

D2.4

Version	1.1
Author	CNR
Dissemination	PUBLIC
Date	31-12-2019
Status	FINAL



D2.4 SotA revision document v2

Project acronym	ELASTEST
Project title	ElasTest: an elastic platform for testing complex distributed large software systems
Project duration	01-01-2017 to 31-12-2019
Project type	H2020-ICT-2016-1. Software Technologies
Project reference	731535
Project website	http://elastest.eu/
Work package	WP2 User-centered agile conception
WP leader	TUB
Deliverable nature	Report
Lead editor	Francesca Lonetti
Planned delivery date	31-12-2019
Actual delivery date	30-12-2019
Keywords	Open source software, cloud computing, software engineering, operating systems, computer languages, software design & development



Funded by the European Union

License

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.

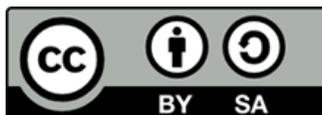
Notices:

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
Malena Donato Cohen	ATOS
Guglielmo De Angelis	CNR
Felicita Di Giandomenico	CNR
Emilia García	ATOS
Francisco Gortázar	URJC
Varun Gowtham	TUB
Piyush Harsh	ZHAW
Guiomar Tuñón Hita	NAEVATEC
Eduardo de la Iglesia Monje	NAEVATEC
Eduardo Jiménez	URJC
Magda Kacmajor	IBM
Francesca Lonetti	CNR
Kimon Moschandreou	REL
Michael Pauls	TUB
Tran Quang	TUB
Christos Roupas	REL
Avinash Sudhodanan	IMDEA Soft

Version history

Version	Date	Author(s)	Description of changes
0.1	10/01/2019	Francesca Lonetti Antonia Bertolino Guglielmo De Angelis	Initial ToC
0.2	26/07/2019	Francesca Lonetti Guglielmo De Angelis Antonia Bertolino	First draft version of Grey Literature
0.3	11/11/2019	Malena Donato Cohen	First draft version of Market Analysis
0.4	20/11/2019	ALL	First draft version of Technical SotA
0.5	22/11/2019	Guglielmo De Angelis	First draft version of projects related to ElasTest
0.6	25/11/2019	Francesca Lonetti	Executive Summary, Introduction, Conclusion
0.7	27/11/2019	Francesca Lonetti Guglielmo De Angelis Antonia Bertolino	Final version of Grey Literature and Projects
0.8	28/11/2019	ALL	Final Version of Technical SoTA
0.9	29/11/2019	Malena Donato Cohen	Final version of Market Analysis
0.10	6/12/2019	Francesca Lonetti Guglielmo De Angelis	Revision of the document
0.11	19/12/2019	Christos Roupas	Internal review
0.12	23/12/2019	Francesca Lonetti	Revised document following internal review

Table of contents

1	Executive summary.....	15
2	Introduction.....	16
2.1	Objectives	16
2.2	Structure of the document	17
3	Literature Review on Cloud Testing	17
3.1	Scientific Literature review on cloud testing	17
3.2	Grey Literature review on cloud testing.....	17
3.2.1	Methodology	18
	Planning the Review	18
	Sources for the Review	18
	Automated Search.....	19
	Selection based on inclusion/exclusion criteria	19
	Selection based on quality assessment.....	20
3.2.2	The classification framework.....	23
3.2.3	Summary of Results	24
4	Technical Analysis of SoTA.....	31
4.1	Continuous Integration.....	32
4.1.1	Continuous Integration Server - Baseline and comparative analysis.....	32
	AWS_CodePipeline	34
	Jenkins X	34
	Semaphore.....	35
	Team Foundation Server	35
	TeamCity.....	35
4.1.2	Continuous integration - Artifact distribution - Baseline and comparative analysis	36
4.1.3	Progress within ElasTest.....	37
4.2	Performance Testing.....	38
4.2.1	Baseline and comparative analysis.....	38
	StormRunner Load.....	39
	Load Impact	39
	Loader	39
	OctoPerf.....	40

LoadNinja.....	40
Gatling.....	40
Tsunami.....	40
Locust.....	40
4.2.2 Progress within ElasTest.....	41
4.3 Security testing.....	41
4.3.1 Baseline and comparative analysis.....	41
Privacy.net.....	42
Extensions.inrialpes.fr.....	42
4.3.2 Progress within ElasTest.....	42
4.4 Monitoring.....	43
4.4.1 Baseline and comparative analysis.....	43
InfluxData TICK.....	44
Zabbix.....	45
Datadog.....	45
4.4.2 Progress within ElasTest.....	46
4.5 GUI automation and impersonation (IoT Testing).....	47
4.5.1 Baseline and comparative analysis.....	47
Patriot IoT Testing Framework.....	47
Eggplant.....	47
IoT Emulator.....	48
Simple IoT Simulator.....	48
4.5.2 Progress within ElasTest.....	50
4.6 Cloud Instrumentation.....	51
4.6.1 Baseline and comparative analysis.....	51
Marathon.....	54
Google Kubernetes Engine (GKE).....	54
MaaS.....	55
Opensource MANO (OSM).....	55
Ansible.....	56
Chef habitat.....	56
Terraform.....	56
BOSH.....	57

4.6.2	Progress within ElasTest	59
4.7	Dashboard Management	59
4.7.1	Baseline and comparative analysis.....	59
	Prometheus/Grafana.....	59
	Honeycomb.....	60
4.7.2	Progress within ElasTest	61
4.8	WebRTC Testing.....	61
4.8.1	Baseline and comparative analysis.....	61
4.8.2	Progress within ElasTest	62
4.9	Cross-browser Testing	62
4.9.1	Baseline and comparative analysis.....	62
4.9.2	Progress within ElasTest	63
4.10	Mobile Testing.....	64
4.10.1	Baseline and comparative analysis.....	64
4.10.2	Progress within ElasTest	66
4.11	Cognitive Q&A Systems	66
4.11.1	Baseline and comparative analysis.....	67
4.11.2	Progress within ElasTest	69
5	Summary of ElasTest Outcomes, Progresses and Benefits	69
5.1	ElasTest Outcomes and Progresses	69
5.2	ElasTest Main Benefits.....	72
6	Research projects related to ElasTest: an overview.....	74
6.1	CodeSan	74
6.2	GAMMA	74
6.3	SENECA.....	75
6.4	TEFIS.....	75
6.5	SWITCH.....	76
6.6	STAMP.....	76
6.7	ADVANCE	77
6.8	PRECRIME.....	77
6.9	NOBUGS	78
6.10	FI-WARE.....	78
6.11	TESTOMAT.....	79

6.12	GAUSS.....	79
7	Market Analysis	80
7.1	IT budget allocated to testing.....	81
7.2	ICT market.....	83
7.3	Cloud Market	84
7.4	Devops	86
7.5	Test automation.....	86
7.6	Expectations from the market	88
7.7	Comparison of ElasTest with trendy tools.....	90
7.8	Main stream technologies related to ElasTest	96
7.8.1	Kubernetes.....	96
7.8.2	Jenkins and TestLink	96
7.9	Market Perspective for ElasTest	96
8	Conclusions.....	97
9	References.....	98

List of figures

Figure 1: Triangular Isosceles Membership Functions	21
Figure 2: Cloud Testing Framework.....	23
Figure 3: Breakdown of the Primary Studies by Category	25
Figure 4: Number of Publications by Year	25
Figure 5: Number of Publications by Year and Area	26
Figure 6: Distribution of the Papers by Area	26
Figure 7: Breakdown of the Primary Study in Test Perspective	27
Figure 8: Breakdown of the Primary Study in Test Design	28
Figure 9: Breakdown of the Primary Study in Test Objective	29
Figure 10: Breakdown of the Primary Study in Test Execution	29
Figure 11: Breakdown of the Primary Study in Test Evaluation.....	30
Figure 12: Breakdown of the Primary Study in Test Domain	31
Figure 13: Jenkins X architecture.....	34
Figure 14: InfluxData TICK architecture.....	45
Figure 15: Number of developers in the world	81
Figure 16: Portion of the testing budget allocated to QA testing (including testing process, tools and resources).....	82
Figure 17: Factors influencing the portion of IT budget dedicated to testing	83
Figure 18: Worldwide ICT Spending 2016-2020	84
Figure 19: Cloud market prediction.....	85
Figure 20: Functional Testing Trend	87
Figure 21: Main challenges faced while achieving test automation	88
Figure 22: Artificial intelligence and machine learning projects for 2019.	89

List of tables

Table 1: Inclusion and Exclusion Criteria	20
Table 2: Indicators driving the quality score procedure	20
Table 3: Continuous integration server tools	33
Table 4: Semaphore characteristics	35
Table 5: Comparison of CI server tools.....	36
Table 6: CI- Artifact distribution tools	37
Table 7: CI - Artifact distribution tools: Main Features	37
Table 8: Performance Testing Tools	39
Table 9: Comparison of performance tools.....	41
Table 10: Security Testing Tools	42
Table 11: Comparison of Security Testing Tools	42
Table 12: Monitoring Tools	44
Table 13: Comparison of monitoring tools.....	46
Table 14: IoT Testing tools.....	49
Table 15: Comparison of IoT Testing Tools	50
Table 16: Cloud Instrumentation Tools	54
Table 17: Comparison of Cloud Instrumentation Tools	58
Table 18: Dashboard Management Tools	61
Table 19: Comparison of Dashboard Management Tools.....	61
Table 20: WebRTC Testing Tools	62
Table 21: KITE features	62
Table 22: Cross Browser Testing Tools	63
Table 23: Browsersync features	63
Table 24: Mobile Testing Tools.....	66
Table 25: Comparison of Mobile testing Tools.....	66
Table 26: Cognitive Q&A Tools	68
Table 27: Comparison of Cognitive Q&A Tools	69
Table 28: Summary of ElasTest Progresses and Outcomes.....	72
Table 29: Spending Forecast for technology (Billions of U.S. Dollars)	84
Table 30: Public Cloud Service Revenue Forecast (Billions of U.S. Dollars)	85
Table 31: ElasTest vs key tools	95

Glossary of acronyms

Acronym	Description
<i>API (Application Programming interface)</i>	It refers to an interface or communication protocol aiming to simplify the implementation and maintenance of software
<i>AST (Abstract Syntax Tree)</i>	It refers to a tree representation of the abstract syntactic structure of source code written in a programming language
<i>CI (Continuous Integration)</i>	It refers to a software development practice
<i>CLI (Command Line Interface)</i>	It refers to a console or text based representation in which the user types the commands to interact with an operating system or device
<i>CNCF (Cloud Native Computing Foundation)</i>	It refers to communities supporting the growth and health of cloud native open source software
<i>CPS (Cyber-Physical Systems)</i>	They refer to systems integrating sensing, computation, control and networking.
<i>DSL (Domain Specific Language)</i>	It refers to a computer language specialized to a particular application domain
<i>EDM (ElasTest Data Manager)</i>	A core component of ElasTest
<i>EDS (ElasTest Device Emulator Service)</i>	A test support service provided by ElasTest
<i>EMP (ElasTest Platform Monitoring)</i>	A core component of ElasTest
<i>EMS (ElasTest Monitoring Service)</i>	A test support service provided by ElasTest
<i>EOE (ElasTest Orchestration Engine)</i>	A test engine provided by ElasTest
<i>EPM (ElasTest Platform Manager)</i>	A core component of ElasTest
<i>ERE (ElasTest Recommendation Engine)</i>	A test engine provided by ElasTest
<i>ESS (ElasTest Security Service)</i>	A test support service provided by ElasTest
<i>ETM (ElasTest Tests Manager)</i>	A core component of ElasTest

<i>EUS (ElasTest User Impersonation Service)</i>	A test support service provided by ElasTest
<i>GUI (Graphical User Interface)</i>	It refers to a type of user interface that allows users to interact with electronic devices through graphical icons and visual indicators
<i>IaaS (Infrastructure as a Service), PaaS (Platform as a Service) and SaaS (Software as a Service) CaaS (Container-as-a-Service)</i>	This refers to different models of exposing cloud capabilities and services to third parties
<i>ICT (Information and communications technology)</i>	It refers to all devices, networking components, applications and systems that allow people and organizations to interact in the digital world
<i>IoT (Internet of Things)</i>	It refers to a system of interrelated computing devices, and digital objects that are provided with unique identifiers and the ability to transfer data over a network without requiring human-to-human or human-to-computer interaction
<i>IT (Information Technology)</i>	It refers to the use of computers to store, retrieve, transmit, and manipulate data
<i>JSON (JavaScript Object Notation)</i>	It refers to an open-standard file format and data interchange format
<i>LDAP (Lightweight Directory Access Protocol)</i>	It is an open and cross platform protocol used for directory services authentication
<i>LGs (Load Generators)</i>	It refers to a system which is used to simulate load for performance testing
<i>LVM (Logical Volume Management)</i>	It refers to an advanced system of managing logical volumes or filesystems
<i>NFV (Network functions virtualization)</i>	It allows to abstract network functions, allowing them to be installed, controlled, and manipulated by software running on standardized compute nodes
<i>NLP (Natural Language Processing)</i>	It refers to a branch of artificial intelligence dealing with the interaction between computers and humans using

	the natural language
<i>OSS (Open Source Software)</i>	It refers to a type of computer software whose source code is released under a license in which the copyright holder grants users the rights to study, change, and distribute the software to anyone and for any purpose
<i>QA (Quality Assurance)</i>	It refers to a set of activities for monitoring the software engineering processes and methods used to ensure quality
<i>QoE (Quality of Experience)</i>	It is a measure of the overall level of customer satisfaction
<i>QoS (Quality of Service)</i>	It refers to non functional attributes of the system
<i>RAID (Redundant Arrays of Independent Disks)</i>	It is a data storage virtualization technology that combines multiple physical disk drive components into one or more logical units for the purposes of data redundancy as well as performance improvement
<i>RDP (Remote Desktop Protocol)</i>	It is a proprietary protocol which provides a user with a graphical interface to connect to another computer over a network connection
<i>REST (Representational State Transfer)</i>	It refers to an architectural style for providing standards between computer systems on the web, making it easier for systems to communicate with each other
<i>SAML (Security Assertion Markup Language)</i>	It is an XML-based framework for authentication and authorization
<i>SDK (Software Development Kit)</i>	It refers to a collection of software development tools in one installable package
<i>SiL (System in Large)</i>	A SiL is a large distributed system exposing applications and services involving complex architectures on highly interconnected and heterogeneous environments

<i>SOA (Service-Oriented Architecture)</i>	It refers to an architectural style where applications are essentially a collection of services that communicate with each other
<i>SotA</i>	State of the Art
<i>SUT (Software under Test)</i>	This refers to the software that a test is validating
<i>T-Job (Testing Job)</i>	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
<i>TiL (Test in the Large)</i>	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
<i>UI (User Interface)</i>	It represents the point of human-computer interaction and communication in a device
<i>VM (Virtual Machine)</i>	It refers to an emulation of a computer system
<i>VNC (Virtual Network Computing)</i>	It refers to a technology for remote desktop sharing and remote access on computer networks
<i>ZFS (Z File System)</i>	It is a combined file system and logical volume manager designed by Sun Microsystems

1 Executive summary

This deliverable presents the results of the scouting activity carried out in Task 2.1 of WP2 in the second reporting period (from M18 to M36). Specifically, on the one side this document provides an overview of the existing (scientific and grey) literature and technical solutions on the topics covered by the project. On the other side, it reports about an analysis of the market so that to highlight current ICT trends and potential opportunities for ElasTest platform. Both the technical survey and the market analysis in this document represent an update of the scouting performed during the first review period and already provided in D2.2 [7]. The survey on the (scientific and grey) literature in cloud testing concerns scouting activities started in the second reporting period.

The document includes four main parts. The first part (i.e., Section 3) presents an overview of the literature in cloud testing. Specifically, Section 3.1 discusses the activities performed while surveying the scientific literature in M18-36; while Section 3.2 focuses only on the grey literature. The aim of this last analysis is to complement and enhance the results of a systematic review of the scientific literature on the broad field of cloud testing in the last five years and a framework we propose for categorizing all relevant research in this context. As cloud testing is a practitioner-oriented and an application-oriented field, in this document we want mainly to classify in a systematic way also the knowledge coming from all the informal sources (such as blogs, videos, white papers and web-pages) that are different from the formal literature sources rigorously reported in academic settings that we addressed in [6].

The second part (Section 4 and Section 5) of the document includes the results of a technical analysis of the state of the art, aiming to collect the most relevant tools identified in this second review period as well as important updates of the tools already presented in D2.2 [7]. This part also includes main progresses and benefits of ElasTest. The analysis was performed in two main steps: a first overview was conducted in March 2019 and its update in September 2019. The survey is a collective work from the consortium, in which each partner was responsible for the technical analysis of one or more identified aspects according to its expertise. As in D2.2 [7], the tools have been classified according to the main technological areas that are of interest for ElasTest as specified also in the Description of the Action (DoA) [1] that are: Continuous Integration, Performance Testing, Security Testing, Monitoring, GUI Automation and Impersonation (IoT Testing), Cloud Instrumentation, Data Ingestion, Dashboard Management, WebRTC Testing, Cross-browser Testing, Test Execution & Visualization, Mobile Testing, Cognitive Q&A Systems, Test Orchestration, Test Management, Testing Framework and Virtualization. This technical analysis represents an update of the technical analysis of SotA presented in D2.2 [7]. Specifically, in Section 4 of this document we report only the technological areas in which new tools have been found or important updates of tools presented in D2.2 [7] have been identified in the second review period. For each of these technological areas, we present a short description of the tools, a comparison among them and the ElasTest progress in this second review period in this area with respect to the state of the art.

With respect to the technical analysis presented in D2.2 [7], no new tools or relevant updates of the existing ones have been identified in the technological areas of: Data Ingestion, Test Execution & Visualization, Test Management, Testing Framework, Test Orchestration and Virtualization. In Section 5 we present a summary of the main ElasTest outcomes and progresses for all the technological areas that are of interest for ElasTest as well as the main benefits of ElasTest.

The third part (Section 6) shows an overview of research projects covering one or more topics addressed in ElasTest. The aim is to identify common research topics and related technologies among ElasTest and the other ongoing or past projects.

Finally, the fourth part (Section 7) presents an overview of the most important IT market trends in this second review period and shows that the demand of competitive testing solutions as well as the growing investments in cloud testing and test automation will have a positive impact on the development of ElasTest.

2 Introduction

2.1 Objectives

ElasTest developed a comprehensive platform aimed at improving the efficiency and effectiveness of the testing process of large complex systems. The platform supports end-to-end testing in the cloud (TiC) and addresses many research challenges of the literature on cloud testing. After providing the results of a systematic review of the scientific literature in D2.2 [7] and in [6], in this document we present a systematic review of the grey literature related to topics of interest of ElasTest.

ElasTest is a quite ambitious project that aims at evolving current SotA in many technological domains. For the sake of simplicity, in this document we concentrate on the ones which have more relevance for the project. For this, we identified 17 technological areas, covering those specified in the DoA [1] that are: Continuous Integration, Performance Testing, Security Testing, Monitoring, GUI Automation and Impersonation (IoT Testing), Cloud Instrumentation, Data Ingestion, Dashboard Management, WebRTC Testing, Cross-browser Testing, Test Execution & Visualization, Mobile Testing, Cognitive Q&A Systems, Test Orchestration, Test Management, Testing Framework and Virtualization. For each technological area, we provide an update of the technical state of the art with respect to that of D2.2 [7], namely the most relevant tools, a comparison of these tools and the current technological progress of ElasTest with respect to the presented tools. Then, in Table 28 we provide a summary of the ElasTest outcomes and advancements in all the identified technological areas.

The research topics addressed into ElasTest are also covered by other past or current research projects. In this document, we provide an overview of European and national research projects related to ElasTest outlining the main potential synergies and common technological aspects. This overview is useful to foster future dissemination and exploitation of ElasTest results in the context of other research projects.

Finally, a goal of this document is to present an update of the state of the art about market trends with respect to that presented in D2.2 [7]. This document provides a quantitative and qualitative assessment of the market and gives an overview of market trends focusing on the IT areas in which ElasTest can later create impact.

2.2 Structure of the document

This document is structured as follows: in Section 3, we present an analysis of the literature about cloud testing. Then, in Section 4, we provide an analysis of the technical state of the art focusing on the technological areas that are of interest in ElasTest. In Section 5 we summarize the main ElasTest outcomes and progresses with respect to all the technological areas identified in the SotA. In Section 6, we present an overview of the research projects related to ElasTest. Section 7 shows how ElasTest is aligned with the markets trends in the areas in which ElasTest can have an impact. Finally, Section 8 draws conclusions.

3 Literature Review on Cloud Testing

3.1 Scientific Literature review on cloud testing

The systematic review of the scientific literature was carried out in two main steps: a former systematic review was performed in the first review period (M1-M18) analyzing and classifying peer reviewed articles in cloud testing found in IEEE Xplore Digital Library, ACM Digital Library and Scopus electronic sources. The results of this systematic review have been presented in D2.2 [7]; a latter systematic review of the scientific literature in cloud testing was performed in the second review period (M18-M36) by analyzing peer reviewed articles in cloud testing found in ScienceDirect, Wiley Online, and Springer Link digital libraries and by performing snowballing of the primary studies identified in the first review period. We refer to [6] for the whole results of the systematic survey of the scientific literature on cloud testing.

3.2 Grey Literature review on cloud testing

In this deliverable, we present the results of a grey literature review on cloud testing, with the main objective to identify and categorize the literature that is not formally published in sources such as books or journal articles on cloud-based testing. To this purpose, we present in Figure 2 a framework used for classifying the selected primary studies into three (non-overlapping) categories that are: testing in the cloud, testing of the cloud and testing of the cloud in the cloud, and six areas that are: Test Perspective, Test Design, Test Execution, Test Objectives, Test Evaluation and Application Testing. Within each area we list several topics. More details of this framework are in Section 3.2.2. We show in Section 3.2.1 the methodology adopted for conducting our review where Section 3.2.3 depicts the obtained results.

3.2.1 Methodology

This survey combines the guidelines proposed by two major and influential categories of works in this field. On the one side, this work refers to the procedures proposed by Garousi and co-authors [2] for planning and running a multivocal literature review. On the other side, this work also includes aspects following the overall recommendations proposed by Kitchenham and co-authors [3] when conducting systematic literature reviews on software engineering. In the following, we present the research methodology that was applied in this study. Specifically, the resulting approach has been structured around the following 5 phases: planning of the objectives, identification of the sources, entries extraction from the sources, selection of the entries in scope with the study, quality assessment of the resulting entries.

Planning the Review

The objective of this work is an analysis of the current approaches and trends in cloud testing as discussed by the industry, professionals, and SE practitioners, outside of academic forums. Such a goal has been achieved by answering the following research questions (RQs):

1. RQ1: What are the challenges, issues and future directions of cloud testing?
2. RQ2: What type of cloud testing methods has been used?
3. RQ3: What are the main objectives for cloud testing?
4. RQ4: How many primary entries present techniques, tools, or models for cloud testing?
5. RQ5: Which are the main application domains for software testing in the cloud?

Sources for the Review

The applied methodology started from the identification of a set of sources considered relevant for a grey literature survey on cloud testing. The selection was driven following four main categories of sources: Authoritative Grey Literature Sources; Authoritative Industrial Sources; Must-read Cloud Computing Blogs; Must-read Software Engineering Blogs. For each category, the actual sources have been selected by means of joint discussions among colleagues from the ElasTest project coming from both industry and academia. In addition, it was also considered useful to include the outcomes from a general purpose search engine such as Google Search Engine. In the following, we report the final list of the sources adopted in this study:

Authoritative Grey Literature Sources

- OpenGrey EU (<http://opengrey.eu/>)
- Cloud Computing IEEE (<https://cloudcomputing.ieee.org/>)
- Medium Blogging Platform (<https://medium.com/>)

Authoritative Industrial Sources

- ISTQB (<https://www.istqb.org/>)

IBM Blog (<https://www.ibm.com/blogs/>)

AWS Blog (<https://aws.amazon.com/blogs/aws/>)

IDC (<https://www.idc.com/>)

TechRepublic (<https://www.techrepublic.com/>)

InfoQ (<https://www.infoq.com/>)

Must-Read Cloud Computing Blogs

Infoworld Cloud Computing (<https://www.infoworld.com/>)

All Things Distributed (<https://www.allthingsdistributed.com/>)

Diversity Limited (<https://www.diversity.net.nz/>)

Compare the Cloud (<https://www.comparethecloud.net/>)

Network World (<https://www.networkworld.com>)

Cloud Tweaks (<https://cloudtweaks.com>)

Must-Read Software Engineering Blogs

Toptal Engineering Blog (<https://www.toptal.com/blog>)

Martin Fowler Blog (<https://martinfowler.com/>)

Coding Horrors (<https://blog.codinghorror.com/>)

Scott Hanselman Blog (<https://www.hanselman.com/blog/>)

Scott Berkun Blog (<https://scottberkun.com/blog/>)

Steve Yegge Blog (<https://steve-yegge.blogspot.com/>)

Generic Sources from the Web

Google Search Engine (<https://www.google.com/>)

Automated Search

The second step in the methodology concerned the definition of the criteria for launching automated searches on the identified sources. Specifically, it was established to apply a full search on all the fields indexed by the specific source repository (e.g., title, description, body of the content, meta-data, etc.). In order to be as comprehensive as possible, there was an agreement on adopting a quite general search string that was: “{testing} <AND> {cloud}”. The main idea was that entries must concern both testing, and cloud. However, it was also agreed to limit the search to English contributions appeared from 2012 to 2018.

Selection based on inclusion/exclusion criteria

On top of all the entries returned by querying the selected sources, a first screening was planned by taking into account the title of the item, and where possible, by

reading its associated abstract or short description. A set of Inclusion and Exclusion Criteria has been considered in order to uniformly drive such preliminary selection process. The detailed list of the adopted criteria is reported in Table 1. Specifically, exclusion criteria aimed to filter academic or peer-reviewed works, but also commercial or sponsored entries. In addition, it is well known that querying general purpose search engine (e.g., Google Search) may return a huge number of entries so that it is unfeasible to treat all of them. A common approach in grey literature surveys is to rely on the rank algorithm that each search engine includes in order to sort the returned results by relevance. For this reason, as inclusion criteria it was agreed to consider only the first 120 items returned by each search engine.

Exclusion Criteria	Inclusion Criteria
papers published in proceedings/journals after peer-review process	edited between 2012 and 2018
advertisement/sponsored links	thesis
presentations about a company or its profile	tech reports
commercial material about a specific product/solution/service/platform/tool	tools manual
market report/overview	tutorials
no English	webinars
no full and free accessible (for the authors)	blog posts
not in the scope of the survey	video from panels
online courses	opinions
How-to (i.e. Quora, StackOver-flow, FAQ, etc)	slides
job offer	interviews
not included within the first 120 returned items	

Table 1: Inclusion and Exclusion Criteria

Selection based on quality assessment

The actual selection of any identified entry has been done by considering its whole contribution (e.g. reading the article, watching the video, etc.) and evaluating it according to a quality checklist. Specifically, the checklist aggregated the 8 different indicators reported in Table 2.

Indicator	Indicator description
11	The publishing organization is supposed to be authoritative
12	The individual author is associated with a renowned organization
13	The author published other work either in the area of testing or cloud
14	The author has a clear expertise either in the area of testing or cloud
15	The document is focusing on cloud testing
16	The statements in the source are as much objective as possible
17	The source has a clearly stated aim
18	The source has a stated methodology

Table 2: Indicators driving the quality score procedure

All the entries have been processed by two reviewers that assigned to each indicator a quality score expressed in terms of the fuzzy Likert approach proposed in [4]. In detail, each fuzzy quality score is composed by two consecutive judgements measuring the level of agreement with the statements associated to the considered indicator. This level of agreement can be: *Fully Disagree* (value =1), *Partially Disagree* (value =2), *Neutral* (value =3), *Partially Agree* (value =4), *Fully Agree* (value =5). Also, a reviewer was requested to give a confidence level for each of these two judgements. The sum of the two confidence levels expressed per a quality score is assumed to be 1. For instance, let us consider the indicator I5: a reviewer can have a clear opinion about the focus of the contribution (e.g., she/he somehow agrees with the indicator), or she/he tends to disagree with the statement associated to the indicator but without a crystal opinion. The former case could be represented by assigning to the fuzzy quality score a single judgment (e.g., $QS1_{I5}=f[4; 1:0]$), while in the latter the reviewer can express the fuzzy quality score by means of a pair of judgments with different confidence degrees (e.g., $QS2_{I5}=f[1; 0:3]; [2; 0:7]$). For each fuzzy quality score expressed by a reviewer (i.e., one per indicator), the de-fuzzification process described in [4] has been applied. A de-fuzzification process represents the transformation procedure that maps back a fuzzy input (i.e., the pairs of quality scores expressed as judgement and confidence) into a scalar value. More specifically, in [4] the de-fuzzification process has been instantiated by taking into account the triangular isosceles membership functions; the same approach has been adopted also in this study. Specifically, the output of the de-fuzzification process is calculated as a combination of the two consecutive judgments in a fuzzy quality score (i.e., J_i and J_{i+1} in Eq. 1) weighted with the area of the trapezoids resulting from their respective confidence degrees (i.e., $A(C_i)$ and $A(C_{i+1})$ in Eq. 1).

$$Output = \frac{J_i A(C_i) + J_{(i+1)} A(C_{i+1})}{A(C_i) + A(C_{i+1})} \quad (1)$$

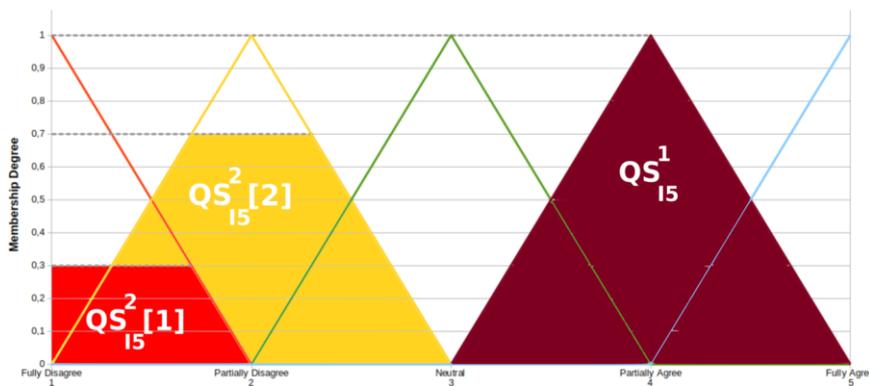


Figure 1: Triangular Isosceles Membership Functions

Figure 1 depicts the triangular isosceles membership functions. Reviewers could have very different interpretations on the indicators in Table 2; in addition they could refer very different standards or metrics in order to express their feelings. For these

reasons, a set of guidelines have been agreed by the reviewers before accessing the results from the automated search procedure. Among the others, such guidelines established criteria supporting the judgements about: an authoritative organisation (i.e., I1); the number of publications (i.e., I3); the expertise of an author (i.e., I2, and I4). The overall opinion of each reviewer about an entry has been calculated by taking into account all the quality scores from the 8 indicators. Specifically, as all the indicators have the same relevance for this study, thus the aggregated entry's quality score per reviewer has been computed as the average on the resulting fuzzy Likert values. The decision about adding or rejecting an entry from the set of primary studies has been taken by estimating the level of agreement reached by both the reviewers. Specifically, a measure of the consensus [5] on the aggregated quality scores has been considered. Intuitively, consensus is modelled as a function over a set of different opinions about some statement expressed on the basis of a pre-defined scale (e.g., Likert scale, or fuzzy Likert scale) and that ranges from 0 (i.e., complete disagreement of opinions), to 1 (i.e., for complete agreement). Following the formulation reported in [5], the consensus model adopted in this study is given with Eq. 2.

$$Cns \left(\begin{matrix} r_1 \\ r_2 \end{matrix} \right) = 1 + \sum_{i=1}^2 1/2 \log_2 \frac{|r_i - \bar{R}|}{d} = 1 + \log_2 \frac{|r_1 - \bar{R}|}{d} + \log_2 \frac{|r_2 - \bar{R}|}{d} \quad (2)$$

where r_i is the aggregated quality score from the reviewer i on the considered entry; $d = L_{\max} - L_{\min} = 4$ is the width of categories on the referred fuzzy Likert scale (i.e., $5 - 1 = 4$), and \bar{R} is the mean on the aggregated agreements by the two reviewers.

On the one hand, when the two reviewers show a high agreement on a given entry (i.e., the consensus scored at least 0.85) then the verdict on their evaluations is assumed to be significant. In this case, if the average on both the aggregated quality scores results greater than 3 the entry is added to the set of the primary studies; otherwise it is rejected. On the other hand, when the agreement between the two reviewers is not significant (i.e. the consensus scored less than 0.85) then the entry is processed by a third reviewer who decides if it deserves to be included or not within the set of primary studies.

Specifically, we applied the automated search described before to all the identified sources, obtaining an initial set of 250 entries. Duplicated entries among Google Search Engine and the other sources have been deleted, and not considered in the initial set of entries found in Google Search Engine. To this set of 250 entries, we applied the inclusion and exclusion criteria described in Table 1, obtaining a set of 141 included entries. Out of these 141 entries, according to the quality assessment procedure described above, 51 were included, 80 were excluded, whereas the remaining 10 entries (for these entries the consensus scored was less than the threshold) were processed by a third reviewer. Out of these 10 entries, applying the same quality assessment procedure, the third reviewer decided to include 6 and exclude 4 entries, obtaining a total set of 57 primary studies analyzed in our study.

3.2.2 The classification framework

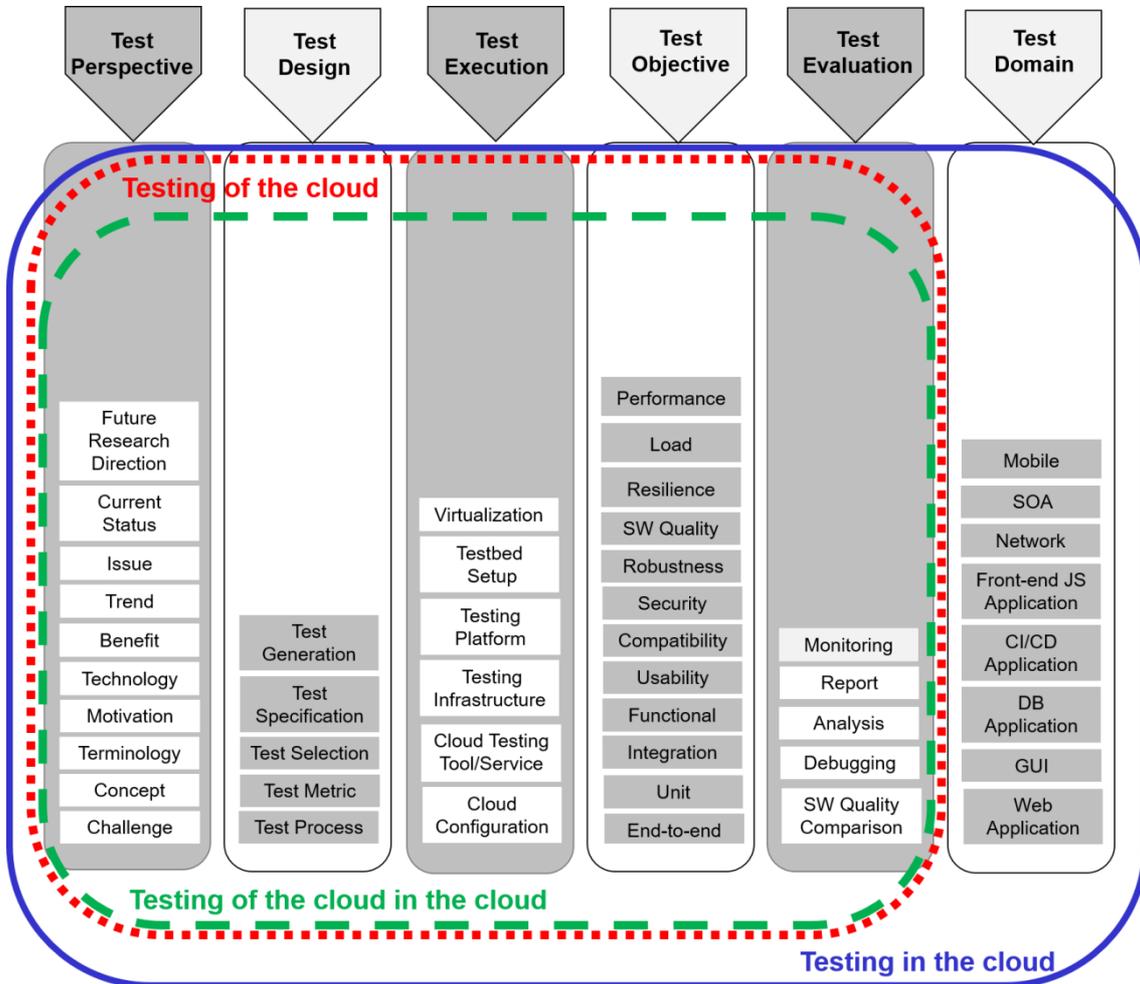


Figure 2: Cloud Testing Framework

In Figure 2, we present the framework we developed to characterize the results of our grey literature review on cloud testing. This framework was inspired by a similar framework we developed to classify the papers of a systematic review in cloud testing presented in [6]. Specifically, we initially considered as draft scheme the framework presented in [6] including six areas and several topics for each area. Then, we used this draft scheme to classify the primary entries, adding new topics within each area during the reading of the full text of the primary entries. Finally, we removed the topics not covered by any primary entry. The resulting framework that is presented in Figure 2 includes the following six areas and some topics representing trends and directions of the grey literature on cloud testing:

Test Perspective. The primary entries belonging to this area address topics such as basic concepts, trends, current research directions, terminology, challenges and benefits of cloud testing.

Test Design. This area includes approaches addressing the design phase of testing activity. This area shows less topics than the same area in the framework of [6], namely test cases generation or selection, test specification and test metric, evidencing a low interest of the grey literature for test design approaches.

Test Execution. The primary entries belonging to this area present artifacts involved into the execution phase of the testing activity, such as platforms or infrastructures or tools or services for cloud testing as well as visualization or cloud configuration approaches or solutions for testbed setup.

Test Objective. The primary entries belonging to this area address the different objectives of cloud testing such as functional aspects or different types of testing such as unit or integration or end-to end or non-functional properties such as robustness, performance, reliability, security, usability and others.

Test Evaluation. The primary entries belonging to this area address the evaluation stage of the testing activity namely monitoring of the testing activity, report and analysis of test results, as well as support for debugging.

Test Domain. The primary entries belonging to this area present cloud-based testing solutions addressing specific application domains, such as mobile, web applications, CI/CD applications among the other.

As in [6], we classify papers into three different categories that span over the above areas as shown by the colored frames in Figure 2: i) Testing in the cloud (blue continued frame in Figure 2) refers to software testing performed by leveraging scalable cloud technologies, solutions and computing resources to validate non-cloud software/applications; ii) Testing of the cloud (red dotted frame in Figure 2) refers to validating the quality (functional and non-functional properties) of applications and infrastructures that are deployed in the cloud; iii) Testing of the cloud in the cloud (green dashed frame in Figure 2) refers to applications and infrastructures deployed in the cloud and tested by leveraging cloud environment.

3.2.3 Summary of Results

Figure 3 shows the breakdown of primary studies according the three categories of our framework. This figure evidences a high interest of the grey literature for TIC category (41% of the selected primary studies belong to this category) whereas a minor number of primary studies proposes to leverage cloud platform for testing of applications and infrastructures deployed in the cloud (11% of primary studies belong to ToiC category and only few primary studies address TOC, 5%).

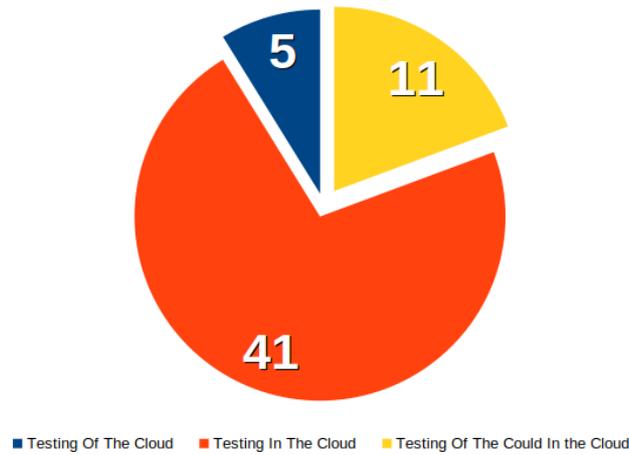


Figure 3: Breakdown of the Primary Studies by Category

Figure 4 shows the number of primary studies per year. As evidenced, this number swings between 10 and 20 until 2016, then, it rapidly grows in the last 2 years (from 14 items in 2016 to 57 in 2018). This confirms the growing attention devoted to the cloud testing in the last years by the grey literature.

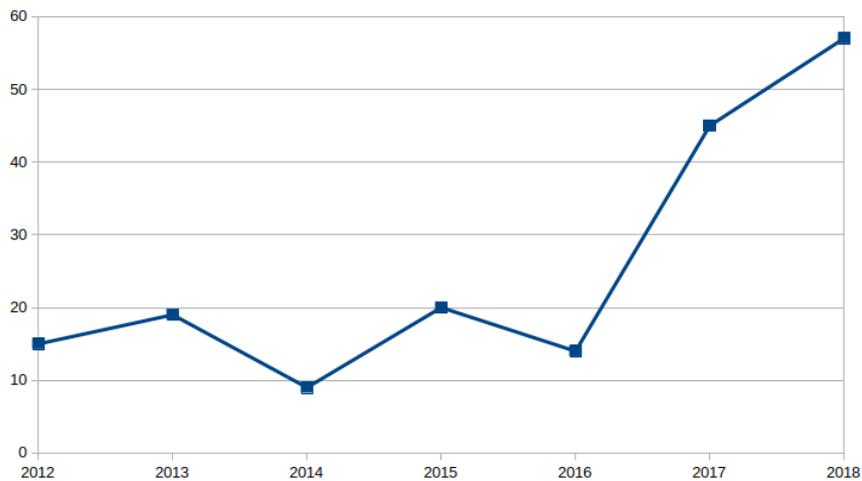


Figure 4: Number of Publications by Year

The data of Figure 5 confirm the trend of Figure 4, showing the number of primary studies per year in each area. These data show that for most of the areas the number of items rapidly grows in the last two years (from 4 to 13 for *Test Execution*, from 4 to 11 for *Test Objective*, from 3 to 16 for *Test Domain* and from 2 to 10 for *Test Perspective*).

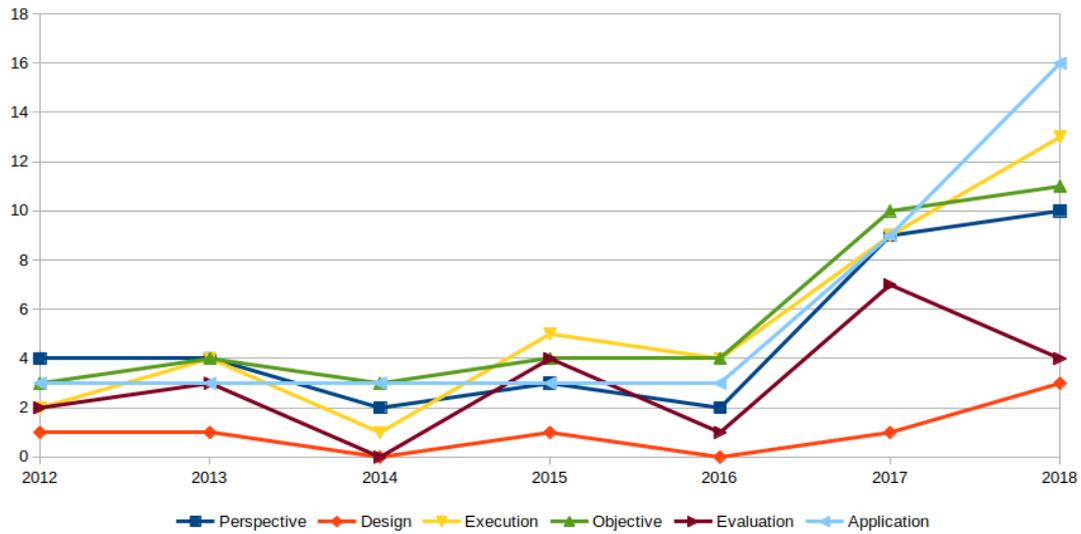


Figure 5: Number of Publications by Year and Area

Figure 6 depicts the distribution of primary studies over the six areas. In particular, the majority of primary studies focus on four of the six areas: *Test Perspective* (34 primary studies), *Test Execution* (38 primary studies), *Test Objective* (39 primary studies), and *Test Domain* (40 primary studies). Data suggest a low coverage from the grey literature on *Test Evaluation* (21 primary studies) and even lower on *Test Design* (7 primary studies). Note that, as each primary study could be classified in multiple areas, any distribution analysis done by area is not a partition of the set of primary studies; thus in each kind of distribution the sum of primary studies could be greater than their number (57).

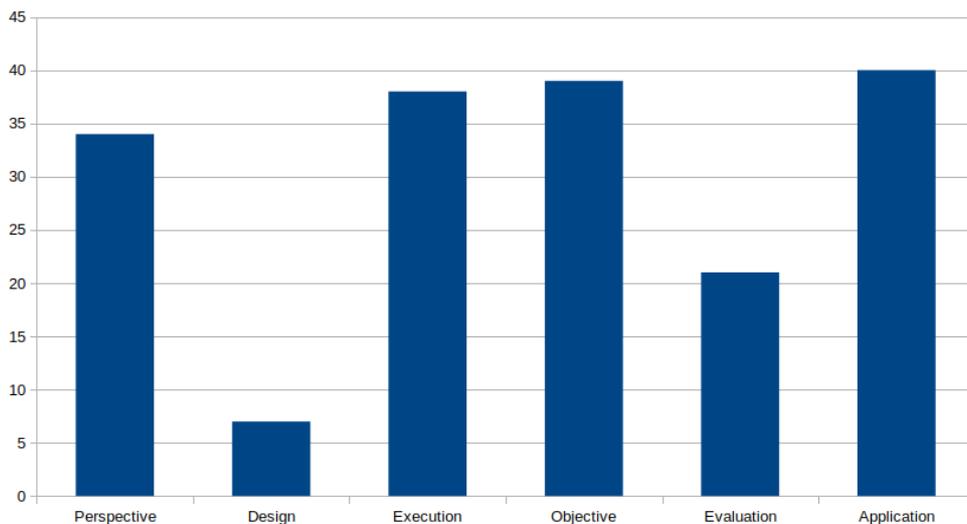


Figure 6: Distribution of the Papers by Area

Comparing the results of Figure 6 with those of the analysis reported in D2.2 (Fig. 3)[7], we can observe that *Test Perspective* and *Test Domain* areas achieve a greater interest from primary studies of our grey literature review than from the research papers analyzed in D2.2[7], evidencing a different audience of the two types of publications.

In the following, we summarize the main results that answer to the research questions defined in Section 3.2.1, by presenting the breakdowns of primary studies over the topics of each area and considering the three categories of *TiC*, *ToC* and *ToiC*.

Research Question1: What are the challenges, issues and future directions of cloud testing?

Figure 7 evidences that the majority of primary studies classified in the *Test Perspective* describe the *concepts* underlying cloud testing (17 primary studies). Other topics that show a good interest are *challenges* and *technology* (13 primary studies for each of them). Lower interest is showed for *benefit* (9 primary studies), *issues* (10 primary studies), *motivation* (8 primary studies) and *terminology* (10 primary studies) topics. Finally, few works address *future research direction* (4 primary studies) as well as *current status* and *trends* (3 primary studies for each one) of cloud testing.

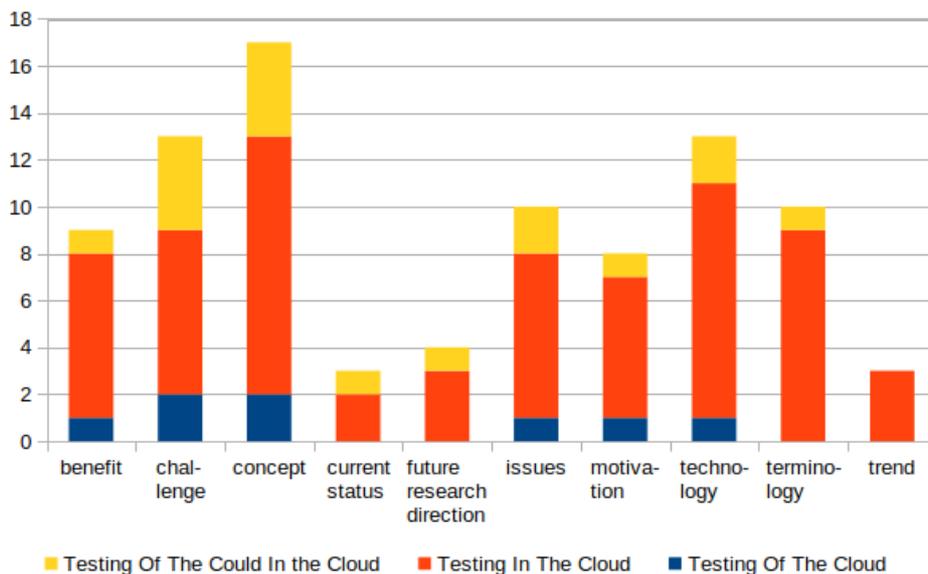


Figure 7: Breakdown of the Primary Study in Test Perspective

Research Question2: What type of cloud testing methods has been used?

Figure 8 shows that the few primary studies included in the area of *Test Design* fairly address the different topics (1 primary study for *test metrics*, *test process*, *test selection*, and *test specification* with the only exception of *test generation* that is the most addressed topic, covered with 3 primary studies).

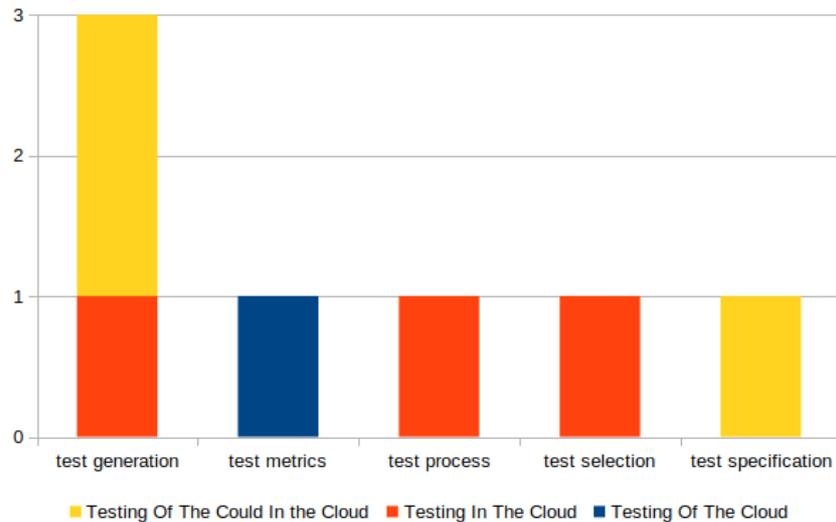


Figure 8: Breakdown of the Primary Study in Test Design

Research Question3: What are the main objectives for cloud testing?

In Figure 9, the primary studies have been classified according to the objectives of the cloud testing. This figure shows that functional and non-functional aspects are fairly addressed. Specifically, the most covered goals are *functional* and *performance* testing (i.e., 12 primary studies) with a similar breakdown on the *TIC*, *TOC* and *TOIC* categories. Other goals that result to be sufficiently covered are *integration testing* (i.e., 7 primary studies), *compatibility* (i.e., 7 primary studies) and *security* (i.e., 8 primary studies). These results also confirm what already evidenced in D2.2[7] (Figure 5) concerning the great effort spent in validating performance and the low interest of cloud testing for the other not functional aspects such as *load* (i.e., 1 primary study), *resilience* (i.e., 2 primary studies), *robustness* (i.e., 3 primary studies), as well as *usability* and *sw quality* in general (i.e., 5 primary studies). Minor attention is also deserved for *end-to-end* and *unit* testing (only 3 primary studies).

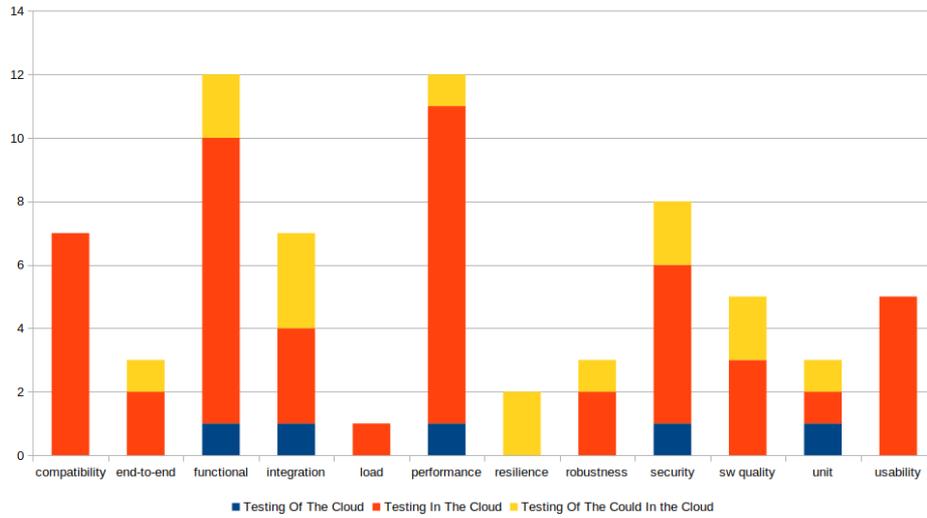


Figure 9: Breakdown of the Primary Study in Test Objective

Research Question4: How many primary entries present techniques, tools, or models for cloud testing?

Many primary studies focus on the execution phase of the testing activity. The distribution of papers addressing *Test Execution* is presented in Figure 10. The reported data show the great interest in this area for testing tools and services (i.e., 22 primary studies), in particular concerning testing in the cloud (i.e., 16 primary studies). Few works address *testbed setup* (i.e., 9), *cloud configuration* and *testing infrastructures* (i.e., 8) whereas still less attention is deserved to *testing platform* (i.e., 3) and *virtualization* (i.e., 2).

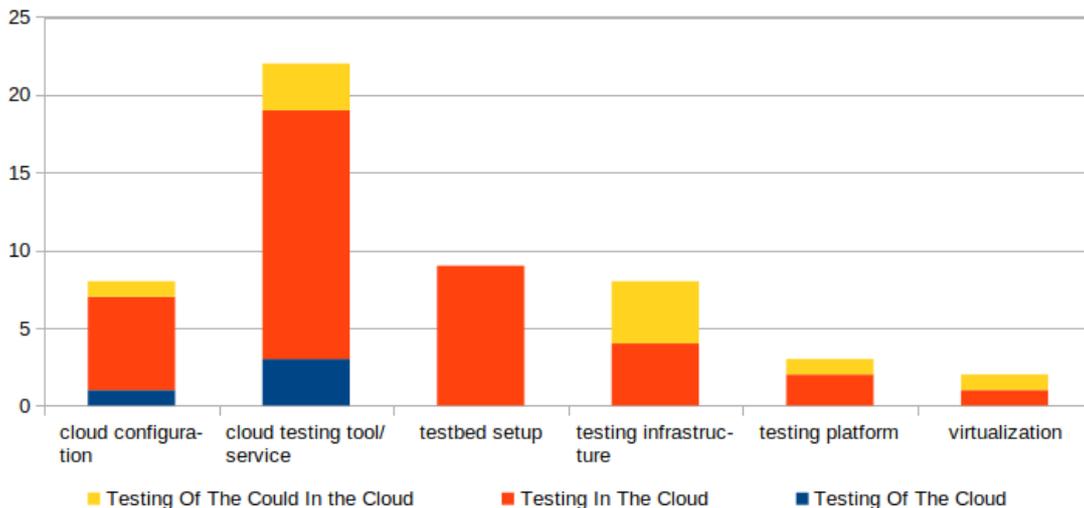


Figure 10: Breakdown of the Primary Study in Test Execution

Figure 11 highlights that the main task supporting the evaluation of the cloud testing activities concerns reporting of test results (i.e., 16 over 27 primary entries address this topic). All the other topics receive a marginal attention, indeed a very few works address *analysis* (i.e., 3), *debugging* (i.e., 4), monitoring (i.e., 4) and *sw quality comparison* (i.e., 1).

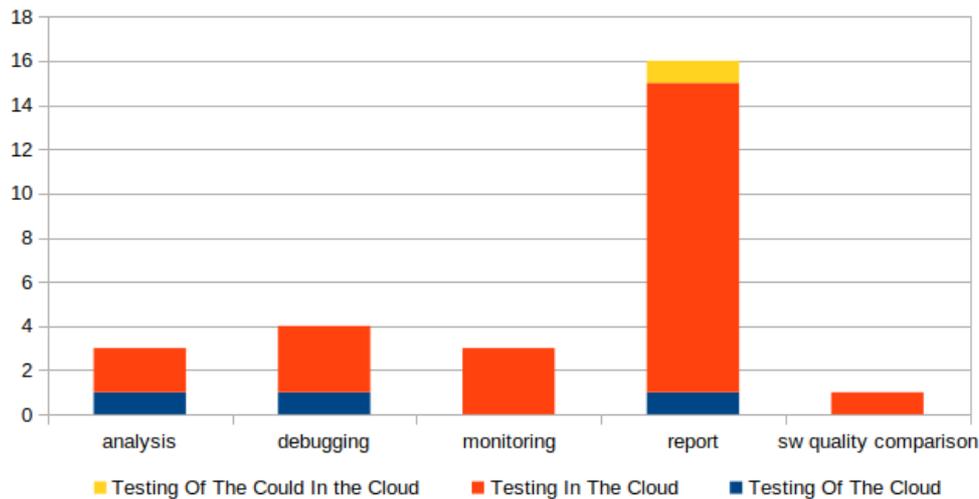


Figure 11: Breakdown of the Primary Study in Test Evaluation

Research Question5: Which are the main application domains for software testing in the cloud?

Figure 12 plots the classification of the primary studies according to the application domains addressed by the testing solutions leveraging scalable cloud technologies. Concerning the domains of cloud testing, the collected data confirm what already evidenced in the survey reported in D2.2 [7] (Figure 10), namely that cloud solutions are mostly used to validate *mobile* (i.e., 18 primary studies) and *web applications* (i.e., 12 primary studies). The other application domains receive less attention: 3 primary studies focus on *CI/CD applications*, 2 primary studies on *DB Applications*, *GUI*, *network* and *SOA* whereas only one study addresses *front-end JS Application*.

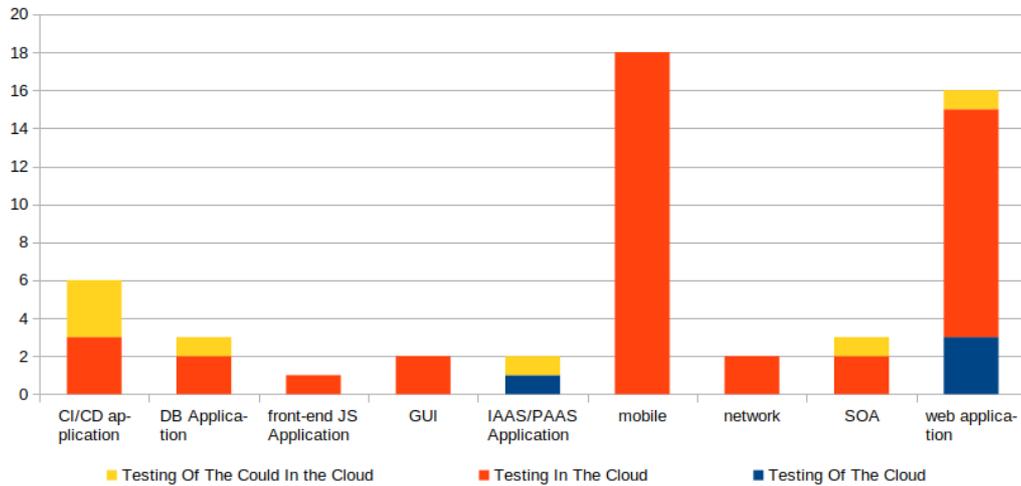


Figure 12: Breakdown of the Primary Study in Test Domain

4 Technical Analysis of SoTA

In this section, we focus on technical analysis of SotA, namely the most important tools that can be of interest for ElasTest. As in D2.2 [7], these tools have been classified according to the following main aspects covering the technological areas in which ElasTest advanced the state of the art, defined also in the DoA [1]:

- Continuous Integration
- Performance Testing
- Security Testing
- Monitoring
- GUI automation and impersonation (IoT Testing)
- Cloud Instrumentation
- Dashboard Management
- WebRTC Testing
- Cross-browser Testing
- Mobile Testing
- Cognitive Q&A Systems
- Test Orchestration
- Data Ingestion
- Test Execution & Visualization
- Test Management
- Testing Framework
- Virtualization

The analysis includes a different number of tools for each technological area and has been performed along all the second reporting period. Specifically, after the delivery of D2.2 [7], an overview of the state of the art was conducted in March 2019 and its

update performed in September 2019. We describe the most important tools for each technological area in the following sections. We report only the technological areas in which new tools have been found or important updates of tools presented in D2.2 [7] have been identified in the second review period. For each of these technological areas, we present a short description of the tools, a comparison among them and the ElasTest progress in this area with respect to the state of the art.

4.1 Continuous Integration

In this section, we provide the most relevant tools of continuous integration identified in the second review period. Specifically, we divide the tools into two main categories: Continuous integration Server and Continuous integration - Artifact distribution. There are not new relevant tools about Continuous integration - Identity Access Management and Continuous integration – Docker Image Distribution addressed in D2.2 [7].

4.1.1 Continuous Integration Server - Baseline and comparative analysis

At the beginning of the ElasTest project the main use of the CI servers was building and testing software projects/artefacts. Occasionally, only in well established companies or projects where the DevOps trend was adopted early, some activities concerning deployment tasks were also conceived in charge of CI servers. Not all the tools specified in the first version of the SoTA were able to manage the whole DevOps cycle and those that were able, did it with different level of simplicity for the administrators, and giving the user/admin different level of control.

Nowadays, DevOps and Continuous Delivery are the main trends. Tools and frameworks are often integrated in CI/CD pipelines, and at the same time new tools specifically designed to be linked into the CI/CD pipelines have appeared. The integration between CI servers and other tools such as code repositories, development IDEs, code quality, testing tools was successful in different degrees. Most of the CI servers were able of executing a series of tasks, that where triggered either manually, periodically or automatically by an event. The definition of these series of tasks usually takes place by managing of some sort of pipelined infrastructure.

Currently, most of the CI servers have been evolved to the DevOps paradigms; in line with this new paradigm, CI technologies support the explicit management of SW development activities in terms of pipelines. Even though CI Servers could impose a structured organization of the activities on the projects they are managing, nowadays most of the CI servers allows the admin/user to maintain their pipelines as code. Another big evolution of the CI servers is the redefinition of their own architecture towards cloud-native; thus the CI server functionalities have been improved also towards this kind of applications. In addition, due to the proliferation of cloud-native applications, also big public cloud providers like AWS and Azure have launched their own CI servers specifically designed for the DevOps operations in their cloud, and have integrated this product as part of their global offer.

Table 3 shows the most relevant tools identified in this second review period for continuous integration server.

Name	URL	Brief description	Licence
AWS CodePipeline	https://aws.amazon.com/pipeline/	AWS CodePipeline is a fully managed continuous delivery service that helps to automate your release pipelines for fast and reliable application and infrastructure updates. CodePipeline automates the build, test, and deploy phases of the release process every time there is a code change, based on: the release model you define; the release process every time there is a code change.	Proprietary
Azure Pipelines	https://azure.microsoft.com/en-us/services/devops/pipelines/?nav=min	It provides the capacity of creating pipelines to cover all the DevOps cycle stages.	Proprietary but provides free access for OSS
CloudBees CodeShip	https://www.cloudbees.com/products/cloudbees-codeship	Aim to ship faster with CI/CD as a Service.	Proprietary
Jenkins X	https://jenkins-x.io/	Jenkins X is a CI/CD solution for modern cloud applications on Kubernetes and is being proposed as a sub-project via JEP-400.	OSS (Apache License 2.0)
Semaphore	https://semaphoreci.com/product	AutoScalable CI/CD-as-a-service.	Proprietary
Team Foundation Server	https://www.visualstudio.com/tfs/	Microsoft tool that provides source code management (Team Foundation Version Control or Git), reporting, requirements management, project management, automated builds, lab management, testing and release management capabilities.	Proprietary
TeamCity	http://www.jetbrains.com/teamcity/	It allows Java-based build management and CI server from JetBrains.	Proprietary

Table 3: Continuous integration server tools

According to the opinions registered on G2Crowd¹ and Slant², the most relevant tools about continuous integration server developed in this second review period are: AWS

¹ Open Source Initiative, licenses <https://opensource.org/licenses/alphabetical>

² Slant.co <https://www.slant.co/topics/799/~best-continuous-integration-tools>

CodePipeline, Jenkins X, and Semaphore. In the following, we present a brief description of each of them. Other tools, such as Team Foundation Server and TeamCity have been updated with respect to the description provided in D2.2 [7]. We describe in the following their main updates.

AWS_CodePipeline

AWS CodePipeline is part of the AWS Developers tools. The AWS Developers tools is a set of services designed to enable developers and IT operations professionals practising DevOps to rapidly and safely deliver SW.

Specifically, AWS CodePipeline is a continuous integration and continuous delivery service for fast and reliable application and infrastructure updates. CodePipeline builds, tests, and deploys the code every time there is a code change, based on the defined release process models. This enables you to rapidly and reliably deliver features and updates. You can easily build out an end-to-end solution by using the pre-built plugins for popular third-party services like GitHub or integrating your own custom plugins into any stage of your release process.

AWS CodePipeline has a simple rate of \$1.00 per active pipeline per month.

Jenkins X

Jenkins X provides pipeline automation, built-in GitOps and preview environments to help teams collaborate and accelerate their software delivery at any scale. As shown in Figure 13, Jenkins X is not exactly a new CI Server but an architecture around Jenkins over Kubernetes, so while it makes use of the very mature functionality of Jenkins it adapts to the Cloud natively.

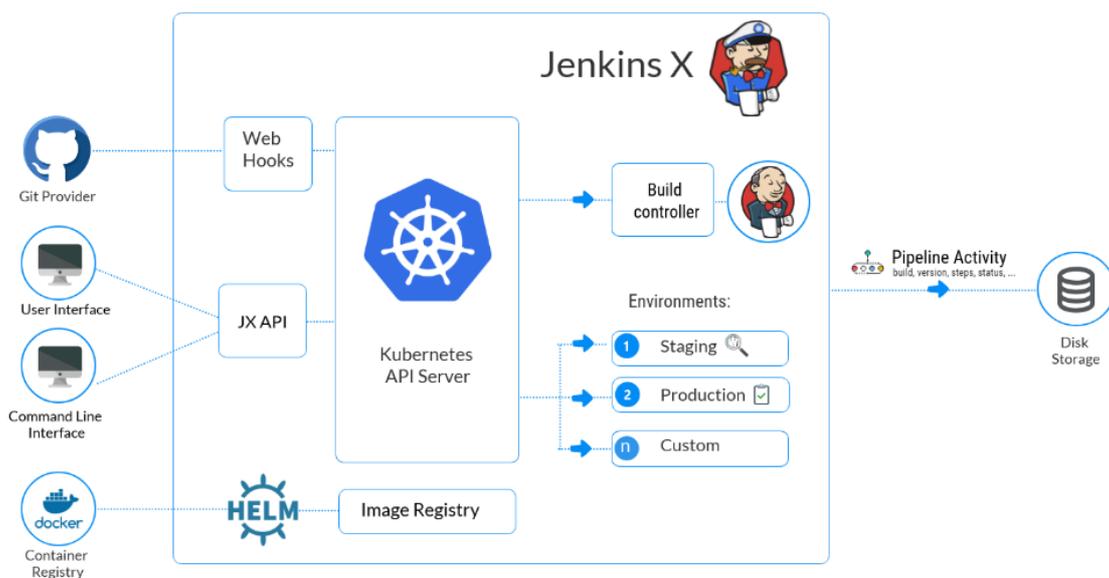


Figure 13: Jenkins X architecture

Semaphore

Semaphore [5] just claims to be a CI/CD server that “just works”. It expresses well that even it doesn’t stand out for something ground-breaking it does what is expected: automate any continuous delivery pipeline with customizable stages, parallel execution, control flow switches, secrets and dependency management; and provides auto-scalability while offering a pricing system of pay per use. The pricing offering is based on the characteristics of the underlying infrastructure where the pipeline is executed and the time of execution is detailed in Table 4.

Instance name	Characteristics	Price
e1-standard-2	2 vCPU @ 3.4GHz (Turbo 4.0GHz), 4GB RAM, 25GB RAM	\$0.00025/sec
e1-standard-4	4 vCPU @ 3.4GHz (Turbo 4.0GHz), 8GB RAM, 35GB RAM	\$0.00050/sec
e1-standard-8	8 vCPU @ 3.4GHz (Turbo 4GHz), 16GB RAM, 45GB RAM	\$0.00100/sec

Table 4: Semaphore characteristics

Team Foundation Server

As described in D2.2 [7], Team Foundation Server (TFS) is an enterprise-grade server for teams to share code, track work, and ship software for any language, all in a single package. TFS works better as a part of the Microsoft CI tools, but multiple tools (proprietary and open source) can be integrated with TFS.

Team Foundation Server is available in two different forms: on-premises and online.

TFS is quite easy to install and configure while working within the Microsoft Suite and Microsoft Cloud platform (Azure). TFS has evolved to separate the on premise environment, still called Team Foundation Server, and the hosted one now called Azure DevOps. The CI Server inside Azure DevOps has been renamed as Azure Pipelines and evolved into a more “standard” CI server where the tasks are organised as pipelines and can be managed as code.

TeamCity

As described in D2.2 [7], TeamCity is a continuous integration and deployment server from JetBrains. It is known to be quite harsh to setup, but once it is done, building projects using Ant or MSBuild is incredibly easy, whereas for many other languages the initial setup of the build configuration can be a bit daunting. As many environments are supported such as Java or Python, TeamCity shines while used in .Net environments.

TeamCity provides a wide range of additional plugins to provide integration with most used CI tools. TeamCity has evolved in the same line of most CI servers; it has focused in the UI and usability.

Finally, Table 5 shows a comparison of CI server tools according to some common dimensions such as simplicity of usage, acceptance by the developer community, expertise of partners, integration with other CI tools, costs. The table also compares the CI server tools described before with Jenkins that has been integrated into ElasTest. We refer to D2.2 [7] for the Jenkins description.

	 AWS CodePipeline	 Jenkins	 Jenkins X	 Semaphore	 Microsoft TFS	 TeamCity
Simplicity of usage	Bad	Good	Medium	Good	Good	Medium
Hosted	Yes	No	No	Yes	Yes/No	Yes/No
Supported types of projects	multiple	multiple	multiple	multiple	multiple	multiple
Integration with other CI tools	Poor ³	Very Good	Good	Average	Good	Good
Acceptance by the developer community	Niche	High	Medium	Medium	Medium-High	Low
Expertise of partners	Low	High	Low	Low	Low	Low
Costs	Average (pay per use)	Open Source + hosting	Open Source + hosting	Average (pay per use)	High	Average (License + Hosting)

Table 5: Comparison of CI server tools

4.1.2 Continuous integration - Artifact distribution - Baseline and comparative analysis

In the context of continuous integration-artifact distribution, a new tool has been identified in the second review period that is Jfrog Artifactory + Bintray. In the following a brief description and main features of this tool are provided (see Table 6 and Table 7).

Name	URL	Brief description	License
Jfrog Artifactory + Bintray	https://jfrog.com/artifactory/ https://jfrog.com/bintray/	It allows for fast release and Universal Artifact Management	Private (Pay per use and License)

³ AWS CodePipeline has poor integration with tools outside the AWS Code* tools.

for DevOps Acceleration.

Table 6: CI- Artifact distribution tools

Jfrog Artifactory + Bintray - The DevOps integrated option

Jfrog Artifactory + Bintray can work together in order to provide not only a storage for artifacts but a set of rules and integrations for deliver and release these artifacts.

As a universal repository manager, Artifactory integrates with the existing ecosystem supporting end-to-end binary management that overcomes the complexity of working with different software package management systems, and provides consistency to CI/CD workflow. In the other hand, Bintray that can be integrated with Artifactory would provide the access control to the artifacts, statistics of access and downloads of the artifacts.

These two tools are specifically designed to be integrated in DevOps pipelines for Rapid Continuous Delivery.

	Jfrog Artifactory + Bintray
Simplicity of usage	High
Access and Permission management	Very good
Acceptance by the developer community	Medium-Low
Previous knowledge and experience of partners	Low
Integration with other CI tools	Medium
Costs	Pay per use and license (high)

Table 7: CI - Artifact distribution tools: Main Features

4.1.3 Progress within ElasTest

During this last review period, the consortium decided to avoid drastic replacement about the main CI tools (including the server) already included within the ElasTest platform. In fact, any update of that pool of CI tools would have also required to consider different infrastructure and to put in place different IT procedures. Nevertheless, as motivated in D2.2 [7], most of the CI tools adopted within the ElasTest platform during the first review period of the project are still valid as their new releases took into account new trends about CI. About Jenkins (i.e. the CI server adopted within the ElasTest platform), although nowadays there are more advanced CI servers on the market, Jenkins still provides good functionality and it covers the needs

of the project. Thus, there was not any justified gain for risking a major evolution of the ElasTest platform by adopting a more recent technology as CI server.

Looking forward into the ElasTest integration with CI servers, the current available integration can be used as an example of how this integration can be made, as ElasTest is providing an Open API that other servers, or plugins can make use of in order to integrate ElasTest functionality into their core.

4.2 Performance Testing

In this section, we provide an overview and a brief description of the most important tools concerning performance and scalability testing. As remarked since the DoA [1], supporting both the aspects is one of the main objectives for the ElasTest project.

4.2.1 Baseline and comparative analysis

In the following, we provide a short description of the performance tools identified in this second review period (Table 8). An overall comparison of the most relevant tools is reported in Table 9.

Name	URL	Brief description	License
StormRunner Load	https://www.microfocus.com/en-us/products/stormrunner-load-agile-cloud-testing/overview	It allows for load testing of mobile and web applications.	Proprietary
Load Impact	https://loadimpact.com/	It allows to run larger tests on global cloud infrastructure and makes stress testing of websites and web apps to ensure they can handle peak traffic conditions.	Proprietary
Loader	https://loader.io/pricing	It allows for stress testing of web-apps & apis as well as monitoring of the tests in real-time.	Proprietary (free for a reduced number of tests)
Octoperf	https://octoperf.com/	It is a SaaS-based performance testing tool powered by Jmeter.	Proprietary
LoadNinja	https://loadninja.com/	It uses real browsers for load tests, monitors the CPU Usage and allows to create different performance scenarios as well as to destroy session data captured in a load test.	Proprietary
Gatling	https://gatling.io/	Gatling is an open source load testing framework that is built based on Netty, Akka and Scala.	Open source
Tsung	http://tsung.erlang-projects.org/	It is an open source load testing tool that runs on multiple	Open source

		protocols and can be used to stress WebDAV, HTTP, MySQL, LDAP and many other servers.	
Locust	https://locust.io/	Locust is an easy-to-use, distributed, user load testing tool. It allow to write very expressive scenarios in Python.	Open source (MIT license)

Table 8: Performance Testing Tools

StormRunner Load

It is a cloud-based load testing tool for mobile and web applications. It can create voluminous virtual users as required to identify the breaking point of any application. The tool is customizable and has pre-loaded testing scenarios within the tool. The main benefits of this tool are: i) reduce effort and skill level: it provides easy to record and playback protocol support for developers and testers with TruClient and TruAPI protocols. Agile teams can quickly automate the user stories and start running performance tests in quick time. ii) geographical location simulation: all cloud based solutions offer simulating traffic from various geographical locations by choosing the LGs (Load Generators). StormRunner Load makes it simpler by just choosing the user load against a geographical location without having the need to calculate the number of load generators needed to simulate the target user load. iii) real time analysis: StormRunner Load identifies the SLA violations and anomalies during the test based on the script, geographical location and elapsed time. iv) monitoring servers: StormRunner can be integrated with monitoring tools like Sitescope to provide the resource utilization on the servers.

Load Impact

It reduces development costs, improves customer satisfaction and provides a new approach for supporting performance testing throughout the development lifecycle. It supports load testing early in the software development life cycle, on the developer's machine. It allows easily analyzing load test results and fixing performance issues. It allows running large tests in the cloud and catching API performance problems before production. It allows using the same test scripts across local and cloud execution modes. It allows to run larger tests on global cloud infrastructure and to make stress testing of websites and web apps to ensure they can handle peak traffic conditions. It allows generating loads from multiple load zones around the world to test performance for all global customers.

Loader

It is a load testing service that allows stress testing of web-apps & apis with thousands of concurrent connections. It allows to monitor the test in real-time, then share the results with the team.

OctoPerf

It is a SaaS-based performance testing tool powered by JMeter for the web, API, REST & mobile app. It allows for designing, monitoring, executing and analyzing how the website performs through a web browser. Each test comes with an option to download the report including: rate and response time, request details, response time breakdown, average response time, throughput.

LoadNinja

It uses real browsers at scale for load tests, creating the most realistic and accurate representation of load on the infrastructure supporting the web application under test. It monitors the CPU Usage of the load generating servers to ensure there are no bottlenecks. It allows to create different performance scenarios as well as to destroy session data captured in a load test after the test is run for privacy reasons.

Gatling

Gatling is an open source load testing framework based on Netty, Akka and Scala. It is a high-performance framework that offers ready-to-present HTTP reports. It offers scenario recorder and developer-friendly DSL.

Tsung

This is an open source load testing tool that runs on multiple protocols. It is free software that is released under GPLv2 license. It can be used to stress WebDAV, HTTP, MySQL, LDAP and many other servers.

Locust

Locust is an easy-to-use, distributed, load testing tool. It allows for load-testing web sites (or other systems) and figuring out how many concurrent users a system can handle. Locust is completely event-based, and able to support thousands of concurrent users on a single machine. In contrast to many other event-based apps, it doesn't use callbacks and allows for writing very expressive scenarios in Python. Locust has a HTML+JS user interface that shows relevant test details in real-time. It is cross-platform and easily extendable. Even though Locust is web-oriented, it can be used to test almost any system.

	LoadNinja	Gatling	Tsung	Locust
OS	Any	Any	Linux/Unix	Any
GUI	Yes	Recorder Only	No	No
Test Recorder	-	HTTP	HTTP, Postgres	No
Test Language	Scripting Language	Scala	XML	Python
Extension Language	Scripting Language	Scala	Erlang	Python
Load Reports	HTML	HTML	HTML	HTML
Protocols	HTTP, HTTPS, SAP GUI Web,	HTTP, JDBC,JMS	HTTP, WebDAV Postgres MySQL,	HTTP

	WebSocket, Java based protocol, Google Web Toolkit, Oracle forms		XMPP WebSocket, AMQP, MQTT, LDAP	
Host Monitoring	No	No	Yes	No
Open Source	No	Yes	Yes	Yes
Detailed Documentation	No	No	Yes	Yes

Table 9: Comparison of performance tools

4.2.2 Progress within ElasTest

As stated also in D2.2 [7], ElasTest enables the combination of performance testing with scalability aspects. Such an objective was achieved by means of the ElasTest configuration management mechanisms, and the Test Support Services. The main progress of ElasTest with respect to the tools reviewed in this second review period is that it allows testers and developers to assess their SiL by running different configurations and comparing the results among them. This will enable teams to choose the most appropriate scalability approach. Existing tools for load testing don't enable running the SuT under different conditions and don't enable testers to compare results from different configurations. We used these tools to gather ideas for the implementation/features/user interface of ElasTest. ElasTest allows integrating some existing performance tools. For example, ElasTest could execute JMeter performance tests and show the results in its web interface. This work has been reinforced by our collaboration with H2020 STAMP Project that allowed bringing generated test configurations to ElasTest.

4.3 Security testing

This section surveys the technological area about security testing. The analysis reports the list of tools identified, and an overview on their main features with respect to ElasTest.

4.3.1 Baseline and comparative analysis

The survey in this second review period revealed two security testing services: Privacy.net and Extensions.inrialpes.fr. Table 10 resumes their main information, while Table 11 provides a comparison between them.

Name	URL	Brief description	License
Privacy.net	https://privacy.net/analyzer/	It is an online tool that can be leveraged to check the web logins of a user and also to determine whether a user can be tracked across browsers.	Free (Not OSS)
Extensions.inrialpes.fr	https://extensions.inrialpes.fr	In this online tool, a user can check whether he can be	Free (Not OSS)

uniquely fingerprinted or not using web logins and browser extensions installed.

Table 10: Security Testing Tools

Privacy.net

It is an online tool that leverages several user fingerprinting techniques such as canvas fingerprinting, login leak and user agent settings in order to identify whether a user can be uniquely identified or not. A user can visit the tool from their browser and run the tests in order to calculate the unique fingerprint of the user.

Extensions.inrialpes.fr

It is an online service that can be leveraged by a user to identify whether a web site can uniquely identify them or not. The web site uses techniques such as installed browser extensions and login leak in order to calculate the unique fingerprint of the user.

	Privacy.net	Extensions.inrialpes.fr
Purpose	Browser Fingerprinting	Browser Fingerprinting
# of Attack Classes	5	2
License	Free	Free
Target Users	Web users	Web users

Table 11: Comparison of Security Testing Tools

4.3.2 **Progress within ElasTest**

ElasTest Security Services (ESS) supports security testing of cloud-based web applications. Compared to the security testing provided by the above tools, the ESS supports the detection of Cross-Origin State Inference (COSI) attacks, that are not currently supported by the others. Although the online services from *privacy.net* and *extensions.inrialpes.fr* leverage some COSI techniques, the ESS in ElasTest platform considers more sophisticated attack classes (40 attack classes including object properties, AppCache and postMessage) and it provides more advanced functionalities such as automatic attack page generation and combining of multiple user states. The main beneficiaries of these two online services are web users, the main beneficiaries of ESS are testers. Additionally, ESS also comes with OWASP ZAP integration for identifying common web application weaknesses.

4.4 Monitoring

This section surveys the technological area about monitoring frameworks and infrastructures. The analysis reports the list of tools identified, and an overview on their main features with respect to ElasTest.

4.4.1 Baseline and comparative analysis

We present in Table 12 the most relevant monitoring tools identified in this second review period. Among them, InfluxData TICK, Zabbix, and Datadog have been described and compared in D2.2 [7] (see Section 4.4), we present here the most important updates of these tools whereas in Table 13 we present a comparison of the new identified monitoring tools in this second review period.

Name	URL	Brief description	License
InfluxData TICK	https://www.influxdata.com/time-series-platform/	Building monitoring and analytics applications.	MIT Licensed
ZABBIX	https://www.zabbix.com/	Zabbix is open source enterprise monitoring tool to track the availability and performance of IT infrastructure components.	OSS
Datadog	https://www.datadoghq.com/	It allows for quickly search, filter, and log analysis. It enables events correlation as well as generation and upload of JSON-formatted dashboards.	MIT and GPL 2
Heapster	https://github.com/kubernetes/heapster	It allows to compute resource usage analysis and monitoring of container clusters.	ASL 2.0
OpenTSDB	http://opentsdb.net/overview.html	Time series database written on top of HBase or hosted Google Bigtable service. It is released under LGPLv2.1+ and GPLv3+ licenses. It allows fast handling of large amount of monitoring metrics.	LGPLv2.1+
Weave Scope	https://www.weave.works/os/scope/	Open source monitoring and visualization engine for Docker and Kubernetes. It can perform automatic topology identification and clustering of containers as well as tracking of key metrics assisting developers.	ASL 2.0
Sentry	https://sentry.io/welcome/	Sentry is open source distributed tracing framework which tracks and reports errors as and when they occur. It supports alerting, notification, issue assignments.	BSD 3-clause
Graylog	https://www.graylog.org/	Graylog supports monitoring of logs and other metrics at scale. The community version components are	GPL 3.0

		released under mixed GPL-3.0 and ASL 2.0 licenses. Core components are under GPL-3.0 license. The solution is developed over Elasticsearch and MongoDB clusters.	
Jaeger	https://www.jaegertracing.io/	A CNCF supported ASL2.0 distributed tracing framework open sourced by Uber. It is especially suited for microservices based distributed system.	ASL 2.0
Netdata	https://my-netdata.io/	Netdata is a lightweight monitoring, visualization solution for tracking large amount of metrics across physical machines, VMs, containers, etc. It supports alarms, and data archival. Netdata is released under GPL-3.0 license.	GPL v3+

Table 12: Monitoring Tools***InfluxData TICK***

The system architecture of InfluxData TICK generally kept the structure we presented in deliverable D2.2 (Section 4.4.1.1) [7] but a slight refocus on stream processing of monitored data has happened. The revised architecture is shown in Figure 14. The components names have been modified whereas the overall process workflow remains the same.

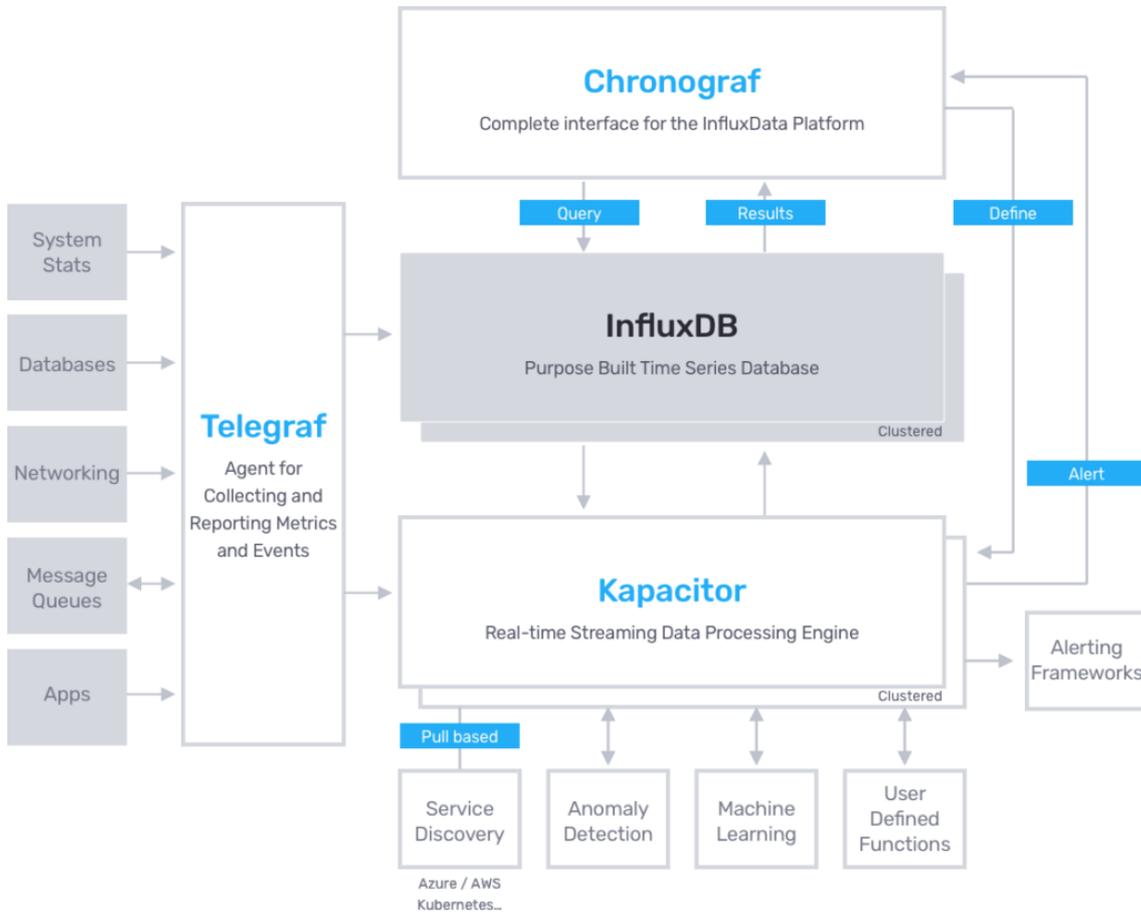


Figure 14: InfluxData TICK architecture

Zabbix

The overall architecture for Zabbix has undergone no major updates. The framework itself has advanced filtering capabilities but in line with the analysis presented in D2.2 (Section 4.4.1.2) [7].

Datadog

Datadog has extended the range of capabilities by adding application performance monitoring and tracing capabilities in general. They now support integration with Prometheus and open-metrics. No major updates to architecture were observed since our initial report in D2.2 (Section 4.4.1.3) [7].

	Heapster	OpenTSDB	Weave Scope	Netdata	Sentry	Graylog	Jaeger
OpenSource	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Community activity indicators	1000+ stars on github	3800+ stars on github	3500+ stars on github	40000+ stars on github	21500+ stars on github	4900+ stars on github	8600+ stars on github
Main features	Metrics collection and reporting of kubernetes resources	Time series data store with space optimization backed by HBase store	Automatic real time resource view for Docker and Kubernetes	Real time performance monitoring with support for many platforms	Application monitoring with focus on error tracking	Focuses on log management	A distributed tracing platform
Main users	Kubernetes framework	Yahoo, Pinterest, Cloudflare	Weaveworks in their Weavecloud	Atos, Cisco, Google, Amazon, Newrelic, SAP, and many more	Uber, Airbnb, Paypal, and more	CircleCI, AppBrain	A lot of smaller SaaS providers, IBM in their app cloud deployments

Table 13: Comparison of monitoring tools

4.4.2 Progress within ElasTest

The development of monitoring tools has been especially driven by the activities in CNCF, such as fast adoption of containers and CI/CD driven development. ElasTest project identified these trends early on, the homegrown monitoring solutions - EMS and EMP have already provided key features now available within the ecosystem of monitoring tools. The tools described in both the previous [7] and the current SoTAs are primarily designed for handling system metrics; in particular, InfluxData TICK stack and Zabbix are agent push based systems. In ElasTest, the monitoring tool that has been designed and implemented supports many of the features used in the tools described above, and made it capable of handling both metric data as well as log streams. Most of the described tools support collector/server style architecture. Regarding this aspect, all the collectors which were developed in ElasTest could be easily modified to send metrics into any of the identified platforms described above.

One trend which has become prominent is the increasing availability of tracing solutions easing the tracing of both control, and data flows in large scale distributed systems deployment. The assessment of the monitoring capabilities offered by the EMP on the verticals in the ElasTest project demonstrates how the proposed approach is an innovative means for tracing nowadays strongly demanded solutions in ICT such as microservices, or serverless architectural styles.

4.5 GUI automation and impersonation (IoT Testing)

This section surveys GUI Automation and Impersonation tools. More specifically, the technological area refers to two main aspects: i) general GUI test automation tools; and ii) device emulation tools. The former aspect has been extensively considered in D2.2 [7] and no relevant changes in the SoTA have been identified during this second review period. The latter specifically refers to emulation of IoT devices in order to realize and test IoT applications. In this sense, the rest of the section only focuses on reporting updates for the tools and frameworks in the context of IoT Testing.

4.5.1 Baseline and comparative analysis

Device emulation refers to the approach where a virtual device can take the role of a real device. The process of emulation can help in establishing a device to completely replace the functionalities of a real device. It depends on the use case, for which the user decides the level of emulation required. The level of emulation here refers to the extent the virtual device mimics the real device in terms of its overall aspects. Devices in the context of IoT, refer to sensors, actuators and smart devices. IoT solutions can be complex and may require many devices to come to effect. IoT testing with device emulation aims to reduce the cost to test IoT solutions. In order to sufficiently address the number of devices required to realize a complex application, device emulation helps in providing a solution to first test concepts with virtual devices.

The view point considered here is that, the sensors or actuators cannot be used directly to realize an IoT application. Associated to the sensors and actuators, a nodal device is required which acts as an interface for the network. Sensor and actuator data is obtained by the nodal device and converted to for example data packets or data visualized on a GUI. The main purpose is to emulate the behaviour of the nodal devices.

The services or tools identified in this second review period that are available in the state of the art and that can assist or provide emulated IoT devices in order to realize and test an IoT application are: Patriot IoT Testing Framework; Eggplant; IoT Emulator; Simple IoT Simulator. In the following there is a short description of these tools. Table 14 shows an overview of these tools.

Patriot IoT Testing Framework

It is a framework which enables testing of distributed IoT applications by providing testers with features such as network virtualization and/or interaction with hardware devices all integrated into a single tool. It allows communication with a Docker container and uses Java programming language for network simulation and device emulation. It runs on an OS where Docker is available. The interaction method is with Docker using Java.

Eggplant

Eggplant is a UI based automated testing tool for IoT applications on the SuT where there are already physical IoT devices connected. The advantage that Eggplant offers is

that of automated testing of applications. The interaction is mainly using UI based approach where a machine running Eggplant connects to a SuT using RDP or VNC. A user is able to connect to the Eggplant instance on the SuT in order to configure tests using UI. It has a closed platform. It uses a GUI based programming approach to automate and configure tests and provides data analytics and visualization facilities.

IoT Emulator

It represents an online platform to emulate IoT applications and accelerate prototyping. It is a closed platform. Users can create, customize, integrate and test IoT applications online. It allows on-line debug and re-configuration and provides facilities for collaborating online.

Simple IoT Simulator

It is a simulator that focuses on capturing device data from sensors, actuators and gateways. It allows to learn from the captured data and to replicate them so that the traffic from such devices can be simulated. The simulated traffic is used to test and aid rapid prototyping of applications. The main idea is to simulate the capability of networking technologies that are standard in the current industries. It is a closed platform. It captures and replicates device traffic to test networking technologies. A core feature is to ascertain the capabilities of handling network data from many devices.

Name	URL	Brief description	License
Patriot IoT Testing Framework	https://patriot-framework.io	A framework which enables testing of distributed IoT applications by providing test developers with features such as network virtualization and/or interaction with hardware devices all integrated into a single tool.	NA
Eggplant	http://docs.testplant.com/eggplant-documentation-home.htm	Eggplant is a UI based automated testing tool for IoT applications on the SuT where there are already physical IoT devices connected.	Private
IoT Emulator	http://iot-emulator.weebly.com/	An online platform to emulate IoT applications and accelerate prototyping.	Private
Simple IoT Simulator	https://www.smplsft.com/SimpleIoTSimulator.html	A simulator that focuses on capturing device data from sensors, actuators and gateways. Learn from the captured data and furthermore replicate them so that the traffic from such devices can be	Private

simulated to test networking technologies.

Table 14: IoT Testing tools

As in D2.2 [7], these tools are compared according to the following aspects:

Prerequisite: Any application that the user needs to run and access the facilities to get started with device emulation.

Language: Language/s in which the application is written.

Codeless testing tool: Whether there exists a codeless testing tool such as buttons or another graphical interface which could be used to turn on/off or change behavior of the device.

Type of device emulated: It refers to the type of device emulated. Type of device refers to sensor/actuator hardware or platform.

Runtime environment emulated device: It refers to where the application used to emulate the device is run.

Runtime environment IoT application: It refers to where the IoT application which makes use of the emulated devices is run ultimately.

Interface: The type of interface used for communication between the IoT application and the emulated devices.

Table 15 shows the comparison among these IoT testing tools.

	Patriot IoT Testing Framework	Eggplant	IoT Emulator	Simple IoT Simulator
Prerequisite	Java, Availability of Framework	Availability of the licensed framework	Purchase of a licensed software	Purchase of a licensed software
Language	Java	NA	NA	NA
Codeless testing tool	No	YES	NA	NA
Type of emulated device	Framework emulates network, data generation and network routing	The framework provides automated testing features.	NA	Data exchanges on the network are learnt which can be manifested into the many devices and gateways. Concentrates mainly on networking.

Runtime environment for emulated device	Windows, OS X, Linux	Windows	Cloud/online virtual environment	NA
Runtime environment for IoT application	Docker containers, Windows, OS X, Linux	Via RDP or VNC connected UI	Cloud/online virtual environment	NA
Interface	API	UI	NA	NA

Table 15: Comparison of IoT Testing Tools

4.5.2 Progress within ElasTest

Firstly, the main progress is to increase the ease of device emulation and using the emulated device to form IoT applications. Secondly, testing such IoT applications should be relatively easier as compared to the tools presented. The emphasis is towards rapid prototyping and testing of IoT solutions. The test support service, ElasTest Device Emulator Service (EDS) focuses on the above mentioned improvements compared to the tools already mentioned in this document. The ease of deploying and testing IoT solutions for a user further narrows down the approach of ElasTest to use virtualization technologies such as Docker which can be used to implement tests and solutions at user defined platforms. Emphasis is given towards end to end testing which signifies that emulated device mimic the nodal behavior rather than hardware details. Device emulation acts as an economical approach towards validating the features of an IoT solution. Typically, an IoT application is realized by implementing a test bed containing physical devices connected in small scale to establish a proof of concept. The challenge of testing large scale test beds involves the usage of multiple physical devices towards establishing a proof of concept which can be expensive. ElasTest brings together device emulation and test orchestration. The feature of device emulation provides facilities to emulation of physical devices. Many of the tools listed before provide the feature of device emulation as cloud services or proprietary software. ElasTest provides the feature of device emulation as an open source implementation that is implemented using OpenMTC, yet another open source reference implementation of industrial machine to machine communication protocol called oneM2M. Accompanying the strength of open source device emulation feature, ElasTest provides the facility of test orchestration which simplifies the activity of IoT testing.

4.6 Cloud Instrumentation

Cloud Instrumentation refers to the management and orchestration of virtualized resources exposing services over cloud environment. The crucial importance of this technological area is grounded on the ongoing and constantly increasing trend in offering both cloud-based and cloud-native applications.

4.6.1 Baseline and comparative analysis

Table 16 shows a list of cloud instrumentation tools. Among them, the most relevant ones are: Ansible, Chef habitat, MaaS, Terraform, OSM, Marathon, GKE, BOSH.

Several tools have been utilized in the ElasTest Platform Manager (EPM) so far. In the latest version, a new Ansible-based EPM adaptor has been developed that allows deploying Kubernetes cluster on OpenStack.

Table 17 shows a comparison of these cloud instrumentation tools. As in D2.2 [7], the following characteristics are considered for proper comparison of the proposed solutions:

- Model of abstraction: Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS), Container-as-a-Service (CaaS);
- Type of virtualized resources: Virtual Machines, Containers;
- Monitoring support: internal, external;
- Runtime Management: autoscaling of resources, update of services;
- Activity;
- License: OSS, Proprietary;
- Management: API, CLI, SDK, Dashboard.

In the following, a short description of these tools is presented.

Name	URL	Brief description	License
Marathon	https://mesosphere.github.io/marathon/	A production-grade container orchestration platform for Mesosphere's Datacenter Operating System (DC/OS) and Apache Mesos.	Apache-2.0
ServiceNow Cloud Management	https://www.servicenow.com/products/cloud-management.html	The solution offers a standard operating approach to hybrid and public clouds. It provides on-demand access to multi-cloud resources which is fully automated.	Commercial
OSM	https://osm.etsi.org/	Open Source MANO is an ETSI-hosted project to develop an Open Source NFV Management and Orchestration (MANO) software stack aligned with ETSI NFV.	Apache-2.0

CloudForms	https://www.redhat.com/en/technologies/management/cloudforms	Red Hat® CloudForms® is an infrastructure management platform that allows IT departments to control users' self-service abilities to provision, manage, and ensure compliance across virtual machines and private clouds.	Commercial
ManageIQ	http://manageiq.org/	ManageIQ is an open source management platform for Hybrid IT. It can manage small and large environments, and supports multiple technologies such as virtual machines, public clouds and containers.	Apache-2.0
ECS	https://aws.amazon.com/ecs/	Amazon Elastic Container Service (Amazon ECS) is a highly scalable, high-performance container orchestration service that supports Docker containers and allows to easily run and scale containerized applications on AWS. Amazon ECS eliminates the need to install and operate the own container orchestration software, manage and scale a cluster of virtual machines, or schedule containers on those virtual machines.	Amazon license
AKS	https://azure.microsoft.com/en-us/services/kubernetes-service/	The fully managed Azure Kubernetes Service (AKS) makes deploying and managing containerized applications easy. It offers serverless Kubernetes, an integrated continuous integration and continuous delivery (CI/CD) experience, and enterprise-grade security and governance.	Microsoft license
GKE	https://cloud.google.com/kubernetes-engine/	Kubernetes Engine (GKE) is a managed, production-ready environment for deploying containerized applications. It brings latest innovations in developer productivity, resource efficiency, automated operations, and open source flexibility to accelerate time to market.	Google license (Free for master node)
Terraform	https://www.terraform.io	Terraform is a tool for building,	MPL-2.0

		changing, and versioning infrastructure safely and efficiently. Terraform can manage existing and popular service providers as well as custom in-house solutions. The infrastructure Terraform can manage includes low-level components such as compute instances, storage, and networking, as well as high-level components such as DNS entries, SaaS features, etc.	
BOSH	https://bosh.io/docs/	It is a project that unifies release engineering, deployment, and lifecycle management of small and large-scale cloud software. BOSH supports multiple Infrastructures as a Service (IaaS) providers like VMware vSphere, Google Cloud Platform, Amazon Web Services EC2, Microsoft Azure, and OpenStack.	Apache-2.0
Chef habitat	https://www.habitat.sh/	Habitat is a new approach to automation that focuses on building, deploying, and managing applications that can be run anywhere, from bare metal servers and VMs to containers and PaaS solutions.	Apache-2.0
MaaS	https://maas.io/	Self-service, remote installation of Windows, CentOS, ESXi and Ubuntu on real servers turns data center into a bare-metal cloud (Metal-as-a-service).	AGPL-3.0
Pulumi	https://www.pulumi.com/docs/index.html	A platform for building and deploying cloud infrastructure and applications in the favorite language on any cloud. Pulumi is open source, free to start, and has plans available for teams.	Apache-2.0
Puppet	https://puppet.com/products/why-puppet	It delivers infrastructure faster, no matter where it lives, using leading agile practices.	Apache-2.0
Ansible	https://www.ansible.com/overview/how-ansible-works	It is a radically simple IT automation engine that automates cloud provisioning, configuration management, as	GPL-3.0

well as application deployment
and intra-service orchestration.

Table 16: Cloud Instrumentation Tools

Marathon

Marathon is a production-grade container orchestration platform for Mesosphere's Datacenter Operating System (DC/OS) and Apache Mesos. The main features offered by Marathon are described as follows:

- High Availability: Run as an active/passive cluster with leader election for 100% uptime.
- Multiple container runtimes: Support both Mesos containers (using cgroups) and Docker.
- Stateful applications: Can bind persistent storage volumes to the application.
- Beautiful and powerful UI
- Constraints: Allow to e.g. place only one instance of an application per rack, node, etc.
- Service Discovery & Load Balancing
- Health Checks: Evaluate the application's health using HTTP or TCP checks.
- Event Subscription: Supply an HTTP endpoint to receive notifications.
- Metrics: Query them in JSON format, push them to systems like Graphite, StatsD and DataDog, or scrape them using Prometheus.
- REST API: for easy integration and scriptability.

Google Kubernetes Engine (GKE)

GKE is a managed, production-ready environment for deploying containerized applications. It allows users to get up and running with Kubernetes in no time, by completely eliminating the need to install, manage, and operate the own Kubernetes clusters. The main features offered by GKE are:

- Wide Variety of Applications: It isn't just for stateless applications; it allows attaching persistent storage, and even running a database in the cluster.
- Hardware accelerators: makes it easy to run Machine Learning, General Purpose GPU, High-Performance Computing, and other workloads that benefit from specialized hardware accelerators.
- Operate Seamlessly with High Availability: it allows to control the environment from the built-in Kubernetes Engine dashboard in Google Cloud console. It uses routine health checks to detect and replace hung, or crashed, applications inside deployments. Container replication strategies, monitoring, and automated repairs help ensure that the developed services are highly available and offer a seamless experience to the users.
- Scale Effortlessly: Go from a single machine to thousands to meet customer demands.

MaaS

MAAS is freely available, open source server provisioning software from Canonical. The main features offered by MaaS are:

- Speed: Zero-touch deployment of Ubuntu, CentOS, Windows and RHEL. Full deployment time is approximately two boot cycles plus two minutes for disk imaging.
- Cloud metadata: Reuse standard cloud operations with cloud-init and metadata services.
- Storage layouts: Create advanced filesystem layouts with RAID, bcache, LVM, ZFS and more. Automate storage configuration through APIs.
- Network monitoring: Continuously observes network traffic and catalogs every active IP address of unknown origin. Discovers rogue devices, IPs and MAC addresses. Drives active scanning of network ranges.
- Authentication and Identity: Integrate with LDAP, Active Directory or SAML for central identity management and single-sign-on across multiple MAAS regions.
- DevOps on bare-metal: Integration with Chef, Puppet, SALT, Ansible, Conjure-up, and Juju.
- REST API, CLI and Python bindings: Enable full lifecycle and project automation.

Opensource MANO (OSM)

Open Source MANO is an ETSI-hosted project to develop an Open Source NFV Management and Orchestration (MANO) software stack aligned with ETSI NFV. The goal of OSM is the development of a community-driven production quality E2E Network Service Orchestrator (E2E NSO) for telco services, capable of modelling and automating real telco-grade services, with all the intrinsic complexity of production environments. It provides a way to accelerate maturation of NFV technologies, enable a broad ecosystem of VNF vendors, and test and validate the joint interaction of the orchestrator with the other components it has to interact with: commercial NFV infrastructures (NFVI+VIM) and Network Functions (either VNFs, PNFs or Hybrid ones). The main features offered by MANO are:

- Well-known Information Model (IM): aligned with ETSI NFV, that is capable of modelling and automating the full lifecycle of Network Functions (virtual, physical or hybrid), Network Services (NS), and Network Slices (NSI).
- Unified northbound interface (NBI): it enables the full operation of system and the Network Services and Network Slices under its control.
- Extended concept of “Network Service”: An NS can span across the different domains identified (virtual, physical and transport) and therefore control the full lifecycle.
- Network slicing for 5G: Manage the lifecycle of Network Slices.
- Monitoring Metrics and Alarms: Metrics collection is now supported from both the infrastructure (VIM) and directly from VNFs. OSM is now able to create, manage and trigger alarms based on infrastructure or VNF events and metrics.

- Service Assurance with Autoscaling VNFs: Through the Policy Manager and in coordination with LCM orchestrator, OSM is now capable of automating the horizontal scaling decisions.

Ansible

Redhat Ansible is an open source simple IT automation engine that automates cloud provisioning, configuration management, application deployment, intra-service orchestration, and many other IT needs. It uses a simple language (YAML, in the form of Ansible Playbooks) that allows for describing the automation jobs and includes hundreds of modules to support a wide variety of integrations. Its main features are:

- Simple to set up and use: No special coding skills are necessary.
- Powerful: Ansible allows to model even highly complex IT workflows.
- Flexible: It is possible to orchestrate and customize the entire application environment no matter where it's deployed.
- Agentless: It is not needed to install any other software or firewall ports on the client systems to automate.
- Cloud Integration: Ansible can deploy to bare metal hosts, virtualized systems and various cloud environments.

Chef habitat

Habitat is open source software that creates platform-independent application artifacts and provides built-in deployment and management capabilities. Its main features are:

- Fast: Package services easily with a simple Plan file to build the application as a Habitat artifact.
- Flexible Deployments: Publish Habitat artifacts via the build service, and run on traditional systems, or export to containers to deploy anywhere.
- Traceable Content: Audit the configuration, dependencies, and health of each artifact built with Habitat via the Supervisor API.
- Intelligent Run-time Management: Cluster components through gossip based service groups to enable dynamic configuration updates, leader elections, rolling deployments, and more.

Terraform

Terraform is a tool for building, changing, and versioning infrastructures safely and efficiently. The key features of Terraform are:

- Infrastructure as Code: Infrastructure is described using high-level configuration syntax. This allows a blueprint of the datacenter to be versioned and treated as any other code. Additionally, infrastructure can be shared and re-used.

- Execution Plans: Terraform has a "planning" step where it generates an execution plan. The execution plan shows what Terraform will do when you call apply. This allows avoiding any surprise when Terraform manipulates infrastructure.
- Resource Graph: Terraform builds a graph of all resources, and parallelizes the creation and modification of any non-dependent resource. Because of this, Terraform builds infrastructure as efficiently as possible, and operators get insight into dependencies in their infrastructure.
- Change Automation: Complex changesets can be applied to the infrastructure with minimal human interaction. With the previously mentioned execution plan and resource graph, it is possible to know exactly what Terraform will change and in what order, avoiding many possible human errors.

BOSH

BOSH is a project that unifies release engineering, deployment, and lifecycle management of small and large-scale cloud software. BOSH can provision and deploy software over hundreds of VMs. It also performs monitoring, failure recovery, and software updates with zero-to-minimal downtime.

BOSH was developed to deploy Cloud Foundry PaaS, it can also be used to deploy almost any other. BOSH is particularly well-suited for large distributed systems and supports multiple Infrastructure as a Service (IaaS) providers.

	Marathon	GKE	MaaS	OSM	Ansible	Chef habitat	Terraform	BOSH
Abstraction	CaaS	Caas	MaaS	VNFaaS, CaaS	CaaS, PaaS, MaaS	CaaS, MaaS, PaaS,	IaC, CaaS, MaaS, PaaS,	PaaS
Model Virtualized resources	Container	Container	VM, bare-metal	VM, Container	Container, VM, bare-metal	Container, VM, bare-metal	Container, VM, bare-metal	VM
Monitoring support	Built-in	Built-in	Built-in	Built-in	(Modules)	Built-in	Built-in	Built-in
Runtime Management	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Activity	High	High	High	High	high	high	high	high
License	OSS (Apache 2.0)	Google Software License	OSS (AGPL 3.0)	OSS (Apache 2.0)	OSS (GPL 3.0)	OSS (Apache 2.0)	OSS (MPL 2.0)	OSS (Apache -2.0)
Management	APIs, GUI, SDKs	CLI, GUI, APIs/SDKs	CLI, GUI, APIs	RESTful API, CLI, GUI, SDKs	APIs, CLI, SDKs	APIs, CLI, SDKs	RESTful API, CLI	CLI

Table 17: Comparison of Cloud Instrumentation Tools

4.6.2 Progress within ElasTest

The ElasTest platform and, in particular, the ElasTest Platform Manager (EPM) abstracts several cloud instrumentation technologies. Specifically, the EPM is an intermediate component within the whole platform that exposes a well-defined interface that is technological agnostic and that hides the actual binding with several cloud services. As a consequence, the EPM is able to deploy and to execute seamlessly cloud services in the target cloud infrastructures where the consumer of the EPM does not need to care about the underlying infrastructure. The EPM makes use of provided capabilities (e.g. autoscaling, healing functionalities, QoS) of certain cloud technologies whereas missing capabilities of those are compensated within the EPM. To make the EPM fully agnostic of the underlying infrastructure and technology in use, the information of deployed services is also abstracted and managed by means of a common information model.

4.7 Dashboard Management

The following sections survey the technological area of Dashboard Management. The area includes tools and frameworks supporting both the generation of charts from collected logs, and the reaction to the filtering/aggregation requests from the users.

4.7.1 Baseline and comparative analysis

As shown in Table 18, the most relevant dashboard management tools are: Prometheus/Grafana in Kubernetes environments and Honeycomb.io. We provide a short description of them in the following and a comparison of these tools in Table 19 .

Prometheus/Grafana

Prometheus is the defacto standard monitoring tool for Kubernetes clusters. It can ingest logs and metrics from the different applications and services running within the Kubernetes cluster. This information can be queried and visualized using Grafana, a dashboard tool with a good integration with Prometheus, although Prometheus also offers an interface to visualize this information. Main features of Grafana are:

- Time-series oriented. Grafana is specifically designed to visualize time-series data, which is very appropriate for applications and services, and allows a proper visualization of the different events that happen in a running system.
- Alerting: It visually defines alert rules for the most important metrics. Grafana continuously evaluates and sends notifications to systems like Slack, PagerDuty, VictorOps, OpsGenie.
- Dynamic Dashboards: Create dynamic & reusable dashboards with template variables that appear as dropdowns at the top of the dashboard.

- Explore Metrics: It allows: i) to explore data through ad-hoc queries and dynamic drilldown; ii) to split view and compare different time ranges, queries and data sources side by side.
- Explore Logs: It allows the possibility of switching from metrics to logs with preserved label filters. It allows to quickly search through all logs or streaming them live.
- Mixed Data Sources: Mix different data sources in the same graph. It is possible to specify a data source on a per-query basis.
- Annotations: Annotate graphs with rich events from different data sources. Hover over events shows the full event metadata and tags.
- Ad-hoc Filters: Ad-hoc filters allow creating new key/value filters on the fly, which are automatically applied to all queries that use that data source.

Honeycomb

Honeycomb is a tool for introspecting and interrogating the production system. It can gather data from any source: from the clients (mobile, IoT, browsers), vendored software, or the own code. Single-node debugging tools miss crucial details in a world where infrastructure is dynamic and ephemeral. Honeycomb is a new type of tool, designed and evolved to meet the real needs of platforms, microservices, serverless apps, and complex systems. Main features of Honeycomb are:

- Instrumentation: It gives a quick start that automatically sends raw event data to Honeycomb and ingests structured log data.
- Intuitive UI/UX: Makes it easy to proactively observe user impact as code is released. Dive in to investigate any production incident using interactive real-time charts.
- Rich, Blazing Fast Query: Gives incredible flexibility and customization to help rapidly find and understand problems by letting you ask any question.
- Distributed Tracing Navigation: Efficiently navigate the full breadth of tracing data without missing crucial details that can help resolve issues.
- BubbleUp: Delivers instant, automatic detection of often-hidden outliers that are behaving differently from the baseline.
- Team Collaboration: Preserve and share how others have resolved problems previously, saving time and elevating everyone.

Name	URL	Short Description	License
Prometheus / Grafana	https://prometheus.io/docs/visualization/grafana/	Prometheus is an open-source systems monitoring and alerting toolkit.	Open source (Apache License 2.0)
HoneyComb	https://www.honeycomb.io/	Honeycomb is a tool for introspecting and interrogating the production systems about	-

the log data stored in
Elasticsearch clusters.

Table 18: Dashboard Management Tools

	Prometheus/Grafana	Honeycomb
OS	Any	-
GUI	Yes	Yes
Test Language	Scripting Language	-
Extension Language	Scripting Language	-
Protocols	HTTP, HTTPS, Socket	-
Open Source	Yes	No
Detailed Documentation	Yes	Yes

Table 19: Comparison of Dashboard Management Tools

4.7.2 Progress within ElasTest

Most of the dashboard tools were thought to ingest and visualize metrics coming from a running system. However, in a continuous testing environment, these tools are not able to tell whether the metrics correspond to one test or another. ElasTest enables testers to segregate data depending on the test that was exercising the SUT.

4.8 WebRTC Testing

4.8.1 Baseline and comparative analysis

A relevant tool for webRTC testing identified in this second review period is KITE tool⁴, which has been developed by Google and CoSMo. It is an open source testing engine that allows implementing automated interoperability tests for WebRTC. With KITE it is possible to implement tests using Java or Javascript and run them on almost any platform. KITE supports:

- all web browser: Chrome, Firefox, Safari, Edge, Opera, on all OS (Linux, Windows, Mac, iOS and Android)
- Mobile Native Apps on Android, iOS
- Desktop Native Apps on Windows and MacOS
- Electron Apps

⁴ <https://webrtc.org/testing/kite/>

KITE also provides additional features such as: load testing; network instrumentation in the cloud or on premises, for all platforms (browser and native). KITE is intended to be a tool to help the user to implement its WebRTC test effortlessly. Table 20 and Table 21 provide further details about KITE.

Name	URL	Short Description	License
Kite	https://github.com/webrtc/KITE	KITE is an open source test tool to test interoperability of WebRTC across browsers. KITE makes it easy to test interoperability of WebRTC applications and detect regressions early.	Apache License Version 2.0

Table 20: WebRTC Testing Tools

Kite	
OS	Linux, Windows, Mac, iOS and Android
GUI	No
Support for	Browser/Native Mobile Apps/Desktop Apps/Electron Apps
Test Recorder	No
Load Testing	Yes
Test Language	Java/Javascript
Extension Language	Java
Open Source	Yes
Detailed Documentation	Yes

Table 21: KITE features

4.8.2 Progress within ElasTest

In addition to the information provided by KITE, ElasTest is also able to provide both QoE metrics, and a whole set of infrastructural measures (e.g. CPU, memory, or network traffic, etc.) for the SUT or any other device involved in the test.

4.9 Cross-browser Testing

4.9.1 Baseline and comparative analysis

A cross-browser testing tool identified in the second review period is Browsersync⁵⁶. It is an automation tool that makes web development faster, keeping multiple browsers and devices in sync when building websites. Main features of Browsersync are:

⁵ <https://www.browsersync.io/>

⁶ <https://www.sitepoint.com/improve-workflow-browsersync-2-0/>

- Install and run anywhere: Built on Node.JS_ENTRY to support Windows, MacOS and Linux. It allows setup in less than 5 minutes.
- Free to run and reuse: Browsersync is an open source project available to use under the Apache 2.0 License.
- Build-tool compatible: Easily integrated with task runners like Grunt and Gulp, or included in other Node projects.
- Network Throttle: It allows testing the website against a slower connection, even when devices are connected to wifi.
- Interaction sync: Your scroll, click, refresh and form actions are mirrored between browsers while you test.
- File sync: Browsers are automatically updated as you change HTML, CSS, images and other project files.
- UI or CLI control: Run the new browser-based UI for quick control, or stick with the original command line usage.
- Sync customisation: Toggle individual sync settings to create your preferred test environment.
- URL history: Records your test URLs so you can push them back out to all devices with a single click.
- Proxy server: If you're already using a local web server or need to connect to a live website, you can start BrowserSync as a proxy server. Table 22 and Table 23 show further details about Browsersync.

Name	URL	Short Description	License
Browsersync	https://www.browsersync.io/	Time-saving synchronised browser testing.	Open source (Apache License 2.0)

Table 22: Cross Browser Testing Tools

Browsersync	
OS	Any
GUI	Yes
Test Recorder	does not apply
Test Language	Javascript
Extension Language	Javascript
Protocols	HTTP, HTTPS
Open Source	Yes
Detailed Documentation	Yes

Table 23: Browsersync features

4.9.2 Progress within ElasTest

ElasTest promotes the distributed adoption of cross-browser testing. As from our technical SoTA, *browsersync* only works locally. Differently, ElasTest cross-browser capabilities can be enacted on top of any distributed infrastructure, for example based

on Docker or Kubernetes. Therefore, while testing with ElasTest, testers are able to plan and to emulate user actions in many different browsers at the time in the same or different machines.

4.10 Mobile Testing

This section reports about the technological area covering automated services and tools for in-depth functional tests of mobile apps. The area also includes solutions that allow for the validation/reactiveness of software systems against a variety of mobile devices.

4.10.1 Baseline and comparative analysis

The list of mobile testing tools has been updated with the new players on market, following the same strategy adopted in D2.2 [7]. In addition to the characteristics included in the comparison of mobile testing tools of D2.2 [7], the current version includes “Cloud version” as a new feature to take into account, according with the trends of the new tools for Mobile Testing. For completing this comparison there has been considered the rankings registered on “TOP 15 Best Mobile Testing Tools In 2019 For Android & IOS”⁷, “14 Best Mobile App Testing Tools for Android & iOS (2019)”⁸ and “A Complete List of Mobile Application Testing Tools”⁹. Since the delivery of D2.2 [7], the mobile testing tools have been maturing and evolving from Open Source and on premise installation to more professionalized platforms, with on cloud versions and assuming the SAAS model of commercialization. The variants of the service are based on the number of devices, number of nominal or simultaneous users, the assignment of a dedicated lab or shared devices for mobile testing. All these parameters allow customizing the service to the requirement of every customer.

The mobile testing tools identified in this second review period are: TestComplete; Experitest; Test IO; Kobiton; iOS UI Automation; UIAutomator (Android); KepptFunctional (KIF). Table 24 shows an overview of these tools whereas Table 25 provides a comparison among them.

Name	URL	Short Description	License
TestComplete	https://smartbear.com/product/testcomplete/mobile-testing	It is a platform that allows creating and running repeatable and robust UI tests across native or hybrid mobile apps. It also allows for automation of UI actions or user scenarios on real devices or emulators using script-free record and replay or by creating automated scripts in different languages, such as Python,	Proprietary

⁷ <https://www.softwaretestinghelp.com/best-mobile-testing-tools/>

⁸ <https://www.guru99.com/mobile-testing-tools.html>

⁹ <https://www.edureka.co/blog/mobile-testing-tools/>

		VBScript, Jscript, and JavaScript.	
Experitest	https://experitest.com/mobile-testing/	It is a platform for creating Appium tests with ease using a device reflection, test recorder, and object spy. It allows for automation tests for native, hybrid and web apps. It allows for exportation of Appium tests to any Appium client (e.g. Java, C#, Ruby, Python). It is integrated with any IDE and testing framework.	Proprietary
test IO	https://get.test.io/mobile-testing/	test IO is a leading SaaS platform for software crowd testing: it allows the continuous testing of web and mobile applications by skilled human testers using real devices on iOS, Android and web versions.	Proprietary
Kobiton	https://kobiton.com/	Kobiton is a mobile device cloud platform that provides access to real devices for running manual and automated tests on native, web and hybrid Android/iOS apps. Built on top of the Appium open-source framework. It allows testing across devices without script modifications.	Proprietary
iOS UI Automation (iOS)	https://help.apple.com/instruments/mac/current/	OS UI Automation is Apple's open-source test automation framework specifically for iOS apps. It helps to automate interface tests through test scripts. JavaScript programming interface is used to specify actions to be performed on device UI. It does not work well with other tools, methodology, and framework as it is a proprietary tool. It helps to reduce procedural efforts and time needed for software product development.	Open Source
UIAutomator (Android)	https://developer.android.com/training/testing/#UIAutomator	UI Automator is an open-source framework which allows testing the UI using automated functional test cases. Able to run against an app on one or more devices. The UI Automator API is packaged in the UI Automator.jar file under the /platforms/ directory. This API includes classes interfaces and exceptions. UI Automator framework uses the scripts that are written in JavaScript.	Open Source
KeepItFunctional (KIF)	https://github.com/kif-framework/KIF	KeepItFunctional (KIF) is an iOS integration test framework used for Functional Testing that builds and performs test cases using standard XCTest testing target. It is an Open-Source framework designed to test	Open Source

mobile app UI and allows easy automation testing of iOS apps.

Table 24: Mobile Testing Tools

	Open Source	iOS	Android	Native	Hybrid	Web Apps	Cloud version
Test Complete	-	+	+	+	+	+	-
Experitest	-	+	+	+	+	+	+
Test IO	-	+	+	+	+	+	+
Kobiton	-	+	+	+	+	+	+
iOS UI Automation (iOS)	+	+	-	+	-	-	-
UIAutomator (Android)	+	-	+	+	+/-	-	-
KeepItFunctional (KIF)	+	+	-	+	-	-	-

+: Available, -: Not Available, +/-: Partially available

Table 25: Comparison of Mobile testing Tools

4.10.2 Progress within ElasTest

As it was mentioned in DoA [1], among the possible research directions covered by the ElasTest project there was the development of specific modules/components for Mobile Testing and to embed them into Elastest platform. However, as already clarified in the deliverable D2.2 [7], the evolution of Elastest with respect to Mobile Testing was temporarily suspended.

In the period after the last project review meeting, the consortium decided to focus on improving all the features developed so far, thus the activities about Mobile Testing features were not resumed. In conclusion, the ElasTest project does not contribute to any progress in SotA regarding Mobile Testing, as the released platform does not include any dedicated component/module supporting these features.

4.11 Cognitive Q&A Systems

This section addresses a technological area that focuses on automatic recommender systems specifically conceived or trained in order to be applied in the context of some activity about software testing. Differently from the survey in other technological areas, a major evidence from the following analysis is that few industrial tools/frameworks have been actually identified. Indeed, most of the outcomes resulted from this survey were advanced research approaches or prototypes rather than well established solutions ready to be applicable.

4.11.1 Baseline and comparative analysis

Table 26 presents five tools developed after ElasTest project has commenced. All of them except for Aroma [18] target specifically testing domain. Aroma is designed for general code recommendation, and can be applied to test code. It has been included because its purpose is very similar to one of the functionalities implemented in ElasTest Test Recommendation Engine (ERE), namely using machine learning to retrieve relevant programming methods from a large repository. The core difference is that the input to Aroma is a code snippet written by a human user, whereas ERE asks for a natural language description of a test case, generates test code and then uses that generated code as a search query. In Table 27 these five tools have been compared according to the problem description, the used data and the implemented algorithm.

Name	URL	Brief description	License
Aroma[18]	https://ai.facebook.com/blog/aroma-ml-for-code-recommendation/	A tool for code recommendation via structural code search. It takes a code snippet as a query and searches an indexed source code repository for method bodies containing the query snippet. Clusters and intersects the results.	Copyright hold by the authors
TestDescriber[19]	https://zenodo.org/record/45120#.XUi8tZJKiqQ	A tool that combines summarization approaches with code coverage information. It automatically generates test case summaries (natural language descriptions of JUnit test cases and the portion of the target classes they are going to test).	MIT License
Weak-Assert[20]	http://congwang92.cn/weakassert/	A weakness-oriented assertion recommendation toolkit for program analysis of C code. It matches abstract syntax trees of source code to pre-defined weakness patterns and inserts assertions into programs.	Copyright held by the authors; third party CVE - copyright held by Mitre
GuideGen[21]	https://github.com/hotomski/guidegen https://youtu.be/4uXqP3mwmAo	A web application to support requirement management. It analyses changes in requirements, automatically generates guidance on how to adapt the affected acceptance tests and send notifications to subscribed users.	Copyright held by the authors
SoTesTeR[22]	-	A content-based recommender system that offers a ranking of	Copyright held by the authors

software testing techniques based on a target project characterization and evaluation of testing techniques in similar projects.

Table 26: Cognitive Q&A Tools

	Aroma	TestDescriber	Weak-Assert	GuideGen	SoTesTeR
Problem Description	Given a partial code snippet, search a large source code repository and return code representing idiomatic coding patterns to extend/complete the snippet.	Improve readability of automatically generated unit test cases.	Given the code, recommend appropriate assertion patterns and where to insert the recommended assertions.	Given a requirement change, adapt associated acceptance tests.	Given project characteristics, recommend most appropriate testing techniques.
Data	Large source code repositories	Code under test	Code under test	A collection of existing requirements and acceptance tests	Characterization scheme for software testing techniques as similarity attributes and performance attributes; Repository storing characterized techniques and historical projects.
Algorithm	Search based on human-engineered features; Prune and re-rank (Jackard distance as the similarity metric); Cluster and intersect.	Test generation using a chosen external automation tool; Test coverage analysis based on Cobertura; Summary generation based on SWUM (Software Word Usage Model).	Parse code to extract abstract syntax trees (using an external library); Match manually crafted assertion patterns to nodes in ASTs.	Sentence- and word-level analysis of the requirements to identify modifications/deletions/additions, using off-the-shelf NLP tools (Stanford CoreNLP, Text_Diff, SyntaxNet); Rule-based algorithm for determining	Characterize new target project: Ranking elaboration: TOPSIS method for determining similarity between the target project and historical projects – use weighted similarity attributes kNN to select k similar projects; use performance attributes for

relevance and generating guidance.	final ranking: Characterize instantiated techniques to provide feedback.
------------------------------------	--

Table 27: Comparison of Cognitive Q&A Tools

4.11.2 Progress within ElasTest

A recent systematic review on recommender systems applied to testing [23] shows that the solutions designed to support test cases creation are concerned with finding existing code suitable for reuse rather than with generating new code. One exception is a system [20] (developed after ElasTest project commencement) generating template-based code snippets that implement weakness-oriented assertions. The main limitations of this solution are the narrow scope of application: only assertion statements are supported, and only those that match predefined, human-crafted patterns.

ElasTest Recommendation Engine (ERE) generates complete implementations of unit test cases based on short descriptions in natural language. To our knowledge, there is no existing tool offering this type of support. Rather than using predefined templates, ERE leverages deep learning to generate unseen code token by token. The core of ERE is Neural Translation Model which can be trained on vast amounts of data stored in online software repositories, and also fine-tuned to the needs of a specific project.

Furthermore, ERE solution offers a novel way of recommending existing reusable test code. While other systems determine similarity of code by measuring token overlap or finding longest common subsequence of tokens, ERE learns deep vector representations (embeddings) of test cases and computes the distance between them in the semantic vector space. Such vector representations reflect not only individual tokens or fixed sequences of tokens, but are able to capture subtle patterns and broad context of tokens and sequences.

5 Summary of ElasTest Outcomes, Progresses and Benefits

This section presents the main outcomes, progresses and benefits of ElasTest with respect to the SotA. Specifically, Section 5.1 summarizes for each technologic area addressed in the project, the main project outcome and the progress of ElasTest for that specific area. Section 5.2 shows the overall main benefits of ElasTest.

5.1 ElasTest Outcomes and Progresses

Table 28 presents for each technological area addressed during the overall project duration, the ElasTest outcome and progress with respect to the SotA. Finally, the last column of the table reports references to more detailed information of ElasTest outcomes and progresses.

Technological Area	ElaSTest Outcome	ElaSTest Progress	Evidences and further information
Continuous Integration	ElaSTest eases the application of continuous testing principles inside CI platforms. Specifically, ElaSTest exposes the whole set of its functionalities for managing tests by means of an open API. In this sense, the ElaSTest Jenkins Plugin is a dedicated module of ElaSTest that directly enables the interaction between any Jenkins CI instance with ElaSTest.	ElaSTest is providing an Open API that other servers, or plugins can make use of in order to integrate ElaSTest functionality into their core.	D2.3 [25]
Performance Testing	ElaSTest Orchestration Engine (EOE) supports configuration management mechanisms and allows the parallel combination of test cases so that to achieve the execution of test cases in realistic operative testing scenarios.	Testing SiL by running different conditions (configurations) and comparing the results among them.	D2.3[25] D4.3[26]
Security Testing	ElaSTest Security Services (ESS) supports security testing of cloud-based web applications. It supports the identification of common web application weaknesses and provides advanced functionalities such as automatic attack page generation.	Detection of sophisticated attack classes and Cross-Origin State Inference (COSI) attacks.	D5.1[28]
Monitoring	ElaSTest Monitoring Service (EMS) used for inspecting executions of a System Under Test. Elastest Monitoring Platform (EMP) allows to monitor the health of various components of ElaSTest platform as well as correlated queries aiding the fault location within the platform in an optimized manner.	Monitoring solution capable of handling both metric data as well as log streams. Tracing nowadays strongly demanded solutions in ICT such as microservices, or serveless architectural styles.	D5.1[28] D3.1[31]
IoT Testing	ElaSTest Device Emulator Service (EDS) offers capability of emulating sensors, actuators and smart devices on all types of IoT SUTs.	Device emulation as an open source implementation leveraging OpenMTC.	D5.1[28]
Cloud Instrumentation	ElaSTest Platform Manager (EPM) allows to deploy and execute seamlessly cloud services in the target cloud infrastructure. It provides an interface between ElaSTest components and the cloud infrastructure where ElaSTest is deployed by abstracting and managing the information of deployed services by means of a common information model.	Full abstraction of the underlying infrastructure and cloud instrumentation technologies.	D3.1[31]
Dashboard Management	ElaSTest Tests Manager (ETM) provides a dashboard with charts generated by	ElaSTest enables testers to segregate	D5.1[28]

	metrics and logs gathered during test execution.	data depending on the test that was exercising the SUT.	
WebRTC Testing	ElasTest User Impersonation Service (EUS) provides GUI automation basing on open source paradigms and enables also the evaluation of the perceived quality of users on relevant scenarios such as real-time multimedia applications.	WebRTC Testing solution able to simulate different WebRTC network topologies; addressing interoperability issues for the SUT or any other device involved in the test.	[17]
Cross-browser Testing	ElasTest User Impersonation Service (EUS) provides capability to impersonate browsers and mobile devices.	Emulation of user actions in many different browsers at the time in the same or different machines.	[17]
Mobile Testing	The released ElasTest platform does not include any dedicated component/module supporting mobile testing.	ElasTest did not advance in mobile testing domain.	-
Cognitive Q&A Systems	ElasTestTest Recommendation Engine (ERE), asks for a natural language description of a test case, generates test code and then uses that generated code as a search query.	Full implementation of unit test cases based on short descriptions in natural language. Learning of deep vector representations (embeddings) of test cases and computation of the distance between them in the semantic vector space.	D4.2[30] D4.4[27]
Test Orchestration	ElasTest Orchestration Engine (EOE), which is responsible of selecting, ordering, and executing a group of tests in ElasTest.	Test augmentation consisting in introducing new TJobs to the original one to reproduce custom operational conditions of the SUT. This allows to test, in addition to functional features of the SUT, other non-functional attributes (such as performance, scalability or reliability).	D4.3 [26]
Data Ingestion	ElasTest Data Manager (EDM) is responsible for installing, managing, and uninstalling the different persistent services available for ElasTest platform.	Providing a persistence layer and a big data processing layer supporting data-agnostic caching and stable persistence	D2.3[25]

		service bundle with auto-scaling facility.	
Test Execution & Visualization	<p>ElaSTest Orchestration Engine (EOE), which is responsible for selecting, ordering, and executing a group of tests in ElaSTest.</p> <p>ElaSTest Tests Manager (ETM) provides facilities for the test case visualization and comparison after their execution. For example, ETM enables the visualization/comparison of infrastructural measures such as CPU, memory and IO consumption of the SuT as well as to inspect all the logs gathered during test execution.</p>	Supporting visualization of the execution of tests in the large (TiLs) as well as simultaneous playback of video recordings and logs.	D4.3 [26]
Test Management	ElaSTest Tests Manager (ETM) gives users the ability to manage the execution of end to end tests in order to verify complex distributed applications.	Automatic collection of logs and metrics during test case execution. Comparison of different test executions.	D4.3 [26]
Testing Framework	ElaSTest Tests Manager (ETM) also gives users the ability to create new execution context for test cases (i.e. TJobs) and to instantiate them by configuring parameters referring both the execution context and test case internal set-ups.	Reusability of test code in different tests cases and easy parameterization of tests.	D4.3 [26]
Virtualization	<p>ElaSTest Platform Manager (EPM) provides the ability to instantiate execution entities (like docker containers or virtual machines).</p> <p>ElaSTest itself has been developed in order to be deployed over several kinds of virtualization technologies. Specifically, the platform is currently available for: Docker, Amazon Web Services, Kubernetes.</p>	Compatibility and adaptability to the most popular virtualization solutions.	D3.1 [31]

Table 28: Summary of ElaSTest Progresses and Outcomes

5.2 ElaSTest Main Benefits

The technical analysis of the SotA (see Section 4) evidenced a lot of tools addressing test automation. Most of them provide advanced facilities such as test annotation while testing, archiving of complete test sessions recording, customizable dashboards and test tracking. ElaSTest integrates the advanced facilities of these tools, and provides a new testing solution aiming to improve the efficiency and effectiveness of the testing process of large software systems, leveraging cloud resources. The main goal of ElaSTest is to provide a holistic and comprehensive integrated end-to-end

testing platform to test distributed large systems. Differently from existing testing tools, the very motivation behind ElasTest is to improve the software testing process as a whole, rather than improving the cost-effectiveness of one specific functional or not functional testing approach. Indeed, ElasTest represents a solution for test automation all along the test process, including: SUT deployment, test execution, SUT monitoring during test execution, and test reporting.

ElasTest offers an open source cloud-based testing service platform OS independent, very flexible and compatible with current Continuous Integration (CI) tools and methodologies so that testers/developers can use it without disrupting their common practices. It is easy to deploy and easy to use and offers capabilities to test end-to-end different types of applications including web, mobile, real-time video communications and Internet-of-Things. The tests can be done under different configurations and environments allowing testers to reproduce real world conditions. ElasTest aims to provide advanced testing capabilities to increase the scalability, robustness, security and quality of experience of large distributed systems. It allows gathering information from the tests and the software under test and presenting the information unified and integrated, enabling easy comparison of tests results from different executions, analysis of the logs, and compatibility of the tested applications with different browsers. ElasTest offers observability and visibility for every test integrating end-to-end tests and monitoring options to automatically and continuously identify different bugs in the software system.

Moreover, the innovation pillars of the project with respect to existing testing platforms are: i) test orchestration, namely the definition of a general test orchestration topology notation and orchestration rules applied to large systems; ii) test recommendation, namely the ability of supporting personalized recommender system able to recommend the tester with the specific T-Job combinations to be included into a TiL.

From a practical point of view, the main benefits that testers can have using ElasTest deal with the many facilities that the platform provides for the validation of the SUT behaviour when interacting with different 3rd party clients or services. Specifically, the main ElasTest benefits for testers can be summarized as:

- Reduction of the time invested from QA team to-fix any software bug. This is achieved by the ElasTest observability capabilities that allow for reducing bug localization time by performing log collection and analysis, aggregation of large volumes of events, comparison with previous executions (previous tests run) as well as metrics collection and comparison.
- Shorten time-to-market. ElasTest allows easy and fast writing of complex tests leveraging the ElasTest support services.
- Improve communication in QA teams. ElasTest allows the instantaneous availability of all the information for the QA teams, developers and managers. This is supported by log and metrics collection, test session recording, signaling of errors by logs.

6 Research projects related to ElasTest: an overview

The following section reports a collection of research projects overlapping with one or more topics covered by ElasTest. Specifically, for each project are reported: its main fact-sheet information (e.g., Official Title, URL, Founding Schema, and Period), a brief description of its main objectives, and some highlights on its relation with the ElasTest project and related technologies.

Among the others, the interactions documented among the ElasTest dissemination activities (see D8.2 [29]) remark the evidence of the relation between ElasTest and the STAMP, the FI-WARE, and the GAUSS projects.

6.1 CodeSan

Title: CodeSan: Code Sanitization for Vulnerability Pruning and Exploitation Mitigation

URL: <https://cordis.europa.eu/project/rcn/225307/factsheet/en>

Founding Schema: ERC Starting Grant

Period: March 2020 - February 2025

Brief description: CodeSan proposes a comprehensive approach to improve code quality. CodeSan sanitises software by automating bug discovery during development through software testing and by protecting deployed software through the activation of smart runtime checks only where they are actually needed. CodeSan complements formal approaches by protecting software that is currently out of reach due to its size, or complexity. In this sense, CodeSan promotes automatic test case generation strategies increasing testing coverage for large programs without the need for pre-existing test cases.

Relation with ElasTest: Both CodeSan and ElasTest aim to improve the efficiency and effectiveness of the testing process of large and complex software systems. ElasTest could offer to CodeSan a suitable testing platform for launching and governing the execution of the test suites automatically generated by means of the strategies that CodeSan investigates.

6.2 GAMMA

Title: The Artificial Intelligence Code Analysis & Recommendation Engine to drive software development speed & reliability for global corporations

URL: <https://cordis.europa.eu/project/rcn/223522/factsheet/en>

Founding Schema: EIC Short SME Financing

Period: June 2019 - November 2019

Brief description: The Gamma Recommendation Engine is an (AI)-based platform for software engineers. Gamma can scan source code repositories detecting a very wide range of software bugs in real-time and immediately proposing fixes for them.

Relation with ElasTest: As in the Gamma project, the ElasTest platform provides specific components leveraging both cognitive computing and machine learning mechanisms. Such components are able to generate testing recommendations (e.g. propose structure for test cases) or to answer natural language questions about the testing process. The specific technological results from both the projects could be potentially combined in order to enhance the traditional software testing processes by identifying issues and possibly prevent faults before they appear.

6.3 SENECA

Title: Software ENgineering in Enterprise Cloud Applications systems

URL: <https://cordis.europa.eu/project/rcn/193968/factsheet/en>

Founding Schema: Marie-Curie

Period: January 2015 - December 2018

Brief description: The SENECA project aimed to address key issues in the software engineering of cloud-based systems, including a disciplined approach to their development and operation. Specifically, the project investigated cloud-based development process and it proposed related tools for quality assurance.

Relation with ElasTest: Both the projects propose solutions aiming at assessing the quality of cloud-based systems. On the one hand, the SENECA project could benefit from the ElasTest platform in order to deploy a cloud-based system and testing it while producing real-world operational conditions. On the other hand, as ElasTest itself is a cloud platform, the ElasTest project could benefit from some of the QA approaches for cloud environments investigated within the SENECA project.

6.4 TEFIS

Title: TEstbed for Future Internet Services

URL: <https://cordis.europa.eu/project/rcn/96812/factsheet/en>

Founding Schema: FP7-ICT

Period: June 2010 - February 2013

Brief description: TEFIS provided an open platform to support experimentation at large-scale of resource demanding Internet services in conjunction with the so-called Future Internet networking technologies and user-oriented living labs.

Relation with ElasTest: Both the projects aim at significantly improving the efficiency and effectiveness of the testing process by means of an extensible testing platform that could combine testing services on-demand and that could take into account realistic operational conditions. Despite the similar long term vision, the wide gap in time between the two projects would possibly limit the actual technological compatibility between the proposed solutions.

6.5 SWITCH

Title: Software Workbench for Interactive, Time Critical and Highly self-adaptive cloud applications

URL: <https://cordis.europa.eu/project/rcn/194122/factsheet/en>

Founding Schema: H2020 ICT-RIA

Period: February 2015 - January 2018

Brief description: The SWITCH project addresses the industrial need for developing and executing time critical applications in Clouds. Specifically, SWITCH aims at improving the existing development and execution model of time critical applications by considering aspects of QoS/QoE, together with the programmability and controllability of the Cloud environments, since the early stage of the applications lifecycle.

Relation with ElasTest: As the SWITCH project, ElasTest investigated tools and methods for the software development of critical applications running in the cloud. In this sense, potential follow-up from the SWITCH project could refer the ElasTest platform for its capabilities to reproduce real-world operational conditions. For example, the ElasTest platform could be used in order to assess the QoS/QoE contacts of a time critical application by means of specific built-in services acting on the testing process.

6.6 STAMP

Title: Software Testing AMPLification

URL: <https://cordis.europa.eu/project/rcn/206167/factsheet/en>

Founding Schema: H2020 ICT-RIA

Period: December 2016-November 2019

Brief description: Leveraging advanced research in automatic test generation, STAMP aims at pushing automation in DevOps one step further through

innovative methods of test amplification. It will reuse existing testing artefacts, in order to generate more test cases and test configurations each time the application is updated. Acting at all steps of development cycle, it will bring amplification services at unit level, configuration level and production stage.

Relation with ElasTest: Both the projects aim to enhance automation in software testing by proposing solutions that can be combined within CI/CD pipeline. In this sense, the test artefacts generated with some of the approaches developed within the STAMP project can be managed and launched by means of the ElasTest platform.

6.7 ADVANCE

Title: Addressing Verification and Validation Challenges in Future Cyber-Physical Systems

URL: <https://cordis.europa.eu/project/rcn/219168/factsheet/en>

Founding Schema: Marie-Curie

Period: January 2019 - December 2022

Brief description: The ADVANCE project investigates new approaches to support the Verification and Validation (V&V) of Cyber-Physical Systems (CPS). In particular, it will focus on techniques that can both collect evidence useful for V&V techniques in CPS, and also analyze data of the system under analysis.

Relation with ElasTest: The ElasTest platform includes capabilities for the instrumentation of the Software under Test so that observe and properly react to evidence from the testing session. Reaction can include changes on the testing environments or the decisions about the next test to execute. Furthermore, the ElasTest platform includes native services for device/sensor emulation, logs ingestion and analysis, and for testing observability in general. All these features could be fruitfully exploited during the ADVANCE project.

6.8 PRECRIME

Title: Self-assessment Oracles for Anticipatory Testing

URL: <https://cordis.europa.eu/project/rcn/216587/factsheet/en>

Founding Schema: ERC Advanced Grant

Period: January 2019-December 2023

Brief description: The PRECRIME project promotes a new and disruptive view on testing, called anticipatory testing and aimed at fixing bugs before they even manifest themselves in the field. The goal of anticipatory testing is to anticipate any failure that might occur in the field due to unexpected execution contexts.

Relation with ElasTest: The ElasTest platform could be referred by the PRECRIME project as operative platform where to explore the behaviour of a SUT in several alternative configurations or environments. Also whenever a self-assessment oracle suggests a deeper testing for monitored execution context, the ElasTest platform could be used in order to resume specific configuration observed in the field, and then starting to test the SUT from there, by exploring uncovered states.

6.9 NOBUGS

Title: Toward Zero-Defect Software Through Automatic Cooperative Self-Improvement

URL: <https://cordis.europa.eu/project/rcn/102167/factsheet/en>

Founding Schema: ERC Starting Grant

Period: February 2012-January 2018

Brief description: NOBUGS investigates techniques and formalisms for automatically recouping and aggregating information from everyday software use. Such information is then turned into tests and proofs that aim to automatically assess the correct behaviour of a system, and to explore behaviours for which information is lacking.

Relation with ElasTest: Potential follow-up from the NOBUGS project could refer the ElasTest platform as operative platform where to explore the behaviour of a SUT in several alternative configurations or environments. Also, the ElasTest platform could be used in order to resume specific configurations observed in the field, and then starting to test the SUT from there, by exploring behaviours potentially uncovered from previous testing activities.

6.10 FI-WARE

Title: Future Internet Core Platform

URL:

- <https://cordis.europa.eu/project/rcn/99929/factsheet/en>
- <https://www.fiware.org/>

Founding Schema: FP7-ICT Collaborative project

Period: May 2011 - December 2014

Brief description: The goal of the FI-WARE project will be an open architecture and a reference implementation of a novel service infrastructure, building upon generic and reusable building blocks developed in earlier research projects.

Relation with ElasTest: One of the vertical demonstrators adopted for the validation of the ElasTest platform concerned the Fraunhofer FOKUS Open5GCore toolkit. The results from the application of the ElasTest technologies in the context of 5G environments suggest that similar testing campaign could be also developed and applied in the context of the FI-WARE infrastructure.

6.11 TESTOMAT

Title: The Next Level of Test Automation

URL: <https://www.testomatproject.eu/>

Founding Schema: ITEA 3 Call 3

Period: October 2017-September 2020

Brief description: The TESTOMAT project supports software teams to strike the right balance by increasing the development speed without sacrificing quality. The project will ultimately result in a Test Automation Improvement Model, which will define key improvement areas in test automation, with the focus on measurable improvement steps.

Relation with ElasTest: The TESTOMAT project has a tool-intensive approach to test automation. In this sense, the ElasTest platform could be referred to as a public available open-source platform so that to be possibly exploited in one of the activities of the TESTOMAT Project.

6.12 GAUSS

Title: Governing Adaptive and Unplanned Systems of Systems

URL: <http://www.lta.disco.unimib.it/GAUSS/>

Founding Schema: Italian MIUR PRIN 2015 Project

Period: September 2017 - January 2020

Brief description: The GAUSS project investigates methodological enablers required to identify, integrate, and manage emergent System-of-Systems (eSoS). These require dynamic and opportunistic engineering due to their intrinsically variable nature tied to their scale and heterogeneity. GAUSS developed a set of integrated techniques in order to address these engineering problems of eSoS at run-time, when specific execution contexts may invalidate design-time solutions.

Relation with ElasTest: The specific domain of investigation addressed by the GAUSS project often requires for modular CI/CD pipelines that could be injected with feedback observed from the field. In this sense, the techniques from the

GUASS project could benefit from the ElasTest platform in order to deploy a cloud-based system and testing it while injecting/reproducing in-vitro real-world operational observations.

7 Market Analysis

This section is a brief update of the market analysis presented in D2.2 [7]. It provides quantitative and qualitative assessment of the IT market for the present year 2019 and some trends and predictions for 2020. It investigates the key trends in software testing market showing its volume and value, potential growth, customer segments, and the key competition. The main goal is to identify the various areas of the market in which ELasTest can later create impact and be sold, in order to ensure ElasTest sustainability.

The main trends of the market related to testing activities are: i) more automation of regression testing; ii) digital transformation with agile; iii) continuous integration to ensure the best quality of the software; iv) increasing adoption of Devops; v) artificial intelligence offering new opportunities for services; vi) shortening Delivery Cycle with Selenium; vii) microservices architecture. Moreover, the cloud testing market is growing fast. According to a recent report [8], published in October 2019, the cloud testing market worldwide is projected to grow by US\$8.3 Billion, driven by a compounded growth of 13.1%. The testing platforms and tools represent one of the key segments analyzed that foresee the potential to grow at over 11.8%. Poised to reach over US\$8.5 Billion by the year 2025, testing tools/platforms will bring in healthy gains adding significant momentum to global growth. In Europe, which continues to remain an important element in the world economy, Germany will add over US\$337.4 Million in the next 5 to 6 years. Over US\$396.5 Million worth of projected demand in the region will come from the rest of the European markets. As the world's second largest economy and the new game changer in global markets, China exhibits the potential to grow at 13% over the next couple of years and add approximately US\$1.5 Billion in terms of addressable opportunity for the picking.

Another important aspect to be considered in the market analysis is the growing number of software developers in terms of potential ElasTest users. As shown in Figure 15, the number of programmers is growing in different speeds and places. A report by Evans Data Corporation states that last year there were 23 millions of software developers in 2018, this number is expected to be 26,4 millions by the end of 2019 and 27,7 millions by 2023. On top of this, according to IDC calculations [9], in 2018 the number of software developers in the world grew to 22,3 million, while in 2014 there were only 18,5 million of programmers. And according to Stack Overflow, the number of software developers in Europe was reported to stand at 4,7 millions in 2016, which in 2018 reached 5,5 millions. According to these data and considering that ElasTest aims to be released and maintained totally open source, the potential of usage for ElasTest end-to-end testing platform is huge.

In order to focus on the main ElasTest market position, we revise in the following of this section: the IT budget allocated to testing; the trends in ICT and cloud market considering the agile and devOps development; the challenges in test automation and finally the expectations from the market. Finally, we show how ElasTest is aligned with the key trendy tools and technological advancements of the market and will define the market perspectives for ElasTest.

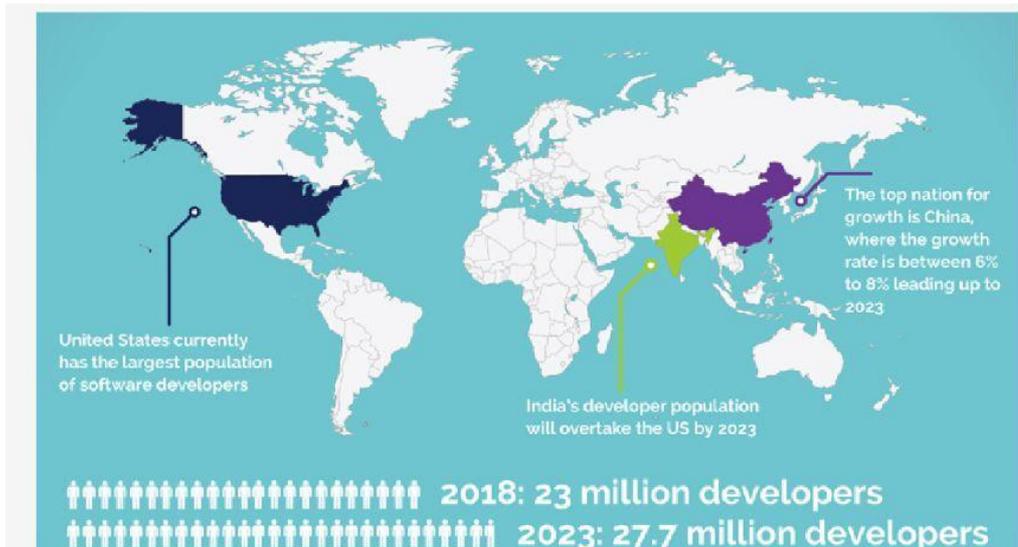


Figure 15: Number of developers in the world

7.1 IT budget allocated to testing

The aim of software testing is to detect any error in the development of high quality software products and validate the software components in a given system before launching them to the market. The software testing process allows for validating that software products behave as these were planned: that is the application gives the user what it is designed for and that the QA team ensures that the quality of the system is stable enough to be massively used. For supporting developers there are many software testing tools, solutions, and services used across a range of industry verticals to build high-quality software. The importance of testing is clear and every year IT budget allocated to testing grows. In fact, a recent report [11] says that 25% of budget within companies is allocated to the testing and by 2025, it may be around 33%.



Figure 16: Portion of the testing budget allocated to QA testing (including testing process, tools and resources)

Figure 16 clearly shows that the 23% is steadily a percentage of the total testing budget allocated from organizations to QA testing which accounts for testing process, tools and resources. However, as new efficient tools are introduced, the testing process is becoming more efficient and costs are reduced. Therefore, OSS tools such ElasTest present an opportunity to consider for this purpose.

The factors influencing the portion of IT budget dedicated to testing are shown in Figure 17. This figure evidences a tendency in having more focus on the number of developments and releases with an almost 6% (5.47% out of 7%, 7 being the maximum/most important value). Other important factors are: the need to have agile and devOps methodologies causing more and more test iterations cycles in the software testing process (5.39%), followed by the increased challenges in test environments (5%). ElasTest tackles these challenges supporting testing of IoT, webRTC or web applications among the others.

Moreover, according to 451 Research annual report “Top 10 IT trends for 2019” [13], most of the IT investment in the coming decade will be focused on automation. Specifically, this report identifies the following four top trends for the IT market: 1) Cloud native takes center stage; 2) Successful organizations will be data driven; 3) Focus in digital procurement shifts to optimization; and finally, the most relevant for ElasTest 4) A new Age of Automation. In fact they claim: *“Since the financial crisis of 2008 much of IT investment has been focused on the emancipation of software from the underlying hardware. Most of the investment in the coming decade will be focused*

on automation, building on this virtualized software layer. A large range of technologies, from robotic process automation (RPA) through to Machine Learning to will drive this and Digital Automation Platforms will be the platforms used to build applications that can adapt to change quickly”.

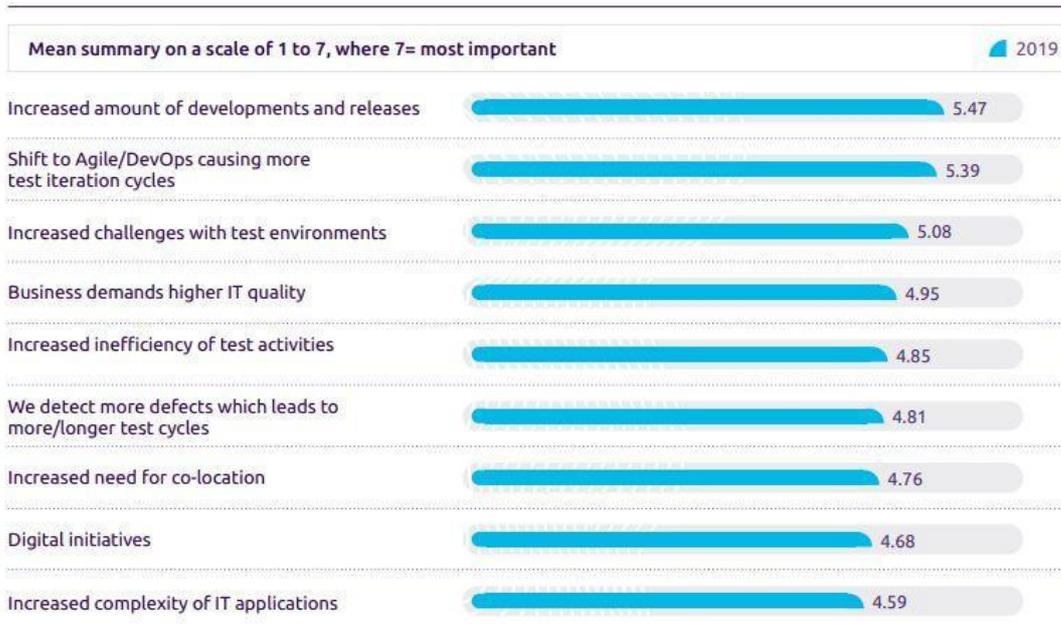


Figure 17: Factors influencing the portion of IT budget dedicated to testing

7.2 ICT market

The ICT market is expected to continue to grow. IDC report [32] forecasts that the worldwide dollar-valued ICT spending will grow. It will grow up to \$4,453,711 billion dollar by 2022, and during 2019 to \$4,099,343. Figure 18 shows the worldwide ICT spending in \$ millions whereas Table 29 shows the ICT spending forecast per technology.

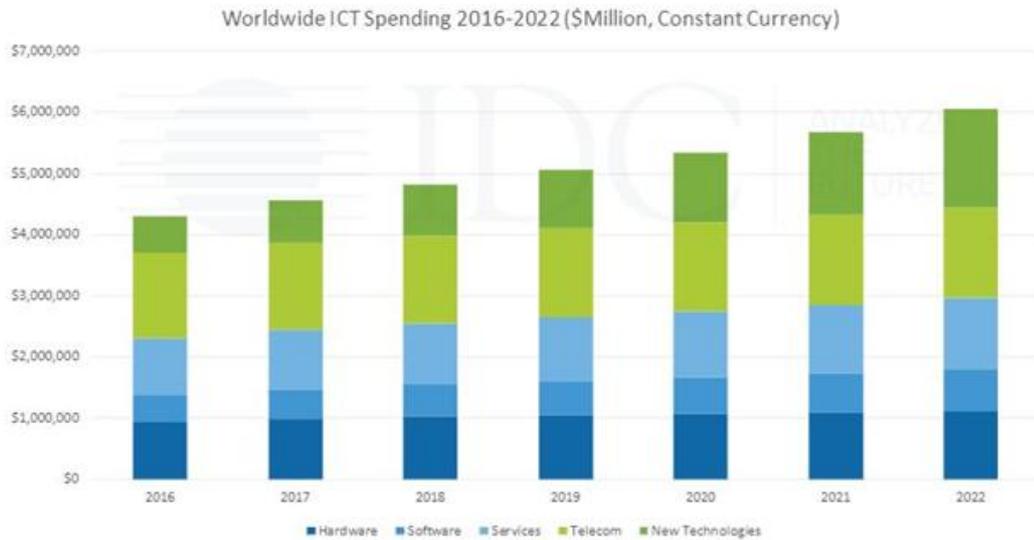


Figure 18: Worldwide ICT Spending 2016-2020

Technology Spending \$M	2018 Spending	2018 Growth	2019 Spending	2019 Growth
Hardware	\$1.033.759	4%	\$1.053.959	2%
Software	\$512.237	7%	\$550.567	7%
Services	\$1.009.573	4%	\$1.048.654	4%
Telecom	\$1.431.128	1%	\$1.446.164	1%
Traditional ICT	\$3.986.977	3%	\$4.099.343	3%
New Technologies	\$825.978	16%	\$961.173	16%
Total ICT	\$4.812.974	5%	\$5.061.106	5%

Table 29: Spending Forecast for technology (Billions of U.S. Dollars)

Another report from IDC about automated Software Quality Forecast, 2018–2022 [12]: states that the growth will be driven by continuous testing and DevOps demand. This IDC report suggests that during 2018–2022 automated software quality market will grow. They also acknowledge the impact of open source software usage and integration by vendors like Selenium (used also in ElasTest) and for example Cucumber. Moreover, the need for testing of IoT applications and the rising importance of DevOps and Agile are factors which are driving the market and continue to grow in adoption as well as quality assurance (QA).

7.3 Cloud Market

The cloud market continues to become mainstream within most organizations. According to Markets and Markets report published in early 2019 [33], the total cloud computing market size is expected to grow from USD 272.0 billion in 2018 to USD 623.3 billion by 2023, at a Compound Annual Growth Rate (CAGR) of 18.0% during that period as shown in Figure 19. There are many and different aspects driving this growth

such as the increasing volume of data gathered from all the services/devices, from websites and/or mobile apps, and the tendency to deliver customer-centric applications for driving customer satisfaction. As showed in Table 30, Infrastructure as a service (IaaS), based on the cloud system infrastructure, seems to be the fastest-growing market segment which is forecast to grow of 27.5 percent in 2020 to reach \$49.1 billion, up to \$76.6 billion in 2022. The second-highest growth rate of 21.8 percent will be achieved by cloud application infrastructure services, or platform as a service (PaaS).



Figure 19: Cloud market prediction

	2018	2019	2020	2021	2022
Cloud Business Process Services (BPaaS)	45.8	49.3	53.1	57.0	61.1
Cloud Application Infrastructure Services (PaaS)	15.6	19.0	23.0	27.5	31.8
Cloud Application Services (SaaS)	80.0	94.8	110.5	126.7	143.7
Cloud Management and Security Services	10.5	12.2	14.1	16.0	17.9
Cloud System Infrastructure Services (IaaS)	30.5	38.9	49.1	61.9	76.6
Total Market	182.4	214.3	249.8	289.1	331.2

Table 30: Public Cloud Service Revenue Forecast (Billions of U.S. Dollars)

These data show that there are different and attractive opportunities in the cloud market. This cloud market will grow further in North America and Europe, whereas it is expected to hold a significant growth rate in Asia and Latin America. Furthermore, the factors that are expected to drive the market growth are: increased automation and agility, need to deliver enhanced customer experience, and increased cost savings and return on investment.

7.4 Devops

A report published by Research and Markets [34] predicts a global investment of \$19 Billion in Automation Testing market for the period 2018-2023. The global automation testing market size foresees to grow from USD 8.52 Billion in 2018 to USD 19.27 Billion by 2023, at a Compound Annual Growth Rate (CAGR) of 17.7% during the forecast period. This report indicates *DevOps methodology* as one of the main factors driving this growth. Indeed, they claim that: *“The major factors that are expected to drive the growth of the market include the increasing adoption of mobile devices and technologies, increasing adoption of the DevOps methodology, and transforming testing by digital transformation”*.

The report in [16] offers good recommendations for those organizations supporting wider agile and DevOps adoption such as: i) build a smart and connected testing ecosystem deploying intelligent analytics; ii) introduce security testing early in the lifecycle — during design; iii) expand AI-related skillsets within the test team by onboarding data science, statistics, mathematics, and more; iv) re-imagine test automation as a platform; v) raise awareness and visibility of test environments; vi) adopt a center of excellence approach for test data management.

In addition, Mark Buenen, Global Leader, Digital Assurance and Quality Engineering for the Capgemini Group said [38]: *“We are continuing to see Testing and QA move from a discrete area within an organization to one that is more fundamental to enterprise operations and business outcomes. At the same time, change brings its own challenges – two of the most important being orchestration of Testing and QA in agile and DevOps development and access to the required skillsets. To stay ahead, organizations need to embrace new approaches, including a connected and holistic approach to testing, raising organizational awareness of test environments, and adopting a center of excellence approach to test data management.”*

Considering the API Testing (as it is done in ElasTest by the REST API), a report from Research and Markets [14], it reveals that the global API testing market size is estimated to grow from USD 447.4 Million in 2017 to USD 1,099.1 Billion by 2022. Moreover, the current evolving adoption of both Devops and Agile in software development and the strategies in companies to use open APIs are key factors driving API testing market.

7.5 Test automation

The test automation demand in the market is augmenting more and more every year. In fact, according to the report by markets and market [15], the automation testing

market is expected to grow from USD 8 billion in 2018 to USD 60.4 billion by 2026, at a Compound Annual Growth Rate (CAGR) of 33.4% during the forecast period. The functional testing segment is expected to have a larger market size from 2019-2024, and the prediction reflects an increase in all areas, not only in Europe, as shown in Figure 20.

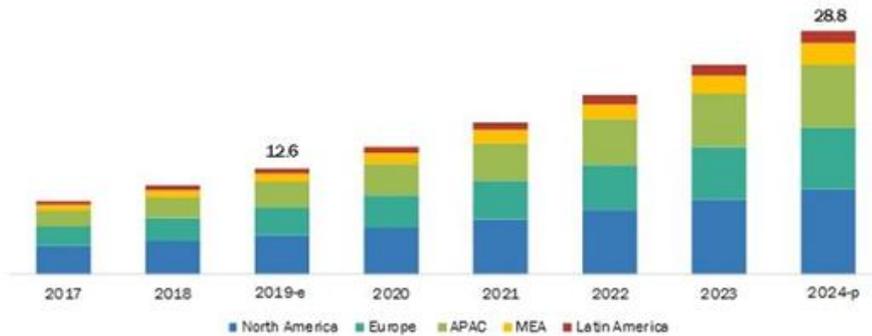


Figure 20: Functional Testing Trend

To address the challenges of testing many organizations are looking for more dynamic ways to test their processes and systems and this makes automation very relevant. Moreover, as the competition among market players is increasing and the information is freely available for the users, customers are becoming more empowered. They can choose from a wide array of options. This encourages companies to deploy advanced testing tools and services able to increase the response time and provide a better experience to their customers. The automation testing services are key factors for industry and these cover and fulfill the growing need of QA. These services help increase the efficiency and effectiveness of applications, replicate testing across different platforms, minimize manual intervention, and reuse test scripts in various testing scenarios. Automated testing services include: suitable test tools; identification of test scenarios; test script maintenance; and generation of automated test reports.

The report in [16] says that test automation has delivered many benefits including: improved control and transparency of test activities (63%), better detection of defects (56%), and reduction of test costs (56%). When asked about the technical challenges the developers face in developing applications: the 63% respondents say that there's a "lack of end-to-end automation from build to deployment", up from 55% in last year's survey.

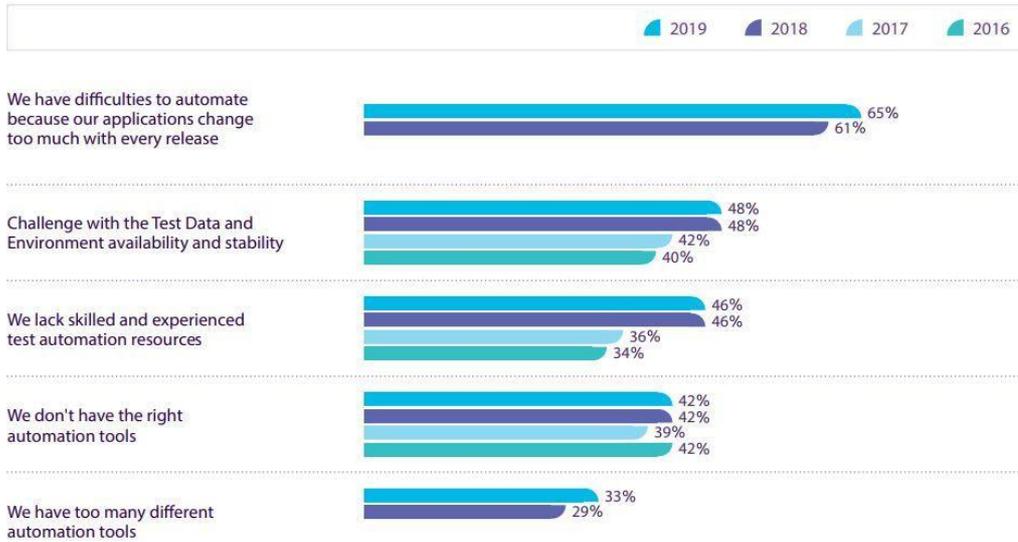


Figure 21: Main challenges faced while achieving test automation

They say at the same time there’s a need for more use of automation and artificial Intelligence in some organizations. From the respondents, 41% identified a ‘*lack of proper skills for QA & Testing*’ as a technical challenge. Moreover, 58% of organizations that took part in the survey looked for external AI expertise, either because it’s not part of their core business (23%), or they needed AI knowledge fast (24%) or it was a requirement for a limited amount of time (11%).

Figure 21 shows main challenges about achieving test automation. Among them, there are the lack of automation tools and difficulties due to tracing of test data and environment changes. In this context, ElasTest is able to address most of these challenges and then provide a good value in the market. Indeed, it supports realistic operative testing scenarios and allows the logs and metrics visualization as well as test data recording and management.

7.6 Expectations from the market

The report published by CapGemini [16] surveys different organisations and predicts what are the trends in software quality. The authors of [16] consider most important (almost 6%, 6 out of 7 scale, 7 being the maximum) the investments in testing IT systems. For that, there is a huge need of robust end-to-end software testing tools. However, one of the testing strategy goals is end-user satisfaction. The report claims that testing must consider customer satisfaction and should be aligned with business goals to ensure market share in companies. Also, it shows that 99% of respondents use DevOps in at least some of their projects. Therefore, more automation is required to speed-up the testing process. Indeed, the report reveals that automation is the biggest bottleneck that is holding back QA and testing today. The idea is how to

optimise testing that needs to be done, in order to shorten test cycles while increasing their effectiveness and ensure software quality. This can be reached by: tools extracting information from application lifecycle management; end-to-end testing tools such as ElasTest; monitoring systems; production monitoring systems; processing of information in a timely manner.

Moreover, there’s a huge room for investigations and innovation on applying AI to QA, as well as testing AI algorithms and products. The World Quality Report [35] finds that a lot of organizations are experimenting how AI can be applied in the whole testing and QA process. In addition, 55% of respondents are struggling with identifying where and how to apply AI, whereas 51% say they have experienced difficulty integrating AI with their existing applications. Figure 22 shows the percentage of projects addressing artificial intelligence and machine learning for 2019 in the different software development areas. Specifically, 38% of these artificial intelligence and machine learning projects address quality assurance. This implies a set of additional challenges, namely that AI tester must master an additional set of highly technical and mathematical skills, such as mathematical optimization, and algorithmic knowledge.

ElasTest is in line with the application of artificial intelligence and machine learning to testing and may have for this a good place in the future testing market. Indeed, one of the ElasTest key advancements is the investigation and implementation of new algorithms of artificial intelligence for tests generation and recommendation.



Figure 22: Artificial intelligence and machine learning projects for 2019.

7.7 Comparison of ElasTest with trendy tools

This section provides a comparison of ElasTest with key trendy tools according to their similarity and/or relevance to ElasTest, focusing on specific market aspects, such as ease of use, business model and cost. Table 31 shows an overview of these tools compared to ElasTest and provides an update of the analysis of similar tools to ElasTest, performed in D2.2 [7]. As in D2.2, we performed this comparison in terms of: strengths and weaknesses of the tool, percentage of similar functionalities with ElasTest, deployment model, automation, type of application under test, acceptance by market/developers, numbers of users and customers, number of tests, target market, ease of installation and use, finally business model and cost. Many of these tools were already identified in the first review and we provide in this document an update of their information, other tools such as Scope and Honeycomb have been identified in this second review period. These tools aim to solve only a part of the problems covered by ElasTest. Moreover, some of them such as Jenkins and TestLink have been successfully integrated with ElasTest.

	ElasTest	Test Complete ¹⁰ 	Tricentis Qtest ¹¹ 	PractiTest ¹² 	TestLink ¹³ 	HP ALM Quality Center ¹⁴ 	Jenkins ¹⁵ 	Travis CI ¹⁶ 	EasyQA ¹⁷ 	Scope ¹⁸ 	Honeycomb ¹⁹ 
Value Proposition (strength)	ElasTest: bringing observability to your test and making complex tests simple. Improving the efficiency, productivity and code	Gives testers the ability to create automated tests in Windows, Web & Android and IOs. Screen capture to create scripts	Simple and intuitive to use. Tricentis qTest streamlines software testing in agile and DevOps environments and centralize	PractiTest is an end-to-end QA and Test management solution. Solution for Agile Testing, Regression Testing, MicroServices and DevOps. Reports and	Open Source Test Management Application. Testlink is one of the best and useful test tracking tools	The software quality management component of the highly renowned HP application lifecycle management (ALM) software suite	The leading open source automation server, Jenkins provides hundreds of plugins to support building, deploying and automating	Travis CI is a leading provider of continuous integration and delivery services and empowers software development teams to test and deploy their applications	EasyQA offers test management tools for IT Development teams to improve software quality. Manage the testing process, run up to 50 projects simultaneously, manage 3	Test management and monitoring platform.	Tool designed for debugging and understanding complex systems, microservices, distributed systems, including the data layer, with the industry's only DB-agnostic raw query

¹⁰ <https://smartbear.com/product/testcomplete/>

¹¹ <https://www.tricentis.com/products/agile-dev-testing-qtest/>

¹² <https://www.practitest.com/>

¹³ <http://testlink.org/>

¹⁴ <https://saas.hpe.com/en-us/software/quality-center>

¹⁵ <https://jenkins.io/>

¹⁶ <https://travis-ci.org/>

¹⁷ <https://geteasyqa.com/>

¹⁸ <https://scope.dev/>

¹⁹ <https://xebialabs.com/technology/honeycomb-io/>

	reusability of the testing process in large complex distributed applications	Easy debug scripts. 1,500 mobile and desktop browsers in more than 65 operating systems	s testing efforts across the enterprise . Easy-to-use UI and low ramp-up time. Records every step and screen of testing execution	dashboards are very customizable so to create metrics			any project. A vibrant DevOps Automation Community	with confidence.	different platforms within one project and have 50 organizations in one account	analyzer	
Weakness of the tool	Integration complexity	None. In fact it was awarded/ leader best tool in 2020 by G2.com ²⁰	Can't call tests from other projects. Manual configuration. UI needs simplification	Hard to integrate questions. Option to copy and paste tests and ability of batch editing tags. Dashboard customization	The design and ergonomic is old	None	None	None	None	It is necessary to wait for access to be given in order to try	Cannot launch executions
% Similarity with ElasTest	N/A	80%	80%	70%	70%	70%	70%	70%	50%	15% at least	30%
Deployment	Cloud, Saas	Cloud,	Cloud,	Cloud, Saas	OS Web-	Cloud, SaaS	Cloud, SaaS	Cloud, SaaS	Cloud, SaaS,	-	-

²⁰ <https://www.g2.com/products/testcomplete/reviews>

model		SaaS	SaaS		based tool				Web		
Automated Test platform	Yes	Yes	Yes	Yes (SaaS)	Yes	Yes	Yes	Yes	Yes (SaaS)	Yes (SaaS)	No
Application under test	web, mobile apps	web, mobile apps	web, mobile apps	web, mobile apps	web, mobile apps	web, mobile apps	web, mobile apps	web, mobile apps	Windows desktop, web, mobile apps	Windows desktop, web, mobile apps	Any
Acceptance by market/developers	Low	Niche	Very High	Very High	Very High	Very High	Very High	Very High	Very High	Very High	Very High
Statistics Of usage, numbers of users	N/A	300,000 users worldwide	No info	No info	No info	No info	15.8 million of developers use Jenkins	>900k open source projects	No info	No info	No info
Nº of test run	33 commits	>10 million tests	No info	>2.5 million tests run on customers >450K test cases	No info	No info	>148.416 installations . 1,000 plugins.>6 million build jobs on Jenkins	>200K active projects	No info	No info	No info
Nº customers	N/A	>6 million individual developers	>500 customers	No info	More than 25 years. No info on number of users	No info	No info	>700k users	No info	No info	There are not much info, but at least 12 companies
Targeted Market	Developers, Community	Companies, SMEs,	SMEs, Large and	Technical teams,	Manager, QA teams,	Companies, SMEs, Large	Developers, SMEs, Large	Developers, Startups, SMEs,	QAs, Developers,	No info	Developers, Companies,

	Testers, Designers, Students, SMEs	Large and medium enterprises, IT professionals, Developers and QA teams	medium enterprises, QA teams, Developers	Developers, Public administration	Developers	and medium enterprises, Free users	and medium enterprises	Big corporations	and PMs.		SMEs, Large and medium enterprises
Ease of installation and use	Mid-user friendly	Easy to setup and run. Easy to use, help desk very active	Easy to setup and run	Not really	Mid-user friendly	Easy to setup and run	Very easy to use	Travis CI is simple to use	Mid-user friendly	No info	Mid-user friendly
Business model	Open Source + free	License and maintenance fees. Pay per use	Free License + Hosting Pay per use	Pay per use + hosting	Open Source + free	Open Source + hosting License, Pay per use	Open Source + hosting	High	Pay per use	No info	Free limited used + pay per use
Cost	Not yet defined	\$1,250 (year)	\$29/user/month	\$39 Professional Tester /user/month 49\$ Enterprise testeruser/month	Upon request	\$9,000 per seat license	Free	Various prices \$69/hosted/month for hobby projects on-premises 4000\$ (per 10 pack users)	\$10/user/month	No info	Professional starts @ \$70/month Enterprise starts @ \$24,000/YEAR

Developer
\$15
user/month

Table 31: ElasTest vs key tools

7.8 Main stream technologies related to ElasTest

ElasTest is currently aligned with the technological advancements of the market. It uses and integrates the modern datacentre technologies as well as the most popular tools such as Kubernetes, Jenkins and TestLink as described below.

7.8.1 Kubernetes

Kubernetes²¹ is a cluster and container management tool that lets deploying containers to clusters, namely a network of virtual machines. It works with different containers, including Docker²². According to a survey of November 2019 [36], by Stackrock, 86% of respondents are using Kubernetes. ElasTest uses Kubernetes in order to deploy and execute seamlessly cloud services in the target cloud infrastructure. ElasTest itselfs has been developed in order to be deployed over clusters controlled by means of Kubernetes.

7.8.2 Jenkins and TestLink

Jenkins²³ is the de-facto CI leader in the market. According to the analyst firm Datanyse [37], Jenkins is the most adopted CI server in the market with 70% of the market share. It accounts with more than 100M installations all over the world. ElasTest exposes the whole set of its functionalities for managing tests by means of an open API. In this sense, it is fully compatible with current Continuous Integration (CI) tools and methodologies commonly used in QA processes. Specifically, the ElasTest Jenkins Plugin is a dedicated module of ElasTest that directly enables the interaction between any Jenkins CI instance with ElasTest. Moreover, ElasTest has been integrated with TestLink²⁴, which is the most widely used open source tool for test management.

7.9 Market Perspective for ElasTest

Nowadays, end-to-end testing of software and distributed applications requires a lot of investments in terms of time and effort and in some cases it is done mostly manually. The market analysis showed a clear tendency to continuous budget allocation for testing. More and more investments are devoted to software testing for quality assurance, fasten time to market and to ensure better quality of software products. The market trends outline that agile methodologies for building software and compatibility with the most used continuous integration environments are adopted by development teams for ensuring quality assurance of products. Also, DevOps, observability and monitoring tools for software development represent key factors that are driving the market. The market analysis outlines that the cost of the management of test environments within companies still remains a challenging task

²¹ <https://kubernetes.io/>

²² <https://www.docker.com/>

²³ <https://jenkins.io/>

²⁴ <http://testlink.org/>

and that companies have an increased need of tools to support developers and QA teams in making proper software testing and improving software quality products. In this context, ElasTest which offers an efficient and effective platform for the entire testing process and it is able to improve the quality of large cloud software systems has good market opportunities. Specifically, the following market opportunities are detected for the ElasTest results:

- ElasTest offers a cloud-based platform and may have a good penetration in the IT market to allow creation of a new generation of software testing services. The platform fully satisfies the cloud and ICT market needs to allow higher quality software.
- Elastest platform has been natively conceived to be integrated with any Agile process, continuous Integration platforms, and DevOps practices. This allow the easy and fast adoption of the ElasTest results in the development environments governed by such key development aspects.
- ElasTest offers observability to the tester and a unique flexible testing platform able to perform rapid and accurate end-to-end testing, reducing the possibility of failures and bugs and at the same time increasing the quality of the resulting software product. The capability of ElasTest of reducing the time-to-market of the software products makes it a very competitor platform in the market of test automation.
- ElasTest project promotes an open cloud platform for software testing. This allows the definition of reusable testing components/services. ElasTest platform has been hosted at GitHub as open source code under the Apache 2.0 license. This ensures a good path for sustainability and enables the adoption of ElasTest services by many stakeholders fostering new business models and market penetration.
- ElasTest has been successfully demonstrated on four different types of applications, including web, 5G mobile, real-time video communications and Internet-of-Things. This represents a good starting point for a future uptake of ElasTest in the market related to these areas.

8 Conclusions

This deliverable presented an update of the analysis of the state of the art performed in the second review period.

The first part of the document covers a systematic review on the literature about cloud testing. Its main objective was to identify and classify all the documents and information belonging to informal sources (such as blogs, videos, white papers and web-pages) that are different from the academic ones addressed in D2.2 [7]. The results from the systematic survey of the grey literature showed a growing attention in the last years for testing in the cloud. With respect to the systematic review of the scientific literature presented in D2.2 [7] and [6], the primary studies of our grey

literature review revealed a great interest for the test perspectives and test domains as well for testing services and tools mainly devoted in validating performance aspects. The intent of the analyzed studies is to identify practical issues in the development and usage of cloud testing focusing on test execution and test evaluation. These outcomes evidence the different audience of the grey literature (i.e., practitioners, and software developers) with respect to that of the scientific literature (i.e., researchers mostly from academia). These results also confirmed that ElasTest is in line with the trends and goals of the current software development environments. Indeed, the comprehensive ElasTest cloud platform is able to address several of the dimensions highlighted by the grey literature survey; among the others there are the capabilities to manage test configuration and execution in the cloud and to validate performances and other non-functional properties such as reliability, security and scalability.

As expected, the review of the technical SoTA during this second observation period did not raise a large number of tools: neither new or that presented major updates. In addition, there were also technological areas that did not spot any new contribution with respect to what already presented in D2.2 [7]. For the sake of completeness, Table 28 recaps, for each technological area, the overall contribution that ElasTest brought to the technical SotA. Specifically, the table reports an overview of all the project outcomes and progresses covered by both D2.2 [7], and this deliverable. As a general outcome from the analysis of the technological areas, the SotA evidenced a lack of a comprehensive platform aimed to end-to-end testing of large complex cloud applications. ElasTest fills this lack by providing such a general platform for automating testing all along the test process cycle, including SUT deployment, test design and execution, SUT monitoring during test run, and reporting of test results, as well as considering different domain such as web applications, mobile, WebRTC, and so on.

The overview of the research projects related to ElasTest revealed several effective and potential relations with other projects. In particular, the scientific and technological outcomes from both ElasTest and its related projects could be potentially combined in order to enhance the efficiency and effectiveness of automation in the testing process. Moreover, the open-source features of the ElasTest platform are public available and they could be fruitfully exploited by other researchers, companies, and practitioners involved in other research projects.

Finally, market analysis in this second review period revealed increasing investments in testing of IT systems which demand for competitive automated solutions in cloud testing. These aspects can positively impact on the exploitation of ElasTest results.

9 References

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

- [2] Garousi, Vahid, Michael Felderer, and Mika V. Mäntylä. Guidelines for including grey literature and conducting multivocal literature reviews in software engineering. *Information and Software Technology*, 106 (2019): 101-121.
- [3] B. Kitchenham and S. Charters. *Guidelines for performing Systematic Literature Reviews in Software Engineering*. 2007.
- [4] Q. Li. A novel likert scale based on fuzzy sets theory. *Expert Systems with Applications* 40 (5) (2013) 1609-1618.
- [5] W. J. Tastle, M. J. Wierman. Consensus and dissention: A measure of ordinal dispersion. *International Journal of Approximate Reasoning* 45 (3) (2007) 531- 545.
- [6] Antonia Bertolino, Guglielmo De Angelis, Micael Gallego, Boni García, Francisco Gortázar, Francesca Lonetti, Eda Marchetti. A Systematic Review on Cloud Testing. *ACM Comput. Surv.* 52(5): 93:1-93:42 (2019)
- [7] ElasTest Project. D2.2 SotA revision document v1 (06/30/18) <https://elastest.eu/deliverables.html>
- [8] Cloud Testing - Market Analysis, Trends, and Forecasts Report. Global Industry Analysts, Inc. ID: 4804261 October 2019. <https://www.researchandmarkets.com/publication/mzrqiu6i7/4804261>
- [9] Arnal Dayaratna. IDC's Worldwide Developer Census. 2018: Part-Time Developers Lead the Expansion of the Global Developer Population. October 2018. <https://www.idc.com/getdoc.jsp?containerId=US44363318>
- [10] Stackoverflow. <https://es.stackoverflow.com/>
- [11] Global Software Testing Services Market Size, Status and Forecast 2019-2025
- [12] Melinda-Carol Ballou. Worldwide Automated Software Quality Forecast, 2018–2022: Growth Driven by Continuous Testing and DevOps Demand. IDC Market Forecast. June 2018 <https://www.idc.com/getdoc.jsp?containerId=US42652718>
- [13] 451 Research. Top 10 Trends for 2019. <https://go.451research.com/Top-10-IT-Trends-For-2019.html>
- [14] API Testing Market by Component (API Testing Software/Tools and API Testing Services), Deployment Type (Cloud Based and On-Premises), Vertical, and Region - Global Forecast to 2022" <https://www.marketsandmarkets.com/PressReleases/api-testing.asp>
- [15] MarketsandMarkets. Automation Testing Market by Component, Services, Endpoint Interface, Organization Size, Vertical And Region - Global Forecast to 2024. September 2019. https://www.reportlinker.com/p05377128/Automation-Testing-Market-by-Technology-Testing-Type-Service-Endpoint-Interface-And-Region-Global-Forecast-to.html?utm_source=PRN
- [16] Capgemini. World Quality Report 2019-20. The future of quality assurance and its role in maximizing growth. <https://www.capgemini.com/research/world-quality-report-2019/>
- [17] Boni García, Francisco Gortázar, Micael Gallego and Eduardo Jiménez. User Impersonation as a Service in End-to-End Testing. *Proceedings of the 6th International Conference on Model-Driven Engineering and Software Development (MODELSWARD 2018)*, pp. 707-714

- [18] S. Luan, D. Yang, K. Sen, and S. Chandra. Aroma: Code Recommendation via Structural Code Search. ArXiv181201158 Cs, Dec. 2018.
- [19] S. Panichella. Summarization techniques for code, change, testing, and user feedback (Invited paper). In Proc. of IEEE Workshop on Validation, Analysis and Evolution of Software Tests (VST), 2018, pp. 1–5.
- [20] C. Wang, Y. Jiang, X. Zhao, X. Song, M. Gu, and J. Sun. Weak-Assert: A Weakness-Oriented Assertion Recommendation Toolkit for Program Analysis. In Proc. of IEEE/ACM 40th International Conference on Software Engineering: Companion (ICSE-Companion), 2018, pp. 69–72.
- [21] S. Hotomski and M. Glinz. GuideGen: a tool for keeping requirements and acceptance tests aligned. In Proc. of the 40th International Conference on Software Engineering Companion Proceedings - ICSE '18, Gothenburg, Sweden, 2018, pp. 49–52.
- [22] R. Ibarra and G. Rodriguez. SoTesTeR: Software Testing Techniques' Recommender System Using a Collaborative Approach. In Information Management and Big Data, 2019, pp. 289–303
- [23] Magda Kacmajor, Francesca Lonetti. Recommender systems applied to software testing: A systematic literature review. To be submitted to Journal of Systems and Software.
- [24] Cheers, Boni García, Francesca Lonetti, Micael Gallego, Breno Miranda, Eduardo Jiménez, Guglielmo De Angelis, Carlos Eduardo Moreira Dos Santos, Eda Marchetti. A Proposal to Orchestrate Test Cases. QUATIC 2018: 38-46
- [25] ElasTest Project. D2.3 ElasTest requirements, use-cases and architecture v1 (06/30/18) <https://elastest.eu/deliverables.html>
- [26] ElasTest Project. D4.3 Test Orchestration basic toolbox v2 (31/12/19) <https://elastest.eu/deliverables.html>
- [27] ElasTest Project D4.4 Test recommendation engines v2 (31/12/19) <https://elastest.eu/deliverables.html>
- [28] ElasTest Project D5.1 ElasTest Test Support Services v1 (06/30/18) <https://elastest.eu/deliverables.html>
- [29] ElasTest Project D8.2 ElasTest dissemination plan and activities v1 (06/30/18) <https://elastest.eu/deliverables.html>
- [30] ElasTest Project D4.2 Test recommendation engines v1 (06/30/18) <https://elastest.eu/deliverables.html>
- [31] ElasTest Project D3.1 ElasTest Platform cloud modules v1 (06/30/18) <https://elastest.eu/deliverables.html>
- [32] ICT Spending Forecast. 2018 - 2022 Forecast. <https://www.idc.com/promo/global-ict-spending/forecast>
- [33] Cloud Computing Market. Cloud Computing Market by Service Model (Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS)), Deployment Model (Public, Private, and Hybrid), Organization Size, Workload, Vertical, and Region - Global Forecast to 2023. <https://www.marketsandmarkets.com/Market-Reports/cloud-computing-market-234.html>

-
- [34]Automation Testing Market. Automation Testing Market worth \$28.8 billion by 2024.<https://www.marketsandmarkets.com/PressReleases/automation-esting.asp>
 - [35]World Quality Report 2019-20. The future of quality assurance and its role in maximizing growth. <https://www.cappgemini.com/research/world-quality-report-2019/>
 - [36]Gartner Report: Kubernetes and Container Security and Adoption Trends. <https://www.stackrox.com/kubernetes-adoption-and-security-trends-and-market-share-for-containers/>
 - [37]Datanyze Universe <https://www.datanyze.com/market-share/ci/jenkins-market-share>
 - [38]Cappgemini. World Quality Report. Lack of alignment between business goals and quality ambitions impedes Agile and DevOps adoption. 31 October 2019 <https://www.cappgemini.com/news/world-quality-report-19/>