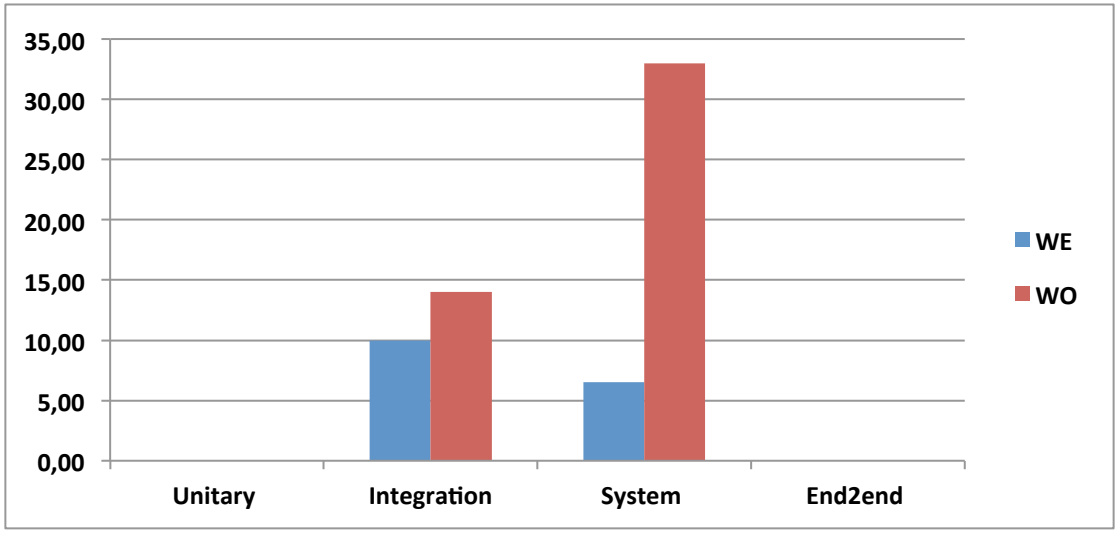

Figure 15. TUB TTM collected during the QE

Figure 16 below shows the same analysis in term of percentage of testing time, i.e. the time of each stage has been normalized to the total amount of time dedicated to the entire test activity. This figure highlights that for the WE are the most influencing stage is the Integration one.

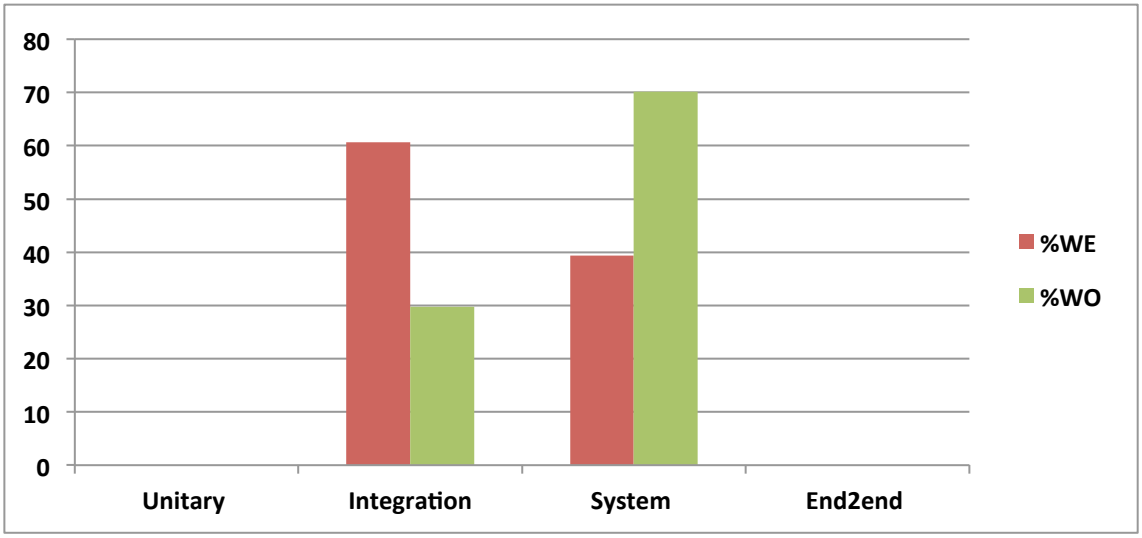

Figure 16. TUB percentage of TTM collected during the QE

A confirmation of these evidences came from a fine-tuned analysis of the testing data. In this case each testing stage has been divided into conceptually separated steps:

- Clock time in human thinking and testing preparation
- Clock time in test coding
- Clock time in test execution
- Clock time in result analysis and debugging

Figure 17 below summarizes the data collected in the four steps for the WE and WO stages. The clock time dedicated to test coding is the most influencing factor during the Unitary stage of WE while that dedicated to the result analysis and debugging is the most critical for the Integration stage during the WO. In the remaining steps the required clock time is always against the WO.

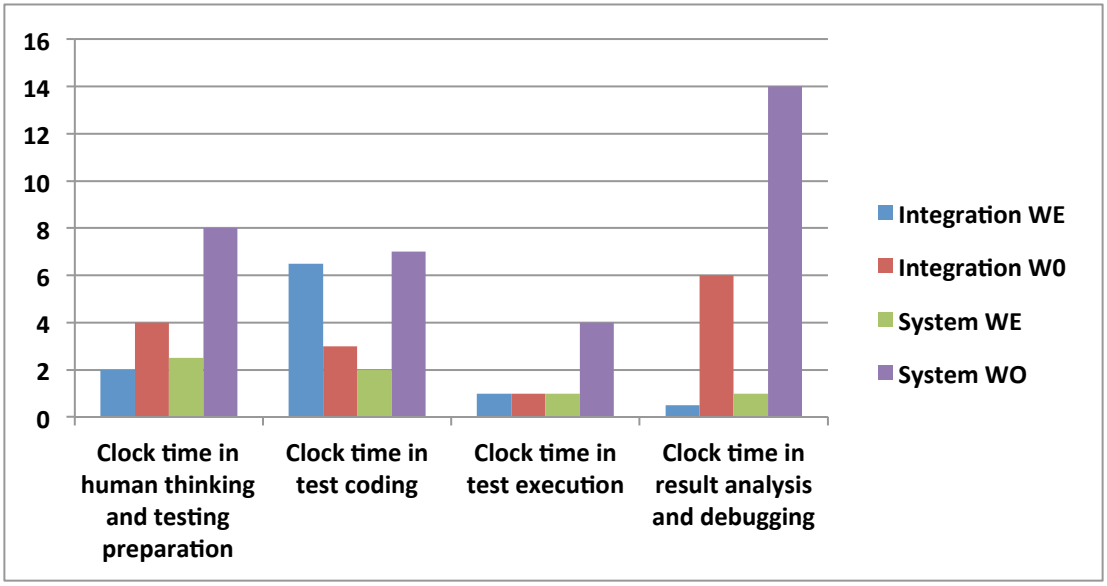

Figure 17. TUB clock time in: human thinking and testing preparation, test coding, test execution, result analysis and debugging

In Figure 18 the total amount of time dedicated to each of the four steps in the WO and WE branches of the experiment is reported.

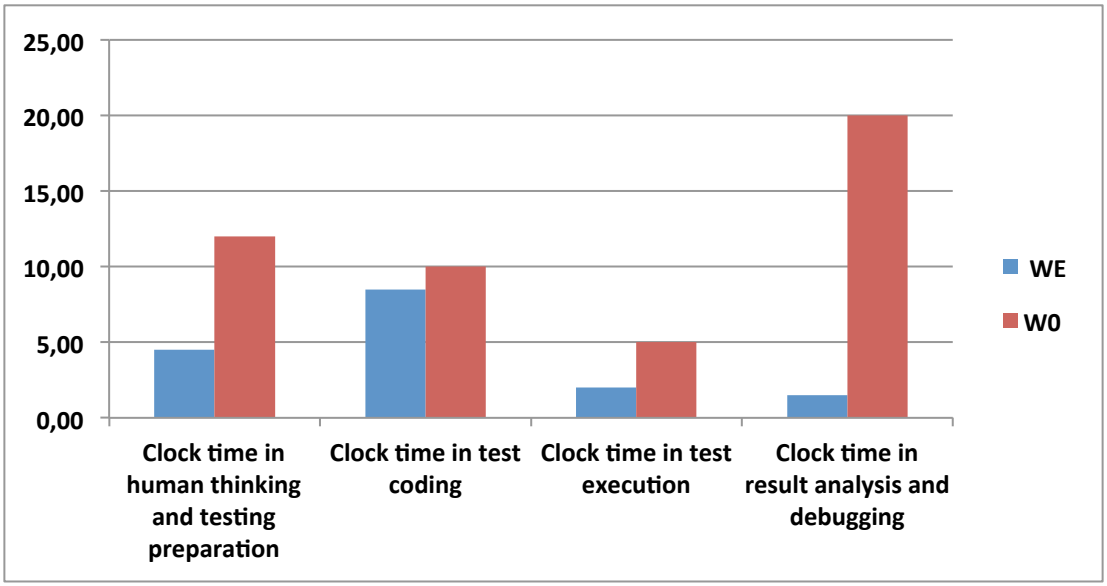

Figure 18. TUB percentage of TTM collected during the QE

8.2.2.2 Productivity through TUB QE

During the testing activity in both the WE and WO the same lines of code have been considered. The same test plan has been developed producing 18 test cases for the WE and 19 for the WO but with no differences evidenced in productivity. Comparing the two settings in number of defects found the WE revealed a fault in EDS while the WO highlighted a possible inconsistency in OpenMTC during the system test. Table 30 below summarizes the results obtained.

	Integr. WE	System WE	End2end WE	Integr. W0	System WO	End2end WO	Total WE	Total W0
Productivity _TU	155	155	155	155	155	155	155	155
N. of test cases	16	2	15	4	18	19	16	2
N. of defects found		1		0	1	0		1
N. of defects closed								
N. of defects reopened								
N. of defects replicated		1			1			1

Table 30. TUB productivity metrics collected during the QE

8.2.3 TUB lessons learnt

During the WO experimentation, the main encountered difficulty was in monitoring lifecycle of applications. Indeed, the monitor facility provided by ElasTest reduces of more than one half the time for analysis and debug.

In the WE side, an important result was the possibility of detecting a fault in EDS component. As general evaluation the WE tester provided positive feedbacks for the features provided by ElasTest useful for the deployment of the applications and in the checking if the SUT is currently up and running. Moreover, ElasTest helped in the prototyping and testing the applications because it allowed to test the EDS between multiple SuTs and TJobs.

EDS helped in reducing the time required to prototype an IoT application.

Several improvements have been suggested in the management of the different log and the image accumulation over time.

9 Pilot empirical survey

In this chapter we report the results collected during the Pilot experimentation of the Empirical Survey. Data presented in this chapter have been collected during the QE from the personnel involved in the WE and WO stage. It is out of scope of this document to provide the final evaluation of the ElasTest as of course the platform is yet preliminary. The data collected through the pilot ES have been useful to evaluate the proposed questionnaires and to collect possible improvements and feedbacks for the next releases of ElasTest itself.

The overall empirical survey experimentation involved 14 testers, 7 in the WO group and 7 in the WE one, having an almost homogeneous expertise as shown in the following figure:

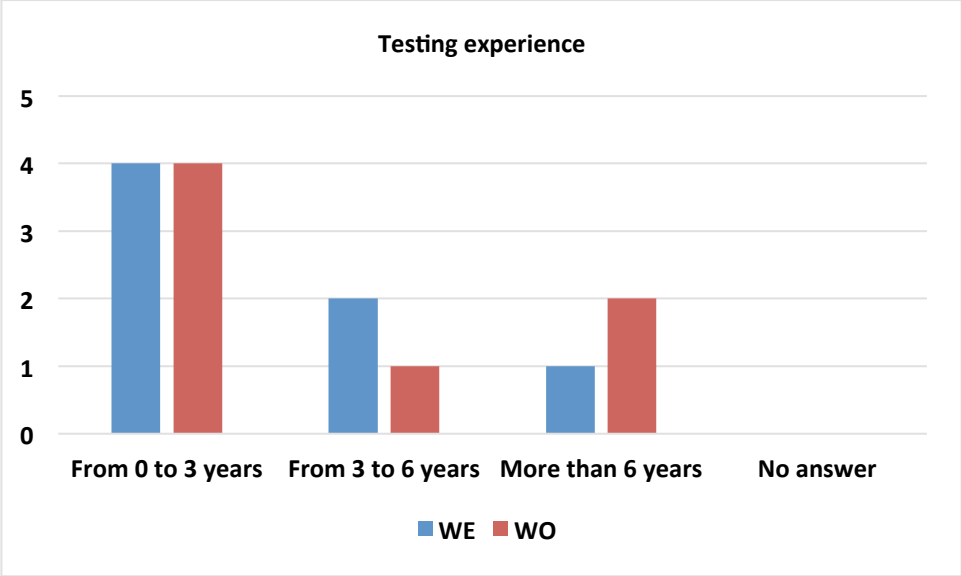

Figure 19. Testing experience

The questionnaire targeted the 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 the following subsection details for each of the above listed characteristics are provided.

9.1 Simplicity

The first group of questions targets the simplicity of the used testing environment. As a first feedback the testers using ElasTest evaluated the testing environment as easier than the standard one (Arithmetic mean 3.86 for WE and 2.86 for the WO). Data collected are summarized in the following figure:

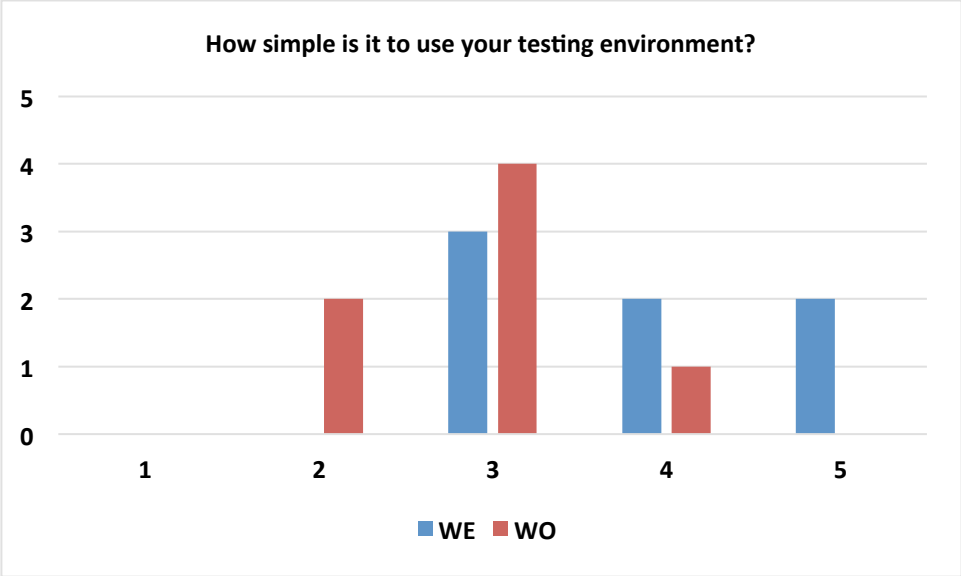

Figure 20. Simplicity

To have a clearer idea of the meaning of simplicity in the two environments additional questions have been proposed to the involved testers as shown in the following figures. In particular testers provided feedback about:

- The simplicity in coding (Arithmetic mean 4.17 and 3.4 for WE and WO respectively);

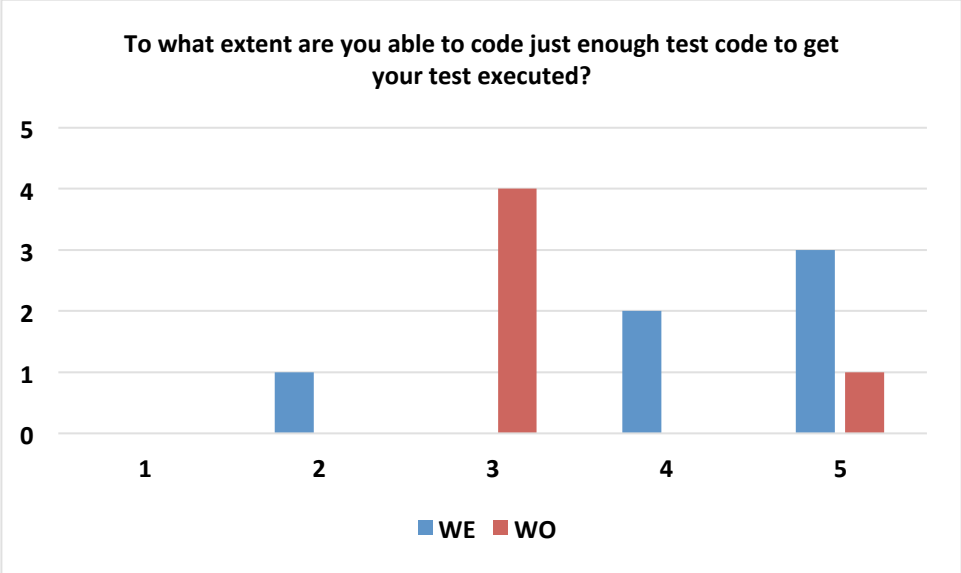

Figure 21. Simplicity in coding

- Interpretation of the test outcome (Arithmetic mean 3.71 and 3.43 for WE and WO respectively);

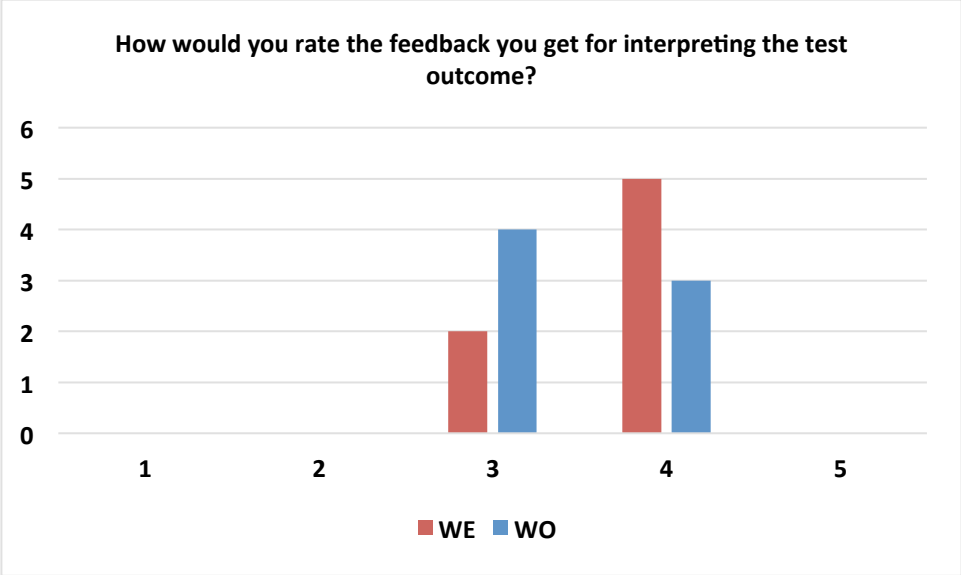

Figure 22. Simplicity in interpretation of the test outcome

- Writing documentation (Arithmetic mean 4 and 2.1 for WE and WO respectively);

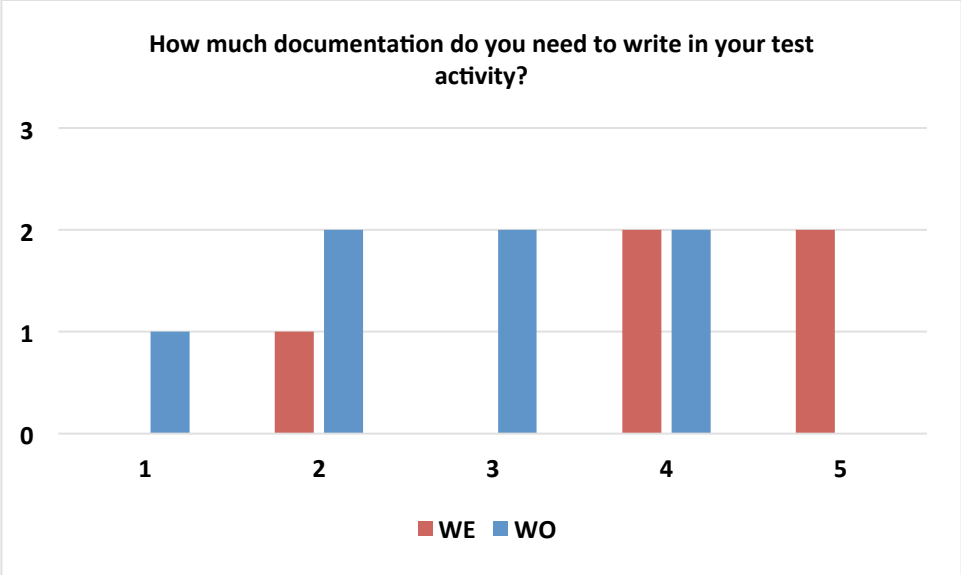

Figure 23. Simplicity in writing documentation

- Reading documentation (Arithmetic mean 3.14 and 2.29 for the WE and WO respectively);

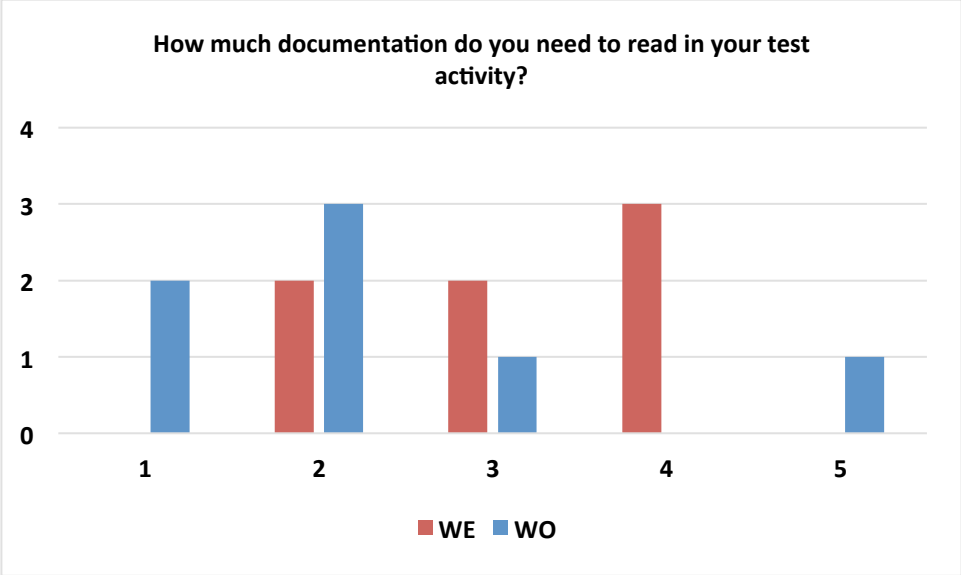

Figure 24. Simplicity in the reading documentation

- Measuring the testing progress (Arithmetic mean 3.67 and 2.57 for WE and WO respectively);

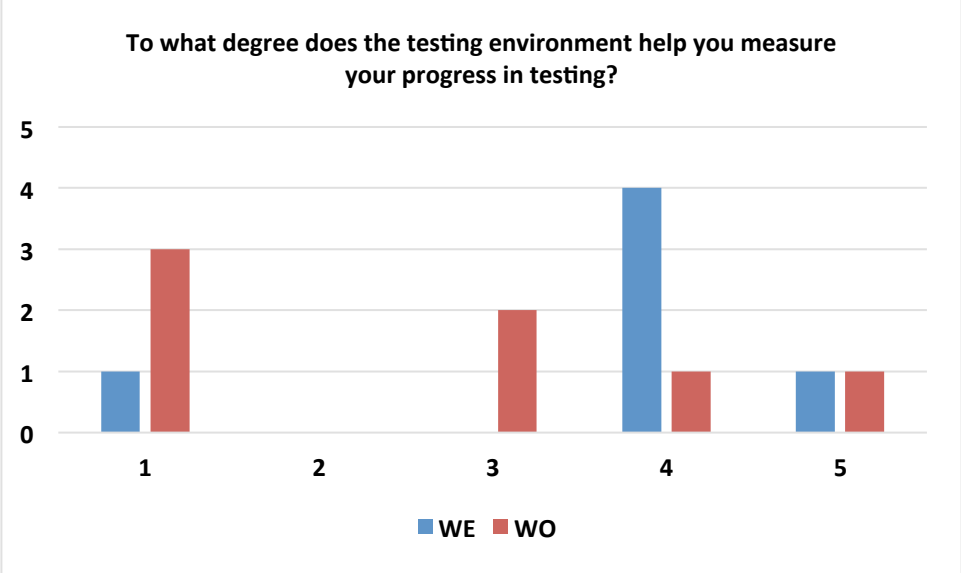

Figure 25. Simplicity in measuring test progress

- Focusing the testing activity on the parts that are relevant (Arithmetic mean 3.43 and 2.86 for the WE and WO cases, respectively).

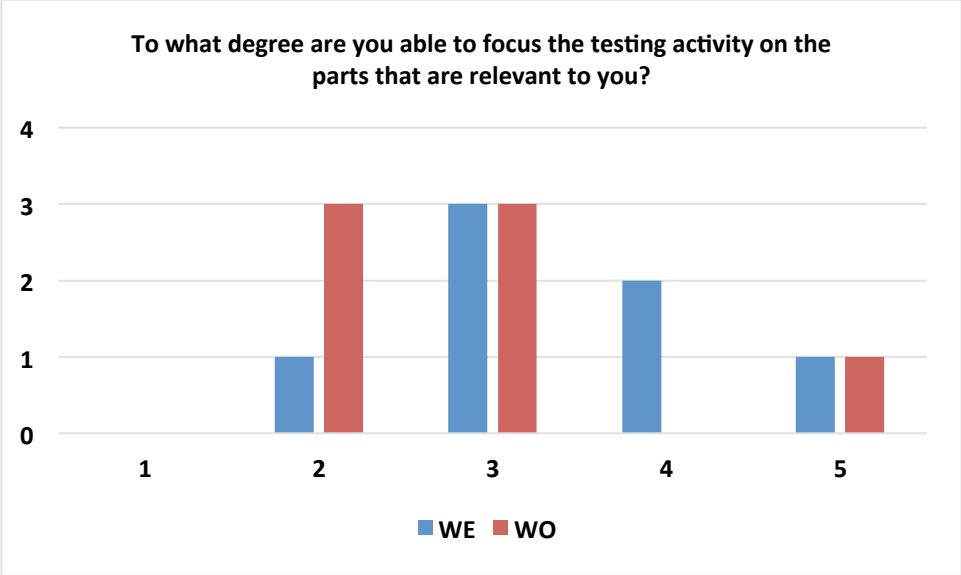

Figure 26. Simplicity in focusing on testing activity parts that are relevant

9.2 Satisfaction

The second group of questions concerns the satisfaction. In this case testers involved in the WE experimentation were more satisfied than those in WO (Arithmetic mean 3.57 and 3 for WE and WO respectively) as summarized in the following figure:

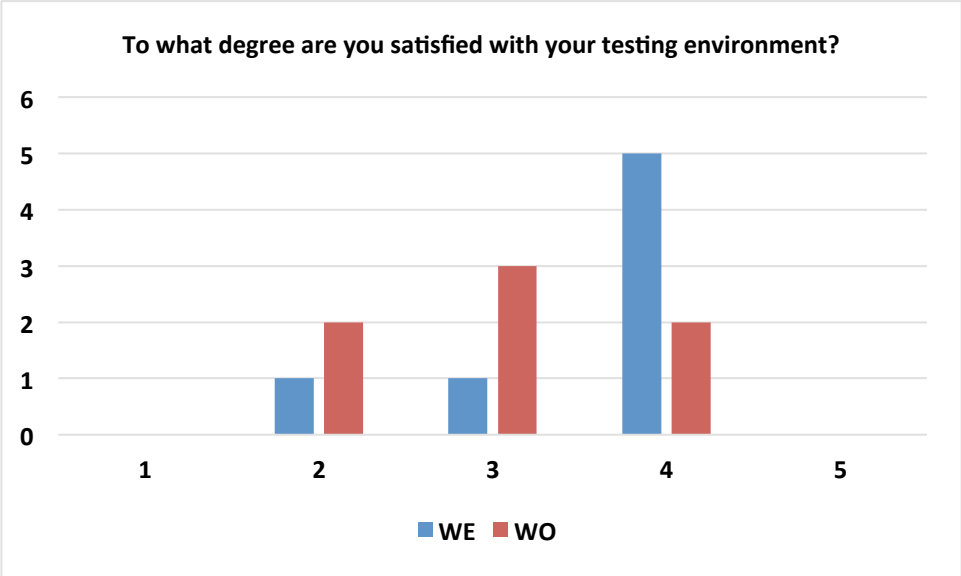

Figure 27. Satisfaction

We try to refine the concept of satisfaction providing additional questions. In particular, as visualized in the following figures, testers provided feedback about:

- The testing activity performed (Arithmetic mean 4 and 3.14 for WE and WO respectively);

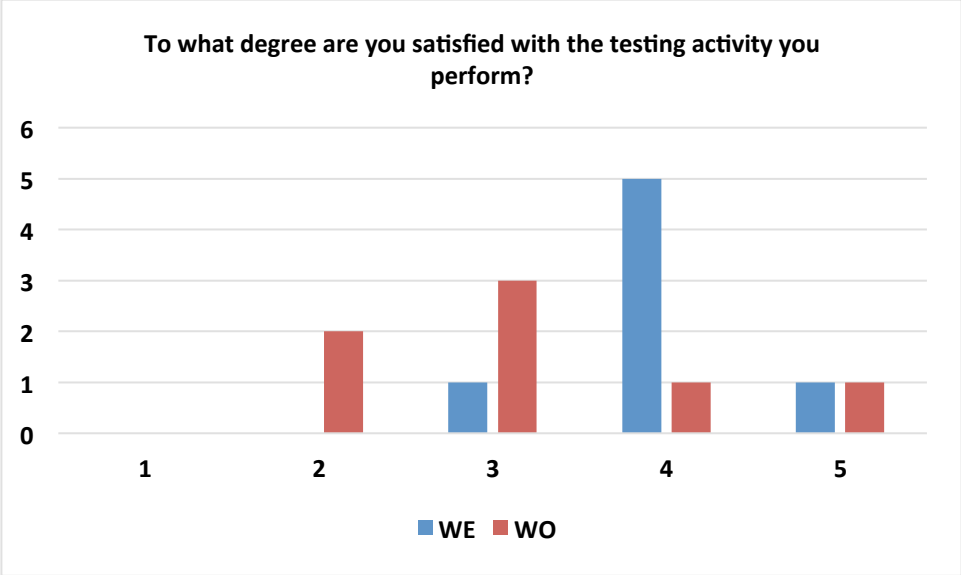

Figure 28. Satisfaction with testing activity performed

- The collaboration with co-workers (Arithmetic mean 4.6 and 4.57 for WE and WO respectively);

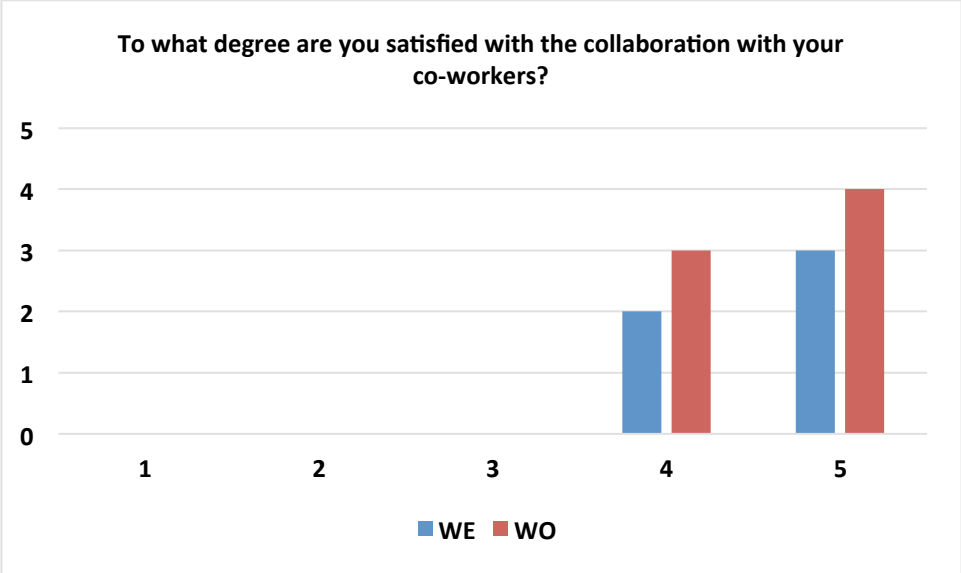

Figure 29. Satisfaction with collaboration with co-workers

- The performance of the results of the testing activity (Arithmetic mean 3.4 and 3.33 for WE and WO respectively);

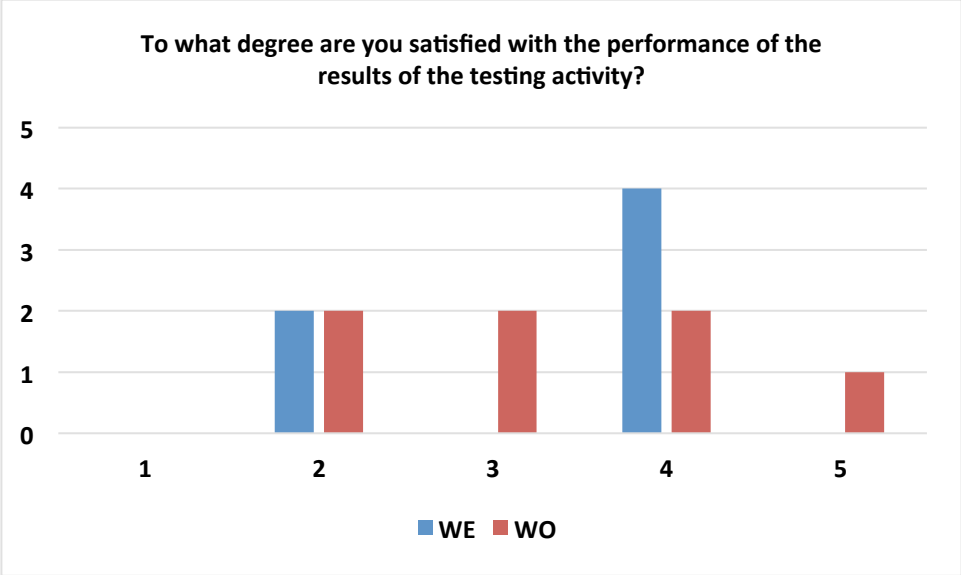

Figure 30. Satisfaction with performance of the results of the testing activity

- The productivity in performing testing job (Arithmetic mean 3.4 and 3.33 for WE and WO respectively);

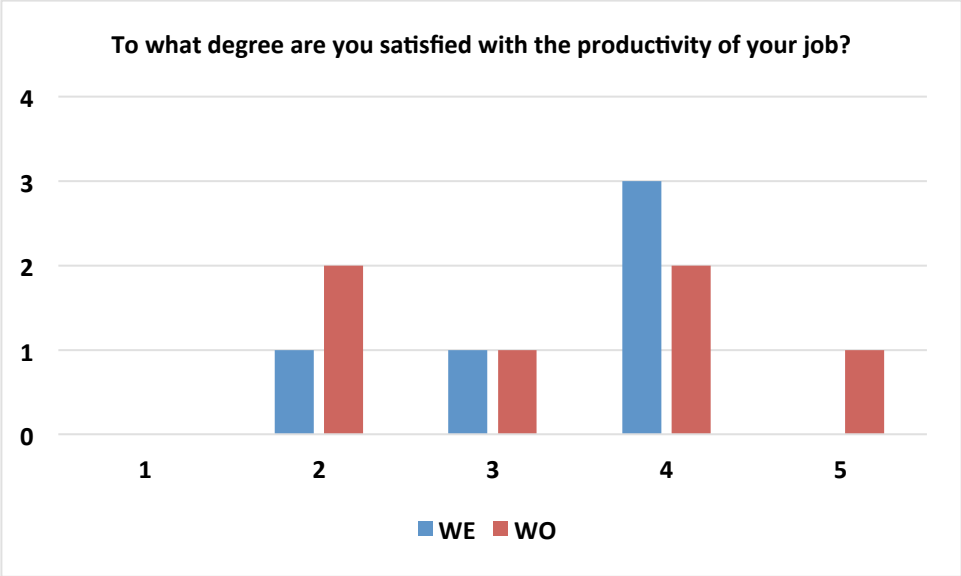

Figure 31. Satisfaction with productivity in performing testing job

- The adopted test process (Arithmetic mean 3.67 and 2.71 for WE and WO respectively).

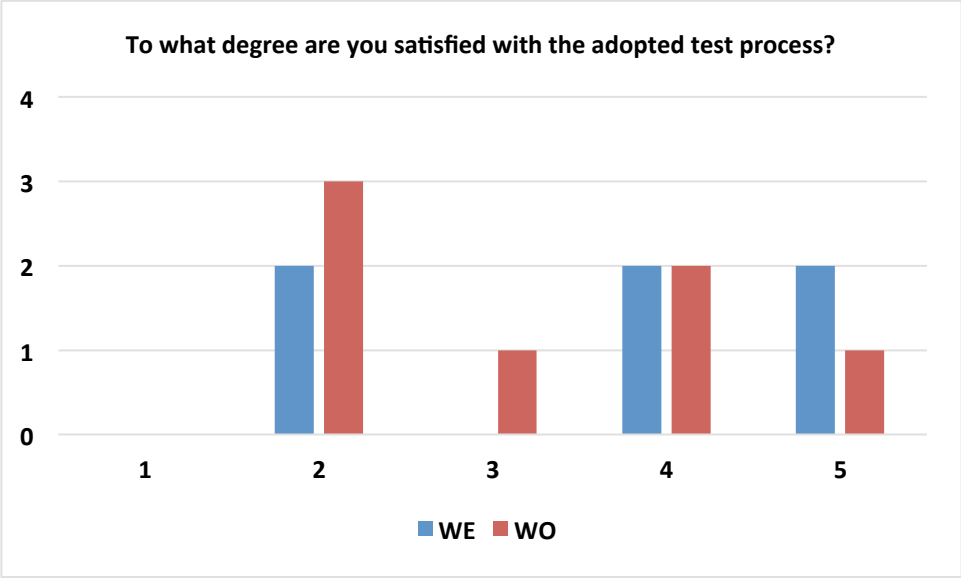

Figure 32. Satisfaction with the adopted test process

The data summarized in the previous figures evidenced very small differences between the WO and WE testing settings.

9.3 Efficacy

The third group of questions concerns the efficacy. In this case testers involved in the WE and WO stages considered the test environment adopted efficacious with a very small difference in favor of the WE one (Arithmetic mean 3.29 and 3.14 for the WE and WO respectively) as summarized in the following figure:

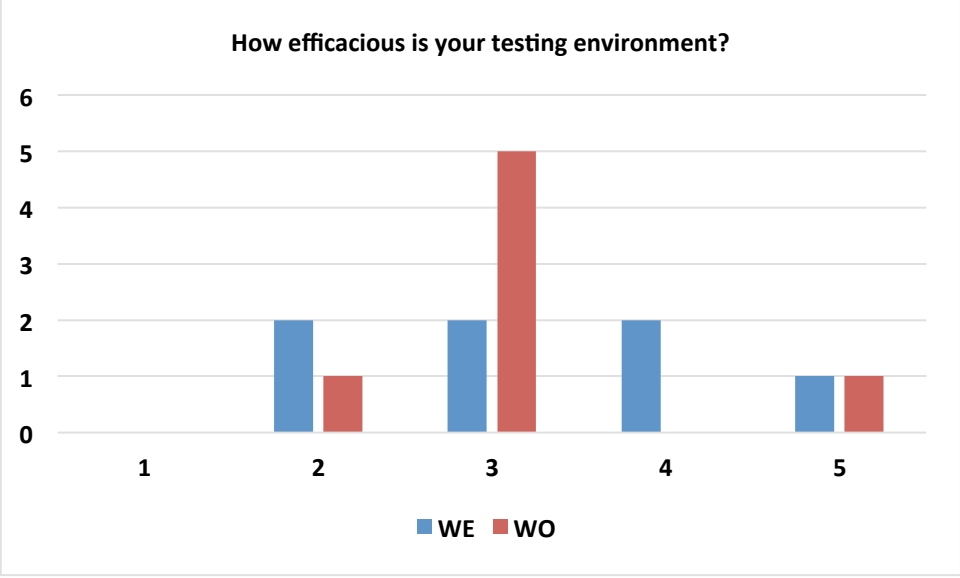

Figure 33. Efficacy

For better investigating this quality attribute we split the efficacy evaluation into two subgroups of questions: Efficiency and Effectiveness. In the remaining of this section details about the two subgroups are provided.

9.4 Efficiency

As a first investigation we asked the testers to evaluate how efficient was the testing environment, i.e. its ability to perform testing activity within the scheduled time. In this case testers involved in the WE experimentation were more satisfied than the WO (Arithmetic mean 3.57 and 2.71 for the WE and WO respectively) as summarized in the following figure:

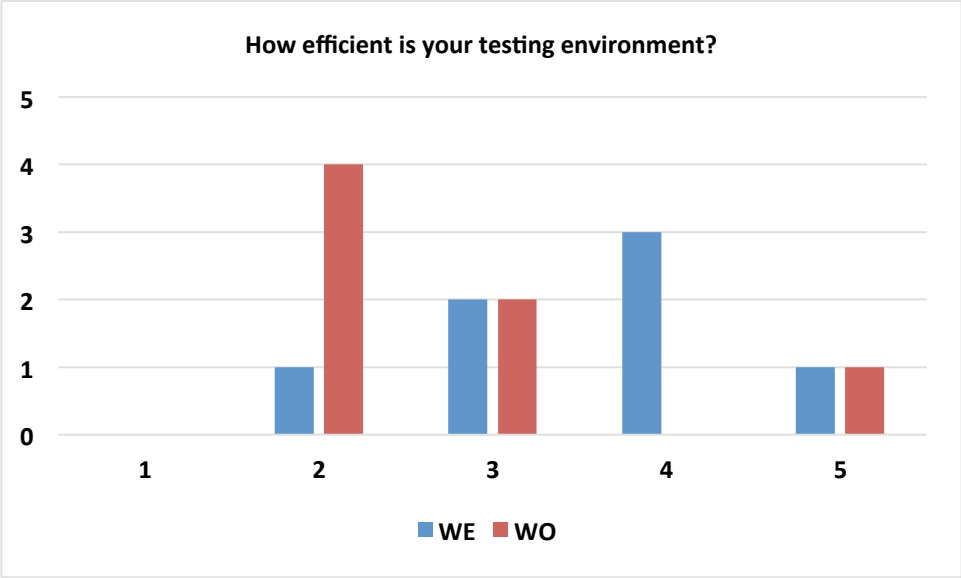

Figure 34. Efficiency

To refine the concept of efficiency an additional set of questions have been required. These focus on the help provided by the testing environment:

- To complete the testing activity within the scheduled time (Arithmetic mean 3.5 and 2.86 for WE and WO respectively);

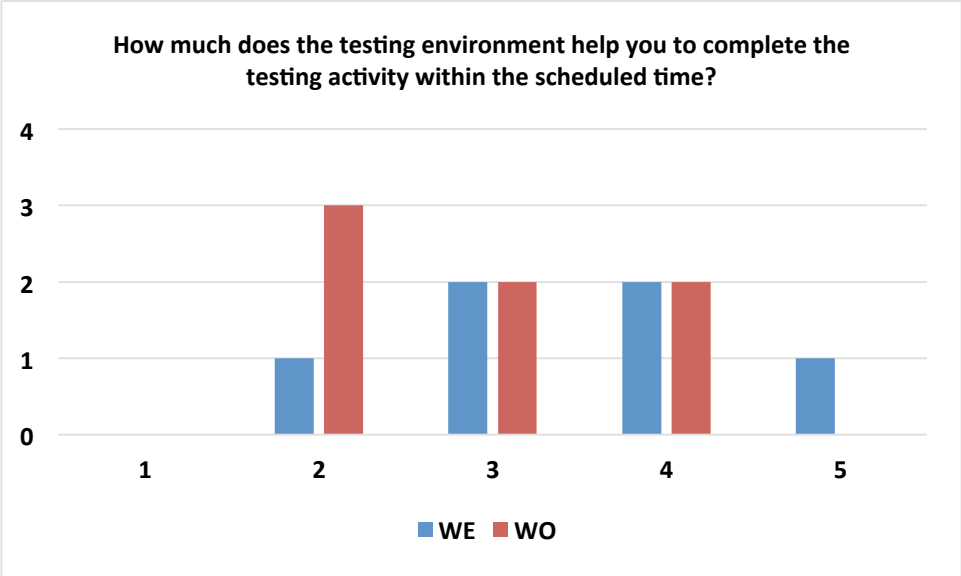

Figure 35. Efficiency in completing test activities in time

- To automated facilities (Arithmetic mean 3.67 and 2.75 for WE and WO respectively);

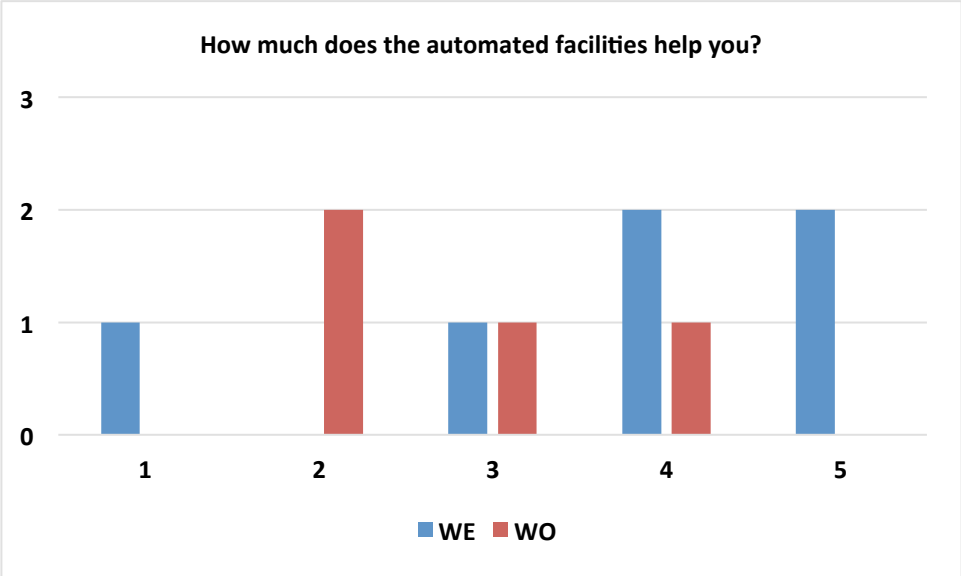

Figure 36. Efficiency in automated facilities

- To manage unexpected problems/failures (Arithmetic mean 3 and 2.71 for WE and WO respectively);

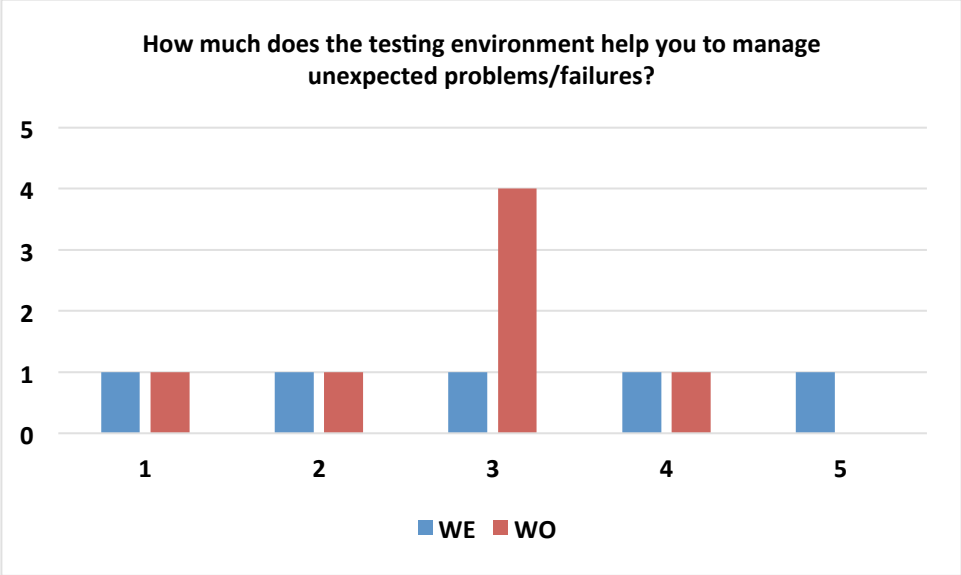

Figure 37. Efficiency in helping to manage unexpected problems/failures

- To identify the quality aspects more relevant (Arithmetic mean 3 for both the WE and WO cases).

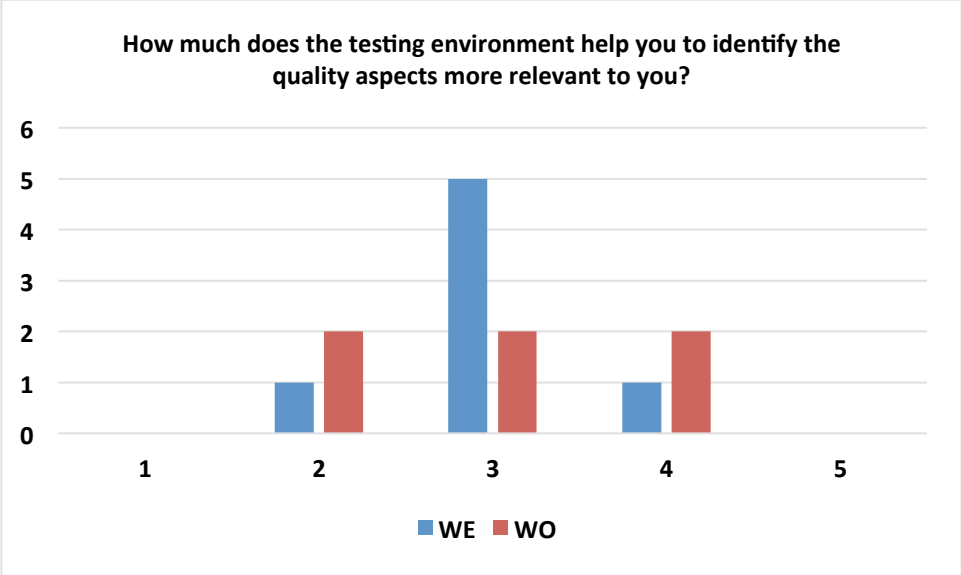

Figure 38. Efficiency in identifying quality aspects

The data, summarized in the previous figures, evidenced a situation in favor of ElasTest application.

9.5 Effectiveness

As a second investigation we asked the testers to evaluate how effective was the testing environment, i.e. its capability of producing a desired result. In this case testers

did not express any difference between WE and WO (Arithmetic mean 3.29 in both cases) as summarized in the following figure:

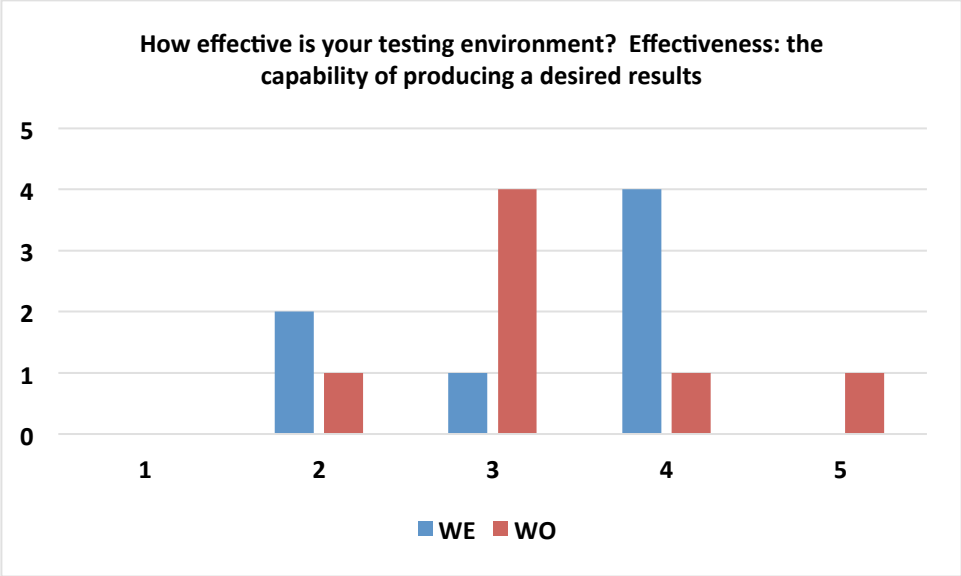

Figure 39. Effectiveness

To refine the concept of effectiveness an additional set of questions has been asked. These focus on the help provided by the testing environment:

- To monitor the testing activity (Arithmetic mean 4 and 3.71 for the WE and WO respectively);

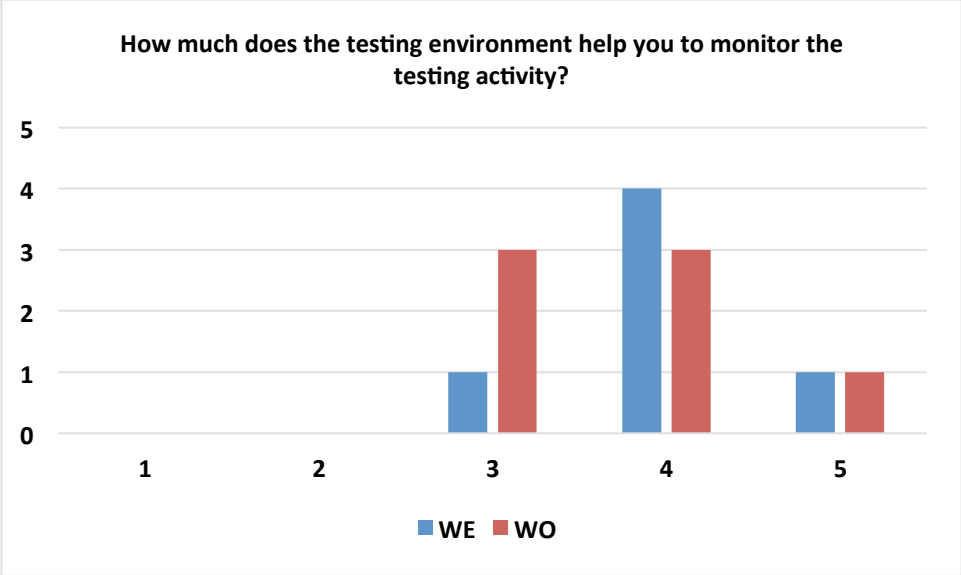

Figure 40. Effectiveness in helping to monitor testing activity

- To measure the quality aspects more relevant (Arithmetic mean 3.17 and 3.29 for WE and WO respectively);

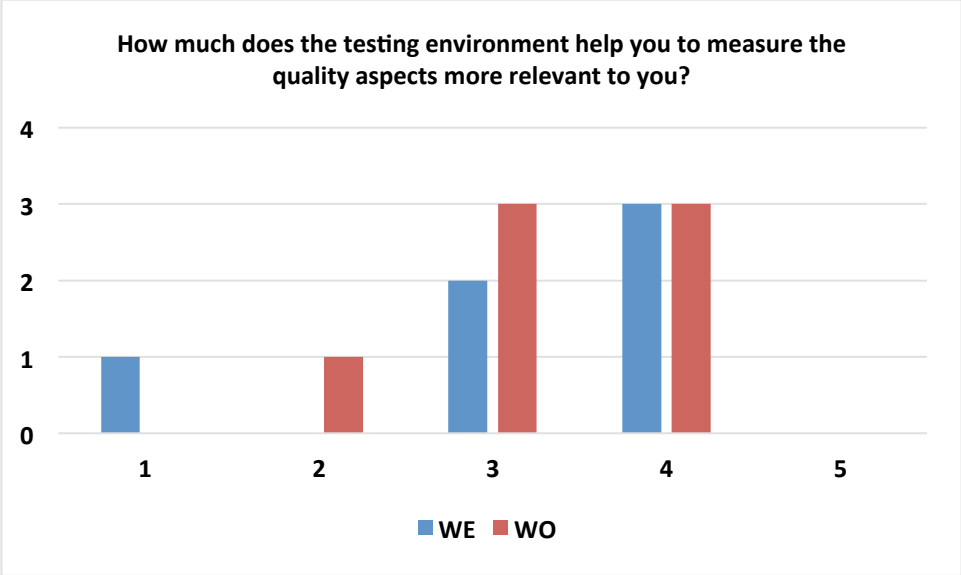

Figure 41. Effectiveness in helping measure quality aspects

- To cover all the prefixed testing features (Arithmetic mean 3.67 and 3.14 for WE and WO respectively);

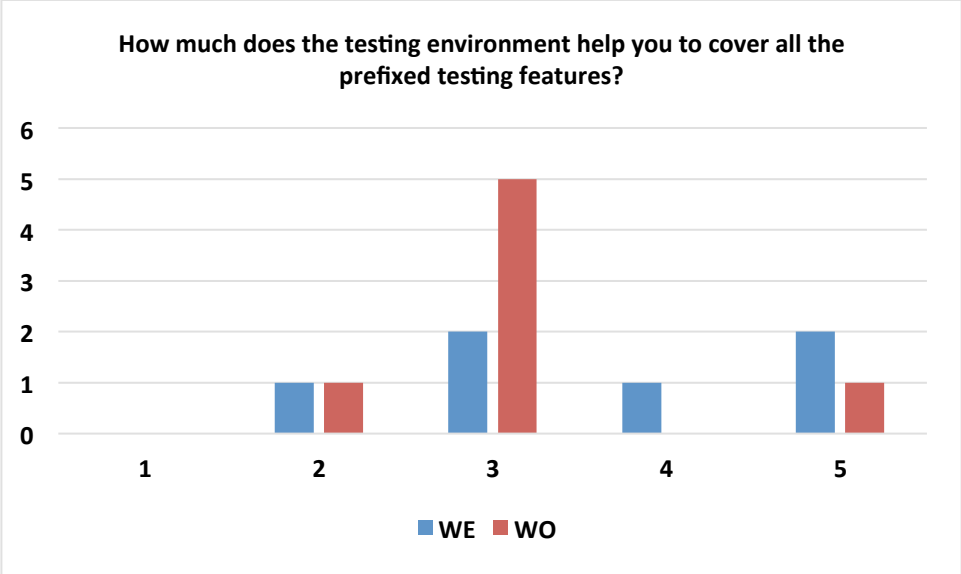

Figure 42. Effectiveness in helping to cover prefixed testing features

- To collect logs of the testing activity (Arithmetic mean 4.5 and 3 for WE and WO respectively).

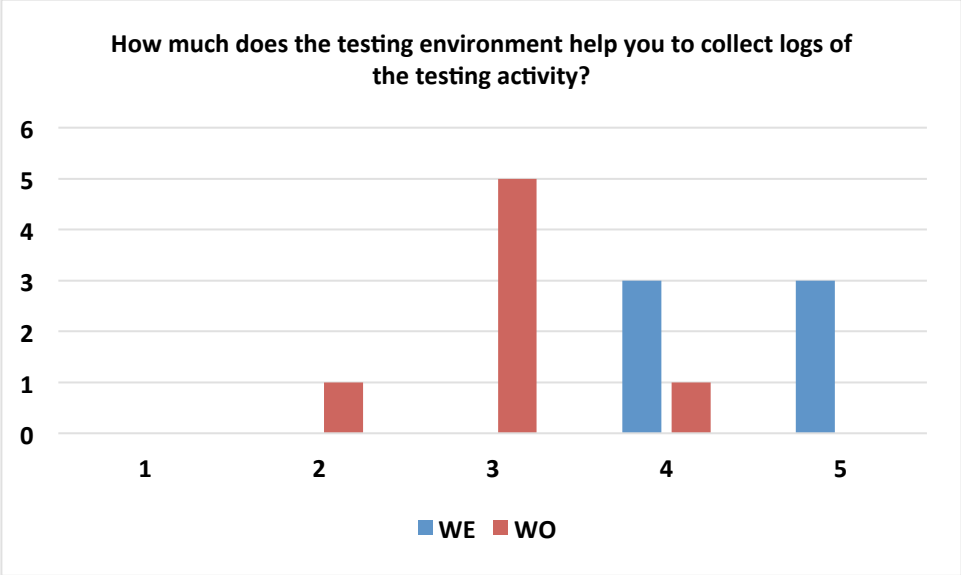

Figure 43. Effectiveness in helping to collect logs

The data, summarized in the previous figures, evidenced a situation in favor of ElasTest application.

9.6 Confidence

The forth group of questions concerns the confidence. In this case testers involved in the WE and WO stages are not completely confident with the environment adopted with a very small in favor of the WE one (Arithmetic mean 2.29 and 2.14 for WE and WO respectively) as summarized in the following figure:

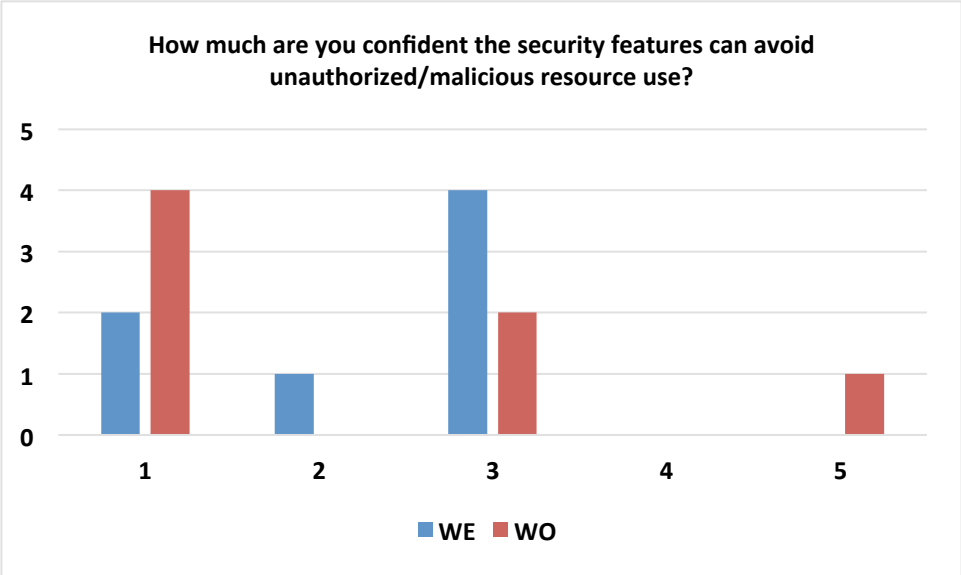

Figure 44. Confidence

To refine the concept of confidence an additional set of questions have been required. These focus on how the tester was confident that he security features can:

- Avoid unauthorized/malicious modifications (Arithmetic mean 2 for both WE and WO);

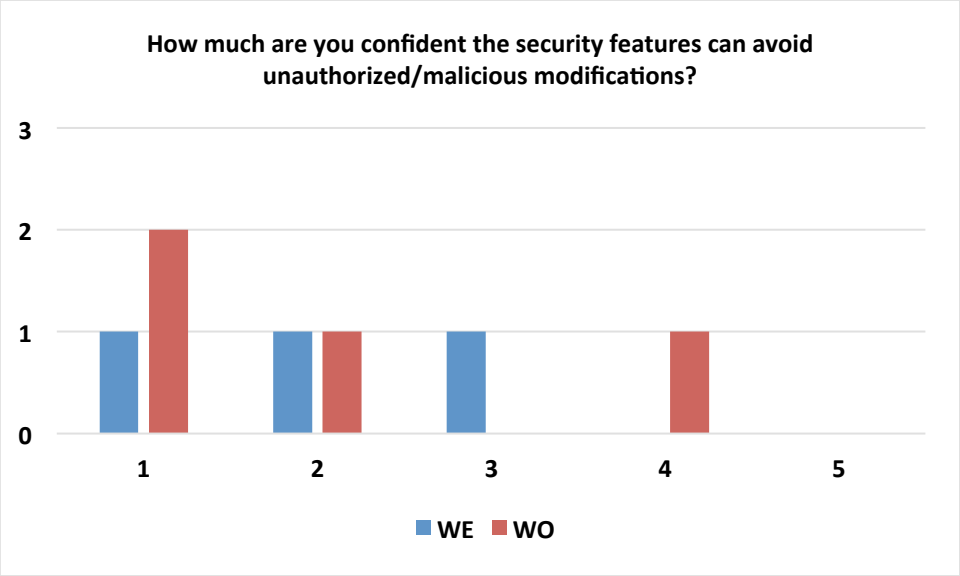

Figure 45. Confidence in avoiding unauthorized/malicious modification

- Avoid unauthorized/malicious destruction (Arithmetic mean 3 and 2.33 for WE and WO respectively);

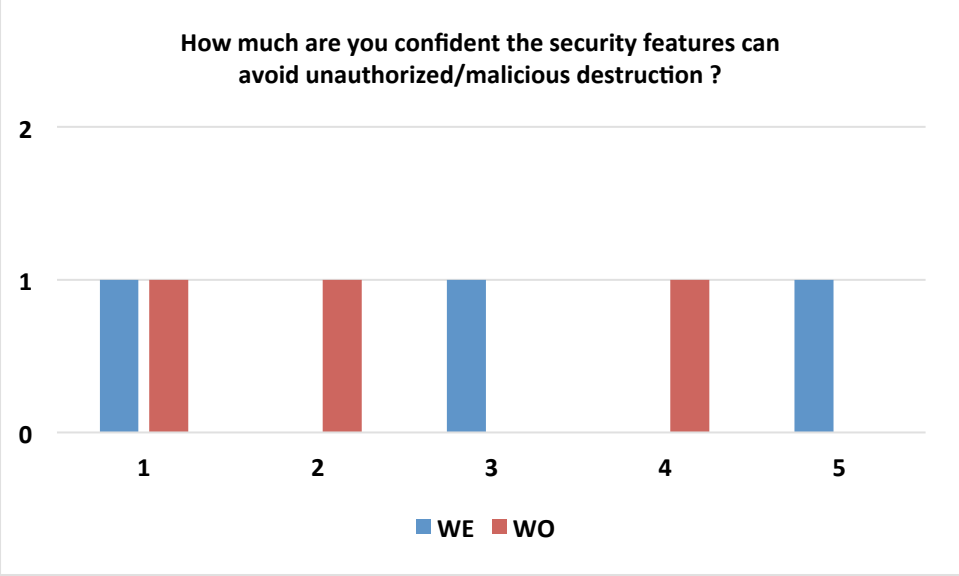

Figure 46. Confidence in avoiding unauthorized/malicious destruction

- Assure testing environment (Arithmetic mean 2 for both WE and WO respectively);

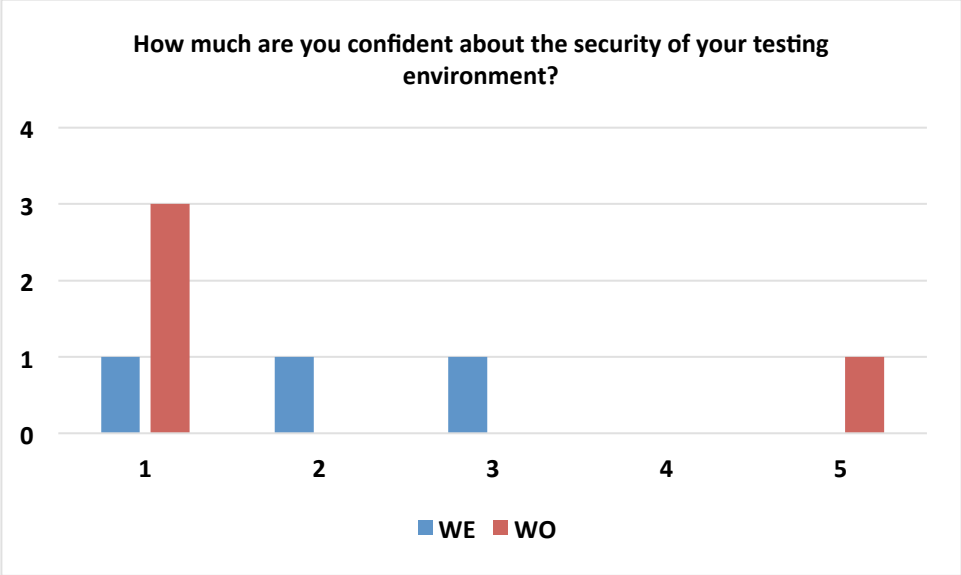

Figure 47. Confidence in the security of testing environment

- Avoid unauthorized/malicious disclosure (Arithmetic mean 3 and 2 for WE and WO respectively).

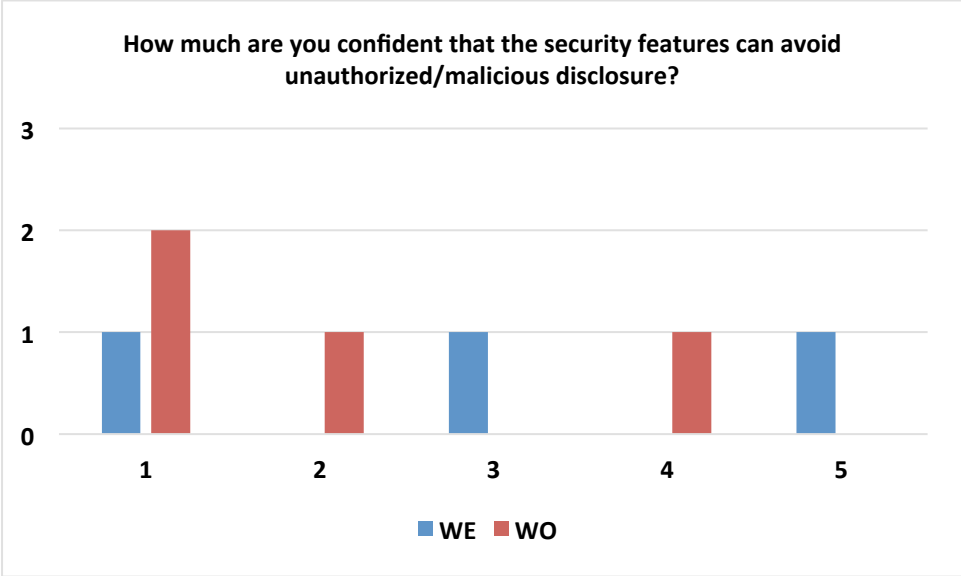

Figure 48. Confidence in avoiding unauthorized/malicious disclosure

The data, summarized in the previous figures, evidenced an overall necessity to improve the confidence of the testing environment even if the situation seems slightly better for the ElasTest application.

9.7 Usefulness

The last group of questions concerns the usefulness. In this case testers involved in the WE and WO stages are quite satisfied about the usefulness of the testing

environment adopted, and preferences are very small in favor of the WE one as summarized in the following figure:

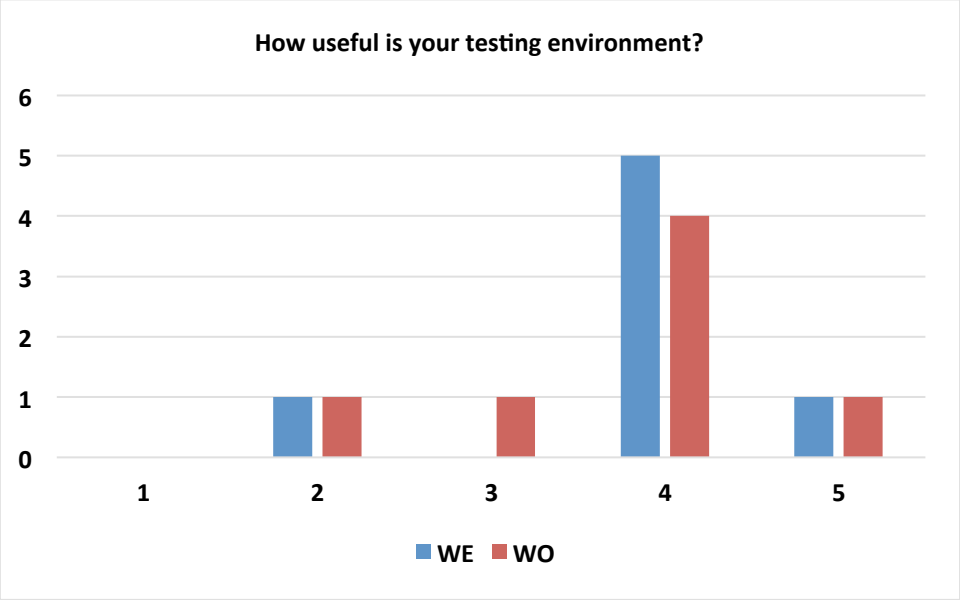

Figure 49. Usefulness

To refine the concept of usefulness an additional set of questions have been required. These focus on:

- The frequency in the usage (Arithmetic mean 5 and 4.14 for WE and WO respectively);

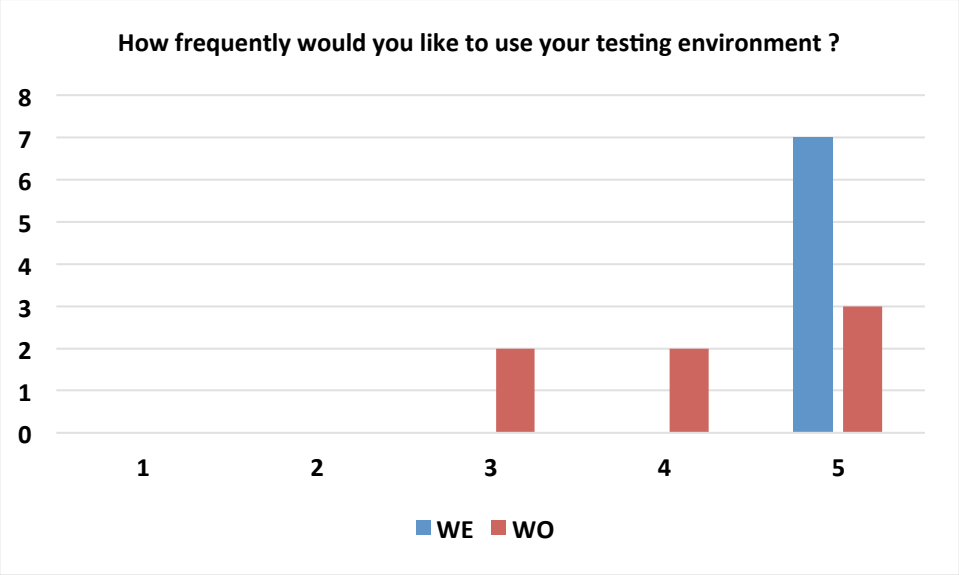

Figure 50. Frequency on usage of the testing environment

- The usage simplicity (Arithmetic mean 4.29 and 2.71 for WE and WO respectively);

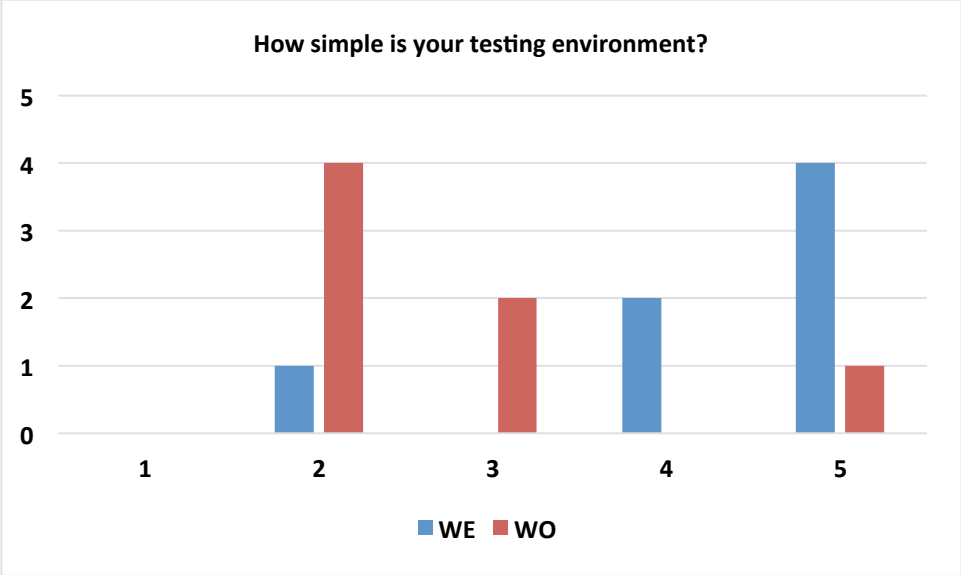

Figure 51. Usage simplicity

- Easy to use degree (Arithmetic mean 4.56 and 2.86 for WE and WO respectively);

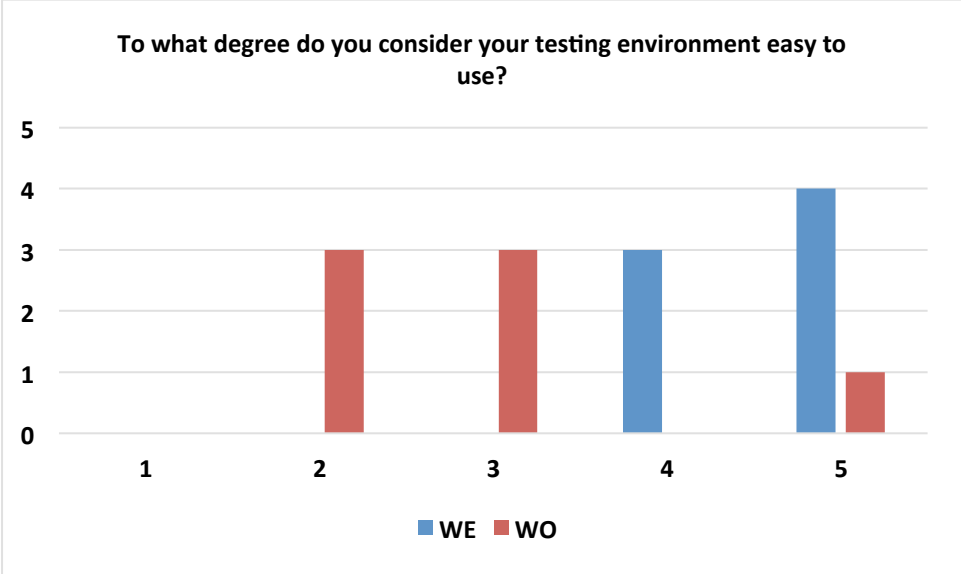

Figure 52. Easy to use degree

- Ability to use the testing environment without the support of a technical expert (Arithmetic mean 3.86 and 2.71 for the WE and WO respectively);

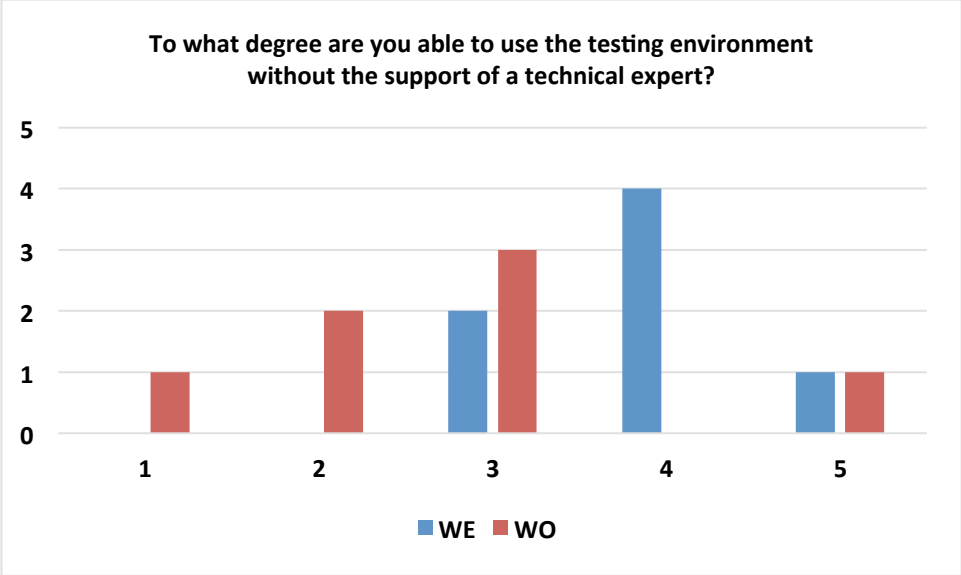

Figure 53. Ability to use without support

- Degree of integration of the different testing facilities (Arithmetic mean 3.57 and 2.86 for WE and WO respectively);

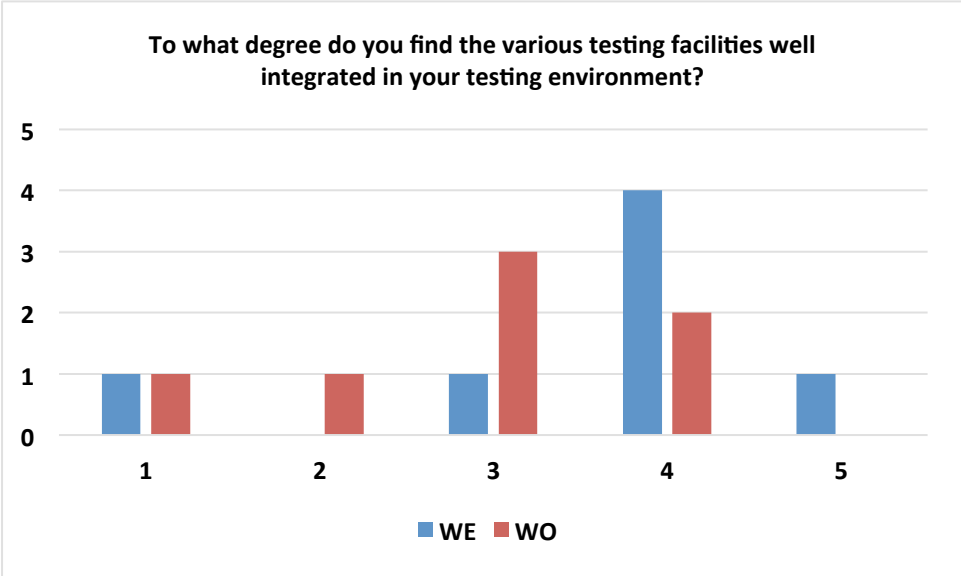

Figure 54. Degree of integration

- Degree in which the testing environment can drive the testing activity (Arithmetic mean 4.5 and 3.33 for WE and WO respectively);

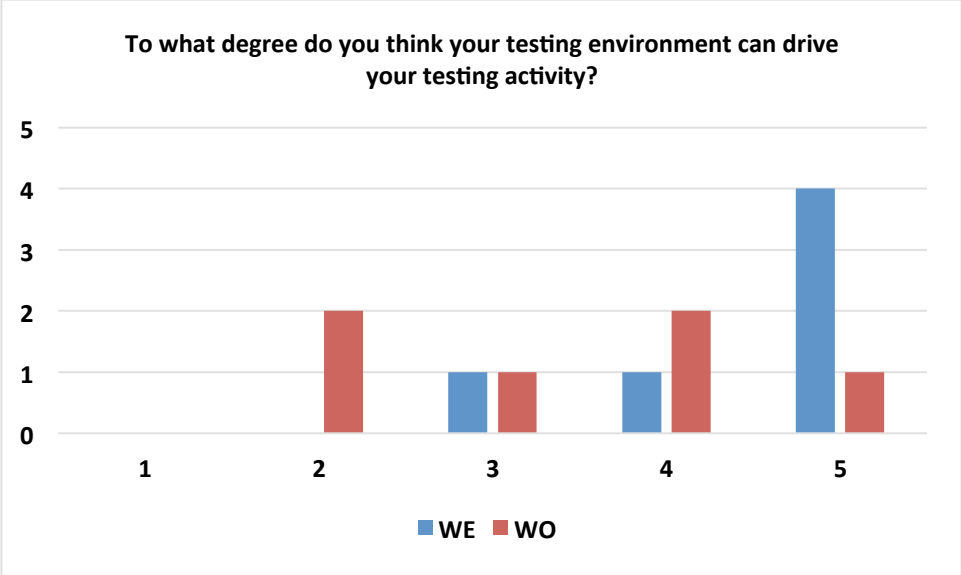

Figure 55. Testing drive testing activity

- Degree in which the testing environment is intuitive (Arithmetic mean 4.17 and 2.71 for WE and WO respectively).

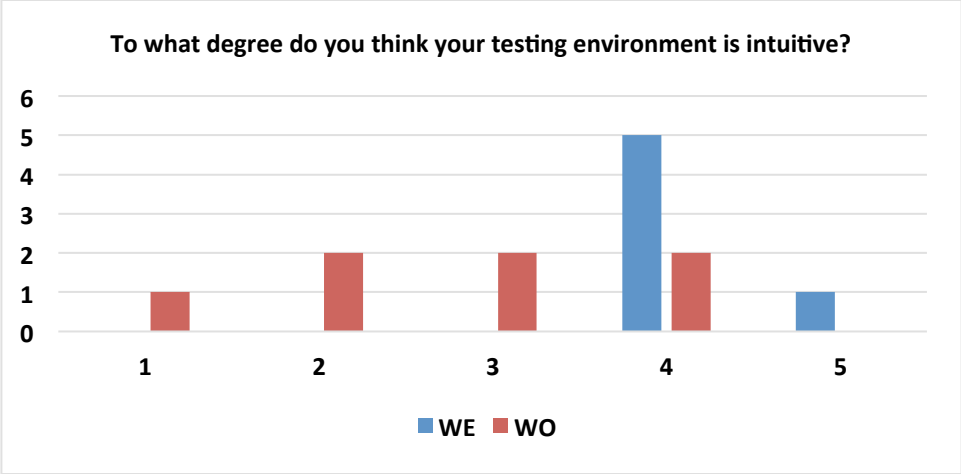

Figure 56. Intuitiveness

The data, summarized in the previous figures, evidenced that the testing environment provided by ElasTest has a better evaluation than the standard one.

10 Conclusions and future work

In this deliverable we reported the activity performed within WP7 while we are at half of the project life-cycle (i.e., M18). Precisely, in line with the project DoA, we presented:

- the strategy devised for the validation of the ElasTest platform, which includes three different kind of studies (Comparative Case Study, Quasi-experiment, Empirical Survey)
- the four vertical demonstrators spanning over four different domains, namely: web applications (Atos), 5G networks (FOKUS), WebRTC (NaevaTec), and IOT applications (TUB).
- the results achieved so far along data collection and a series of pilot studies.

Indeed, while the presented strategy can be considered final, for an actual validation we need to wait until the next release of this deliverable at M36, because the version of the ElasTest platform that we could use is of course not complete (it refers to Release R3 released at M12) and cannot obviously provide all expected benefits.

Nevertheless, as reported in detail in the document across the many included figures and tables, we could already perform with all four demonstrators a series of observations and studies, and we got from all of them both favorable and unfavorable results. We consider **all of them** useful and positive. In particular, we are learning from less favorable results how ElasTest can be improved in future releases.

It was our intent to establish a collaborative attitude among the four involved partners to make all efforts to support a rigorous and continuous data collection process all along the ElasTest project. We wanted to prevent the risk to consider project validation as an easy and last-minute activity because we are aware (also from own experience in other EU projects) that if validation is not built-in since the beginning and considered very important by the consortium, it will not succeed. In this sense, we would like to report the high enthusiasm and extremely responsive work from the partners directly involved in the demonstrators, as well as the constant support and great consideration from the project management and developers.

Of special note are a couple of considerations: concerning the CCS studies, we were expecting to get from the partners their historical data to be then used as a baseline. However, as described in the devoted chapters, some of the chosen demonstrators (FullTeaching and OpenIoTFog) are relatively younger applications for which a procedure of metrics collection was not in place. In such cases, we set up the process of data collection during the project purposely for the task of validation. The issue is that the data may not be as stable as a reference baseline should be. On the other side, it is positive that as a side-effect of WP7 all partners are now rigorously performing process measuring.

Concerning the QEs, we were prepared to have to hardly convince the partners to devote double effort to test the same application (WO and WE). In our experience this is hardly agreed in practice because very costly, in fact actual experiments are usually reported from academic studies, mostly involving students. This was not the case here: while the actual QEs are planned after Release 5, all the four partners easily agreed to

perform a pilot QE with two groups of testers. The reality check here is that in most cases (except for the FOKUS QE), the WO and WE "groups" actually correspond to 1 tester, and due to available resources within the project we don't expect that this may improve much in next iterations.

The future directions of work, according to the WP7 schedule (see Figure 1), include 3 further iterations of the QE, one round of data collection for the CCS, and two stages of ES. The results will be illustrated in D7.2, while the four developed demonstrators will be included in D7.3 and D7.4.

11 References

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

[2] Bertolino, A., 2007, May. Software testing research: Achievements, challenges, dreams. In *2007 Future of Software Engineering* (pp. 85-103). IEEE Computer Society.

[3] Apache 2.0 license terms. https://www.apache.org/licenses/LICENSE-2.0. Accessed on 07 March 2017.

[4] Grant Agreement number: 731535 - ELASTEST - H2020-ICT-2016-2017/H2020-ICT-2016-1. EUROPEAN COMMISSION. Communications Networks, Content and Technology. 11 November 2016.

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

Appendix: Tester survey QE

There are 59 questions in this survey

Agreement

Project information

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

Archiving data

Please enter your name *

Please write your answer here:

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/researchers 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 collaboration with your co-workers?

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

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

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?

(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

Effectiveness

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?

(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 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 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 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

How frequently would you like to use your testing environment?

Please estimate the question concerning your experience.

If not applicable, leave the scale empty.

Scale: 1 never - 3 neutral - 5 very frequent

Please choose **only one** of the following:

✖1 ✖2 ✖3 ✖4 ✖5

How simple is your testing environment?

Please estimate the question concerning 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 consider your testing environment easy to use?

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 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

Thank you for completing this survey.