A Systematic Review on Cloud Testing

ANTONIA BERTOLINO, CNR-ISTI, Italy
GUGLIELMO DE ANGELIS, CNR-IASI, Italy
MICAEL GALLEGO, Universidad Rey Juan Carlos, Spain
BONI GARCÍA, Universidad Rey Juan Carlos, Spain
FRANCISCO GORTÁZAR, Universidad Rey Juan Carlos, Spain
FRANCESCA LONETTI, CNR-ISTI, Italy
EDA MARCHETTI, CNR-ISTI, Italy

A systematic literature review is presented that surveyed the topic of cloud testing over the period (2012-2017). Cloud testing can refer either to testing cloud-based systems (testing of the cloud), or to leveraging the cloud for testing purposes (testing in the cloud): both approaches (and their combination into testing of the cloud in the cloud) have drawn research interest. An extensive paper search was conducted by both automated query of popular digital libraries and snowballing, which resulted into the final selection of 147 primary studies. Along the survey a framework has been incrementally derived that classifies cloud testing research along six main areas and their topics. The paper includes a detailed analysis of the selected primary studies to identify trends and gaps, as well as an extensive report of the state of art as it emerges by answering the identified Research Questions. We find that cloud testing is an active research field, although not all topics have received so far enough attention, and conclude by presenting the most relevant open research challenges for each area of the classification framework.

CCS Concepts: • **Computer systems organization** → **Cloud computing**; • **Software and its engineering** → **Software testing and debugging**; *Cloud computing*; • **Networks** → *Cloud computing*;

Additional Key Words and Phrases: Cloud Computing, testing, systematic literature review

ACM Reference Format:
Antonia Bertolino, Guglielmo De Angelis, Micael Gallego, Boni García, Francisco Gortázar, Francesca Lonetti, and Eda Marchetti. 2019. A Systematic Review on Cloud Testing. *ACM Comput. Surv.* 1, 1, Article 1 (January 2019), 41 pages. https://doi.org/10.1145/3331447

1 INTRODUCTION

At the beginning of this decade, cloud computing has been greeted as the "new frontier of Internet Computing" [111] that will finally realize the dream of delivering computing services as a utility, just as water or electricity. Research in the new technology has gained momentum since, and the IT industry is moving towards the cloud with huge investments and great expectations [150],[26].

Authors' addresses: Antonia Bertolino, CNR-ISTI, Pisa, Italy, antonia.bertolino@isti.cnr.it; Guglielmo De Angelis, CNR-IASI, Roma, Italy, guglielmo.deangelis@iasi.cnr.it; Micael Gallego, Universidad Rey Juan Carlos, Madrid, Spain, micael.gallego@urjc.es; Boni García, Universidad Rey Juan Carlos, Madrid, Spain, boni.garcia@urjc.es; Francisco Gortázar, Universidad Rey Juan Carlos, Madrid, Spain, francisco.gortazar@urjc.es; Francesca Lonetti, CNR-ISTI, Pisa, Italy, francesca.lonetti@isti.cnr.it; Eda Marchetti, CNR-ISTI, Pisa, Italy, eda.marchetti@isti.cnr.it.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2019 Copyright held by the owner/author(s). Publication rights licensed to ACM.
0360-0300/2019/1-ART1 $15.00
https://doi.org/10.1145/3331447

The acclaimed new computing paradigm does not come without challenges, though. To clear the initial confusion around its actual capabilities, Armbrust and coauthors [14] provided a view of the top 10 obstacles to cloud computing. Among them, the difficulty of "*removing errors in these very large-scale distributed systems*" [14] points to a broad and challenging research topic that is *cloud testing*, which refers to "testing and measurement activities on a cloud-based environment and infrastructure by leveraging cloud technologies and solutions" [47]. In recent years, researchers have actively investigated the scientific and technical problems posed by cloud testing, and have developed many techniques and tools for testing cloud-based systems.

Beyond addressing the challenge of testing systems that reside in the cloud, researchers have also realized the potential offered *by* the cloud to mitigate the ancient problem of high testing costs [22]. In fact, the cloud offers the opportunity to develop and maintain costly test infrastructures and to leverage on demand scalable resources for configuration (by using cloud virtualization) and performance (by means of cloud elasticity) testing. Thus the very term "cloud testing" is used in the literature with two different meanings: testing **of** the cloud or testing **in** the cloud.

In front of an active and continuing interest by the community in cloud testing research, there *does not exist a recent and comprehensive classification of research results* that can guide researchers in entering this field. Several authors provided an overview of the issues and the opportunities in testing of the cloud or testing in the cloud, *e.g.*, [16, 78, 126, 152], but not based on a systematic study of literature. A few systematic surveys or mapping studies have also been conducted, *e.g.*, [4, 71, 76, 115, 122, 164]. However, such studies either focus on specific aspects related to cloud testing, or are now several years old (see next section). In particular, the latest comprehensive surveys reviewed the literature until 2012.

Motivated by the above, this survey fills a gap by conducting a systematic literature review (SLR) [83] over the period (2012-2017) with the objective to identify and categorize relevant research on cloud testing. The study covers any aspect of testing of the cloud (ToC), testing in the cloud (TiC), and their intersection, *i.e.*, testing of the cloud in the cloud (ToiC). Our extensive "hunt" of literature included the automated search over six popular digital libraries (Scopus, ACM, IEEE, ScienceDirect, Wiley and Springer) and several snowballing iterations, both backward and forward (over Google Scholar). As a result, a total of 810 primary studies[1] have been scrutinized, of which 147 primary studies eventually passed the selection and are here surveyed.

A classification framework is proposed that divides the selected primary studies into the three (non-overlapping) categories of TiC, ToC, and ToiC. The categories are structured into six main testing research areas, namely: test perspective, design, execution, objective, evaluation, and domain. Each research area includes several topics that emerged from the reading of the studies.

The paper is structured as follows: in the next section we overview related work, *i.e.*, recent surveys in cloud testing. Then in Section 3 we describe our Research Questions (RQs) and the research methodology. In Section 4 we describe the derived classification framework, which is another contribution of this work. In Sections 5 and 6 we present the results from the survey: the former provides some interesting numerical analyses, while the latter includes an extensive discussion of insights gained from the full-text reading of the selected primary studies. Although the focus of this review is the scientific literature, the section also outlines industry trends in this field. Then, in Section 7 we summarize the open research challenges that emerge from the literature. Conclusions and directions for future research are given in Section 8.

[1]Precisely, 810 is the sum of: 655 papers resulting from a 1^{st} automated query, plus 124 papers from three snowballing iterations, plus 31 papers from a 2^{nd} automated query.

2 RELATED WORK

This paper fills a gap in cloud testing research: although several surveys exist, no previous work provides a comprehensive and up-to-date systematic survey of the field. To facilitate comparison, related works are summarized in Table 1 ordered by publication year. For each work, whose reference is in the first column, the table shows: in the second column the focus of the survey (some surveys cover a specific aspect, others are broad); in the third column the year of the most recent referred study; in the fourth column what is the research method (in particular whether the selection of covered studies is done *ad hoc* or following a systematic procedure); and finally in the fifth column either the number of selected primary studies, when available, or otherwise the whole number of paper references. As a general comment, we can see from the last column that this survey includes a quite larger set of studies than every other previous work.

Several papers overview informally existing methods and tools for cloud testing based on an *ad hoc* selection of the literature, without applying a systematic approach [16], [152], [78], [122], [126]. In the early years of the topic, such papers were certainly useful to provide a quick introduction to the field. Some of them contributed to establish a good categorization of relevant research trends, see, *e.g.*, [152], [78], whereas others presented cloud testing practices and tools, see, *e.g.*, [126],[16].

Among the early overview papers, the surveys by Inçki *et al.* [71] and by Priyanka *et al.* [115] make a systematic search of the literature and provide a comprehensive classification of research studies. However both works review the literature until 2012, and much research has been conducted after that year, so a new up-to-date SLR is necessary.

More recent SLRs related to cloud testing exist that cover specific topics within the broad field of cloud testing. For instance, Sakellari and Loukas [122] make a service to researchers by reviewing existing mathematical models, approaches to simulation and testbeds that can be used for conducting research in testing of the cloud. Zein *et al.* [164] survey methods and tools specifically aimed at testing of mobile applications, which also include, among others, cloud-based approaches. Such surveys partially overlap with this work, however none of them provides a survey of the whole cloud testing research field.

A peculiar case is the work by Jia *et al.* [76]: this paper proposes to use the well-known 5W + 1H pattern to guide the structuring of research questions for systematic mapping studies. Then, to demonstrate the approach, the paper conducts as a case study a systematic mapping study of cloud testing research that categorizes 51 primary studies. Although the paper was published in 2016, the set of included papers was selected in 2012, therefore also that work is antecedent to the period we consider here.

Finally, the closest work to this paper is the survey by Ahmad *et al.* [4] that focuses on the empirical studies in cloud testing papers. The work makes a literature search over the period (2010-2015) and provides a systematic mapping study over 69 primary studies (from 75 referred papers). In comparison to [4], this paper surveys a different period (2012-2017), and selects about twice as many papers. Moreover, for the years that are surveyed in both works (*i.e.,* 2012-2015), we can observe different selections of primary studies. This can be due to the usage by Ahmad *et al.* of a different (non-standard) search protocol (manual search on publication venues previously selected by an automated search), and by their more relaxed interpretation of the "testing" term to also include other verification and validation approaches, whereas this survey only focuses on testing approaches.

Paper	Focus of Survey	Newest Ref.	Survey Approach	Size
[16]	Needs and trends for cloud testing tools	2011	Informal	58 Refs
[152]	Methods and tools for cloud testing	2011	Informal	37 Refs
[78]	Methods and tools for cloud testing	2011	Informal	21 Refs
[71]	Research on ToC and TiC	2012	Systematic search (method details not described)	57 Refs
[115]	Research and frameworks for cloud testing	2012	Systematic search	82 Refs
[122]	Models and simulations useful for evaluation	2013	Informal	86 Refs
[126]	Specific goals of cloud testing	2015	Informal	23 Refs
[164]	Testing of mobile applications	2015	Systematic search	79 Refs
[76]	Mapping study of cloud testing	2012	Systematic search	51 Refs
[4]	Empirical studies over cloud testing	2015	Systematic search (manual on selected venues) + snowballing	69 Refs
This survey	Research on ToC, TiC, and ToiC	2017	Systematic search + snowballing + assessment	147 Refs

Table 1. Comparison with Related Works

3 RESEARCH METHODOLOGY

This survey follows the guidelines for *systematic reviews in software engineering research* by Kitchenham and coauthors [83], [24]. Following these guidelines, our research methodology included three main phases: planning the review (as described in Section 3.1), conducting the review (as described in Section 3.2) and reporting the results (we refer to Section 5 and Section 6 for results description).

The literature search included an automated query over three popular digital libraries and several iterations of the *snowballing approach* [155], plus an additional assessment over three more digital libraries of the completeness of results from the above search methodology (as described in Section 3.2.5).

3.1 Planning the Review

The main goal of this study is to understand the current state of art in cloud testing and reviewing the existing approaches. In particular, we identified the following research questions (RQs):

RQ1: What are the main objectives for cloud testing?
RQ2: How are cloud resources exploited for software testing?
RQ3: What are the test methods, techniques and tools mainly used in cloud testing?
RQ4: How are testing results evaluated in cloud testing?
RQ5: What are the research issues and future research directions of cloud testing?
RQ6: Which are the main application domains for software testing in the cloud?

The last question aims at understanding for what type of applications the testing has been migrated to the cloud, hence it only refers to TiC studies.

3.2 Conducting the Review

Conducting the review started with the identification of the relevant primary studies. Overall our search spanned over seven digital libraries that are the most commonly used in similar studies, namely: Scopus, ACM Digital Library, IEEE eXplore, Google Scholar, ScienceDirect, Wiley Online Library and Springer Link. The process was articulated in five steps as described below.

3.2.1 Automated search in digital libraries. In a first step we conducted an automated search in the following electronic sources that are of great relevance for software engineering research:

- Scopus (http://www.scopus.com)
- ACM Digital Library (http://dl.acm.org)
- IEEE eXplore (http://ieeexplore.ieee.org)

Specifically, we searched by title, abstract and keywords selecting *English papers* from 2012 to 2017. To be as comprehensive as possible, we defined a very general search string as shown in Listing 1. We included in the search also the acronym "TaaS" (Testing as a Service) because we noticed that it is commonly used in several cloud testing works. On the other hand, we decided not to search for other terms as, *e.g.*, "analysis", or "evaluation". Although these may be used as synonyms for test or testing, it is unlikely that a paper truly centered on testing would *never* use the words "test" or "testing‘" in its title or abstract or keywords.

```
{ test ␣cloud}␣<OR>␣{cloud␣test }␣<OR>␣{testing ␣over␣cloud}␣<OR>␣{cloud␣testing }␣<OR>␣{TaaS}␣<OR>
{ testing ␣in␣cloud}␣<OR>␣{cloud−based␣test}␣<OR>␣{cloud−based␣testing}␣<OR>␣{testing ␣in␣the␣cloud}
```

Listing 1. Search String

3.2.2 Selection based on inclusion/exclusion criteria. We performed a first selection by reading title, abstract and keywords of the papers and selecting them according to the *Inclusion and Exclusion Criteria* defined in Table 2a. These are quite standard criteria mainly based on relevance of scope. We also excluded works that are not primary studies and works that are not peer-reviewed or too short (*e.g.*, theses or abstracts). We also excluded monographs and books as these tend to present mature work illustrating and merging results that have previously appeared in journals or conferences.

3.2.3 Selection based on quality assessment. We then performed a second selection of the included papers based on the reading of the whole paper. To this purpose, we defined a quality assessment checklist, composed of the five criteria in Table 2b, and a *QualityScore*, given by the sum of the individual scores I_k as shown in Equation 1.

$$QualityScore = \sum_{k=1}^{k=5} I_k \tag{1}$$

The quality assessment procedure followed a conservative approach aimed at excluding only those papers having very low quality. It included two phases. In a first phase, each paper was read by a (randomly selected) author who assigned to each criterion of Table 2b a score I_k between 0 and 1 (precisely, equal to 0 if the paper did not satisfy that criterion, 0.5 if the criterion was partially satisfied, and 1 if it was clearly satisfied), so that the maximum possible *QualityScore* was 5. For each paper: if *QualityScore* was less than 2, it was excluded; if *QualityScore* was greater than or equal to 3, it was included; finally if *QualityScore* was less than 3 and greater than or equal to 2, the paper was labeled as *acceptable*.

In the second phase, another quality assessment was performed for the papers having an *acceptable* quality. For them a second author different from the first one read and assessed the paper following the same process of the first step and producing a second *QualityScore*. Then, all those papers for which the sum of the two *QualityScore*s (in the first and second step) was greater than or equal to 4.5 were included in the survey.

3.2.4 Searching based on snowballing. Snowballing [155] is a search approach commonly used to complement automated queries. We adopted both backward and forward snowballing to identify additional papers that the automated search of might have missed. Precisely, for each selected primary study we examined: for backward snowballing its list of references, and for forward

(a) Inclusion and Exclusion Criteria

Inclusion Criteria
Studies presenting cloud testing architecture/platform/framework
Studies presenting cloud testing strategies
Studies presenting cloud testing issues/goals
Studies presenting cloud testing services
Studies presenting case studies related to cloud testing
Exclusion Criteria
Studies not explicitly presenting testing solutions
Studies presenting surveys
Editorials, abstracts, panels, thesis, monographs, books

(b) Quality Assessment Checklist

	Items
I1	Is the problem of the study clearly defined?
I2	Is the contribution of the study clearly defined?
I3	Are the results clearly validated?
I4	Are limitations and future directions clearly stated?
I5	Is the focus of cloud testing clearly defined?
	Answer Scores for the Items
	No=0; Partially=0.5; Yes=1

Table 2

snowballing its citations in Google Scholar (http://scholar.google.com/). In both cases, we first selected all papers with publication year in the range 2012-2017 that were not already included in the survey; then we applied to the new found primary studies the same quality assessment process previously applied to the automatically retrieved papers.

As detailed in Section 5, the snowballing procedure lasted for three iterations. The first iteration (indicated as Snowball Iteration A in the figure) was applied to the start set of papers selected from the automatically found ones. The remaining iterations (Snowball Iteration B and C in the figure) were applied to the papers derived in iterations A and B of the search procedure, respectively, until no relevant new paper was found.

3.2.5 Assessing the Research Methodology. We finally conducted a further search of the literature with the aim of verifying that all the relevant primary studies have been included. We launched again an automated search over different electronic sources, precisely:

- ScienceDirect (https://www.sciencedirect.com)
- Wiley Online Library (https://onlinelibrary.wiley.com)
- Springer Link (https://link.springer.com)

As detailed in Section 5, this search did not found any relevant primary study to be added, thus confirming the reliability and completeness of the snowballing procedure.

4 A CLASSIFICATION FRAMEWORK FOR CLOUD TESTING RESEARCH

In Fig. 1, we present the framework we developed to characterize the cloud testing research and classify the papers. Aiming at completeness, we derived this framework incrementally. First a draft scheme was obtained based on our reading of titles, keywords and abstracts during paper selection; this scheme included six areas and several topics for each area. We then used this draft scheme to classify the papers while reading the full text, but also continued to add within each area new subtopics as needed. Finally, during data analysis we standardized/unified the new topics. The resulting framework is thus in itself a useful contribution to have a snapshot of trends in cloud testing research.

We now describe the six areas and their topics.

- *Test Perspective.* The papers belonging to this area present novel perspectives on cloud testing research. They address topics such as basic concepts, terminology, challenges, and future research directions of cloud testing, among others.
- *Test Design.* The papers belonging to this area include solutions targeting the design stage of testing activity. Specifically, they present analysis of test requirements, definition of a test model or a test metric as well as different test strategies for test cases generation or

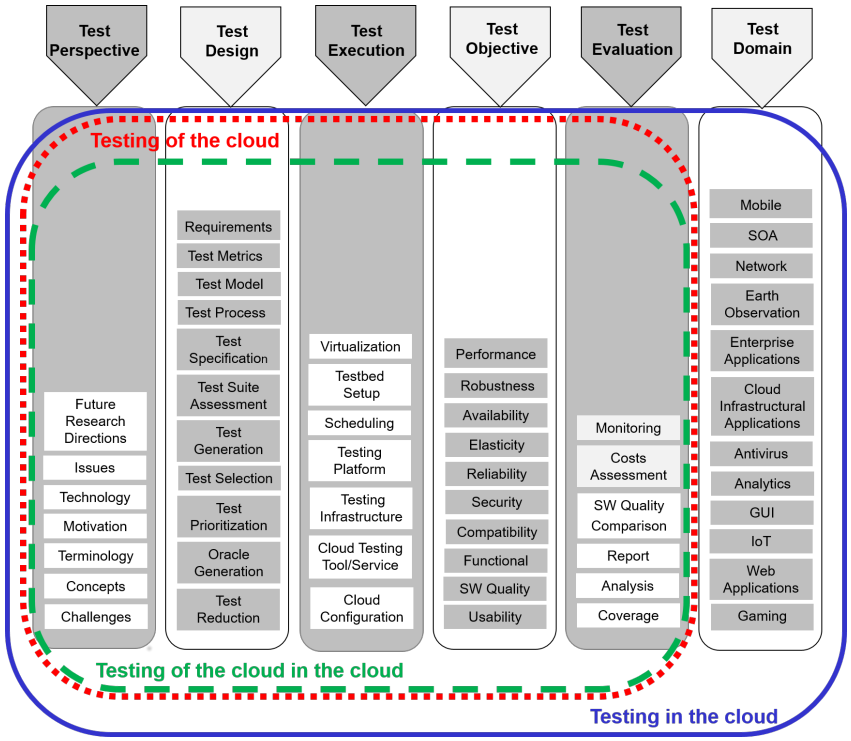


Fig. 1. Cloud Testing Framework

selection, test cases reduction or test suite assessment. In addition, the area also includes papers defining a cloud test process.

- *Test Execution.* The papers belonging to this area present artifacts involved into the execution phase of the testing activity. Specifically, they present 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 papers belonging to this area address the different purposes of cloud testing such as verifying that the systems comply with the functional specifications or show specified non-functional properties such as performance, reliability, robustness, usability, among others.
- *Test Evaluation.* The papers belonging to this area deal with the evaluation of the testing activity and results, providing support for test reports, analysis of test costs, or quality evaluation of different testing solutions.
- *Test Domain.* The papers belonging to this area present cloud-based testing solutions for the needs arising from specific application domains, such as mobile or web applications.

Overall, as anticipated, we classify papers into three different categories that span over the above areas as shown by the colored frames in Fig. 1:

- *Testing in the cloud* (blue continued frame in Fig. 1) refers to software testing performed by leveraging scalable cloud technologies, solutions and computing resources to validate *non-cloud* software/applications. This category includes testing solutions for different application domains, such as mobile or web environments, which are validated exploiting large-scale

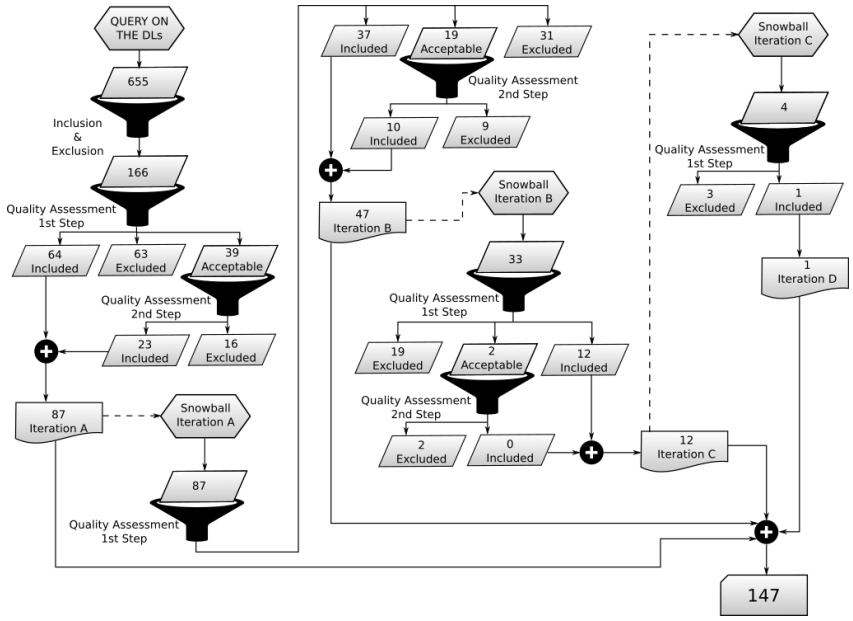


Fig. 2. Flow Chart of the Activities and their Quantitative Outcomes

simulations and elastic resources offered by the cloud. The main benefits of these solutions deal with: i) reducing costs by exploiting apparently unlimited computing resources in the cloud; ii) avoiding to develop and maintain testing infrastructure (scaffolding); iii) on-demand test services provided by a third party to conduct online validation for large-scale software systems.

- *Testing of the cloud* (red dotted frame in Fig. 1) refers to validating the quality (functional and non-functional properties) of applications and infrastructures that are deployed in the cloud. The focus is on the specific testing problems posed by systems residing in the cloud, thus papers belonging to this category aim at checking the provided automatic cloud-based functional services, as well as at validating their performance, scalability, elasticity and security. Moreover, software applications can be deployed on different clouds (*e.g.*, private, public or hybrid), hence testing can also focus on compatibility and interoperability among heterogeneous clouds resources.
- *Testing of the cloud in the cloud* (green dashed frame in Fig. 1) refers to applications and infrastructures deployed in the cloud and tested by leveraging cloud platforms. Papers belonging to this category fill the intersection area between *Testing of the cloud* and *Testing in the cloud.*

5 RESULTS

This section reports the number of primary studies selected in each step and presents several quantitative analyses of the review outcomes.

Table 3. Search Results from the Digital Libraries

Digital Library	Number of Results
Scopus	247
ACM	274
IEEE	134
Total	**655**

(a) Initial Search

Digital Library	Number of Results
ScienceDirect	17
Wiley Online Library	4
Springer Link	10
Total	**31**

(b) Assessment

5.1 Numerical outcomes

From the automated search[2] described in Section 3.2.1, an initial collection of 655 primary studies was found. The detailed results for the digital libraries considered in this step are reported in Table 3a.

This initial collection was filtered according to the inclusion and exclusion criteria stated in Table 2a, obtaining a reduced set of 166 papers. Of these, 87 passed the two-step quality assessment (the detailed results from the quality assessment procedure are reported in Fig. 2).

Applying the forward and backward snowballing (see Section 3.2.4) on the selected 87 papers (*i.e.*, Iteration A), 87 new peer-reviewed papers were identified, of which 47 passed the quality assessment selection.

The snowballing and quality assessment were iterated three more times, collecting a total of 147 (*i.e.*, 87 + 47 + 12 + 1) primary studies. Referring to Iteration B, Iteration C and Iteration D in Fig. 2 we give the number of new papers obtained and selected at each snowballing iteration.

After the snowballing terminated, a second automated query was launched, following the procedure presented in Section 3.2.5. As reported in Table 3b a total amount of 31 primary studies has been collected from the three databases.

By comparing this list with the whole set of papers already analyzed, we found that 10 papers were already included in the previous selection. We evaluated the remaining 21 papers by reading title, abstract and keywords and by applying the same *Inclusion and Exclusion Criteria* defined in Table 2a. As a result, no additional paper was added to the list of the already selected references. Precisely, 20 papers fell into one or more of the following categories: studies not explicitly focusing on testing, editorial contributions, books and surveys. One only last remaining reference was a reprinting of a paper already included. This result confirmed that the snowballing process was exhaustive, so we proceeded to the analysis and reporting phase.

The complete list of 147 selected papers is provided at the end of the paper.

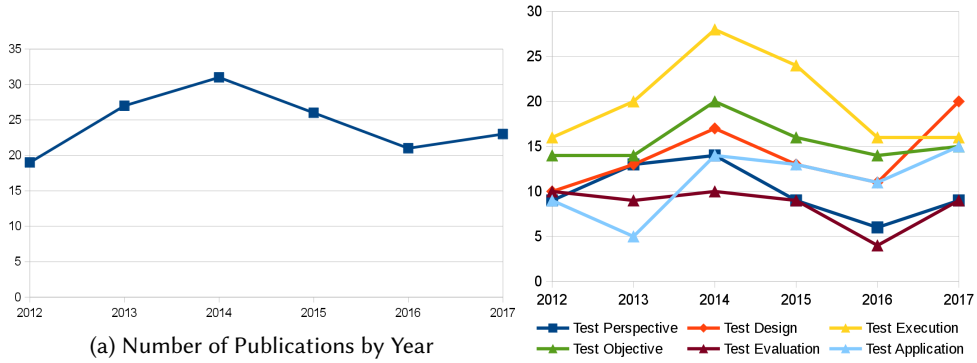
5.2 Quantitative Analyses

Fig. 3a depicts the overall distribution of selected primary studies over the years, whereas Fig. 3b details the trend per each of the six areas.

The overall distribution of primary studies over the six areas is shown in Fig. 4a. In particular, 60 papers were tagged as Test Perspective, 84 as Test Design, 120 as Test Execution, 93 as Test Objective, 51 as Test Evaluation, and 67 as Test Domain. Note that depending on its content each paper could be classified in multiple areas, so the histogram in Fig. 4a (as well as following distributions over the six areas) is not a partition and the sum of papers could be greater than their number (147).

Fig. 4b depicts the distribution of the primary studies among the three categories. As evident, most of the primary studies (*i.e.*, 62.59%) are in the TiC category, while less than 28% of the considered

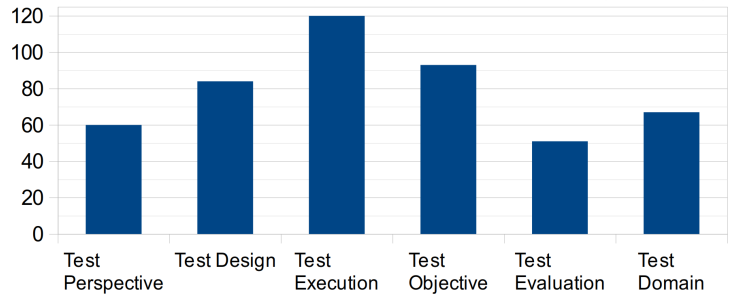
[2]The query was launched in April 2017.



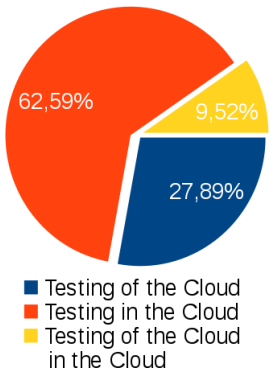
(a) Number of Publications by Year

(b) Trend of the Areas by Year

Fig. 3. Trends in the Primary Study by Year



(a) Distribution of the Papers by Area



(b) Partition of the Primary Study by Category

Fig. 4. Distributions of the Primary Study

papers targeted the problem of testing systems residing in the cloud, and only a minor part (*i.e.*, 9.52%) explicitly refers to testing cloud-based solutions using testing resources on a cloud platform.

The following figures from Fig. 5 to Fig. 10 depict the breakdown of each area into its specific topics (see Fig. 1), distinguishing the three categories of TiC, ToC and ToiC. We remark that, as in the case of classification by area, also multiple tagging by topics was admitted; thus the following breakdowns should not be intended as partitions.

More in detail, Fig. 5 reveals that most of the interest for these primary studies was in describing the challenges subsumed by the cloud testing approaches (*i.e.*, 26). Several primary studies presented concepts (*i.e.*, 12) and potential issues (*i.e.*, 13) of the paradigm. A minor number of primary studies directly addressed future research direction for cloud testing (*i.e.*, 8), while only few focused on aspects such as terminology (*i.e.*, 2), technologies (*i.e.*, 1), or motivations (*i.e.*, 1).

Fig. 6 highlights that among the 90 assigned topics in the area of Test Design, the most addressed topics are Test Generation (*i.e.*, 31) and Test Process (*i.e.*, 16). The topics Test Model (*i.e.*, 8), Test Metrics (*i.e.*, 8), Requirements (*i.e.*, 9), and Test Selection (*i.e.*, 7) were sufficiently covered, while only marginal attention was deserved to Test Reduction (*i.e.*, 3), Test Suite Assessment (*i.e.*, 1),

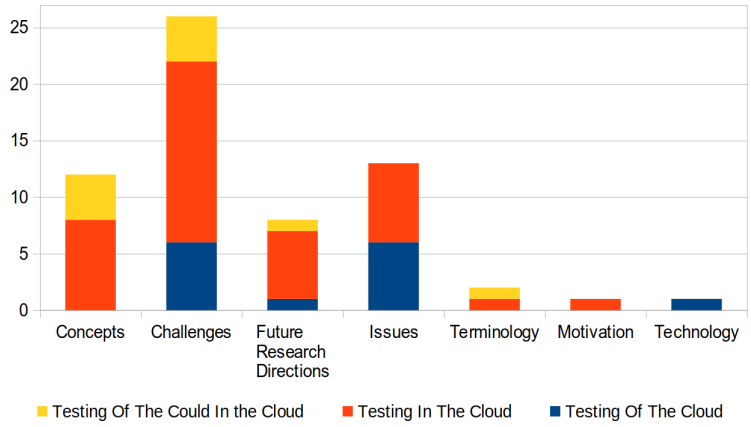


Fig. 5. Breakdown of the Primary Study in Test Perspective

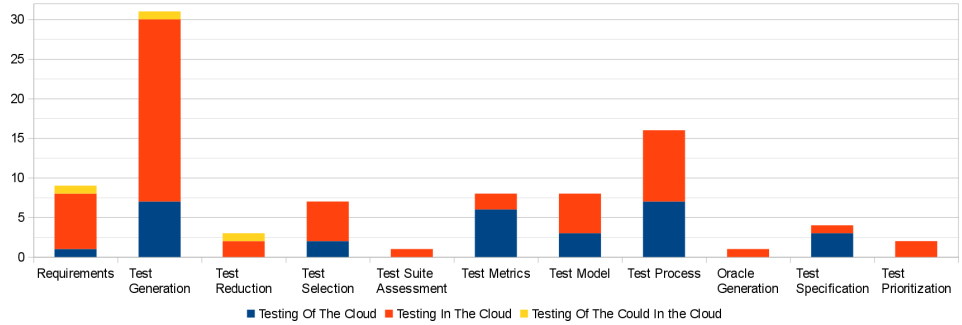


Fig. 6. Breakdown of the Primary Study in Test Design

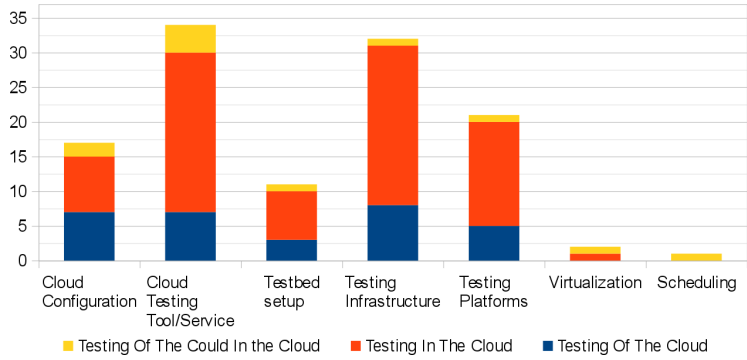


Fig. 7. Breakdown of the Primary Study in Test Execution

Oracle Generation (*i.e.*, 1), Test Specification (*i.e.*, 4), and Test Prioritization (*i.e.*, 2). As a further consideration, a very limited number of primary studies in this area (*i.e.*, 3) are specifically targeting Testing of the Cloud in the Cloud.

The breakdown of the primary studies in Fig. 7 remarks the strong interest towards Test Execution. Out of 118 expressed topics in this Area, several works present cloud testing tools or services (*i.e.*,

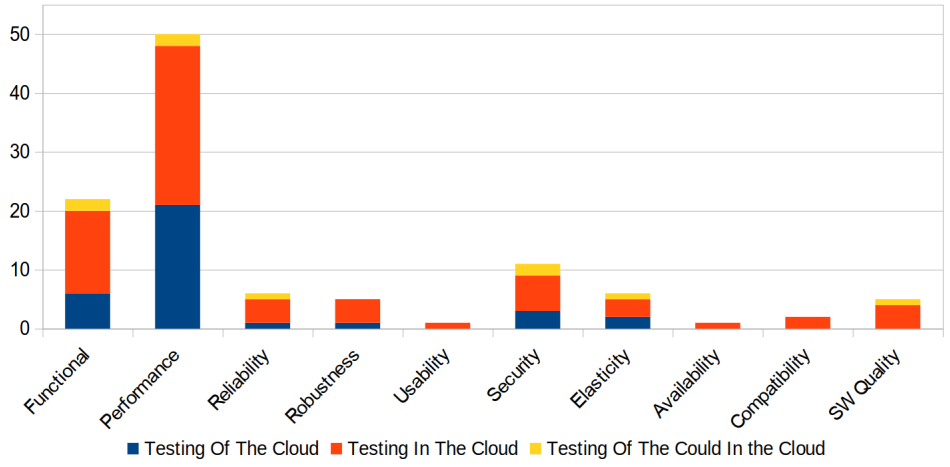


Fig. 8. Breakdown of the Primary Study in Test Objective

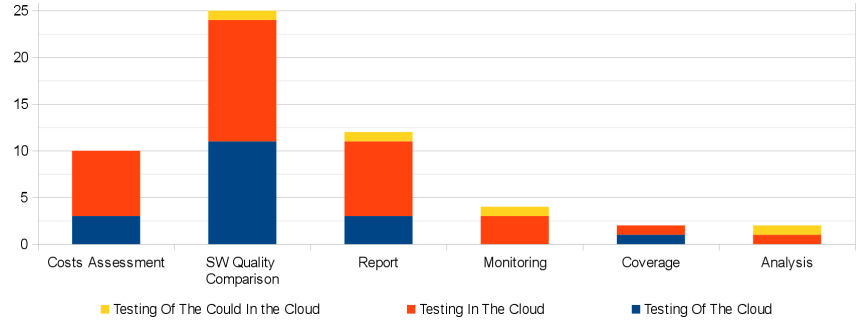


Fig. 9. Breakdown of the Primary Study in Test Evaluation

34), testing infrastructures (*i.e.*, 32) or platforms (*i.e.*, 21), the configuration of cloud instances (*i.e.*, 17). Few works discuss specific testbed setup in/for cloud environments (*i.e.*, 11), others address virtualization (*i.e.*, 2) or scheduling (*i.e.*, 1) aspects.

The analysis of the target goals for cloud testing is given in Fig 8. A good amount of primary studies covers functional testing (*i.e.*, 22), but as expected most of the effort has been spent on approaches that validate performances attributes (*i.e.*, 50 over 108 cumulative classifications in the area). Finally, it is interesting to notice that also other non-functional objectives are covered by the analyzed primary studies: security (*i.e.*, 11), elasticity (*i.e.*, 6), reliability (*i.e.*, 6), robustness (*i.e.*, 5), compatibility (*i.e.*, 2), availability (*i.e.*, 1), usability (*i.e.*, 1), or in general software quality (*i.e.*, 5). Even though the resulting set of non-functional attributes is broad, the papers per topic are not so many. This result highlights the versatility of cloud testing, but conversely evidences that non-functional testing other than performance appears much less mature.

In Fig. 9 the primary studies have been classified according to their capability of supporting the evaluation of the activities related to cloud testing. In this area most of the expressed tags (*i.e.*, 25 over 55) concern means for the comparison of quality attributes of the software-under-test (SUT) in different conditions (*e.g.*, configuration, deployment, load, etc.). Other topics in Test Evaluation such as monitoring (*i.e.*, 4), coverage (*i.e.*, 2), and analysis (*i.e.*, 2) received a marginal consideration;

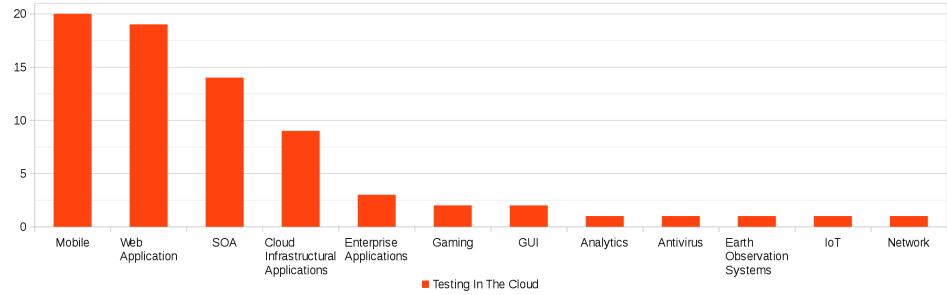


Fig. 10. Breakdown of the Primary Study in Test Domain

while both reporting (*i.e.*, 12), and costs assessment (*i.e.*, 10) were investigated much more. In our interpretation these results enforce the idea that cloud testing is perceived as the modern promise for quantitative analysis of the SUT. At the same time, a considerable attention is given to those methods able to feed the outcomes of the technological experimentation into proper methodological frameworks.

Finally, Fig. 10 reports the classification of the target domain of those primary studies testing a software/application in the cloud. Within this category, the collected data confirm that researchers leverage the cloud mostly for validating web application (*i.e.*, 19) or software specifically targeted to mobile devices (*i.e.*, 20); nevertheless also a relevant number of works referred to the cloud as a mean for testing SOA solutions (*i.e.*, 14). In addition, the review also found a discrete number of papers (*i.e.*, 9) addressing specific application domains such as: antivirus, earth observation, enterprise applications, gaming, GUI, high workload data analytics, IoT, or networks emulation.

6 ANSWERING THE RESEARCH QUESTIONS

We now summarize the main results presented in the primary studies, aiming at answering the research questions introduced in Section 3.1. The discussion is structured into three parts related to the three cloud testing categories. The section also includes a brief summary of recent surveys in industry [26], and an analysis of validity threats.

6.1 Testing in the Cloud

6.1.1 RQ1. As already said, the cloud overcomes the limits of traditional test approaches: indeed, the readiness of huge amount of resources, the possibility to manage big amounts of data and the availability of flexible, elastic environments allow to address testing objectives before considered infeasible. Among them the possibility of performing massive combinatorial testing, measuring performance in several (usage) scenarios, evaluating attributes such as scalability, elasticity, reliability by scaling up and down resources in the most convenient way.

Due to such motivations, several **functional** testing approaches are moved to the cloud, including model based testing, also based on formal specifications [60], [13]; coverage testing [81], [82]; use of combinatorial approaches for concurrently testing different configurations on different servers and in any order [156], [140], [139]. In particular the paradigm Testing-as-a-Service (TaaS) opens new perspectives for functional testing specifically in the mobile context (such as [166], [114], [93]), or GUI testing (such as [31]).

The availability of many and relatively cheap resources in the cloud makes **performance** testing easier, thus many frameworks have been developed addressing the objective of performance testing [35], [116], [75], [103], [112], [153], [58], [102]. However performance evaluation requires a

rigorous planning and the setting of specific configurations to maximize test effectiveness. Thus in this direction we found several research proposals supporting load testing [160], [52], [159]; performing the analysis of usage scenarios for the generation of performance test cases [131]; focusing on the guarantee of specific service level agreements [148]; targeting the management of large test jobs [92]; tracking and analyzing a huge number of events [77].

Elasticity is among the main reasons that make cloud computing an emerging trend. Elasticity testing in turn may have different objectives such as: to control different behaviors, to identify the resources to be (un)allocated, to coordinate events in parallel [9], [8], and of course to target scalability [17],[30], [91], [40].

Reliability testing focuses on the observation of the system under test under the operational usage profile. Proposals focus on measuring the reliability level [59] or on achieving a specific reliability value also through API testing [154], [99], [39].

The growth in complexity of pervasive software-intensive systems goes along the increased concern in the **security** of such systems, especially for mobile applications. Several recent proposals for testing in the cloud include approaches to virtualize, simulate and discover network attacks, protocols vulnerability and other security concerns [135], [147], [161], [66], [98].

6.1.2 RQ2. Cloud infrastructures are used to provide Testing as a Service (TaaS) following a **pay-per-use business policy** [119], [51], [48].

Cloud resources are exploited to achieve test cost saving, scalability and efficient utilization of the test resources, while guaranteeing a quality of service (QoS) level according to a negotiated service level agreement (SLA) [103]. **Efficient resource allocation** approaches and test scheduling solutions are proposed to maximize the utilization of test resources and balance the load among them [11], [60], [44]. Moreover, the work in [101] investigates the possibility of using hierarchical virtual machine fork for optimizing the cloud resources in system testing and saving system configuration effort as well as memory requirements by enabling disk sharing between concurrently executing test cases.

Different strategies are proposed to i) partition the testing tasks; ii) allocate them to different cloud processors for concurrent execution; iii) and collect results. Some proposals focus on **task decomposition** methods and **task scheduling** algorithms to decrease the testing time [93], [90], [92]. The goal is to balance the number of test cases or test suites in each decomposed task or the execution time required to perform each task [90].

By leveraging huge computing resources cloud testing allows for **large scale combinatorial testing** that was not possible in traditional test systems. Large number of processors are used to perform parallel test executions and identify faulty interactions through concurrent test algebra execution and analysis [139], [156]. For instance, in the largest combinatorial testing experiment presented in [145], [141], all 2-wise to 6-wise configurations with 2^{50} components are analyzed.

Cloud-based testing infrastructures are also proposed to **efficiently perform interoperability and compatibility testing**. For example, the work in [36] validates the interoperability among SOA enabled systems by checking the compliance of their communication protocols and the types of exchanged messages, whereas the authors of [93] propose an approach to partition compatibility test suites into concurrent testing tasks that are executed on a set of Android devices.

Dynamic resource adaptation strategies are defined to manage the cloud resources dynamically adding or removing virtual machines based on the workload of the cloud testing platform and the number of available devices [91]. The aim is to **balance the workload among several similar devices** to improve their usage and decrease the testing time.

The computation power of the cloud is leveraged to scale-up fuzz testing of Android applications [98] along the dimensions of code size and test cases number. Platforms such as Cloud

Crawler [35] allow cloud testers to better control the costs of cloud configuration and resources allocation (*e.g.*, by shutting down the VM after each individual test).

Finally, cloud resources are used in heavy testing techniques such as search based software testing using genetic algorithms. MapReduce is the most used model to process distributed data on multiple computers [69], [129], [40]. The goal is to **exploit easy-to-use parallelization mechanisms** for enlarging the solution spaces with respect to sequential search-based techniques and achieving higher efficiency and scalability, thus improving the cost-effectiveness of these approaches [40].

6.1.3 RQ3. As depicted also in Fig. 6 many techniques and tools address **test generation**. In this case, parallelization is used for mitigating the data values explosion problem [84]. Specifically, the Apache Hadoop MapReduce paradigm is used to support parallelization [29], [69], [40] of test data generation techniques such as for instance genetic algorithms [40]. Parallel concolic execution [31] and symbolic execution algorithms [10] have been defined for generating test cases, trying to overcome the path explosion problem by distributing the computation tasks over different workers on private as well as public clouds. Cloud9 is "an automated testing platform that employs parallelization to scale symbolic execution by harnessing the resources of commodity clusters" [27].

Model based testing allows to generate a high number of test cases to be executed on the cloud starting from an abstraction of the SUT. Different model based testing frameworks have been developed [12], [98], [82], [81], such as AUTOMATIC that derives many different QoS configurations to be tested in the cloud [12], or ATCloud that generates test cases based on API models [154], or the proposal of [82] that specifically addresses functional testing for composed services. MIDAS [58] is a model based scalable testing platform leveraging a Domain Specific Language (DSL), based on the Unified Modeling Language (UML) and the UML Testing Profile (UTP). EvoDroid [99] is another model based framework that analyzes the source code of an app, and automatically extracts both a behavioral model, and the APIs of externally referred apps. EvoDroid then exploits these models to automatically generate the tests that are executed concurrently in the cloud on several Android emulators.

Different testing frameworks leverage **combinatorial testing techniques** and use test algebra and adaptive test configuration generation algorithms that identify faulty interactions [145], [146], [140], [142], [144]. In particular, the test results by different processors are combined thanks to test algebra rules that identify those interactions that do not need to be tested. Different solutions address the problem of identifying the configurations to be deployed and tested on a cloud platform with the aim of reducing their number and saving testing effort [38], [12], [154], [145], [141], [146].

Many approaches deal with the **architectural design of cloud-based testing infrastructures** [84], [134], [133], [129], [11]. Some of them are usually tailored to specific application domains, including mobile and web applications. In the context of mobile testing, papers [134] and [133] present the design and implementation of cloud-based infrastructures as a service (known as MTaaS), trying to address the most important issues of testing mobile applications, whereas the work in [129] presents the architecture of a scalable platform for cloud testing of mobile systems allowing to add new testing functionalities such as non-functional testing or test planning.

Concerning web application testing, several works [107], [151], [11], [112] describe the architecture of testing services for analysis of web applications or web services compositions. An open and extensible cloud-based testing platform is MIDAS [58][60] supporting functional, security, and usage-based testing of service orchestrations. The authors of [36] propose a cloud-based multi-layer architecture for interoperability tests among distributed automation systems enabling the configurable compliance testing from protocol to system level. The authors of [17] present the architecture of Vee@Cloud that "serves as a scalable virtual test lab built upon cloud infrastructure services. The resource manager allocates Virtual Machine instances and deploy test tasks, from

a pool of available resources across different Clouds. Another platform is Cloud Crawler [35], which provides a declarative language supporting the description of many different performance evaluation scenarios to be executed in the cloud.

Many solutions present **frameworks and tools implementing the TaaS model** [116], [153], [147], [103], [69], [121], [147], [85], [44]. The common features of these tools are automated tests execution in parallel, computation scaling and configuration test setup. Many of them target mobile applications mainly deployed on Android devices [153], [121], [63], [99], [64], [114], [166], [56], [157], with the goal of performing performance and compatibility testing. They share features like test script generation, configuration of real or emulated devices on the cloud, automatic execution of the distributed tests, or test report generation including error location and error snapshots. Few solutions propose framework and tools addressing other domains such as GUI testing [31], security testing [147], [135], stress testing [69], web browser testing [57], or performance testing [103].

Different proposals focus on the **specification of cloud-based testing processes**. Some of them define common process steps, *e.g.*, selecting the types of testing to be executed, executing the test scripts and reporting the test results [153], whereas others define specific steps for cloud-based parallel test execution [92], such as specifying the test jobs and test deadlines, determining the number of virtual machines, computing test time/cost, partitioning of tasks based on the specified strategy and merging of test results. A specific test process is defined for mobile applications in [51] and [49]: it specifies as main steps unit and integration testing, tenant-based functional and QoS testing as well as continuous testing and testing of specific features of mobile systems. Another test process [59], [58], [112] specifically focuses on SOA applications identifying as main steps SOA monitoring, usage profile inference, test data repository creation, test model definition, and test generation and execution. Finally, the work in [81] shows that testing is an important part of the service life-cycle and proposes a model to "describe systematically the relevant processes governing services in the context of cloud brokerage".

Some papers address **testbed setup** [30], [1]. Specifically, the work in [30] proposes a testbed implementing sCloud that adaptively allocates the available resources to heterogeneous workloads in distributed data-centers, taking into account QoS requirements as well as real green power and workload traces. The work in [100] provides a test environment where mobile applications can be tested on different smart devices and mobile platforms, providing more realistic results than emulators. New objectives of benchmarking in the cloud are addressed in [42], whereas the work in [1] proposes new benchmarking solutions where controlled experiments are run on several clouds sharing a common orchestrator interface, and several multitiered applications are deployed fully automatically. Finally, an open-source and extensible testbed is PHINet [123], which supports development, testing, and analysis of Health-IoT in the cloud: it enables to run experiments under live traffic and various health sensors and allows users to control data acquisition and delivery.

6.1.4 RQ4. Several cloud testing solutions considered in this survey include a persistence layer responsible for storing historical data about past test executions, or applied configurations (*e.g.*, [129], [12], [107], [93], [140], [82], [40], [154], [60]).

In some cases, such a layer is wrapped by dedicated software components/modules offering a finer perspective about these data, rather than considering them as a mere collection of informative test reports. For example, we found that such components can enable the **comparison of the quality attributes** for a considered SUT under different conditions [116], [20], [1], [131], [77].

Among others, CloudPerf [103] integrates a native and extensible reporting infrastructure offering both rule-based querying and statistics providers. Also, the paper remarks the importance to properly control the format of the reports, for example by means of custom XML templates.

The framework in [153] supports developers and testers of mobile applications to assess the quality of their solutions on several mobile devices. For this reason it includes a web-based reporting front-end enabling the comparison of the results obtained during the tests executions.

The evaluation of test results in [159] and [160] is addressed by means of a reporting infrastructure that periodically retrieves, organizes, and stores the data produced by the allocated testing tasks. The main objective of such infrastructure is to analyze and to compare the impact of load testing for a target Web Service.

In [52] test results can be evaluated by means of a set of dedicated components that are responsible for gathering quality metrics (*e.g.*, throughput, response time, etc), for displaying online comparison charts, or for exporting test reports. Those components are exposed both as RPC interfaces, and by means of a presentation layer where a set of tenants can manage test result information.

The Cloud Crawler platform [35] collects and stores the performance results from each executed test scenario, and aggregates them in a spreadsheet document that can be used to compare the performance of alternative configurations, or the impact of different workloads.

In [85] the authors enrich test reports with structural **coverage** information about the source code achieved during the test sessions, which can be continually used during code development, but also as a means for structuring accounting of billing policies.

Live **monitoring** of test executions is also considered as a key feature for the evaluation of test results [103], [107], and [160]. The reason is to anticipate troubleshooting of issues, rather than waiting for the completion of a testing session.

Test results in cloud testing are also evaluated in terms of the **analysis of test costs**, and the works [170], [29], and [85] remark the importance of a predictive costing model. Different business models for cloud testing have been proposed. For example, there are approaches addressing cost-reduction by spreading common costs across parallel test cases execution [38], multiple tenants/renters [68], or implementing the *pay-as-you-test* model [51]. The latter framework explicitly includes components for accounting and billing [51].

6.1.5 RQ5. Many primary studies transfer approaches, methods and infrastructures conceived for traditional testing to the cloud [27]. For instance, the work in [85] migrates concolic test generation, the work in [40] focuses on the use of genetic algorithms for test data generation, the work in [131] proposes usage scenario for the formulation of performance test cases, the work in [102] targets the use of design patterns for performance testing, the work in [60] proposes black-box and gray-box techniques for test input and oracle generation. In addition several papers provide tutorials and informal surveys in the context of cloud testing such as [49], [68], or [119].

The rapid advance of cloud computing, with its deep impact on mobile and web application development, rises interesting **open problems that span all over the development process**, from requirements definition up to the final release [170], [108]. In this context, different works provide informative discussions and possible solutions about the issues of testing-as-a-service (TaaS) [132], [81], [129], [79], [82], [48].

In turn, test case specification and generation [29], and the definition of the most appropriate configurations to be tested [156], [140], [39], are still crucial questions in the cloud context. Some authors target the issues related to frameworks for **parallel tests execution** [116], problems on **effective and efficient allocation** of the available resources [30], [52], [92], [42], specific testing aspects such as security [161], [66] or Android devices and applications [99].

The works in [49] and [108] discuss perspectives related to migrating software testing in the cloud, including: i) the **availability of an elastic environment**, *i.e.*, the ability of dynamically scaling up and down testing resources as needed [30]. Such an environment may address different functional and non-functional aspects [103] and test monitoring facilities [8], [77]; ii) **speeding**

up of the testing activity, *i.e.*, the possibility offered by the cloud to execute in parallel different groups of tests or share and reuse testing resources [129], [60], [28]; iii) **self-service testing**, *i.e.*, the possibility to freely select or customize tools and platforms, configure the test settings, or remotely launch testing [29], [68]; and finally iv) **emulation** of real world scenarios [51].

Important **future directions** in cloud testing are represented by the possibility to develop and validate mobile applications and SaaS applications on mobile web [51], [112] and the availability of frameworks for measuring and certifying performance, quality, applicability, and usability in real world scenarios [13].

As emerged by this investigation, research on testing in the cloud is very active. Approaches, methods and infrastructures are continuously evolving to tackle both well-known and new issues, perspectives and future directions, especially in mobile or web application development. Moreover the possibility to validate the SUT under various configurations and different conditions allows testers to better assess and certify performance and quality attributes [36].

6.1.6 RQ6. From a detailed analysis of the papers labeled under the topic "**web application**", several types of cloud-based testing services/architectures can be distinguished. Some approaches [11], [69], [116], [160], [151] propose a TaaS architecture addressing non-functional requirements (*e.g.*, performance [11], [151], stress [69] and load [160] testing, scalability [151], security [66], etc.). Among the others, the work in [116] proposes an abstraction framework enabling the parallel execution of tests for web application on a local development workstation as both the tests and the application are deployed in the cloud. It provides faster test feedback to developers that can seamlessly apply the same operations both locally and on the cloud without having to care about deployment.

Some works focus on security aspects: the work in [147] validates remote web applications by means of cloud scanners, and the one in [13] formalizes an adaptive assurance technique based on online certification, foreseeing a certification authority regulating the on-line certification processes, and their related trust model for the cloud. The overall perspective is to enable chains of trust supported by the verifiable (non-)functional properties of cloud-based services.

An interesting perspective about the *lock-in* problem for users of TaaS platforms and services is discussed in [39]. Finally, the authors of [57] propose an approach aiming to detect potential cross-browser incompatibilities within web applications, impersonating users accessing a web application from different browsers.

Compatibility testing [121] is one of the most investigated areas within the **mobile** application domain [132]. A contribution common to the papers on this topic is that of checking the same application running on several kinds of mobile devices emulated in the cloud. Similarly, in [133] and its related papers (*e.g.*, [51], [134]), a comprehensive TaaS system is conceived aiming to validate complex mobile scenarios (*i.e.*, MTaaS). According to [133], MTaaS includes both a IaaS and a PaaS layer. The reference implementation of the MTaaS-IaaS supports the resource provisioning, monitoring and billing services.

Several works less ambitious than MTaaS aim to validate the functional behavior of one or more applications when running on different target devices, possibly under different configurations. Specifically, the work in [166] proposes a TaaS platform enabling the automatic generation of functional tests for mobile devices, which are then launched over several kinds of mobiles; the one in [153] proposes an approach that conforms to a set of international testing criteria; the works [93], [63] and [56] propose architectures that improve the efficiency of compatibility testing on mobile devices.

Concerning security, the work in [135] presents an automated approach for security testing of software in mobile phones: the authors adopted the full virtualization technology (*i.e.*, KVM) in

order to easily emulate terminals in the cloud. Each device hosts actual applications that are the target of vulnerability scanning frameworks enabled in the platform.

Then, there are works for mobile testing that are actually **agnostic** from any specific paradigm [99], [98]: however the authors explicitly validated their approach by leveraging the cloud paradigm, with the motivation to achieve several orders of magnitude improvement in execution time by running tests in parallel and on device emulators deployed on-demand.

Another considerable set of primary studies proposes a **TaaS architecture for SOA** (*e.g.*, [113], [112]). Among them, the papers [58], [60], and [59] belong to a series of works that perform SOA testing leveraging the already mentioned MIDAS platform [37], [18].

A methodological support is given in [81], which presents an approach for functional testing based on a Service Lifecycle Model. The contribution aims to "support providers during the service engineering phase, and consumers during the operation phase". Also the papers [156], [139], [140], [142], [143], and [144] are a series of methodological works leveraging an algebraic approach for testing SaaS.

An example of SOA testing of non functional attributes in the cloud is given in [148]. Here the authors validate proposed SLAs (*e.g.*, levels of availability, performances, reliability and other attributes) against the implementation of software services. The framework runs in the cloud sets of test cases designed according to a prescribed quality model.

Papers tagged as **Cloud Infrastructural Applications** include works aiming to test specific applications that could be used as building blocks for some cloud solutions. Under this topic, the work in [103] presents a performance testing framework designed on purpose for multi-tenant dynamic environments; the one in [52] migrates an existing load testing tool (Bench4Q) to the cloud; the one in [9] proposes an approach to reproduce elasticity testing in a deterministic manner; and the one in [75] presents a code generation framework for automated configuration and performance testing in several alternative scenarios.

About **Enterprise Application Software**, the authors of [119] investigate the adoption of cloud testing in practice by SMEs, and propose a structured approach for adopting cloud testing. Nevertheless from a practical perspective, each enterprise has different applications that can be tested in multiple different ways. The frameworks in [102] and [28] abstract most of the concepts of Enterprise Application testing, and discuss various solutions for leveraging cloud testing of non-functional properties (*e.g.*, performance [102], elasticity and reliability [28]).

6.2 Testing of the Cloud

6.2.1 RQ1. Analyzing the results of this systematic survey we can conclude that **performance** is the main objective for testing cloud-based systems. This happens 20 times in the list of selected papers [32], [97], [167], [2], [70], [124], [55], [118], [117], [158], [87], [127], [34], [7], [96], [6], [74], [65], [95], [67]. Some performance indicators assessed in these studies include response time, average latency, or execution time, to name a few.

Other different objectives exist to carry out software testing of the cloud, although in light of the survey results, these objectives appear as secondary in comparison with performance. These objectives are listed as follows, from highest to lowest order of appearance: i) **Functional** aspects, reported in 5 studies [45], [80], [25], [137], [128]; ii) **Security**, reported in 3 studies [105], [21], [163]. iii) **Elasticity**, reported in 2 studies [5], [62]. iv) **Reliability**, reported in 1 study [104].

6.2.2 RQ2. Cloud resources are used in different ways in the selected studies. The work in [124] presents a way for load generation for online testing on the cloud. The work in [33] presents a method for robustness testing of IaaS cloud platforms: test cases are generated by leveraging all the

combinations of input and state levels, applying various constraints. The authors of [6] create test sequences for detecting configurations that decrease cloud-based software systems performance.

Another significant number of studies uses cloud resources in different manners with the aim of assisting in the **testing process**, for instance the work in [105] presents an evaluation of different encryption algorithms (RC4, RC6, MARS, AES, DES, 3DES, Two-Fish, and Blowfish) on both desktop computer and Amazon EC2. The works [80] and [67] present BonFIRE, a multi-site testbed exposing cloud resources across different sites via a web Portal. Among other features, BonFIRE allows to create custom network configurations at scale based on cloud infrastructures. The work in [128] proposes a testing methodology together with a tool (called Elvior TestCast T3, TTCN-3) for automating use case testing.

Testing metrics are also widely used, see, *e.g.*, the study in [167], which presents a cloud framework for anomaly detection called eCAD. This tool internally uses an evolutionary data clustering algorithm called DBSCAN (Density-Based Spatial Clustering of Applications with Noise) to detect cloud anomalies, monitoring different performance indicators, such as CPU, memory, or input/output. Another example of the use of metrics is in [70], in which the authors present a generic cloud performance model for assessing IaaS, PaaS, SaaS, and mashup or hybrid clouds. This work uses different performance metrics, such as speedup, efficiency, latency, or bandwidth. Then, the work in [86] proposes a resource monitoring and management service for OpenStack-based cloud testing platforms for Android devices, using metrics such as average time or number of unit tests in their experiments. Finally, paper [62] proposes a framework to evaluate IaaS elasticity on mixed workloads. To that aim, the authors designed a workload using different patterns for the infrastructure cloud, and then derived aggregated performance metrics for elasticity, including resource, service and cost aspects.

6.2.3 RQ3. The test methods, techniques and tools used for testing of the cloud range over a rich variety of approaches. To start with, different types of **cloud configuration** are available: for instance, the work in [2] analyzes the scalability of the NoSQL database Cassandra using the Yahoo Cloud Serving Benchmark. The authors conclude that scaling the number of nodes does not guarantee an improvement on the performance, even for large data-sets. The work in [138] proposes an adaptive combinatorial testing of SaaS multi-tenant applications based on an Adaptive Reasoning (AR) algorithm using a small subset of the cloud configurations. The work in [7] proposes an approach that controls the resource variations when providing elasticity to web applications in the cloud and is validated using Amazon EC2. The work in [97] introduces a cloud simulation tool, called CloudAnalyzer, aimed at scaling applications deployed in the cloud under different configurations. The tool provides multi-threading and database support, as well as an algorithm editor window.

Cloud testing tools and services are presented in several papers. The authors of [89] present a tool to derive test cases from a formal specification expressed as a DSXM model. The authors of [45] showcase AUToCLES, a TaaS tool of cloud-based elastic systems: from a JOpera (*i.e.*, a visual composition language) specification of the SUT and a set of test cases defined using JMeter, AUToCLES instantiates the SUT, configures the testing environment, generates test input data and executes the test cases. The authors of [163] propose a model-based and change-driven solution for security testing: the approach identifies possible intrusion points by applying malicious users' techniques to the system interface. The work in [127] presents CloudBench, an open source IaaS cloud benchmark that is compatible with multiple cloud providers, such as, *e.g.*, Amazon EC2, OpenStack or Google Compute Engine. The one in [34] presents Cloud Crawler, a declarative environment for the description and execution of automated application performance tests. To

that aim, a DSL (called Crawl) is presented, supporting the description of many different IaaS performance evaluation scenarios.

Testbed setup is spread in some studies. For instance, the authors of [149] present the C-MART benchmark able to emulate modern web cloud applications. The tool is able to generate workloads emulating the access to the website. QoS performance measurement based on response time is proposed. The work in [3] presents aDock, a set of tools for creating sandboxes of cloud environments (based on OpenStack and Docker) that expose varying performance and configurable properties.

6.2.4 RQ4. In terms of test results evaluation, **performance** is again the most influential quality attribute considered by practitioners. This is in line with the results for RQ1, in which performance was also a key motivation to carry out testing of the cloud. In the analyzed studies, a common factor is the evaluation of some performance indicators (such as response or execution time among others), *e.g.*, comparing the results in terms of number of cloud nodes [2], cloud provider [70], type of cloud nodes (*e.g.*, small, medium, large, etc.) [70][55], type of SUT [74], desktop vs. cloud [105], or number of virtual users [158].

Another aspect of test results for applications deployed in the cloud is the **testing cost**. For instance, paper [104] presents the comparison of costs and footprint among different cloud configurations. In the same way, paper [95] studies the impact on testing costs of sequential vs. parallel execution on cloud infrastructures.

The data generated by tests are often gathered as **test reports**. These results are typically generated by testing frameworks or tools [137], [95], [104]. Similarly, **test coverage** is another way of evaluating test results, as shown in study [94].

6.2.5 RQ5. Different studies report **research directions** of testing of the cloud. For instance, the work in [25] presents a formal model for validating firewall configurations including packet filtering or NAT features. This approach has been implemented as a test framework that derives test cases generated on the basis of the formal model. Paper [137] explains "why using state-of-the art model-driven engineering (MDE) and model-based testing (MBT) tools is not adequate for testing uncertainties of cyber-physical systems in IoT cloud infrastructures". The authors of [104] explore the use of cloud technologies for debugging. The authors of [46] introduce new test approaches for elastic systems, mapping metaphors from elastic materials (such as deformation, plasticity, necking or fatigue) with analogy in elastic computing systems. The authors conclude that further research (from requirements engineering to programming language to maintenance) is needed to test elastic systems properly.

A number of papers report different types of **issues**. About [2], some issues have been already introduced in Section 6.2.3 while discussing about RQ3. The work in [55] presents an approach to support application capacity planning in IaaS clouds. It assumes that the application performance under defined configurations and workloads can be inferred from the resource configuration provided by the IaaS, using the total response time as the performance metric. The authors of [87] propose key software architectural drivers for cloud testing. These drivers are divided into two groups: i) Traditional testing management environment (non-cloud testing techniques, non-cloud testing methodologies, non-cloud standards ISO/IEC 25000 and IEEE Std 829-2008); and ii) Cloud test management environment (migration strategies, testing techniques and relevant factors, cloud infrastructure and architectural principles, standards, and collaboration and the best practices). The work in [74] evaluates some open source performance testing tools (Apache JMeter, Load Focus, Nouvula) on Flipkart, Snapdeal, and Amazon online shopping websites. The one in [65] proposes a lightweight test algorithm that can check if a cloud provider meets the agreed SLA in terms of CPU speed.

6.3 Testing of the Cloud in the Cloud

6.3.1 RQ1. The main objectives for moving software testing of the cloud in the cloud are **performance** [130], [41] and **functional** motivations [136], [19]. **Security** is the target of both the studies [88] and [109], whilst paper [50] covers multiple aspects: performance, elasticity, robustness, and reliability.

6.3.2 RQ2. In [165], stub cloud models are used for **test generation** aimed to achieve high structural coverage for cloud-based applications. These models allows to simulate real environment conditions using fake stubs that provide user-defined return values.

A **test case reduction** approach is presented in [19]. In this work, a validation method called Context-Assisted Test Case Reduction (CATCR) for cloud-based systems is proposed. This approach assesses cloud-based applications taking into account geographical context information, considering the relative importance of the test cases in their geographical cloud-location.

6.3.3 RQ3. **Cloud configuration** is addressed in [136]. This work presents a high level DSL to define the deployment process and resource requirements of a software system. This DSL is later transformed in a set of deployment and instantiation scripts for different cloud providers. Another approach in a similar context can be found in [101], which provides an overview of the configurable Chameleon Cloud testbed for researchers.

Cloud testing tools and services are addressed in different studies. For instance, the authors of [50] propose CTaaS, a cloud-based TaaS environment aimed at supporting SaaS performance and scalability testing. The authors of [41] present PEESOS-Cloud, an architecture for conducting experiments in services using the characteristics of the workloads. Paper [43] presents an API called IoTCloud that allows developers to create scalable high-performance IoT applications. Finally, paper [37] discusses the cloud-based software architecture in the MIDAS project in Amazon AWS.

Testing infrastructures aspects are addressed in different ways. The work in [19] introduces a validation method called Context-Assisted Test Case Reduction (CATCR) that supports test reduction based on the context. [72] proposes a cloud framework called PCTF that enables the integration of different independent test components.

6.3.4 RQ4. **Performance comparison** is the evaluation mechanism in study [130]. This paper studies the performance of Eucalyptus and OpenStack in terms of number of VMs and launch time.

6.3.5 RQ5. **Research Objectives** in ToiC are addressed in several papers, such as [88], [109], [37], [165]. A group of papers describes **basic research concepts** [50], [136]. The work in [125] provides a comprehensive cloud testing overview covering the relevant concepts, issues, benefits and goals. Finally, paper [19] discusses **future research directions**.

6.4 Industry surveys

While this survey focuses on the scientific literature on cloud testing, it may be interesting to also consider industrial trends and perspectives. A review of cloud testing approaches, tools and objectives in industry would however entail a quite different research methodology, such as a survey of gray literature or, better, directly interviewing practitioners and managers. Performing such a type of study goes beyond the scope and extent of this work, but for the sake of completeness we provide below a short summary of the results from existing studies.

Riungu-Kalliosaari and coauthors [120][3] have recently conducted an extensive and well-structured survey among practitioners investigating how and why the industrial software testing is moved

[3]Note that this work is also included among the 147 selected primary studies

to the cloud. Their study was based on semi-structured interviews, and involved thirty-five respondents from 20 organizations. The results revealed that industry is on high demand for testing resources, so that the main motivation to cloud testing adoption is improving the cost-effectiveness of testing process. In line with our findings, the study highlighted that the cloud is mainly used for performance and scalability testing. Practitioners find that a cloud-based environment enables CPU-intensive tasks, multi-platform and crowd-sourced testing and brings the benefits of reduced maintenance effort, while security is perceived as a risk.

We also consulted the latest report by CapGemini [26], published with Sogeti and Microfocus, which is considered as one of the most important sources of information about current practices and future trends in software quality in the industry. The research study involves 1,660 IT executives of different companies around the world.

The report highlights three main objectives for the next years: intelligent test automation and smart analytics, smart test platforms, and agile organization of Quality Assurance (QA) and test function. All three objectives involve in some way or another cloud infrastructures. Intelligent test automation and test analytics are topics that involve machine learning into the test execution and reporting. Smart test platforms leverage cloud resources to provide test environment and tools, including self-remediation approaches. Agile organization requires managing several testing environments at different points in the software life-cycle, something that can be provided by the huge amount of resources available in the cloud. Indeed, according to the report, 73% of organizations are already using environments deployed in the cloud, and 15% are using containers.

In relation to testing cloud-based applications the report points out several approaches. Around 63% of organizations mainly do performance testing, whereas security testing is mentioned by 62% of respondents. Assessing peak load requirements is another common testing scenario, approached by 57% of the organizations. Finally, 33% of respondents do not use any specific approach to test cloud-based applications.

One of the domain areas that have been mentioned in many of the selected primary studies in our survey is IoT. When we look at the status of IoT testing in the industry, numbers fall apart when compared with web or mobile testing. Only 32% of respondents having IoT products have a mature IoT testing environment. Around 51% of companies working on IoT products do not have yet a testing environment, although some of them are planning to invest in such an environment. This is one of the areas where researchers and industry are aligned in seek for better and more mature solutions.

In conclusion, we see several points of agreement between the trends in research and practice, although it would be desirable that a tighter collaboration is established between the two worlds [53].

6.5 Threats to validity

This section discusses threats to the internal, construct and external validity of our empirical study. Internal validity is concerned with the confidence on the reported results, and in this study the following aspects can be considered:

Authors' expertise. Our own expertise may have influenced papers selection and classification. In particular, the first step of the screening against quality criteria has been performed by only one author (randomly selected) and might have produced wrong exclusions (false negatives). To reduce this risk, the selection was articulated along several phases adopting a conservative approach, as described in Section 3. Only those papers receiving a very low score have been excluded, whereas for all the others considered of acceptable quality, an additional screening was performed by a second author. Concerning paper classification, along the process we run several meetings of all the authors in which we compared and aligned the respective assignments.

Framework definition and adoption. We contributed to both the definition of the classification framework and its usage for papers classification. This is an unavoidable threat of these studies. However, we make available the classification data to allow other researchers to evaluate the results validity.

Framework inclusiveness. The adopted classification framework might not be inclusive of all areas and topics characterizing cloud testing research. To overcome this risk, the framework has been derived incrementally. As described, starting from a first draft framework obtained reading only title, abstract and keywords, new subtopics have been added within each area after reading the whole paper.

Construct validity includes those threats concerning the correspondence of measures utilized to the related properties. In our study the following threats can be identified:

Identification of primary studies. To identify primary studies of this survey, we defined a search string. A different search string might have produced different results. While this is an intrinsic threat of all systematic surveys, we tried to mitigate this issue by defining a very general search string to be as comprehensive as possible. Another threat to our proposal related to the primary studies identification is due to the considered digital libraries. We initially used three very popular libraries, but it is likely that searching on other sources might have produced different results. This risk has been mitigated by using different iterations of backward and forward snowballing as a search procedure for complementing the search in digital libraries. Finally, we also performed a verification by launching a second automated search on three more libraries, and the results confirmed the validity of the search results. Thus, we see it as very unlikely that we have missed relevant studies.

Selection of primary studies. Exclusion or inclusion of papers was made by first reading title, abstract and keywords according to defined inclusion and exclusion criteria and then reading the whole paper according to a two-steps quality assessment procedure. There is the possibility that papers have been missed due to the defined inclusion/exclusion criteria or to the defined quality assessment checklist of the above selection procedure. However, for defining this selection procedure we followed the guidelines for systematic reviews in software engineering [83], [24] as well as the selection procedure followed in similar studies [73].

Finally, external validity threats descend from potential issues preventing result generalization. In our study, only a subset of papers concerning cloud testing research has been targeted, *i.e.*, only papers published in a five years period from 2012 to 2017. Therefore the results might not well represent the overall research in the field. We believe the risk is low because cloud testing is a new research topic and thus recent years can likely include most relevant advances in the field. A related threat is that the automated search was performed on April 26, 2017: as the field is growing fast, a later search would clearly find more recent papers that were not yet included in the digital libraries. This threat has been partially mitigated by forward snowballing that allowed us to include many additional papers citing the primary studies with publication year in the range 2012-2017, *i.e.*, spanning the whole 2017.

7 SUMMARY OF FINDINGS AND CHALLENGES

With the aim of providing cloud testing researchers with a compendium of the state of the art as emerging from the 147 primary studies, in this section we provide two summaries drawn from two different perspectives. In the next subsection we provide a one-page summary that recaps in tabular form the answers to the six research questions that we present in extended form in Sections 6.1, 6.2, and 6.3. In Section 7.2 we instead summarize the main research challenges that we identify along the six areas of the classification framework proposed in Fig. 1.

7.1 Main findings

Table 4 recaps concisely the main findings of the survey, classified along the six RQs.

TiC	ToC	ToiC
RQ1: main objectives		
Using cloud huge resources to overcome practical limits of testing. Assessing performance, elasticity, reliability and security.	Mostly assessing performance attributes such as execution time or latency, but also other aspects as elasticity or security.	Mostly assessing performance attributes and security, but also functional aspects as traditional application.
RQ2: resources exploited		
Allocating dynamically tasks and computing resources for efficiency (load balancing and resource utilization).	Customizing testing workloads and network configurations, also leveraging cloud performance models for evaluating target metrics.	Simulating the real environment or exploiting geographical context information.
RQ3: methods, techniques and tools		
Mostly model-based and combinatorial techniques, tools implementing the TaaS model, and testbed setup facilities.	Mainly tools and techniques for cloud configuration setup, test execution, test generation, and performance monitoring.	Mainly tools and techniques for cloud configuration setup.
RQ4: result evaluation		
Comparing mainly quality attributes but also business costs, through either a persistence layer or a live monitoring infrastructure.	Comparing mainly performance attributes but also business costs, often gathered as test reports.	Comparing performance attributes.
RQ5: research issues and future directions		
Many novel approaches, methods and infrastructures, especially for mobile or web applications, allowing various test configurations and conditions.	Many new issues and goals for moving existing model-based testing and debugging techniques to cloud, and for scalability and capacity planning.	Quite heterogeneous issues and research directions depending on the specific nature of the study.
RQ6: application domains		
Mostly mobile or web applications, and SOA solutions. But also specific applications as multi-tenant or data distribution infrastructures, network emulation and gamings.		

Table 4. Main findings

7.2 Research challenges along the six cloud testing areas

We now refer to the proposed framework (see Fig. 1) for classification of research in cloud testing, to make here a summary of the main challenges ahead as they emerge from the primary studies. We start by noticing that some of the challenges concern general aspects of the cloud testing problem that have a wide impact and thus return across more areas. Such common challenges include:

(i) **evolution**, related to the continuous and rapid evolution of cloud technology. This impacts transversally all areas, in that testing activities must continuously face novel challenges (test perspective and test objective) and adapt to new environment constraints and conditions (test design, execution, evaluation), while the test domain evolves itself as well.
(ii) **cost**, descending from the large dimensions and high complexity of cloud systems and their many possible configurations. Such high cost heavily impacts the areas of test design, execution and evaluation, and also affects test perspective.
(iii) **lack of standards**, clearly related to the newness of the cloud computing discipline. We find this challenge in studies concerning text execution and evaluation, as well as concerning the portability across test domains.

(iv) **elasticity**, relative to testing the capability of provisioning and de-provisioning cloud resources. This challenge impacts above all test execution and the area of test perspective as well, as it requires novel specific testing approaches.
(v) **security**, which is a crucial concern in cloud systems and entails even more difficulties than in traditional testing. This challenge clearly spans across all areas, mainly impacting test objective and test evaluation.

In the following we instantiate such cross-challenges within the relevant areas, and also present more challenges that are specific to each area.

7.2.1 Challenges in test perspective. Cloud testing is a novel field bringing **several new specific concepts, issues and technologies** related to many different testing aspects, spanning from resources management, performance evaluation, quality and risk assessment, computational infrastructures management and maintenance. As said, cloud technology is very dynamic and evolves incessantly. Hence, a big challenge is that any proposed **TiC solution needs to be continuously revised and adapted** to this evolution. Additionally, the **complexity** of applications and infrastructures that are deployed and tested in the cloud is increasingly higher. In our vision this continuous modification, growing and revision of the cloud ecosystem goes side by side with the continuous discovering of new challenges, perspectives and issues in cloud testing. One main ToC challenge due to the growing complexity is related to the capability of assessing the cloud application as a whole by means of **end-to-end** tests. As long as the applications under test grow, they tend to become more difficult to set up and configure. The automation and maintenance of the proper setup to support end-to-end testing is a challenge for practitioners. Moreover, end-to-end testing of cloud applications is usually a time-consuming activity, which yields high costs. **Elasticity** is commonly identified as a core property provided by cloud-based systems, and is one of the common challenges across areas. A potential ToC challenge in this aspect is the use of proper **workloads** to evaluate the elasticity of an application deployed on a given cloud solution. This challenge is usually divided in different parts, namely workload generation (typically using a given pattern or algorithm), scheduling and execution of load tests, measurement/monitoring (assessment on how the elasticity is actually behaving in the cloud application), and finally follow-up activities (quantification and improvement).

7.2.2 Challenges in test design. Concerning the challenges of test design in the cloud, existing techniques do not **take properly into account specific cloud environment features** such us heterogeneity, scalability, load balancing, communication, frequent failures, and synchronization between distributed components. Parallel algorithms for test data generation, such as parallel concolic execution, graph search heuristics or combinatorial solutions leverage efficient distributed computing architectures such as Apache Hadoop MapReduce to distribute the test generation tasks over public cloud, mitigating both event sequence explosion and data value explosion. An important challenge in this respect is identifying an **abstract representation of the evolving cloud environment**. The adoption of model-driven engineering and generative programming techniques help application developers to identify an abstract representation of the test scenario and to define the right combinations of configuration options for deployment and testing of their applications. However, well-defined test models and coverage criteria to address the constraints of the different cloud technologies or providers are lacking.

Maximizing the **effectiveness** of different kinds of tests for cloud applications is usually identified as a challenge. There are different techniques carried out to achieve this optimization, for example by means of test selection, prioritization, or reduction approaches. Another common research direction in this area is the use of **metaheuristic** search algorithms.

In cloud infrastructures, engineers usually integrate different SaaS and applications based on their provided APIs and connectivity protocols. This **integration** is challenging from a testing point of view due to the extra costs and difficulties that directly impact the design and implementation of the underlying tests.

In view of the dynamic and heterogeneous cloud environments, providing a rigorous **test plan that can take into account the costs of using a cloud environment** from utilization periods through disassembly remains a challenge. Public cloud providers have their operating models and pricing mechanisms, but offer very little interpretability when testers need to change vendors. Moreover, a good test plan should consider also associated hidden costs, such as the cost of encrypting data, before moving testing to a cloud environment, as well as the cost of monitoring the utilization of cloud resources to prevent over-usage and over-payment. Another important aspect to be addressed in the test plan is the **management of test data**, in particular appropriate security policies ruling the supplying of confidential or production data to third parties should be adopted whereas some strategies for filtering or scrutinizing data before testing in the cloud should be foreseen.

7.2.3 Challenges in test execution. According to test engineers feedback, the construction of a test environment in the cloud is tedious, time-consuming, and still involves high costs and complexity. More attention should be given to **making test execution in the cloud cost-efficient**, also trying to reduce the costs due to setting up the test environment on all the machines in the cloud. Indeed, improper sizes of the allocated virtual machines or unbalanced loads can result in low resource utilization or increased response time. Efficient strategies are also needed to **execute complex test scenarios by leveraging dynamic scalability and elasticity** of underlying computational resources. Important aspects to be further investigated are test decomposition policies, test allocation and test scheduling methods. These are needed to decompose the test jobs into more test tasks that can be executed concurrently so to improve resource utilization and computation time.

Concerning TiC, the **cloud-based test environment configuration** is still hard to realize: testers need to deal with combinations of various SaaS and applications according to the offered APIs and connectivity protocols. This task appears even more complex when legacy test software is migrated to the cloud. Traditional test configuration practices do not consider the **heterogeneity and complexity** of the cloud. The challenge in this direction is to investigate the development of a holistic testing framework as an integrated solution with a core TaaS infrastructure, enabling the ease of adding and scaling additional capabilities such as non-functional testing or test planning approaches. This testing framework could support the construction and deployment of **on demand virtual test lab** in a TaaS infrastructure enabling efficient tests execution as well as resource and tool license sharing. This allows to overcome the limitations of some cloud providers that offer only a reduced set of configurations, technology, storage, networking and bandwidth. The main obstacles to the realization of this framework still remain: the **lack of standard interfaces and connections to test tools** and third-party solutions as well as the complexity of connectivity with other clouds.

Another important issue is the lack of automated facilities for **dealing with test failures** or detected bugs during large-scale test execution. Effective test execution solutions as well as self re-settable and auto-recoverable test scripts are needed to support and process any test failures during automated test execution.

Concerning ToC, **multi-tenancy** is a common aspect of cloud-based applications. Complex scenarios involving SaaS multi-tenancy remain an open challenge. The use of **load balancing** technologies aimed at decoupling client traffic from application services is typically used for preventing data loss and network outages.

Finally, a relevant aspect of cloud technologies is **resource usage** (e.g. CPU, memory, disk, or network) and its corresponding **cost**. A potential challenge in this domain, especially for ToiC, is resource contention required for a given test suite execution on the different cloud providers.

7.2.4 Challenges in test objective. The need to provide viable solutions to meet testing needs within organizations and industries will push research of more effective means to support practitioners during development and testing activities, and the interest for TiC solutions will increase. In this direction, new objectives will involve the **decision-making processes and the management** of cloud-based testing.

Other important aspects descend from the need to **increase the confidence in the cloud system** and its components, from its infrastructure to the hosted applications. In our vision, certification, consistency, assurance, and assessment of the cloud environment can become the future keywords for the test objective in the TiC context.

Security has become a hot research topic in the testing community. Testing security aspects is especially challenging when the system under test is deployed in the cloud. Several open questions can lead to further investigation in this domain. For instance, how to assure and assess user privacy or business data privacy hosted in cloud infrastructures.

Achieving higher **scalability** and **performance** of ToiC approaches is a common challenge in the current state of the art of cloud testing. These two quality attributes are closely related to different challenges already presented, such as elasticity, multi-tenancy, or load balancing.

7.2.5 Challenges in test evaluation. In non-functional testing (e.g., load, performance, or stress testing) factors like network bandwidth or workload conditions that can affect the validation must be considered. Thus, in order to achieve meaningful test evaluation, it is important to be able to **properly control and trace such influencing conditions**. Even Service Level Agreements (SLAs), i.e., formal contracts that guarantee a negotiated QoS, are not always sufficient. Although SLAs are supposed not to be violated, it may happen that some violation occurs and impacts the outcome of the TiC session. In conclusion, there should be more emphasis on **linking the cloud test reports with information/metrics** that could help testers.

Most primary studies provide support for evaluating canonical IaaS indicators such as CPU or memory usage, and for reporting summary information. An interesting direction for **reporting capabilities** in TiC is providing native support for **customizable aggregation** of the monitored indexes.

The state of the practice in TiC reveals a large adoption of ad-hoc solutions for measuring/certifying different quality attributes. In the long term such a practice could result in some form of technical debt for customers relying on TiC (e.g. vendor lock-in). As an additional impact, the **lack of standardised reporting approaches** could also limit the possibility to move toward the creation of a concrete cloud brokerage ecosystem for TiC. More effort should be spent in promoting the application of well-known design pattern when structuring/scripting solutions specifically tailored for managing the results produced by a specific TaaS framework.

About security, a well-know obstacle to the adoption of TiC concerns the upload of a SUT in third-party premises. From our perspective it is important to remark that **confidentiality and protection** in both public and private clouds should be related to the whole set of testing artefacts. In this sense, technological or legal means (e.g. cryptography, obfuscation, or features enabling the "*right to be forgotten*") should not be limited to those artefacts loaded in the cloud to be executed, but also to the whole set of historical reports resulting from their execution.

The data gathered during testing and monitoring can be used to learn correlations between the expected test behavior and the observed one. Such correlations can be used to find bottlenecks and defects in the system under test. Facilitating decision making by means of different **machine**

learning approaches based on test reports and metrics can lead to promising research for testing cloud-based applications.

Similar challenges related to test data analysis are likely to happen also in the ToiC arena. In this domain, an additional problem is the inherent complexity of the cloud testing testbed, and as a result, the data volume can be increasingly higher. All in all, existing **big data** technologies can be useful for data discovery, integration or advanced analysis in the evaluation of test results.

7.2.6 Challenges in test domain. When adopting cloud-testing solutions the **impact of costs** is usually under-estimated. The pay-as-you-go business model is often referred, but not with sufficient consideration. For example, in order to launch testing sessions on fresh environments allocated on the cloud, a staging process is required for uploading the code that will be remotely executed with all its required dependencies. Moreover further computational resources are needed to properly configure the environment. Often there are hidden testing costs (e.g. due to packages set-up over the network) that increase with the usage of the monitoring and the logging resources of the specific TiC solution. The consequence is that a requested level of detail in the analysis of the test should better correspond to a clear understanding of its related costs and how such amount of information could be properly consulted and dug.

The rapid evolution of the cloud technologies and the lack of standards make also difficult the **portability** of specific application domains to different cloud providers. This problem usually forces practitioners to create custom testing solutions, for instance following TaaS or BaaS (Benchmark as a Service) on-demand approaches.

Testing complex cloud scenarios in specific domains (e.g. mobile apps) usually involves huge efforts for provisioning the proper infrastructure and configuration required for tests. The challenges in this arena is two-fold. On the one hand, the lack of **automation** can lead to high maintenance and operation costs of the proper testing testbeds. On the other hand, the lack of **open source** solutions can be a potential problem, especially for small or medium size projects aimed to develop and test specific applications (e.g. web, mobile).

8 CONCLUSIONS AND FUTURE RESEARCH DIRECTIONS

As emerged by our systematic review of the literature, in recent past years much research has been devoted to testing in the cloud, testing of the cloud and testing of the cloud in the cloud. The research related to testing activity in all the three categories is in continuous evolution and new approaches, methods and infrastructures are still proposed, especially in mobile systems, web application and SOA contexts.

We have developed a classification framework that, by construction, reflects what have been the main areas of research in the past recent years, namely test perspective, test design, test execution, test objective, test evaluation, and test domain. Within each of these areas we have also identified those topics that drew the greatest interest and in which several solutions have been proposed.

In particular, test execution is the most actively investigated area: indeed, the cloud offers the possibility to develop and maintain costly test infrastructures and to leverage on demand scalable resources for configuration (by using cloud virtualization) and performance (by means of cloud elasticity) testing.

The second most investigated area is test objective, with performance, functional, security, reliability and elasticity, in this order, being the most frequently covered ones in the surveyed primary studies. Indeed, the flexibility, the efficiency and the computational power of the cloud open new interesting testing possibilities considered infeasible before: massive combinatorial testing, huge amount of parallel executions, simulation and emulation, dynamic scheduling, allocation and

adaptation of resources, as well as a more effective and efficient evaluation of quality attributes such as scalability, elasticity, reliability, security and so on.

Considering the migration of testing to the cloud, not surprisingly the domains in which this happens most often are mobile and web applications. Other domains that could certainly benefit of the cloud potential but have not yet done so in large measure are IoT and networking.

The field still lacks proper conceptualization: a minority of papers covered test perspectives, within which -paradigmatically- the most covered topics are by far the open problems and issues. Topics as terminology and technology are almost not considered: we believe the field hardly needs a theoretical treatment, and we hope that this survey can provide a good input for such type of studies.

Our survey also revealed that very important components of any testing activity, such as test monitoring, coverage measurement and analytical techniques, useful for test evaluation, have received scant attention. Innovative testing infrastructures are needed that can support the assessment of cloud testing outcomes, possibly along different validation metrics.

If the great potential and the apparently unlimited resources that the cloud discloses open the way for pursuing innovative and more effective solutions for the testing activity in all its aspects, on the other hand controlling and managing them while testing also rise many new challenges. As a contribution to guide future research in cloud testing, we have provided a taxonomy of most relevant challenges that researchers could consider for future work.

From the results collected in this paper it seems clear that the future of software testing research will be more and more strictly intertwined with the progress of research and developments in cloud computing: the former providing approaches and methodologies for developing, validating, measuring and certifying applications, frameworks, tools and infrastructures, the latter providing the resources and facilities to assess, simulate or emulate real world scenarios.

As an example of a promising research effort in this direction we can refer to the H2020 European Project ElasTest [23] that has developed a comprehensive platform aiming 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) addressing several of the challenges we summarize in the previous section. For instance, it supports elastic end-to-end testing (test perspective), and provides different testing services in order to adapt to different test scenarios. Among others, it provides a cost engine to estimate the costs of using a test environment (test design), an instrumentation manager that can induce controlled failures into the infrastructure to simulate real world conditions (test execution), a security service to find vulnerabilities in the application (test objective), a big data service to analize test results (test evaluation), and emulation of Internet of Things devices, thus increasing portability and automation when testing IoT applications (test domain). On top of all services, ElasTest also provides specific visualization tools aimed at helping testers and developers in root cause localization for those bugs found during the testing process.

ElasTest is just an example of how specific tooling leveraging technologies can be developed to ease Toc, Tic and ToiC. We expect much more interesting research to appear in the coming years, so to disclose the whole potential of the cloud to defeat testing barriers.

ACKNOWLEDGMENTS

This paper describes research work mostly undertaken in the context of the European Project H2020 731535: ElasTest.

This work has also been partially supported by: the Italian MIUR PRIN 2015 Project: GAUSS; the Regional Government of Madrid (CM) under project Cloud4BigData (S2013/ICE-2894) cofunded by FSE & FEDER; and the Spanish Government under project LERNIM (RTC-2016-4674-7) cofunded by the Ministry of Economy and Competitiveness, FEDER & AEI.

REFERENCES

[1] Sameera Abar, Pierre Lemarinier, Georgios K Theodoropoulos, and Gregory MP OHare. 2014. Automated dynamic resource provisioning and monitoring in virtualized large-scale datacenter. In *Advanced Information Networking and Applications (AINA), 2014 IEEE 28th International Conference on*. IEEE, 961–970.

[2] Veronika Abramova, Jorge Bernardino, and Pedro Furtado. 2014. Testing cloud benchmark scalability with cassandra. In *Services (SERVICES), 2014 IEEE World Congress on*. IEEE, 434–441.

[3] Lorenzo Affetti, Giacomo Bresciani, and Sam Guinea. 2015. aDock: a cloud infrastructure experimentation environment based on Open Stack and Docker. In *Cloud Computing (CLOUD), IEEE 8th International Conference on*. IEEE, 203–210.

[4] Amro Al-Said Ahmad, Pearl Brereton, and Peter Andras. 2017. A Systematic Mapping Study of Empirical Studies on Software Cloud Testing Methods. In *Companion of Software Quality, Reliability and Security*. IEEE, 555–562.

[5] Michel Albonico, Amine Benelallam, Jean-Marie Mottu, and Gerson Sunyé. 2016. A DSL-based approach for elasticity testing of cloud systems. In *Proc. of the International Workshop on Domain-Specific Modeling (DSM 2016)*. 8–14.

[6] Michel Albonico, Stefano Di Alesio, Jean-Marie Mottu, Sagar Sen, and Gerson Sunyé. 2017. Generating Test Sequences to Assess the Performance of Elastic Cloud-based Systems. In *Proc. of Inter. Conf. on Cloud Computing*. IEEE, 383–390.

[7] Michel Albonico, Jean-Marie Mottu, and Gerson Sunyé. 2016. Controlling the elasticity of web applications on cloud computing. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing*. ACM, 816–819.

[8] Michel Albonico, Jean-Marie Mottu, and Gerson Sunyé. 2016. Monitoring-based testing of elastic cloud computing applications. In *Companion Publication for International Conference on Performance Engineering*. ACM, 3–6.

[9] Michel Albonico, Jean-Marie Mottu, Gerson Sunyé, and Frederico Alvares. 2017. Making Cloud-based Systems Elasticity Testing Reproducible. In *7th International Conference on Cloud Computing and Services Science*.

[10] Nassima Aleb and Samir Kechid. 2012. Path coverage testing in the cloud. In *Communications and Information Technology (ICCIT), 2012 International Conference on*. IEEE, 118–123.

[11] Amira Ali and Nagwa Badr. 2016. Performance testing as a service for web applications. In *2015 IEEE 7th International Conference on Intelligent Computing and Information Systems, ICICIS 2015*. 356–361.

[12] Kyoungho An, Takayuki Kuroda, Aniroddha Gokhale, Sumant Tambe, and Andrea Sorbini. 2014. Model-driven generative framework for automated OMG DDS performance testing in the cloud. *ACM Sigplan Notices* 49, 3 (2014), 179–182.

[13] Marco Anisetti, Claudio Ardagna, Ernesto Damiani, and Filippo Gaudenzi. 2017. A semi-automatic and trustworthy scheme for continuous cloud service certification. *IEEE Transactions on Services Computing* (2017).

[14] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy Katz, Andy Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, Ion Stoica, and Matei Zaharia. 2010. A View of Cloud Computing. *Commun. ACM* 53 (2010), 50–58.

[15] Sunitha Badanahatti and Yelisetty Satya Sree Rama Murthy. 2016. Optimal Test Case Prioritization in Cloud based Regression Testing with Aid of KFCM. *system* 97 (2016).

[16] Xiaoying Bai, Muyang Li, Bin Chen, Wei-Tek Tsai, and Jerry Gao. 2011. Cloud testing tools. In *Proceeding of 6th International Symposium on Service Oriented System Engineering*. IEEE, 1–12.

[17] Xiaoying Bai, Muyang Li, Xiaofei Huang, Wei-Tek Tsai, and Jerry Gao. 2013. Vee@Cloud: The virtual test lab on the cloud. In *2013 8th International Workshop on Automation of Software Test, AST 2013 - Proceedings*. 15–18.

[18] Miguel Angel Barcelona, Laura García-Borgoñón, and Gonzalo López-Nicolás. 2017. Practical experiences in the usage of MIDAS in the logistics domain. *Int. Journal on Software Tools for Technology Transfer* 19, 3 (2017), 325–339.

[19] Feras A Batarseh, Avelino J Gonzalez, and Rainer Knauf. 2013. Context-assisted test cases reduction for cloud validation. In *International and Interdisciplinary Conference on Modeling and Using Context*. Springer, 288–301.

[20] Jonathan Becedas. 2014. The geo-cloud experiment: Global earth observation system computed in cloud. In *Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2014 Eighth International Conference on*. IEEE, 9–15.

[21] Elhadj Benkhelifa and Thomas Welsh. 2013. Security testing in the cloud by means of ethical worm. In *2013 IEEE Globecom Workshops, GC Wkshps 2013*. 500–505.

[22] Antonia Bertolino. 2007. Software Testing Research: Achievements, Challenges, Dreams. In *Int. Workshop on the Future of Software Engineering, (FOSE @ ICSE 2007)*. IEEE, Minneapolis, MN, USA, 85–103.

[23] Antonia Bertolino, Antonello Calabró, Guglielmo De Angelis, Micael Gallego, Boni García, and Francisco Gortázar. 2018. When the Testing Gets Tough, the Tough Get ElasTest. In *Companion Proc. of the 40th International Conference on Software Engineering (ICSE '18)*. ACM, New York, NY, USA, 17–20.

[24] Pearl Brereton, Barbara A Kitchenham, David Budgen, Mark Turner, and Mohamed Khalil. 2007. Lessons from applying the systematic literature review process within the software engineering domain. *Journal of systems and software* 80, 4 (2007), 571–583.

[25] Achim D Brucker, Lukas Brügger, and Burkhart Wolff. 2015. Formal firewall conformance testing: An application of test and proof techniques. *Software Testing, Verification and Reliability* 25, 1 (2015), 34–71.

[26] Mark Buenen and Govind Muthukrishnan. 2017. World Quality Report: 2017-2018.

[27] Inderveer Chana and Priyanka Chawla. 2013. Testing perspectives for cloud-based applications. In *Software Engineering Frameworks for the Cloud Computing Paradigm*. Springer, 145–164.
[28] Muhammad Aufeef Chauhan, Muhammad Ali Babar, and Christian W Probst. 2016. A Process Framework for Designing Software Reference Architectures for Providing Tools as a Service. In *Int. Conf. on Product-Focused Software Process Improvement*. Springer, 111–126.
[29] Priyanka Chawla, Inderveer Chana, and Ajay Rana. 2016. Cloud-based automatic test data generation framework. *J. Comput. System Sci.* 82, 5 (2016), 712–738.
[30] Dazhao Cheng, Changjun Jiang, and Xiaobo Zhou. 2014. Heterogeneity-aware workload placement and migration in distributed sustainable datacenters. In *Proc. of the Parallel and Distributed Processing Symposium*. IEEE, 307–316.
[31] Lin Cheng, Jialiang Chang, Zijiang Yang, and Chao Wang. 2016. GUICat: GUI testing as a service. In *Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering*. ACM, 858–863.
[32] Chuan-Yen Chiang, Che-Pin Chang, Hung-Yu Chen, Yen-Lin Chen, Shyan-Ming Yuan, and Che Wang. 2016. ATP: A Browser-Based Distributed Testing Service Platform. In *Computer Symposium (ICS), 2016 International*. IEEE, 192–197.
[33] Domenico Cotroneo, Flavio Frattini, Roberto Pietrantuono, and Stefano Russo. 2015. State-based robustness testing of IaaS cloud platforms. In *Proceedings of the 5th International Workshop on Cloud Data and Platforms*. ACM, 3.
[34] Matheus Cunha, Nabor Mendonca, and Americo Sampaio. 2013. A declarative environment for automatic performance evaluation in iaas clouds. In *Proc. of Inter. Conf. on Cloud Computing*. IEEE, 285–292.
[35] Matheus Cunha, NC Mendonça, and Américo Sampaio. 2017. Cloud Crawler: a declarative performance evaluation environment for infrastructure-as-a-service clouds. *Concurrency & Computation: Practice and Experience* 29, 1 (2017).
[36] Wenbin William Dai, Laurynas Riliskis, Valeriy Vyatkin, Evgeny Osipov, and Jerker Delsing. 2014. A configurable cloud-based testing infrastructure for interoperable distributed automation systems. In *Proc. of Annual Conf. of Industrial Electronics Society (IECON)*. IEEE, 2492–2498.
[37] Alberto De Francesco, Claudia Di Napoli, Maurizio Giordano, Giuseppe Ottaviano, Raffaele Perego, and Nicola Tonellotto. 2014. A soa testing platform on the cloud: The midas experience. In *Intelligent Networking and Collaborative Systems (INCoS), 2014 International Conference on*. IEEE, 659–664.
[38] Gustavo Sávio De Oliveira and Alexandre Duarte. 2014. A framework for automated software testing on the cloud. In *Parallel and Distributed Computing, Applications and Technologies, PDCAT Proceedings*. 344–349.
[39] Ricardo Ramos de Oliveira, Rafael Messias Martins, and Adenilso da Silva Simao. 2017. Impact of the Vendor Lock-in Problem on Testing as a Service (TaaS). In *Proc. of Int. Conf. on Cloud Engineering (IC2E)*. IEEE, 190–196.
[40] Sergio Di Martino, Filomena Ferrucci, Valerio Maggio, and Federica Sarro. 2012. Towards migrating genetic algorithms for test data generation to the cloud. *Software Testing in the Cloud: Perspectives on an Emerging Discipline* (2012), 113–135.
[41] Carlos HG Ferreira, Luiz H Nunes, Lourenço A Pereira, Luis HV Nakamura, Julio C Estrella, and Stephan Reiff-Marganiec. 2016. PEESOS-Cloud: a workload-aware architecture for performance evaluation in service-oriented systems. In *Services (SERVICES), 2016 IEEE World Congress on*. IEEE, 118–125.
[42] Enno Folkerts, Alexander Alexandrov, Kai Sachs, Alexandru Iosup, Volker Markl, and Cafer Tosun. 2012. Benchmarking in the cloud: What it should, can, and cannot be. In *Technology Conference on Performance Evaluation and Benchmarking*. Springer, 173–188.
[43] Geoffrey C Fox, Supun Kamburugamuve, and Ryan D Hartman. 2012. Architecture and measured characteristics of a cloud based internet of things. In *Proc. of Int. Conf. on Collaboration Technologies and Systems (CTS)*. IEEE, 6–12.
[44] Alessio Gambi, Alessandra Gorla, and Andreas Zeller. 2017. O! Snap: Cost-Efficient Testing in the Cloud. In *Software Testing, Verification and Validation (ICST), 2017 IEEE International Conference on*. IEEE, 454–459.
[45] Alessio Gambi, Waldemar Hummer, and Schahram Dustdar. 2013. Automated testing of cloud-based elastic systems with AUToCLES. In *Proc. of Int. Conf.on Automated Software Engineering (ASE)*. 714–717.
[46] Alessio Gambi, Waldemar Hummer, Hong-Linh Truong, and Schahram Dustdar. 2013. Testing elastic computing systems. *IEEE Internet Computing* 17, 6 (2013), 76–82.
[47] Jerry Gao, Xiaoying Bai, and Wei-Tek Tsai. 2011. Cloud testing-issues, challenges, needs and practice. *Software Engineering: An International Journal* 1, 1 (2011), 9–23.
[48] Jerry Gao, Xiaoying Bai, Wei-Tek Tsai, and Tadahiro Uehara. 2013. Testing as a service (TaaS) on clouds. In *Service Oriented System Engineering (SOSE), 2013 IEEE 7th International Symposium on*. IEEE, 212–223.
[49] Jerry Gao, Xiaoying Bai, Wei-Tek Tsai, and Tadahiro Uehara. 2014. Mobile application testing: a tutorial. *Computer* 47, 2 (2014), 46–55.
[50] Jerry Gao, K Manjula, P Roopa, E Sumalatha, Xiaoying Bai, Wei-Tek Tsai, and Tadahiro Uehara. 2012. A cloud-based TaaS infrastructure with tools for SaaS validation, performance and scalability evaluation. In *Cloud Computing Technology and Science (CloudCom), 2012 IEEE 4th International Conference on*. IEEE, 464–471.
[51] Jerry Gao, Wei-Tek Tsai, Ray Paul, Xiaoying Bai, and Tadahiro Uehara. 2014. Mobile testing-as-a-service (MTaaS) - Infrastructures, issues, solutions and needs. In *Proc.of the Int. Sym. on High-Assurance Systems Engineering*. 158–167.

[52] Qiang Gao, Wei Wang, Guoquan Wu, Xuan Li, Jun Wei, and Hua Zhong. 2013. Migrating load testing to the cloud: a case study. In *Service Oriented System Engineering (SOSE), 2013 IEEE 7th International Symposium on*. IEEE, 429–434.
[53] Vahid Garousi and Michael Felderer. 2017. Worlds Apart: Industrial and Academic Focus Areas in Software Testing. *IEEE Software* 34, 5 (2017), 38–45. https://doi.org/10.1109/MS.2017.3641116
[54] Alim Ul Gias, Asif Imran, Rayhanur Rahman, and Kazi Sakib. 2013. IVRIDIO: Design of a software testing framework to provide Test-first Performance as a service. In *Proc. of Int. Conf. on Innovative Computing Technology, (INTECH)*. 520–525.
[55] Marcelo Gonçalves, Matheus Cunha, Nabor C Mendonça, and Américo Sampaio. 2015. Performance inference: A novel approach for planning the capacity of iaas cloud applications. In *Cloud Computing (CLOUD), 2015 IEEE 8th International Conference on*. IEEE, 813–820.
[56] Pablo Graubner, Lars Baumgärtner, Patrick Heckmann, Marcel Müller, and Bernd Freisleben. 2015. Dynalize: Dynamic Analysis of Mobile Apps in a Platform-as-a-Service Cloud. In *Proc. of Inter. Conf. on Cloud Computing*. IEEE, 925–932.
[57] Meimei He, Guoquan Wu, Hongyin Tang, Wei Chen, Jun Wei, Hua Zhong, and Tao Huang. 2016. X-check: a novel cross-browser testing service based on record/replay. In *Proc. of Int. Conf. on Web Services (ICWS)*. IEEE, 123–130.
[58] Steffen Herbold, Alberto De Francesco, Jens Grabowski, Patrick Harms, Lom M Hillah, Fabrice Kordon, Ariele-Paolo Maesano, Libero Maesano, Claudia Di Napoli, Fabio De Rosa, et al. 2015. The MIDAS cloud platform for testing SOA applications. In *Proc. of Int. Conf. on Software Testing, Verification and Validation (ICST)*. IEEE, 1–8.
[59] Steffen Herbold, Patrick Harms, and Jens Grabowski. 2017. Combining usage-based and model-based testing for service-oriented architectures in the industrial practice. *International Journal on Software Tools for Technology Transfer* 19, 3 (2017), 309–324.
[60] Lom Messan Hillah, Ariele-Paolo Maesano, Fabio De Rosa, Fabrice Kordon, Pierre-Henri Wuillemin, Riccardo Fontanelli, Sergio Di Bona, Davide Guerri, and Libero Maesano. 2017. Automation and intelligent scheduling of distributed system functional testing. *International Journal on Software Tools for Technology Transfer* 19, 3 (2017), 281–308.
[61] Jhen-Jia Hu. 2014. The Verification and Validation of a Large-Scale System: Equipment TaaS as an Example. In *Computer, Consumer and Control (IS3C), 2014 International Symposium on*. IEEE, 13–18.
[62] Yazhou Hu, Bo Deng, Yu Yang, and Dongxia Wang. 2016. Elasticity Evaluation of IaaS Cloud Based on Mixed Workloads. In *Parallel and Distributed Computing (ISPDC), 2016 15th International Symposium on*. IEEE, 157–164.
[63] Jun-fei Huang. 2014. AppACTS: Mobile app automated compatibility testing service. In *Mobile Cloud Computing, Services, and Engineering (MobileCloud), 2014 2nd IEEE International Conference on*. IEEE, 85–90.
[64] Jun-fei Huang and Yun-zhan Gong. 2012. Remote mobile test system: a mobile phone cloud for application testing. In *Cloud Computing Technology and Science (CloudCom), 2012 IEEE 4th International Conference on*. IEEE, 1–4.
[65] Qiang Huang, Lin Ye, Xinran Liu, and Xiaojiang Du. 2013. Auditing cpu performance in public cloud. In *Services (SERVICES), 2013 IEEE Ninth World Congress on*. IEEE, 286–289.
[66] Vincent Shi-Ming Huang, Robert Huang, and Ming Chiang. 2013. A DDoS mitigation system with multi-stage detection and text-based turing testing in cloud computing. In *Advanced Information Networking and Applications Workshops (WAINA), 2013 27th International Conference on*. IEEE, 655–662.
[67] Alastair C Hume, Yahya Al-Hazmi, Bartosz Belter, Konrad Campowsky, Luis M Carril, Gino Carrozzo, Vegard Engen, David García-Pérez, Jordi Jofre Ponsatí, Roland Kűbert, et al. 2012. Bonfire: A multi-cloud test facility for internet of services experimentation. In *International Conference on Testbeds and Research Infrastructures*. Springer, 81–96.
[68] Hind Husni and Ahmad A Saifan. 2017. Cloud Testing: Steps, Tools, Challenges. *New Trends in Information Technology* (2017), 34.
[69] Gwan-Hwan Hwang, Chi Wu-Lee, Yuan-Hsin Tung, Chih-Ju Chuang, and Syz-Feng Wu. 2014. Implementing TaaS-based stress testing by MapReduce computing model. In *Proceedings of the IEEE International Conference on Software Engineering and Service Sciences, ICSESS*. 137–140.
[70] Kai Hwang, Xiaoying Bai, Yue Shi, Muyang Li, Wen-Guang Chen, and Yongwei Wu. 2016. Cloud Performance Modeling with Benchmark Evaluation of Elastic Scaling Strategies. *IEEE Transactions on Parallel and Distributed Systems* 27, 1 (2016), 130–143.
[71] Koray Inçki, Ismail Ari, and Hasan Sözer. 2012. A Survey of Software Testing in the Cloud. In *Sixth International Conference on Software Security and Reliability, SERE 2012, Gaithersburg, Maryland, USA, 20-22 June 2012 - Companion Volume*. IEEE, Gaithersburg, MD, USA, 18–23. https://doi.org/10.1109/SERE-C.2012.32
[72] Ganesh Neelakanta Iyer, Jayakhanna Pasimuthu, and Ramesh Loganathan. 2013. Pctf: An integrated, extensible cloud test framework for testing cloud platforms and applications. In *Quality Software (QSIC), 2013 13th International Conference on*. IEEE, 135–138.
[73] Pooyan Jamshidi, Aakash Ahmad, and Claus Pahl. 2013. Cloud migration research: a systematic review. *IEEE Transactions on Cloud Computing* 1, 2 (2013), 142–157.

[74] Vajjiram Janani and K. Krishnamoorthy. 2015. Evaluation of cloud based performance testing for online shopping websites. *Indian Journal of Science and Technology* 8, 35 (2015).

[75] Deepal Jayasinghe, Galen Swint, Simon Malkowski, Jack Li, Qingyang Wang, Junhee Park, and Calton Pu. 2012. Expertus: A generator approach to automate performance testing in IaaS clouds. In *Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on*. IEEE, 115–122.

[76] Changjiang Jia, Yan Cai, Yuen Tak Yu, and TH Tse. 2016. 5W+1H pattern: A perspective of systematic mapping studies and a case study on cloud software testing. *Journal of Systems and Software* 116 (2016), 206–219.

[77] Changjiang Jia, Chunbai Yang, and Wing Kwong Chan. 2015. Architecturing dynamic data race detection as a Cloud-based Service. In *Web Services (ICWS), 2015 IEEE International Conference on*. IEEE, 345–352.

[78] A. Vanitha Katherine and K. Alagarsamy. 2012. Software Testing in Cloud Platform: A Survey. *International Journal of Computer Applications* 46, 6 (May 2012), 21–25.

[79] Manveen Kaur. 2016. Testing in the cloud: New challenges. In *Computing, Communication and Automation (ICCCA), 2016 International Conference on*. IEEE, 742–746.

[80] Konstantinos Kavoussanakis, Alastair C. Hume, Josep Martrat, Carmelo Ragusa, Michael Gienger, Konrad Campowsky, Gregory van Seghbroeck, Constantino VÃązquez, Celia Velayos, Frederic Gittler, Philip Inglesant, Giuseppe Carella, Vegard Engen, Michal Giertych, Giada Landi, and David Margery. 2013. BonFIRE: The clouds and services testbed. In *Proceedings of the International Conference on Cloud Computing Technology and Science, CloudCom*, Vol. 2. 321–326.

[81] Mariam Kiran, Andreas Friesen, Anthony J.H. Simons, and Wolfgang K.R. Schwach. 2014. *Model-based testing in cloud brokerage scenarios*. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Vol. 8377 LNCS. 192–208.

[82] Mariam Kiran and Anthony JH Simons. 2014. Model-Based Testing for Composite Web Services in Cloud Brokerage Scenarios. In *European Conference on Service-Oriented and Cloud Computing*. Springer, 190–205.

[83] Barbara Kitchenham. 2004. Procedures for performing systematic reviews. *Keele, UK, Keele University* 33, 2004 (2004), 1–26.

[84] Chorng-Shiuh Koong, Chih-Hsiong Shih, Chang-Chung Wu, and Pao-Ann Hsiung. 2013. The architecture of parallelized cloud-based automatic testing system. In *Complex, Intelligent, and Software Intensive Systems (CISIS), 2013 Seventh International Conference on*. IEEE, 467–470.

[85] Nikolai Kosmatov, Nicky Williams, Bernard Botella, and Muriel Roger. 2013. Structural Unit Testing as a Service with PathCrawler-online.com. In *Service Oriented System Engineering (SOSE), 2013 IEEE 7th International Symposium on*. IEEE, 435–440.

[86] Jong Yih Kuo, Chien-Hung Liu, and Wei Ting Yu. 2015. The Study of Cloud-Based Testing Platform for Android. In *Proceedings - 2015 IEEE 3rd International Conference on Mobile Services, MS 2015*. 197–201.

[87] Etiene Lamas, Luis A.V. Dias, and Adilson Marques Da Cunha. 2012. Software architectural drivers for cloud testing. In *VALID 2012 - 4th International Conference on Advances in System Testing and Validation Lifecycle*. 114–120.

[88] Junwon Lee, Jaeik Cho, Jungtaek Seo, Taeshik Shon, and Dongho Won. 2013. A novel approach to analyzing for detecting malicious network activity using a cloud computing testbed. *Mobile Networks and Applications* 18, 1 (2013), 122–128.

[89] Raluca Lefticaru and Anthony JH Simons. 2014. X-Machine based testing for cloud services. In *European Conference on Service-Oriented and Cloud Computing*. Springer, 175–189.

[90] Chien-Hung Liu and Shu-Ling Chen. 2016. Evaluation of cloud testing strategies based on task decomposition and allocation for improving test efficiency. In *2016 International Conference on Applied System Innovation, IEEE ICASI 2016*.

[91] Chien-Hung Liu, Shu-Ling Chen, and Woei-Kae Chen. 2015. Improving resource utilization of a cloud-based testing platform for android applications. In *Mobile Services (MS), 2015 IEEE International Conference on*. IEEE, 202–208.

[92] Chien-Hung Liu, Shu-Ling Chen, and Woie-Kae Chen. 2017. Cost-benefit evaluation on parallel execution for improving test efficiency over cloud. In *Applied System Innovation (ICASI), 2017 International Conference on*. IEEE, 199–202.

[93] Chien-Hung Liu, Woie-Kae Chen, and Shu-Ling Chen. 2016. A Concurrent Approach for Improving the Efficiency of Android CTS Testing. In *Computer Symposium (ICS), 2016 International*. IEEE, 611–615.

[94] Huai Liu, Maria Spichkova, Heinz W Schmidt, Andreas Ulrich, Horst Sauer, and Jan Wieghardt. 2015. Efficient testing based on logical architecture. In *Proceedings of the ASWEC 2015 24th Australasian Software Engineering Conference*. ACM, 49–53.

[95] Qi Liu, Marcio A Silva, Michael R Hines, and Dilma Da Silva. 2012. Hardware-in-the-loop simulation for automated benchmarking of cloud infrastructures. In *Simulation Conference (WSC), Proceedings of the 2012 Winter*. IEEE, 1–12.

[96] Win-Tsung Lo, Xiao-Long Liu, Ruey-Kai Sheu, Shyan-Ming Yuan, and Chun-Yu Chang. 2015. An Architecture for Cloud Service Testing and Real Time Management. In *Computer Software and Applications Conference (COMPSAC), 2015 IEEE 39th Annual*, Vol. 3. IEEE, 598–603.

[97] Komal Mahajan and Deepak Dahiya. 2014. A cloud based deployment framework for load balancing policies. In *Contemporary Computing (IC3), 2014 Seventh International Conference on*. IEEE, 565–570.
[98] Riyadh Mahmood, Naeem Esfahani, Thabet Kacem, Nariman Mirzaei, Sam Malek, and Angelos Stavrou. 2012. A whitebox approach for automated security testing of Android applications on the cloud. In *Automation of Software Test (AST), 2012 7th International Workshop on*. IEEE, 22–28.
[99] Riyadh Mahmood, Nariman Mirzaei, and Sam Malek. 2014. Evodroid: Segmented evolutionary testing of android apps. In *Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering*. ACM, 599–609.
[100] A. Malini, N. Venkatesh, K. Sundarakantham, and S. Mercyshalinie. 2014. Mobile application testing on smart devices using MTAAS framework in cloud. In *International Conference on Computing and Communication Technologies, ICCCT 2014*.
[101] Joe Mambretti, Jim Chen, and Fei Yeh. 2015. Next generation clouds, the chameleon cloud testbed, and software defined networking (sdn). In *Cloud Computing Research and Innovation (ICCCRI), 2015 International Conference on*. IEEE, 73–79.
[102] Krishna Markande and Sridhar J Murthy. 2013. Leveraging Potential of Cloud for Software Performance Testing. In *Cloud Computing*. Springer, 293–322.
[103] Nicolas Michael, Nitin Ramannavar, Yixiao Shen, Sheetal Patil, and Jan-Lung Sung. 2017. Cloudperf: A performance test framework for distributed and dynamic multi-tenant environments. In *ICPE 2017 - Proceedings of the 2017 ACM/SPEC International Conference on Performance Engineering*. 189–200.
[104] Chandru Mirchandani. 2014. Cloud computing as a debug tool. In *Procedia Computer Science*, Vol. 36. 359–366.
[105] Eman M Mohamed, Sherif El-Etriby, and Hatem S Abdul-kader. 2012. Randomness testing of modern encryption techniques in cloud environment. In *Informatics and Systems (INFOS), 2012 8th International Conference on*. IEEE, CC–1.
[106] Bashir Mohammed and Mariam Kiran. 2015. Analysis of cloud test beds using opensource solutions. In *Future Internet of Things and Cloud (FiCloud), 2015 3rd International Conference on*. IEEE, 195–203.
[107] Shraddha Mungekar and Dhanashree Toradmalle. 2015. W TaaS: An architecture of website analysis in a cloud environment. In *Next Generation Computing Technologies (NGCT), 2015 1st International Conference on*. IEEE, 21–24.
[108] Subramanian Nachiyappan and Selwyn Justus. 2015. Cloud Testing Tools and its Challenges: A Comparative Study. *Procedia Computer Science* 50 (2015), 482 – 489. https://doi.org/10.1016/j.procs.2015.04.018
[109] Crescencio Rodrigues Neto and VinÃŋcius Cardoso Garcia. 2013. Cloud testing framework. In *Proceedings of the 17th International Conference on Evaluation and Assessment in Software Engineering*. ACM, 252–255.
[110] Tomáš Oberle and Csaba Szabó. 2015. An architectural prototype for testware as a service. In *Applied Machine Intelligence and Informatics (SAMI), 2015 IEEE 13th International Symposium on*. IEEE, 15–19.
[111] George Pallis. 2010. Cloud Computing: The New Frontier of Internet Computing. *IEEE Internet Computing* 14, 5 (Sept 2010), 70–73. https://doi.org/10.1109/MIC.2010.113
[112] Dessislava Petrova-Antonova, Sylvia Ilieva, and Denitsa Manova. 2016. Automated Web Service Composition Testing as a Service. In *International Conference on Model-Driven Engineering and Software Development*. Springer, 114–131.
[113] Dessislava Petrova-Antonova, Sylvia Ilieva, and Denitsa Manova. 2016. TASSA: A testing as a service framework for web service compositions. In *AMARETTO 2016 - Proceedings of the International Workshop on DomAin Specific Model-Based AppRoaches to vErificaTion and validaTiOn*. 33–42.
[114] C Mano Prathibhan, A Malini, N Venkatesh, and K Sundarakantham. 2014. An automated testing framework for testing Android mobile applications in the cloud. In *Advanced Communication Control and Computing Technologies (ICACCCT), 2014 International Conference on*. IEEE, 1216–1219.
[115] Priyanka, Inderveer Chana, and Ajay Rana. 2012. Empirical Evaluation of Cloud-based Testing Techniques: A Systematic Review. *SIGSOFT Softw. Eng. Notes* 37, Article 3 (May 2012), 9 pages. https://doi.org/10.1145/180921.2180938
[116] Mazedur Rahman, Zehua Chen, and Jerry Gao. 2015. A service framework for parallel test execution on a developer's local development workstation. In *Proceedings - 9th IEEE International Symposium on Service-Oriented System Engineering, IEEE SOSE 2015*, Vol. 30. 153–160.
[117] Kaliappa Ravindran, Arun Adiththan, and Michael Iannelli. 2014. SLA evaluation with on-the-fly measurements of distributed service implementation over clouds. In *Proceedings of the 6th International Workshop on Principles of Engineering Service-Oriented and Cloud Systems*. ACM, 1–10.
[118] Kaliappa Ravindran and Michael Iannelli. 2014. SLA evaluation in cloud-based data-centric distributed services. In *Computer Communication and Networks (ICCCN), 2014 23rd International Conference on*. IEEE, 1–8.
[119] Leah Riungu-Kalliosaari, Ossi Taipale, Kari Smolander, and Ita Richardson. 2016. Adoption and use of cloud-based testing in practice. *Software Quality Journal* 24, 2 (2016), 337–364.
[120] Leah Riungu-Kalliosaari, Ossi Taipale, Kari Smolander, and Ita Richardson. 2016. Adoption and use of cloud-based testing in practice. *Software Quality Journal* 24, 2 (2016), 337–364.

[121] Isabel K Villanes Rojas, Silvia Meireles, and Arilo Claudio Dias-Neto. 2016. Cloud-Based Mobile App Testing Framework: Architecture, Implementation and Execution. In *Proceedings of the 1st Brazilian Symposium on Systematic and Automated Software Testing*. ACM, 10.

[122] Georgia Sakellari and George Loukas. 2013. A survey of mathematical models, simulation approaches and testbeds used for research in cloud computing. *Simulation Modelling Practice and Theory* 39 (2013), 92–103.

[123] Clint Seales, Trung Do, Eugene Belyi, and Suman Kumar. 2015. PHINet: A plug-n-play content-centric testbed framework for health-internet of things. In *Mobile Services (MS), 2015 IEEE International Conference on*. IEEE, 368–375.

[124] Itai Segall and Rachel Tzoref-Brill. 2015. Feedback-driven combinatorial test design and execution. In *Proceedings of the 8th ACM International Systems and Storage Conference*. ACM, 12.

[125] Akash Shrivastva, Shubham Gupta, and Rinki Tiwari. 2014. Cloud based Testing Techniques (CTT). *International journal of computer applications* 104, 5 (2014).

[126] Tamanna Siddiqui and Riaz Ahmad. 2015. Cloud Testing–A Systematic Review. *International Research Journal of Engineering and Technology* (2015).

[127] Marcio Silva, Michael R Hines, Diego Gallo, Qi Liu, Kyung Dong Ryu, and Dilma Da Silva. 2013. Cloudbench: Experiment automation for cloud environments. In *Cloud Engineering (IC2E), 2013 IEEE International Conference on*. IEEE, 302–311.

[128] Stelios Sotiriadis, Andrus Lehmets, Euripides GM Petrakis, and Nik Bessis. 2017. Unit and Integration Testing of Modular Cloud Services. In *Advanced Information Networking and Applications (AINA), 2017 IEEE 31st International Conference on*. IEEE, 1116–1123.

[129] Oleksii Starov and Sergiy Vilkomir. 2013. Integrated TaaS platform for mobile development: Architecture solutions. In *2013 8th International Workshop on Automation of Software Test, AST 2013 - Proceedings*. 1–7.

[130] Dylan Steinmetz, Brian W Perrault, Ross Nordeen, Jacob Wilson, and Xinli Wang. 2012. Cloud computing performance benchmarking and virtual machine launch time. In *Proceedings of the 13th annual conference on Information technology education*. ACM, 89–90.

[131] Muhammad Dhiauddin Mohamed Suffian, Fairul Rizal Fahrurazi, and Suhaimi Ibrahim. 2014. The design and execution of performance testing strategy for cloud-based system. *International Journal of Software Engineering and Technology* 1, 2 (2014).

[132] Chuanqi Tao and Jerry Gao. 2016. Cloud-based mobile testing as a service. *International Journal of Software Engineering and Knowledge Engineering* 26, 1 (2016), 147–152.

[133] Chuanqi Tao and Jerry Gao. 2017. On building a cloud-based mobile testing infrastructure service system. *Journal of Systems and Software* 124 (2017), 39–55.

[134] Chuanqi Tao, Jerry Gao, and Bixin Li. 2016. Cloud-Based Infrastructure for Mobile Testing as a Service. In *Proceedings - 2015 3rd International Conference on Advanced Cloud and Big Data, CBD 2015*. 133–140.

[135] Dan Tao, Zhaowen Lin, and Cheng Lu. 2015. Cloud platform based automated security testing system for mobile internet. *Tsinghua Science and Technology* 20, 6 (2015), 537–544.

[136] Adrien Thiery, Thomas Cerqueus, Christina Thorpe, Gerson Sunyé, and John Murphy. 2014. A DSL for Deployment and Testing in the Cloud. In *Software Testing, Verification and Validation Workshops (ICSTW), 2014 IEEE Seventh International Conference on*. IEEE, 376–382.

[137] Hong-Linh Truong and Luca Berardinelli. 2017. Testing uncertainty of cyber-physical systems in IoT cloud infrastructures: combining model-driven engineering and elastic execution. In *Proceedings of the 1st ACM SIGSOFT International Workshop on Testing Embedded and Cyber-Physical Systems*. ACM, 5–8.

[138] Wei-Tek Tsai, Qingyang Li, Charles J Colbourn, and Xiaoying Bai. 2013. Adaptive fault detection for testing tenant applications in multi-tenancy SaaS systems. In *Cloud Engineering (IC2E), 2013 IEEE International Conference on*. IEEE, 183–192.

[139] Wei-Tek Tsai, Jie Luo, Guanqiu Qi, and Wenjun Wu. 2014. Concurrent test algebra execution with combinatorial testing. In *Service Oriented System Engineering (SOSE), 2014 IEEE 8th International Symposium on*. IEEE, 35–46.

[140] Wei-Tek Tsai and Guanqiu Qi. 2015. Integrated adaptive reasoning testing framework with automated fault detection. In *Service-Oriented System Engineering (SOSE), 2015 IEEE Symposium on*. IEEE, 169–178.

[141] Wei-Tek Tsai and Guanqiu Qi. 2016. Integrated fault detection and test algebra for combinatorial testing in TaaS (Testing-as-a-Service). *Simulation Modelling Practice and Theory* 68 (2016), 108–124.

[142] Wei-Tek Tsai and Guanqiu Qi. 2017. Adaptive Fault Detection In Multi-tenancy Saas Systems. In *Combinatorial Testing in Cloud Computing*. Springer, 25–36.

[143] Wei-Tek Tsai and Guanqiu Qi. 2017. Adaptive Reasoning Algorithm with Automated Test Cases Generation and Test Algebra in Saas System. In *Combinatorial Testing in Cloud Computing*. Springer, 83–99.

[144] Wei-Tek Tsai and Guanqiu Qi. 2017. Integrated Taas with Fault Detection and Test Algebra. In *Combinatorial Testing in Cloud Computing*. Springer, 115–128.

[145] Wei-Tek Tsai, Guanqiu Qi, and Kai Hu. 2015. Autonomous decentralized combinatorial testing. In *Autonomous Decentralized Systems (ISADS), 2015 IEEE Twelfth International Symposium on*. IEEE, 40–47.
[146] Wei Tek Tsai, Guanqiu Qi, Lian Yu, and Jerry Gao. 2014. Taas (testing-as-a-service) design for combinatorial testing. In *Software Security and Reliability (SERE), 2014 Eighth International Conference on*. IEEE, 127–136.
[147] Yuan-Hsin Tung, Chen-Chiu Lin, and Hwai-Ling Shan. 2014. Test as a service: A framework for web security TaaS service in cloud environment. In *Proceedings - IEEE 8th International Symposium on Service Oriented System Engineering, SOSE 2014*. 212–217.
[148] Yuan-Hsin Tung, Shian-Shyong Tseng, and Yung-Yu Kuo. 2015. A testing-based approach to SLA evaluation on cloud environment. In *Network Operations and Management Symposium (APNOMS), 2015 17th Asia-Pacific*. IEEE, 495–498.
[149] Andrew Turner, Andrew Fox, John Payne, and Hyong S Kim. 2013. C-mart: Benchmarking the cloud. *IEEE Transactions on Parallel and Distributed Systems* 24, 6 (2013), 1256–1266.
[150] Rob van der Meulen and Christy Pettey. 2017. Press Release: Gartner forecasts worldwide public cloud services revenue to reach $260 billion in 2017. https://www.gartner.com/newsroom/id/3815165 Accessed 9-April-2018.
[151] Martti Vasar, Satish Narayana Srirama, and Marlon Dumas. 2012. Framework for monitoring and testing web application scalability on the cloud. In *Proceedings of the WICSA/ECSA 2012 Companion Volume*. ACM, 53–60.
[152] Sergiy Vilkomir. 2012. Cloud testing: A state-of-the-art review. *Information & Security* 28, 2 (2012), 213–222.
[153] Isabel Karina Villanes, Erick Alexandre Bezerra Costa, and Arilo Claudio Dias-Neto. 2015. Automated mobile testing as a service (AM-TaaS). In *Services (SERVICES), 2015 IEEE World Congress on*. IEEE, 79–86.
[154] Junyi Wang, Xiaoying Bai, Linyi Li, Zhicheng Ji, and Haoran Ma. 2017. A Model-Based Framework for Cloud API Testing. In *Computer Software and Applications Conference (COMPSAC), 2017 IEEE 41st Annual*, Vol. 2. IEEE, 60–65.
[155] Claes Wohlin. 2014. Guidelines for snowballing in systematic literature studies and a replication in software engineering. In *Proceedings of the 18th international conference on evaluation and assessment in software engineering*. ACM, London, England, United Kingdom, 38.
[156] Wenjun Wu, Wei-Tek Tsai, Chao Jin, Guanqiu Qi, and Jie Luo. 2014. Test-algebra execution in a cloud environment. In *Service Oriented System Engineering (SOSE), 2014 IEEE 8th International Symposium on*. IEEE, 59–69.
[157] Miguel G Xavier, Kassiano J Matteussi, Gabriel R França, Wagner P Pereira, and Cesar AF De Rose. 2017. Mobile Application Testing on Clouds: Challenges, Opportunities and Architectural Elements. In *Parallel, Distributed and Network-based Processing (PDP), 2017 25th Euromicro International Conference on*. IEEE, 181–185.
[158] Xiaolin Xu, Hai Jin, Song Wu, Lixiang Tang, and Yihong Wang. 2014. URMG: Enhanced CBMG-based method for automatically testing web applications in the cloud. *TSINGHUA science and technology* 19, 1 (2014), 65–75.
[159] Minzhi Yan, Hailong Sun, Xu Wang, and Xudong Liu. 2012. Building a TaaS platform for web service load testing. In *Cluster Computing (CLUSTER), 2012 IEEE International Conference on*. IEEE, 576–579.
[160] Minzhi Yan, Hailong Sun, Xu Wang, and Xudong Liu. 2012. Ws-taas: A testing as a service platform for web service load testing. In *Parallel and Distributed Systems (ICPADS), 2012 IEEE 18th International Conference on*. IEEE, 456–463.
[161] Wei Yan and Nirwan Ansari. 2012. Anti-virus in-the-cloud service: are we ready for the security evolution? *Security and Communication Networks* 5, 6 (2012), 572–582.
[162] Lei Yin, Jin Zeng, Fangwang Liu, and Bo Li. 2013. CTPV: A cloud testing platform based on virtualization. In *Service Oriented System Engineering (SOSE), 2013 IEEE 7th International Symposium on*. IEEE, 425–428.
[163] Philipp Zech, Felderer Michael, and Ruth Breu. 2012. Towards a model-based security testing approach of cloud computing environments. In *Proceedings of the 2012 IEEE 6th International Conference on Software Security and Reliability Companion, SERE-C 2012*. 47–56.
[164] Samer Zein, Norsaremah Salleh, and John Grundy. 2016. A systematic mapping study of mobile application testing techniques. *Journal of Systems and Software* 117 (2016), 334–356.
[165] Linghao Zhang, Xiaoxing Ma, Jian Lu, Tao Xie, Nikolai Tillmann, and Peli De Halleux. 2012. Environmental modeling for automated cloud application testing. *IEEE software* 29, 2 (2012), 30–35.
[166] Shenbin Zhang and Bingfeng Pi. 2015. Mobile Functional Test on TaaS Environment. In *Service-Oriented System Engineering (SOSE), 2015 IEEE Symposium on*. IEEE, 315–320.
[167] Yuchao Zhang, Bin Hong, Ming Zhang, Bo Deng, and Wangqun Lin. 2013. eCAD: cloud anomalies detection from an evolutionary view. In *Cloud Computing and Big Data (CloudCom-Asia), 2013 International Conference on*. IEEE, 328–334.
[168] Yang Zheng, Lizhi Cai, Shidong Huang, and Zhihong Wang. 2014. VM scheduling strategies based on artificial intelligence in Cloud Testing. In *2014 IEEE/ACIS 15th International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, SNPD 2014 - Proceedings*.
[169] Junji Zhi, Sahil Suneja, and Eyal De Lara. 2014. The case for system testing with swift hierarchical vm fork. *TC* 1, S2 (2014), S3.
[170] Junzan Zhou, Shanping Li, Zhen Zhang, and Zhen Ye. 2013. Position paper: Cloud-based performance testing: Issues and challenges. In *Proceedings of the 2013 international workshop on Hot topics in cloud services*. ACM, 55–62.

A CLASSIFICATION OF THE PRIMARY STUDIES BY AREA

Table 5 reports the classification of each Primary Study by Area. In the header of the table: **Persp.** stands for Test Perspective, **Design** stands for Test Design, **Exec.** stands for Test Execution, **Objective** stands for Test Objective, **Eval.** stands for Test Evaluation, and **Domain** stands for Test Domain.

Table 5. Primary Study by Area and Category

Paper ID	Persp.	Design	Exec.	Objective	Eval.	Domain	TiC	ToC	ToiC
[100]			✓	✓		✓	✓		
[11]		✓	✓	✓		✓	✓		
[5]		✓		✓				✓	
[113]		✓	✓			✓	✓		
[169]			✓				✓		
[135]			✓	✓		✓	✓		
[70]		✓		✓	✓			✓	
[69]		✓	✓	✓		✓	✓		
[90]			✓				✓		
[132]	✓					✓	✓		
[38]			✓		✓	✓	✓		
[116]	✓		✓	✓	✓	✓	✓		
[21]		✓		✓				✓	
[134]			✓			✓	✓		
[158]		✓		✓	✓			✓	
[163]			✓	✓				✓	
[86]		✓	✓					✓	
[81]	✓	✓	✓	✓	✓	✓	✓		
[80]	✓	✓	✓	✓	✓			✓	
[87]	✓	✓	✓	✓	✓			✓	
[129]	✓	✓	✓	✓	✓	✓	✓		
[103]	✓	✓	✓	✓	✓	✓	✓		
[29]	✓	✓	✓	✓	✓		✓		
[51]	✓	✓	✓	✓	✓	✓	✓		
[108]	✓	✓	✓	✓			✓		
[147]			✓	✓		✓	✓		
[54]		✓					✓		
[45]			✓	✓				✓	
[17]			✓				✓		
[168]			✓						✓
[124]		✓	✓	✓				✓	

Continued on next page

Continued from previous page

Paper ID	Persp.	Design	Exec.	Objective	Eval.	Domain	TiC	ToC	ToiC
[58]		✓	✓	✓		✓	✓		
[166]		✓	✓	✓		✓	✓		
[10]		✓					✓		
[105]		✓		✓	✓			✓	
[130]			✓	✓	✓				✓
[79]	✓						✓		
[149]			✓					✓	
[160]		✓	✓	✓	✓	✓	✓		
[20]		✓	✓		✓	✓	✓		
[114]			✓	✓		✓	✓		
[84]		✓	✓				✓		
[43]			✓						✓
[25]	✓	✓	✓	✓	✓			✓	
[28]	✓		✓	✓		✓	✓		
[101]	✓		✓		✓				✓
[161]	✓		✓	✓		✓	✓		
[50]	✓		✓	✓	✓				✓
[107]			✓	✓	✓	✓	✓		
[136]	✓		✓	✓					✓
[119]	✓			✓		✓	✓		
[109]	✓	✓	✓	✓					✓
[88]	✓			✓					✓
[167]		✓	✓	✓				✓	
[153]			✓	✓	✓	✓	✓		
[66]	✓			✓		✓	✓		
[19]	✓	✓	✓	✓					✓
[133]			✓			✓	✓		
[1]			✓	✓	✓		✓		
[121]			✓			✓	✓		
[48]	✓	✓					✓		
[30]	✓		✓	✓			✓		
[31]		✓	✓	✓		✓	✓		
[97]			✓	✓				✓	
[117]			✓	✓				✓	
[162]			✓				✓		
[141]		✓	✓				✓		
[118]			✓	✓				✓	
[12]		✓	✓	✓	✓	✓	✓		
[93]		✓		✓	✓	✓	✓		
[145]		✓	✓				✓		
[55]	✓	✓	✓	✓	✓			✓	

Continued on next page

Continued from previous page

Paper ID	Persp.	Design	Exec.	Objective	Eval.	Domain	TiC	ToC	ToiC
[146]		✓	✓				✓		
[32]		✓		✓				✓	
[37]	✓		✓						✓
[72]			✓						✓
[91]			✓	✓		✓	✓		
[85]	✓		✓		✓		✓		
[170]	✓		✓		✓		✓		
[36]	✓	✓	✓				✓		
[110]			✓			✓	✓		
[159]			✓	✓	✓	✓	✓		
[61]		✓						✓	
[106]	✓							✓	
[2]	✓	✓	✓	✓	✓			✓	
[94]		✓			✓			✓	
[8]	✓	✓	✓	✓	✓		✓		
[52]	✓		✓	✓	✓	✓	✓		
[127]			✓	✓	✓			✓	
[138]		✓	✓					✓	
[34]			✓	✓	✓			✓	
[18]		✓	✓				✓		
[6]		✓		✓				✓	
[7]			✓	✓				✓	
[96]			✓	✓				✓	
[137]	✓	✓	✓	✓	✓			✓	
[13]	✓	✓		✓		✓	✓		
[156]	✓	✓	✓	✓		✓	✓		
[140]	✓	✓	✓	✓	✓	✓	✓		
[112]	✓	✓	✓	✓		✓	✓		
[75]	✓	✓	✓	✓		✓	✓		
[128]	✓	✓	✓	✓	✓			✓	
[95]	✓	✓	✓	✓	✓			✓	
[49]	✓	✓	✓	✓		✓	✓		
[39]	✓	✓	✓	✓		✓	✓		
[67]	✓	✓	✓	✓	✓			✓	
[131]	✓	✓	✓	✓	✓	✓	✓		
[104]	✓	✓	✓	✓	✓			✓	
[74]	✓	✓	✓	✓	✓			✓	
[65]	✓	✓	✓	✓	✓			✓	
[82]	✓	✓	✓	✓	✓	✓	✓		
[40]	✓	✓	✓	✓	✓	✓	✓		
[92]	✓	✓	✓	✓	✓	✓	✓		

Continued on next page

Continued from previous page

Paper ID	Persp.	Design	Exec.	Objective	Eval.	Domain	TiC	ToC	ToiC
[68]	✓	✓		✓	✓	✓	✓		
[99]	✓	✓	✓	✓		✓	✓		
[59]		✓	✓	✓		✓	✓		
[35]		✓	✓	✓	✓		✓		
[44]		✓	✓				✓		
[139]		✓	✓	✓		✓	✓		
[27]	✓	✓					✓		
[98]		✓	✓	✓	✓	✓	✓		
[15]		✓		✓			✓		
[148]			✓	✓		✓	✓		
[9]		✓	✓	✓		✓	✓		
[154]		✓	✓	✓	✓	✓	✓		
[62]		✓	✓	✓				✓	
[42]	✓		✓			✓	✓		
[41]			✓	✓					✓
[46]	✓							✓	
[33]		✓	✓					✓	
[123]			✓			✓	✓		
[63]			✓			✓	✓		
[89]		✓	✓					✓	
[77]	✓	✓	✓	✓	✓	✓	✓		
[151]			✓	✓		✓	✓		
[102]	✓		✓	✓		✓	✓		
[125]	✓								✓
[165]	✓	✓							✓
[60]	✓	✓	✓	✓	✓	✓	✓		
[64]			✓			✓	✓		
[142]		✓	✓			✓	✓		
[143]		✓				✓	✓		
[3]			✓					✓	
[56]			✓			✓	✓		
[144]		✓				✓	✓		
[57]			✓			✓	✓		
[157]			✓			✓	✓		

Concluded

Received XXX 201?; revised XXX 201?; accepted XXX 201?