Postprint of article in *Journal of Systems and Software* (2015), doi: 10.1016/j.jss.2015.01.058

5W+1H pattern: A perspective of systematic mapping studies and a case study on cloud software testing*

Changjiang Jia[a,b], Yan Cai[c], Yuen Tak Yu[a], T.H. Tse[d, †]

[a] Department of Computer Science, City University of Hong Kong, Tat Chee Avenue, Hong Kong
[b] Science and Technology on Information Systems Engineering Laboratory, National University of Defense Technology, Changsha, China
[c] State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing, China
[d] Department of Computer Science, The University of Hong Kong, Pokfulam, Hong Kong

ABSTRACT

A common type of study used by researchers to map out the landscape of a research topic is known as *mapping study*. Such a study typically begins with an exploratory search on the possible ideas of the research topic, which is often done in an unsystematic manner. Hence, the activity of formulating research questions in mapping studies is ill-defined, rendering it difficult for researchers who are new to the topic. There is a need to guide them kicking off a mapping study of an unfamiliar domain. This paper proposes a 5W+1H pattern to help investigators systematically examine a generic set of dimensions in a mapping study toward the formulation of research questions before identifying, reading, and analyzing sufficient articles of the topic. We have validated the feasibility of our proposal by conducting a case study of a mapping study on cloud software testing, that is, software testing for and on cloud computing platforms. The case study reveals that the 5W+1H pattern can lead investigators to define a set of systematic, generic, and complementary research questions, enabling them to kick off and expedite the mapping study process in a well-defined manner. We also share our experiences and lessons learned from our case study on the use of the 5W+1H pattern in mapping studies.

Keywords

5W+1H pattern; cloud software testing; systematic mapping study

* This research is supported in part by the General Research Fund of the Research Grants Council of Hong Kong (project numbers 125113, 716612 and 717811) and the National Natural Science Foundation of China (project number 91418206).

† Corresponding author.

Email addresses: cjjia.cs@gmail.com (C. Jia), ycai.mail@gmail.com (Y. Cai), csytyu@cityu.edu.hk (Y.T. Yu), thtse@cs.hku.hk (T.H. Tse).

Highlights

- Novices often find it difficult to kick off a mapping study or literature review
- A 5W+1H pattern is proposed to ease the difficulty of conducting literature surveys
- The 5W+1H pattern helps formulate initial research questions and structure reports
- A case study shows the applicability of the 5W+1H pattern and the lessons learned
- A three-year mapping study on cloud software testing research is presented

1. Introduction

Software testing consumes at least 30% of the whole software development budget [43]. *Cloud computing* [8] aims at providing a highly customizable and resourceful platform to deploy software [7]. The former is resource-hungry while the latter encompasses abundant computing resources. Does cloud computing solve or amplify the issues faced by software testing? For ease of presentation, we refer to the intersection area between *Cloud computing* and *Software Testing* as *CST interface* in this paper.

Systematic literature review (*SLR*) [23][26] is a dominant approach to conducting survey on a research topic. It aims at producing an "engineering" approach with a well-defined methodology so that different investigators can produce survey results effectively and reliably. In particular, *mapping study* (*MS*) [3][5][6][18][24][35][36] is "a more 'open' form of an SLR, intended to 'map out' the research that has been undertaken rather than to answer a detailed research question" [6].

At the core of a typical SLR, including an MS, is a well-defined literature review protocol, which is expected to be conformed to by investigators when they conduct SLRs and MSs [35]. For instance, according to the protocol, a key activity in the planning phase of a typical SLR project [18] is to formulate a set of *Research Questions* (*RQs*) before identifying, reading, and analyzing articles of the topic. For an MS project, the set of RQs to be answered is shaped by an exploratory and yet unstructured search of some selected articles to frame a preliminary impression of the topic. Based on a limited subset of the articles studied, investigators may formulate a set of exploratory RQs without knowing for sure the RQs' relevance to the topic under study. The set of RQs may evolve as more articles are reviewed by the investigators. However, when it comes to a topic with which the investigators are not truly familiar, formulating a comprehensive, coherent, probing, and reliable set of RQs on the topic before understanding a sufficient and unbiased set of relevant articles is indeed challenging, rendering it difficult to kick off an MS.

We observe that existing work on MS methodology [23][25][47][50] focuses on managing the processes of either literature search [23][47][50] or article categorization [25]. To the best of our knowledge, none of the existing work has provided a systematic way to explore the formulation of RQs in an MS. In this paper, we address this problem.

We propose an architectural style based on the 5W+1H (*Who*, *Why*, *What*, *Where*, *When* and *How*) model [16][34], which is widely used in the journalism domain, to define a high-dimensional design space in which the initial set of RQs can be systematically formulated. We validate the feasibility of our proposal via a two-phase case study of the MS of CST interface. Table 1 lists the collection of papers examined in the case study.

Table 1.
List of all the papers in the PVS examined in the case study

Authors and title	Venue	Year
List of papers in the PS (Phase 1)		
[S1] X. Bai, M. Li, B. Chen, W.-T. Tsai, J. Gao. Cloud testing tools.	*SOSE*	2011
[S2] T. Banzai, H. Koizumi, R. Kanbayashi, T. Imada, T. Hanawa, M. Sato. D-Cloud: Design of a software testing environment for reliable distributed systems using cloud computing technology.	*CCGrid*	2010
[S3] S. Bucur, V. Ureche, C. Zamfir, G. Candea. Parallel symbolic execution for automated real-world software testing.	*EuroSys*	2011
[S4] G. Candea, S. Bucur, C. Zamfir. Automated software testing as a service.	*SoCC*	2010
[S5] L. Ciortea, C. Zamfir, S. Bucur, V. Chipounov, G. Candea. Cloud9: A software testing service.	*ACM OSR*	2010
[S6] M.B. Cooray, J.H. Hamlyn-Haris, R.G. Merkel. Test reconfiguration for service oriented applications.	*UCC*	2011
[S7] C. Csallner, L. Fegaras, C. Li. Testing MapReduce-style programs.	*ESEC/FSE*	2011
[S8] X. Ding, H. Huang, Y. Ruan, A. Shaikh, B. Peterson, X. Zhang. Splitter: A proxy-based approach for post-migration testing of web applications.	*EuroSys*	2010
[S9] W. Fang, Y. Xiong. Cloud testing: The next generation test technology.	*ICEMI*	2011
[S10] J. Gao, P. Pattabhiraman, X. Bai, W.-T. Tsai. SaaS performance and scalability evaluation in clouds.	*SOSE*	2011
[S11] H.S. Gunawi, T. Do, P. Joshi, P. Alvaro, J.M. Hellerstein, A.C. Arpaci-Dusseau, R.H. Arpaci-Dusseau, K. Sen, D. Borthakur. FATE and DESTINI: A framework for cloud recovery testing.	*NSDI*	2011
[S12] T. Hanawa, H. Koizumi, T. Banzai, M. Sato, S. Miura, T. Ishii, H. Takamizawa. Customizing virtual machine with fault injector by integrating with SpecC device model for a software testing environment D-Cloud.	*PRDC*	2010
[S13] T. Hanawa, T. Banzai, H. Koizumi, R. Kanbayashi, T. Imada, M. Sato. Large-scale software testing environment using cloud computing technology for dependable parallel and distributed systems.	*ICST-W*	2010
[S14] S. Huang, Z. Li, Y. Liu, J. Zhu. Regression testing as a service.	*SRII*	2011
[S15] W. Jenkins, S. Vilkomir, P. Sharma, G. Pirocanac. Framework for testing cloud platforms and infrastructures.	*CSC*	2011
[S16] P. Joshi, H.S. Gunawi, K. Sen. PREFAIL: A programmable tool for multiple-failure injection.	*OOPSLA*	2011
[S17] T.M. King, A.S. Ganti, D. Froslie. Enabling automated integration testing of cloud application services in virtualized environments.	*CASCON*	2011
[S18] T.M. King, A.S. Ganti. Migrating autonomic self-testing to the cloud.	*ICST-W*	2010
[S19] L. Martignoni, R. Paleari, G.F. Roglia, D. Bruschi. Testing system virtual machines.	*ISSTA*	2010
[S20] A.F. Mohammad, H. Mcheick. Cloud services testing: An understanding.	*Elsevier PCS*	2011
[S21] M. Nagappan. Analysis of execution log files.	*ICST-V2*	2010
[S22] M. Oriol, F. Ullah. YETI on the cloud.	*ICST-W*	2010
[S23] T. Parveen, S. Tilley. When to migrate software testing to the cloud?	*ICST-W*	2010
[S24] S. Patil, M. Polte, K. Ren, W. Tantisiriroj, L. Xiao, J. López, G. Gibson, A. Fuchs, B. Rinaldi. YCSB++: Benchmarking and performance debugging advanced features in scalable table stores.	*SoCC*	2011
[S25] L.M. Riungu, O. Taipale, K. Smolander. Research issues for software testing in the cloud.	*CloudCom*	2010
[S26] L.M. Riungu, O. Taipale, K. Smolander. Software testing as an online service: Observations from practice.	*ICST-W*	2010
[S27] P. Robinson, C. Ragusa. Taxonomy and requirements rationalization for infrastructure in cloud-based software testing.	*CloudCom*	2011
[S28] N. Snellman, A. Ashraf, I. Porres. Towards automatic performance and scalability testing of rich internet applications in the cloud.	*SEAA*	2011
[S29] V. Srivastava, M.D. Bond, K.S. McKinley, V. Shmatikov. A security policy oracle: Detecting security holes using multiple API implementations.	*PLDI*	2011
[S30] M. Staats, C. Pasareanu. Parallel symbolic execution for structural test generation.	*ISSTA*	2010
[S31] W.-T. Tsai, P. Zhong, J. Balasooriya, Y. Chen, X. Bai, J. Elston. An approach for service composition and testing for cloud computing.	*ISADS*	2011
[S32] W.-T. Tsai, Y. Huang, Q. Shao. Testing the scalability of SaaS applications.	*SOCA*	2011
[S33] T. Vengattaraman, P. Dhavachelvan, R. Baskaran. A model of cloud based application environment for software testing.	*IJCSIS*	2010
[S34] J. Wu, C. Wang, Y. Liu, L. Zhang. Agaric: A hybrid cloud based testing platform.	*CSC*	2011
[S35] L. Yu, W.-T. Tsai, X. Chen, L. Liu, Y. Zhao, L. Tang, W. Zhao. Testing as a service over cloud.	*SOSE*	2010
[S36] L. Yu, X. Li, Z. Li. Testing tasks management in testing cloud environment.	*COMPSAC*	2011
[S37] P. Zech. Risk-based security testing in cloud computing environments.	*ICST*	2011
[S38] L. Zhang, Y. Chen, F. Tang, X. Ao. Design and implementation of cloud-based performance testing system for web services.	*CHINACOM*	2011
List of papers in the VS (Phase 2)		
[S39] N. Aleb, S. Kechid. Path coverage testing in the cloud.	*ICCI*	2012
[S40] S. Huang, X. Xu, Y. Xiao, W. Wang. Cloud based test coverage service.	*ICWS*	2012
[S41] K. Incki, I. Ari, H. Sozer. A survey of software testing in the cloud.	*SERE-C*	2012
[S42] D. Jayasinghe, G. Swint, S. Malkowski, J. Li, Q. Wang, J. Park, C. Pu. Expertus: A generator approach to automate performance testing in IaaS clouds.	*Cloud*	2012
[S43] R. Mahmood, N. Esfahani, T. Kacem, N. Mirzaei, S. Malek, A. Stavrou. A whitebox approach for automated security testing of Android applications on the cloud.	*AST*	2012
[S44] S. Malek, N. Esfahani, T. Kacem, R. Mahmood, N. Mirzaei, A. Stavrou. A framework for automated security testing of Android applications on the cloud.	*SERE-C*	2012
[S45] S. Priyanka, I. Chana, A. Rana. Empirical evaluation of cloud-based testing techniques: A systematic review.	*ACM SEN*	2012
[S46] L. Riungu-Kalliosaari, O. Taipale, K. Smolander. Testing in the cloud: Exploring the practice.	*IEEE SW*	2012
[S47] M. Vasar, S. N. Srirama, M. Dumas. Framework for monitoring and testing web application scalability on the cloud.	*WICSA/ECSA*	2012
[S48] S. Versteeg, C. Hine, J.-G. Schneider, J. Han. Emulation of cloud-scale environments for scalability testing.	*QSIC*	2012
[S49] M. Yan, H. Sun, X. Wang, X. Liu. Building a TaaS platform for web service load testing.	*CLUSTER*	2012
[S50] P. Zech, M. Felderer, R. Breu. Towards a model based security testing approach of cloud computing environments.	*SERE-C*	2012
[S51] L. Zhang, X. Ma, J. Lu, T. Xie, N. Tillmann, P. de Halleux. Environmental modeling for automated cloud application testing.	*IEEE SW*	2012

We report evidence that this 5W+1H model-based architectural style can guide investigators to make systematic progress in surveying the state of the art of a research topic with which they are originally unfamiliar.

Our work consists of three parts. In the first part, we proposed a 5W+1H pattern[1] to structure RQs and contrast the questions with the findings from the MS. This model-based pattern investigates a research topic from six different dimensions. Table 2 illustrates the pattern, which consists of six sections, one for each dimension (*Who*, *Why*, *What*, *Where*, *When*, and *How*). In each section, the pattern defines a placeholder for the *RQ*, a placeholder for the corresponding *conjecture*(*s*) relevant to the RQ to be "mapped out" (or verified) via the MS, a list of placeholders for the *major findings* that summarize the facts and statistics found, and a placeholder for *assessment*, which assesses the RQ and conjecture(s) based on the major findings.

In the second part, we applied the 5W+1H pattern to conduct a case study of an MS of CST-interface research. In a preliminary version [21] of this paper, we presented a brief summary of the results obtained in Phase 1 of the case study, which was conducted in June 2012. To improve the RQs and classification scheme used in Phase 1, we progressed to Phase 2 of the case study in June 2013. We then integrated the results obtained in Phase 1 (summarized in Table 2) with the new and consolidated results obtained in Phase 2 (summarized in Table 3) using the same 5W+1H pattern. We only present a summary of the MS results in this paper, while documenting the full details of the MS in our technical report [20].

In the third part, we reflected on the experiences from our two-phase case study on CST interface. We conclude from the complete case study that this 5W+1H pattern can lead us to define a set of generic and complementary RQs, which allow us to easily kick off and expedite the subsequent activities in an MS. This finding is encouraging.

To sum up, the main contribution of this paper, together with its preliminary version [21] that reported a brief summary of the results of Phase 1 of our case study conducted in June 2012, is threefold. (1) It presents, to the best of our knowledge, the first work that defines an architectural style (the 5W+1H pattern) to guide the structuring of RQs for mapping studies. (2) It provides real-life evidence of the feasibility of using the proposed pattern via a two-phase case study of an MS of CST-interface research. (3) It reports our first-hand experiences and reflective lessons learned from applying the 5W+1H pattern to systematically study a new research topic, which we believe would provide insights as well as practical guides to other researchers who plan to conduct an MS on an unfamiliar topic.

The rest of this paper is organized as follows. Section 2 revisits the 5W+1H model and proposes a 5W+1H pattern for applying the model to MS. Section 3 presents the process of our MS. Section 4 reports our experiences gained from the case study of applying the 5W+1H pattern to conduct an MS of CST-interface research. Section 5 discusses related work. Section 6 concludes the paper. Note that the case study is summarized and illustrated in this paper as follows: Figure 1 depicts the process of identifying the collection of papers in Phase 1, which is described in Section 3.2.1, while the process in Phase 2 is slightly simplified, as discussed in Section 3.2.2. Table 1 lists all

[1] Our work is inspired by design patterns in software design. A *pattern* is a template for a general solution to a common design problem. It is documented in seven sections, namely, intent, motivation, applicability, example, code listing, discussion, and assessment consequence.

Table 2.
Summary of research questions, conjectures, major findings, and assessments in Phase 1 based on the 5W+1H pattern

RQ1: *Who*? — Authors and countries
Conjecture C1: Plentiful recent papers have been published by diverse research groups and from different countries across the globe.
Major findings:
1. Our MS in Phase 1 identified a **primary set** (**PS**) of 38 papers published in 2010–2011 on CST interface out of an **initial set** (**IS**) of 2949 records of papers on the broader area of "cloud testing" research (which includes, for instance, hardware or network testing).
2. Twenty-two papers were affiliated with China or USA. In each of the other 10 countries, only one research group published papers in the PS.
3. Some of the top 10 countries that published most cloud testing papers in the IS reported in Scopus contributed no paper to CST interface.

Assessment: CST interface was not widely researched during the period surveyed. Moreover, the author/country distributions of publications on CST interface differed substantially from those on the broader area of cloud testing. We could not find adequate literature evidence to support conjecture C1.

RQ2: *Why*? — Objectives of research
Conjecture C2: Papers on multiple cloud service architectural layers (that is, Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS), and Software-as-a-Service (SaaS)) address the same kind of technical challenge with regard to the software testing topics.
Major findings:
1. Among the total 12 software testing topics identified from papers in the PS, four (1/3 of them) were studied in 14 papers (or 36.8% of all the papers in the PS) on more than one cloud service architectural layer.
2. With regard to the 12 software testing topics and three cloud service architectural layers, only the research on 16 out of all the 36 combinations was reported in the PS, while that of the other 20 combinations was not explored.
3. Six out of the 38 papers in the PS were survey or viewpoint papers on issues such as testing tool features and opinions of practitioners.

Assessment: The first finding demonstrated literature evidence that was in line with conjecture C2. Moreover, more than half of the combinations of software testing topics and cloud service architectural layers was not studied, indicating that much opportunity on CST interface remained to be explored. Furthermore, the notably high proportion of survey and viewpoint papers on different aspects of CST interface seemed to indicate that each aspect was rich and "unclear" enough to warrant a separate study.

RQ3: *What*? — Research ideas
Conjecture C3: Testing research ideas for addressing the challenges in different cloud service architectural layers are very different.
Major findings:
1. Papers on IaaS proposed fault-based testing techniques to expose faults in the virtual machine implementations.
2. Papers on PaaS developed (a) techniques for testing SaaS applications to deal with failure-simulations or issues of nondeterminism in PaaS, (b) a methodology for benchmarking, and (c) strategies to lower testing costs by exploiting the elastic property of PaaS.
3. Papers on SaaS spanned over nine software testing topics. A diverse set of ideas was studied, including performance metrics across multiple cloud service architectural layers, usage-specific and general models of Testing as a Service (TaaS), test parallelization, controllability and observability, test workloads, regression testing, the detection of vulnerability faults, and so on.

Assessment: The only idea common across different layers was fault-based testing. Hence, our data were consistent with conjecture C3. Research at a lower layer involved either randomized or fault-based testing, while many other systematic software testing approaches were not explored.

RQ4: *Where*? — Patterns of papers at different cloud service architectural layers and types of publication venues
Conjecture C4: Every cloud service architectural layer receives good research attention. Also, consistent with the norm for computer science research, the majority of recent papers are published as research articles in conference proceedings, and yet there is a good presence of journal papers.
Major findings:
1. Papers on SaaS contributed to 57.9% of the PS, followed by PaaS (15.8%) and IaaS (10.5%). The rest (15.8%) were survey or viewpoint papers.
2. 92.1% of the articles were published in conference/workshop proceedings. Only a very low percentage (7.9%) of the articles were journal papers.

Assessment: Papers tended to focus on testing challenges at the upper layer (SaaS). Both the ratio (5.5 : 1) between SaaS and IaaS papers and the ratio (11.7 : 1) between conference/workshop and journal papers were notably high, indicating that CST-interface research was not yet mature. Moreover, most articles were published in conference/workshop proceedings. On the issue of publication venues, our data were consistent with conjecture C4. On the other issue, namely, research attention to different cloud service architectural layers, our data did not support conjecture C4.

RQ5: *When*? — Article citation immediacy
Conjecture C5: Many papers are promptly cited by other papers.
Major findings:
1. Among all the 16 papers published in 2010, 43.8% of them received citations within the same year. The proportion of papers published in 2010 and cited in 2011 went up to 68.8%.
2. Among all the 38 papers in the PS, 39.5% of them were cited by papers within the PS.

Assessment: Many papers received prompt research attention in terms of citations, which was consistent with conjecture C5.

RQ6: *How*? — Article interrelevance
Conjecture C6: Many papers on various software testing topics and cloud service architectural layers are interreferenced to evolve CST-interface research.
Major findings:
1. In terms of citation relationships within the PS, the three topic-layer combinations, *robustness testing in the IaaS layer*, *testing parallelization in the SaaS layer*, and *integration testing in the SaaS layer*, were the most cited ones in CST-interface research.
2. Three topic-layer combinations, *fuzzing in the IaaS layer*, *migration testing in the SaaS layer*, and *log analysis in the SaaS layer*, had no citation relationship with others within the PS.
3. Citation relations between papers at the same cloud service architectural layers are much more common than those across layers. Only 39.1% of all citation relationships within the PS involved pairs of papers on different software testing topics at different cloud service architectural layers.
4. Topics on IaaS and PaaS, as well as their intersections with SaaS, were not extensively explored to evolve CST-interface research.

Assessment: The findings indicated that interreferencing to evolve CST-interface research was not yet a mainstream practice. We did not find adequate evidence to support conjecture C6.

Table 3.
Summary of research questions, conjectures, major findings, and assessments in Phase 2 based on the 5W+1H pattern

RQ1: *Who*? — Authors and countries
Conjecture C1: Plentiful recent papers have been published by diverse research groups and from different countries across the globe.
Major findings:
1. Our MS in Phase 2 identified a **validating set** (**VS**) of 13 papers published in 2012 on CST interface. Together with Phase 1, the combined paper set, denoted by **PVS** (= PS ∪ VS), included a total of 51 papers on CST interface published in the period 2010–2012.
2. Six papers in the VS were affiliated with China or USA. In each of the other seven countries, only one research group published one paper in the VS.
3. Some of the top 10 countries that published most cloud testing papers reported in Scopus contributed no paper to CST interface.

Assessment: Compared with Phase 1, we found some new research groups and new countries that conducted research on CST interface. However, at the same time, some countries that published papers in the PS did not publish any paper in the VS. In absolute terms, even if we considered the PVS as a whole, we still found that CST interface had not been widely researched so far. We could not find adequate literature evidence to support conjecture C1. Moreover, similar to Phase 1, the author/country distributions of the VS differed significantly from those on cloud testing.

RQ2: *Why*? — Objectives of research
Conjecture C2: Papers on multiple cloud service architectural layers (that is, Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS), and Software-as-a-Service (SaaS)) address the same kind of technical challenge with regard to the software testing topics.
Major findings:
1. Our MS classified the VS papers into seven testing topics, namely, six of the topics identified in Phase 1 together with one new topic. Two topics were studied at only one cloud service architectural layer in Phase 1 but at different new layers in Phase 2. Thus, treating the PVS as a whole, six software testing topics (involving 49% of the papers) were found in Phase 2 to be studied at more than one cloud service architectural layer.
2. With regard to the 13 software testing topics and three cloud service architectural layers, the VS papers studied two combinations that had not been explored by the PS papers. Treating the PVS as a whole, again 20 out of all the 39 combinations remained unexplored.
3. Three out of the 13 papers in the VS were survey or viewpoint papers on issues such as testing tool features and opinions of practitioners.

Assessment: The first finding shows that our data were in line with conjecture C2. In both Phase 1 and Phase 2, more than half of all combinations of software testing topics and cloud service architectural layers had not been explored. The proportions of survey and viewpoint papers were also high. Thus, the findings in Phase 2 were consistent with those found in Phase 1.

RQ3: *What*? — Research ideas
Conjecture C3: Testing research ideas for addressing the challenges in different cloud service architectural layers are very different.
Major findings:
1. The only paper in the VS on IaaS proposed to build a lightweight cloud component model to scale up the testing of the integration between the deployed software and a large number of cloud components.
2. Papers in the VS on PaaS proposed (a) to create multiple testing instances and distribute them to different cloud nodes to speed up testing execution, (b) a methodology to handle the component dependencies, and (c) strategies to generate workloads for testing.
3. Papers in the VS on SaaS spanned over four software testing topics. A diverse set of ideas had been studied, including generating statically-balanced testing workload distribution, speeding up regression testing with distributed cloud computation framework, identifying vulnerability points from outside of a cloud, and the development of cloud stubs to improve the structural coverage of unit testing.

Assessment: There was no common idea across different layers among the papers in the VS. Our findings were consistent with conjecture C3.

RQ4: *Where*? — Patterns of papers at different cloud service architectural layers and types of publication venues
Conjecture C4: Every cloud service architectural layer receives good research attention. Also, consistent with the norm for computer science research, the majority of recent papers are published as research articles in conference proceedings, and yet there is a good presence of journal papers.
Major findings:
1. Papers on IaaS, PaaS, and SaaS contributed to 9.8%, 21.6%, and 51.0% of the PVS, respectively. The rest (17.6%) were survey or viewpoint papers.
2. 76.9% of the articles in the VS were published in conference/workshop proceedings. Articles in journal papers only accounted for 23.1%.

Assessment: The lowest cloud service architectural layer (IaaS) still received the least attention from the research community. Most articles were published in conference/workshop proceedings. As in Phase 1, our data were consistent with conjecture C4 on the issue of publication venues, but did not support conjecture C4 on the issue regarding research attention to different cloud service architectural layers.

RQ5: *When*? — Article citation immediacy
Conjecture C5: Many papers are promptly cited by other papers.
Major findings:
1. The proportion of papers published in 2010 receiving citations further increased from 68.8% in 2011 to 93.8% in 2012.
2. 49.0% of the papers in the PVS were cited by papers within the PVS.

Assessment: Again, many papers received prompt research attention in terms of citations, which was consistent with conjecture C5.

RQ6: *How*? — Article interrelevance
Conjecture C6: Many papers on various software testing topics and cloud service architectural layers are interreferenced to evolve CST-interface research.
Major findings:
1. In terms of citation relationships within the PVS, the same three topic-layer combinations, *robustness testing in the IaaS layer*, *testing parallelization in the SaaS layer*, and *integration testing in the SaaS layer*, were the most cited ones in CST-interface research.
2. Two new topic-layer combinations, *integration testing in the IaaS layer* and *unit testing in the SaaS layer*, had no citation relationship with others within the PVS.
3. Only 41.1% of all citation relationships in the PVS involved pairs of papers on different software testing topics at different cloud service architectural layers.
4. Topics on IaaS and PaaS, as well as their intersections with SaaS, were not explored extensively to evolve CST-interface research.

Assessment: Although the proportion of citation relationships across different software testing topics and different cloud service architectural layers increased slightly from 39.1% in Phase 1 (2010–2011) to 41.1% in all three years (2010–2012), the two percentages were moderate only, not high. Thus, our data did not provide strong support to conjecture C6.

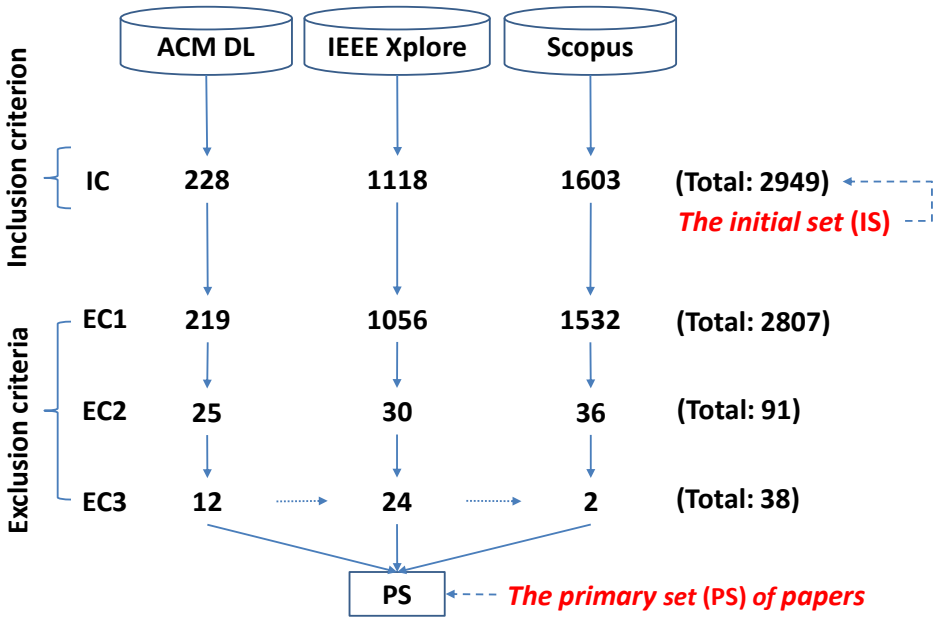

Figure 1. The process of identifying the primary set (PS) of papers in Phase 1.

the papers identified in the case study. Table 2 and Table 3 summarize the research questions, conjectures, major findings, and assessments in Phase 1 and Phase 2, respectively, based on the 5W+1H pattern. Table 4 classifies and enumerates the papers in terms of research topics and ideas. Finally, Figure 2 depicts the citation relationships among research topics to illustrate how the various topics are evolving. Full details of the case study and discussions on individual dimensions of the 5W+1H pattern, the papers reviewed, as well as the tables and figures presented in this paper, can be found in our technical report [20].

2. A 5W+1H pattern-based approach

Mapping study seeks to map out the state of research of a topic. In this section, we revisit the 5W+1H model and elaborate how we apply it to formulate a 5W+1H pattern to refine the free-form exploration in the planning phase of existing typical MS projects.

2.1 The 5W+1H model revisited

The term "5W+1H" is an abbreviation of six keywords: ***W**ho*, ***W**hy*, ***W**hat*, ***W**here*, ***W**hen*, and ***H**ow*. The 5W+1H model represents the majority needs of what people want to know about a news story. Kipling, an English writer, first mentioned the 5W+1H model in his book entitled *Just So Stories* [22] in 1902. Later, journalists widely applied this model to report news. From the perspective of journalists, to report a story, the readers should be supplied with essential information on six questions [16][34]:

(1) *Who* performed the actions in the story (or *who* experienced the results)? [Actor]

(2) *Why* did the actions occur? [Motivation]

Table 4.
Summary of main research ideas by software testing topics and cloud service architectural layers

Topic ID	Software testing topic* (*Cloud service architectural layer*: [references])	No. of papers# studying cloud layer			Proportion† of papers in the			Summary of main research ideas
		IaaS	*PaaS*	*SaaS*	PS	VS	PVS	
1	Fuzzing (*IaaS*: [S19]) (*PaaS*: [S43], [S44])	1:0	0:2		0.026	0.154	0.059	Produce randomized test cases to trigger residual faults in the implementations of virtual machines via protocol-based fuzzing. Speed up the execution of a test suite by simultaneously running several instances of the software under test on different cloud nodes.
2	Robustness testing (*IaaS*: [S2], [S12], [S13]) (*PaaS*: [S11], [S16])	3:0	2:0		0.132		0.098	Generate customizable fault-based execution traces that are reachable by the applications, each execution trace simulating a failure combination of the underlying platforms.
3	Concurrency testing (*PaaS*: [S7])		1:0		0.026		0.020	Generate test cases with respect to the nondeterministic behavior of an underlying infrastructure component reachable from the application.
4	Performance testing (*PaaS* [S24], [S42], [S47], [S49]) (*SaaS*: [S10], [S28], [S32])		1:3	3:0	0.105	0.231	0.137	Distribute workloads to reduce virtual machine rental costs. Design a model of system performance metrics that consider different architectural layers. Modularize the dependency complexities among distributed software components to automate the generation of testing configurations to speed up the testing under different levels of workloads.
5	Testing strategy (*PaaS*: [S27]) (*SaaS*: [S35], [S36])		1:0	2:0	0.079		0.059	Propose models to use or organize Testing-as-a-Service.
6	Context sensitivity (*PaaS*: [S15]) (*SaaS*: [S4])		1:0	1:0	0.053		0.039	Customize testing or Testing-as-a-Service with respect to specific usage scenarios.
7	Testing parallelization (*SaaS*: [S3], [S5], [S22], [S30], [S34], [S38], [S39])			6:1	0.158	0.077	0.137	Parallelize a symbolic or concrete execution so that different fragments can be scheduled to run on a set of virtual machines. Statically model the program into a set of variables with path constraints and distribute the test execution workload evenly before runtime.
8	Integration testing (*IaaS*: [S48]) (*SaaS*: [S17], [S18], [S31], [S33])	0:1		4:0	0.105	0.077	0.098	Control the service discovery mechanism or provide a virtualized testing platform of a service to improve the test controllability and observability of the service or service composition. Build lightweight cloud components to support integration testing of deployed software when the number of interacted cloud components is very large.
9	Regression testing (*SaaS*: [S6], [S14], [S40])			2:1	0.053	0.077	0.059	Identify the changes among different versions of the same SaaS application to select and fix test cases. Apply cloud services (such as BigTable and MapReduce) to scale up the data storage and parallelize the regression testing executions.
10	Security testing (*SaaS*: [S29], [S37], [S50])			2:1	0.053	0.077	0.059	Expose vulnerability faults of an application due to the use of alternate API implementations or defective code accessible by the application. Identify vulnerability points of a cloud computing environment by invoking the cloud public interface with malicious inputs.
11	Migration testing (*SaaS*: [S8])			1:0	0.026		0.020	Translate between test requests induced by the same application on different platforms over the same test case and triage failures to ease failure diagnosis.
12	Log analysis (*SaaS*: [S21])			1:0	0.026		0.020	Construct a generic log format to support different kinds of log analyses.
13	Unit testing (*SaaS*: [S51])			0:1		0.077	0.020	Build a simulated cloud environment that meets the cloud interface specifications to create cloud states for covering specific paths of the application units under test.
Subtotal		**4:1**	**6:5**	**22:4**	**0.842**	**0.770**	**0.824**	
Others: Survey ([S1], [S41], [S45]) and viewpoint papers ([S9], [S20], [S23], [S25], [S26], [S46])		6:3			0.158	0.231	0.176	Either apply a different classification scheme to survey existing cloud software testing work or report the industry's understanding of cloud software testing issues.
Total		**38:13**			**1.000**	**1.000**	**1.000**	

* Our study has examined a total of 51 papers in the PVS: 38 papers [S1]–[S38] in the PS (Phase 1), and 13 papers [S39]–[S51] in the VS (Phase 2).

Each cell contains a pair of values *x*:*y*, where *x* = the number of papers in the PS, and *y* = the number of papers in the VS.

† Each cell contains a ratio *z*, which is the number of papers in the PS, VS, or PVS on the same row divided by 38, 13, or 51, respectively.

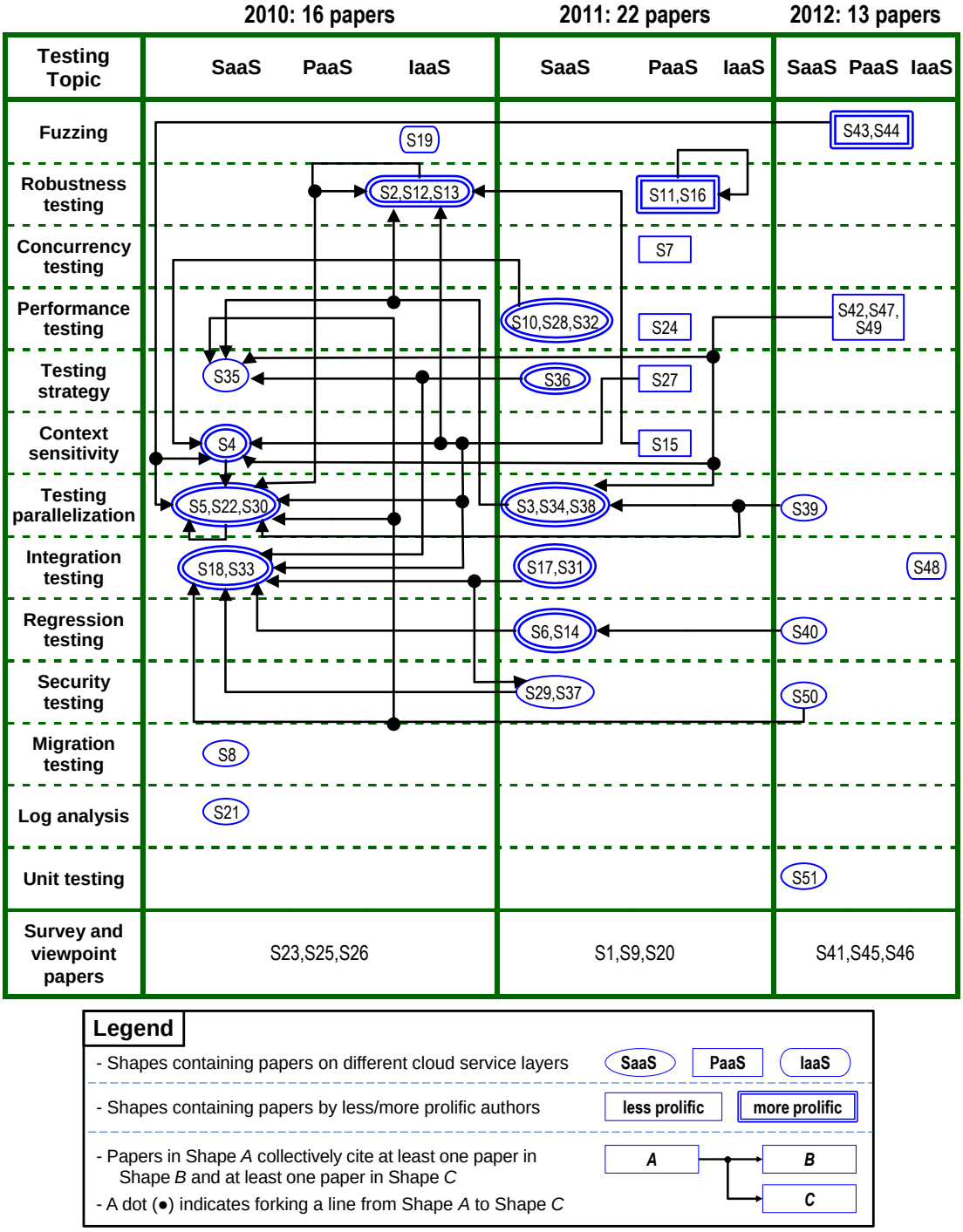

Figure 2. Citation relationships of the PVS papers across years, testing topics, and cloud service layers.

Note: The shapes and edges are interpreted with the following rules. A directed edge from a shape *x* to another shape *y* indicates that the papers referenced in *x* collectively cite the papers referenced in *y*. For more details, please refer to our full technical report [20].

(3) *What* were the actions and *what* happened as a result of the actions? [Content]

(4) *Where* did the actions take place? [Location]

(5) *When* did the actions occur? [Time]

(6) *How* did the actions connect to each other? [Causality]

On the other hand, an MS aims at synthesizing existing work on a research topic to obtain a comprehensive and objective understanding of the topic by mapping out the classifications of work done within the review scope of the MS. Thus, both journalists and MS investigators share the goal of seeking to understand and report certain activities (news events or research undertakings) comprehensively.

Typically, investigators conduct MSs by following a well-formed protocol [23][24] to collect existing literature and then analyze it to categorize findings based on a set of *pre-proposed* RQs. Thus, the understanding of a topic by an investigator when conducting an MS is largely determined by the pre-proposed RQs. Existing MS guidelines [5][6][18] [28][35][36] suggest starting with an exploratory formulation of RQs by reading some selected articles relevant to a topic. For researchers knowledgeable in the domain, proposing a relevant set of coherent and probing RQs thereafter may not be difficult; however, this task can be challenging to investigators new to the research topic.

We recall that a goal of the MS methodology is to provide a well-defined protocol for one to follow so that different investigators can more or less produce similar results (so that the process can be engineering-oriented and repeatable). Simply asking investigators to explore some articles and developing an exploratory set of RQs without a set of concrete guidelines may be too abstract and unsystematic.

The 5W+1H model provides six dimensions to completely report events of interest. We propose that it can benefit MS investigators by relieving their challenges in defining the initial set of RQs and providing them with guides to perspectives that are not necessarily seen from other MSs in similar topics (such as service-based testing versus software testing). In the next section, we will elaborate our view on how to apply the 5W+1H model to MS.

2.2 Applying the 5W+1H model to a mapping study

Generally speaking, the purpose of writing or referencing (instead of publishing) an MS is to quickly understand the research state or progress of a topic as well as to identify the gaps or new problems for further research investigations. Our case study is going to show that the six dimensions of the 5W+1H model have the potential to guide an MS by defining RQs that could easily be missed (due to omission or negligence) or dismissed (due to bias or premature judgment) as "uninteresting". We believe that it is unscientific to presume the lack of interests in certain RQs before soliciting objective grounds or empirical evidences to support the judgment. We also believe that it is unscientific to skip a whole dimension (say, due to the lack of interesting findings perceived by the investigators) in conducting the MS. This is because in either case, the result of the MS will be heavily and subjectively directed toward the selected dimensions and positive findings, resulting in biases in the publications.

Specifically, we adapt the 5W+1H model to the context of conducting an MS of a research topic and formulate a 5W+1H pattern that helps us define and focus on the initial RQs for studying the topic.

(1) *Who*: the researchers

(2) *Why*: the motivations and objectives of proposing the research problems

(3) *What*: the research ideas and issues

(4) *Where*: the locations of the research problems in terms of their positions in the topic context and venues of publication

(5) *When*: the publication date

(6) *How*: the interconnections among individual problems

Our MS case study was framed and guided by the above 5W+1H pattern. Specifically, we formulated a set of RQs for exploring CST-interface research from the six dimensions of the 5W+1H pattern. We extracted author information to report the researchers *Who* were active in research of the topic. We examined the motivations of the reviewed articles to understand *Why* the authors believed the research was necessary. We identified *What* software testing ideas, issues and topics were studied in the reviewed literature. Then, we located *Where* each paper was situated within a two-dimensional classification scheme that integrated the software testing topics with the three-layered cloud service architectural structure [2][31]. We also studied the distribution of papers in different venues *Where* they were published. We counted the papers published and cited within the review period to see *When* the publications of CST-interface research appeared and received attention from the research community, respectively. Finally, we analyzed the citation relationships to explore *How* CST-interface research had interacted and evolved across different software testing topics and cloud service architectural layers.

3. Feasibility case study of applying the 5W+1H pattern: A mapping study of CST interface

In this section, we present the process of our feasibility case study, including the formulation of RQs by applying the 5W+1H pattern, the identification of the paper sets for the two phases of the MS, and the quality assurance and data analysis tasks.

3.1 The 5W+1H pattern

To portray a contemporary picture of CST-interface research, we adopted the 5W+1H pattern developed in Section 2.2 and then instantiated it into the context of studying the topic of CST-interface research to pre-propose one RQ for each of the six dimensions as follows:

RQ1: *Who* (which researchers or groups) were doing research in CST interface?

RQ2: *Why* were the research studies needed? That is, what research objectives were stated in the articles?

RQ3: *What* kinds of software testing research ideas were presented in the articles?

RQ4: *Where* were the articles published? Did the articles appear in typical types of publication venues? On which cloud service architectural layers [2] were the articles focused?

RQ5: *When* did the articles start to show impact? Were the articles immediately cited by other articles?

RQ6: *How* were the articles interreferenced among various software testing topics and cloud service architectural layers?

Using the 5W+1H pattern, we further formulated a conjecture for each RQ according to the common perception of computer scientists. We would like to assess to what extent the collected papers present evidence to support or refute the following conjectures:

C1: Plentiful recent papers have been published by diverse research groups and from different countries across the globe.

C2: Papers on multiple cloud service architectural layers address the same kind of technical challenge with regard to the software testing topics.

C3: Testing research ideas for addressing the challenges in different cloud service architectural layers are very different.

C4: Every cloud service architectural layer receives good research attention. Also, consistent with the norm for computer science research, the majority of recent papers are published as research articles in conference proceedings, and yet there is a good presence of journal papers.

C5: Many papers are promptly cited by other papers.

C6: Many papers on various software testing topics and cloud service architectural layers are interreferenced to evolve CST-interface research.

We started the MS by using the process described in Section 3.2 to identify a set of papers, reviewed them, and mapped out the research according to each RQ and conjecture. We then enumerated the findings (in Table 2 and Table 3) using the 5W+1H pattern as a template, followed by an overall assessment statement with respect to each RQ and conjecture. Details of the findings of the MS can be found in the full technical report [20].

3.2 Paper identification process

In this section, we present the paper selection processes of our case study. We note that the main goal of the case study is to demonstrate the feasibility of adopting the 5W+1H pattern in an MS.

3.2.1 Phase 1: *Identification of the primary set* (PS) *of papers*

Phase 1 of the case study was performed in June 2012. Figure 1 depicts the paper search and selection process.

Databases: Our MS used three popular databases to identify the literature: ACM Digital Library (ACM DL) [1], IEEE Xplore Digital Library (IEEE Xplore) [17], and Scopus abstract and citation database (Scopus) [38].

Search keywords for Phase 1: We used two search keywords "`Cloud`" and "`Testing`" as the base and enumerated their popular variants to construct the following final compound keyword:

"`Cloud AND (Testing OR Analysis OR Test OR Analyzing OR Analyze)`"

Inclusion criterion (IC) and exclusion criteria (EC): Following the practice of Kitchenham et al. [23], we specified an **initial inclusion setting** (**IIS**) by matching the search keywords with the abstract of a paper. We focused on peer-reviewed publications to ensure that the papers in our collection at least reached acceptable and publishable quality. To study the progress on CST-interface research, we included papers published in the two whole years, 2010 and 2011, just before Phase 1 was conducted.

Initial inclusion setting (**IIS**): Search the (compound) keyword in the abstract of a paper that was published in a refereed journal or conference proceedings in 2010 or 2011 and is within the smallest domain that includes computer science.

Specifically, we set the **publication venue** as *journals*, *proceedings*, and *transactions* for ACM DL, *journals & magazines* and *conference publications* for IEEE Xplore, and *journals* and *conference proceedings* for Scopus. We set the **topic domain** as *Computing & Processing* (*Hardware/Software*) for IEEE Xplore and *Computer Science* for Scopus. We could not specify any topic domain for ACM DL as it provided no such option. We then refined the IIS in terms of actual search keywords to form the following **inclusion criterion (IC)**:

IC: Apply IIS using the compound keyword "`(Cloud) AND (Testing OR Analysis OR Test OR Analyzing OR Analyze)`"

By searching via the IC, we extracted an **initial set (IS)** of 2949 paper entries. We then applied the following three **exclusion criteria (EC1–EC3)** in stages to further refine the IS:

EC1: Exclude a paper with fewer than four pages.

EC2: Exclude a paper that mentions no issue on cloud computing or software testing in its abstract.

EC3: Exclude a paper that mentions no issue on cloud computing or software testing in either the introduction or conclusion of the paper. Remove duplications due to multiple records that refer to the same paper.

We followed the practice of Kitchenham et al. [23] and applied EC1, which reduced the size of the IS to 2807. EC2 eliminated a large number of papers on irrelevant topics such as storage, hardware configuration, and network, thereby further reducing the number of records to 91. For EC3, we examined the papers' introductions and conclusions in addition to the abstracts. To filter out duplicated entries, we first kept all

the records from ACM DL. Then, for each record in the two subsequent databases (namely, IEEE Xplore followed by Scopus), we removed a record if it had already been found in a previous database. Figure 1 shows the number of records obtained successively via the IC and then EC1–EC3 in Phase 1. We then applied snowballing [23] to examine the reference lists of the selected papers to see whether we might have missed any important articles. In the last step, we identified one additional article ([S5]).

We finally obtained 38 distinct papers, as listed in Table 1. We refer to this collection of papers as the **primary set** (**PS**). We noted that only 1.3% (= 38 ÷ 2949) of the paper entries in the IS were related to both cloud computing and software testing and, hence, included in the PS.

3.2.2 Phase 2: *Identification of the validating set* (VS) *of papers*

Considering the validation purpose of Phase 2 instead of conducting a new MS, we chose to search papers in Scopus because of its largest literature coverage among the three databases used in Phase 1. With the prior understanding of CST-interface research obtained in Phase 1, we refined the compound search keyword to

"`(Cloud OR IaaS OR PaaS OR SaaS OR TaaS) AND (Testing OR Test)`",

where *IaaS*, *PaaS*, and *Saa*S are abbreviations of the three cloud service architectural layers [2], and *TaaS* is an abbreviation of the term *Testing as a Service*, which refers to the deployment of testing in the form of a software service in the cloud. The four terms IaaS, PaaS, SaaS, and TaaS were added to refine the search keywords because they were so frequently found in cloud computing papers in Phase 1 that we would like to ensure no omission of papers collected in Phase 2 that used only these terms but not the word "cloud" in the abstract or paper title. On the other hand, we noted in Phase 1 that the word "analysis" and its variants were never found alone in papers on CST interface without the co-occurrence of the words "test" or "testing". Accordingly, variants of the term "analysis" were omitted from the search keyword in Phase 2.

Next, we set the search configuration in Scopus as follows: search for keywords in *Article Title*, select document type *ALL*, set the date range as *published 2012 to 2012*, and select *all* subject areas. By executing this search query in June 2013, we extracted 43 papers from Scopus. We refer to this collection of 43 papers, contributed by researchers from 22 countries, as **V-Scopus**. Considering the small size of V-Scopus, we directly read the abstract, introduction, and conclusion of each of the 43 papers, and finally obtained 13 papers published in 2012 that were relevant to CST-interface research, as listed in Table 1. We refer to this set of 13 papers as the **validating set** (**VS**).

In Phase 2, we also needed to combine the two paper sets (PS and VS) to analyze the characteristics of all the papers published in the entire three-year period (2010–2012) of the two phases. We refer to the combined collection of 51 (= 38 + 13) papers in the PS and VS as the **PVS**. The entire PVS is listed in Table 1.

Note that we adopted a slightly different search configuration and a simplified paper selection process in Phase 2 mainly because its purpose was not to exactly replicate Phase 1 but to validate whether the paper classification scheme, analysis process, as well as the findings derived in Phase 1 could be successfully applied to the new set of literature. Despite these differences, we manually read each paper in full detail in Phase 2

to ensure that its quality was acceptable and generally comparable to those collected in Phase 1. More details of the quality assurance measures adopted in Phase 1 and Phase 2 are presented in the next section.

3.3 Quality assurance

We browsed the official website of the publication venue of each paper in the PVS to ascertain that it had been peer-reviewed. Also, by manually reading the full text of each paper in the PVS (in addition to the preceding stages of examining its abstract, introduction, and conclusion), we verified that the paper indeed satisfied the inclusion criterion but was not eliminated by the exclusion criteria, with only a few exceptional papers that we nevertheless decided to include in the MS for the following reasons.

First, we found that although ACM SIGOPS Operating System Review did not specify a paper review process, its paper entitled "Cloud9: A software testing service" [S5] was a published version of [11], which had been peer-reviewed earlier in a workshop with no proceedings. Hence, we decided to include [S5] in the PS. All the other papers in the PS were also verified to have been peer-reviewed, met the inclusion criterion, and were not eliminated by the exclusion criteria. Two of the VS papers [S40][S44] had fewer than four pages, which strictly speaking should have been eliminated according to the first exclusion criterion (EC1). Nevertheless, upon rigorous and thorough scrutiny, we found that the work [S40] was published as a follow-up of [S14] by the same research group, while the other work [S44] was a different version of the paper [S43] from the same research group. Being convinced that analyzing these two closely-related papers could better reveal the relationships among the evolved research ideas, we finally decided to include them in the VS. Every other paper in the VS consists of at least four pages.

As a triangulation check, we also used Google Scholar [15] to reexamine the IC by replacing the search on the abstract setting by a search in the "with all of the words" field in the advanced search menu, setting "where my words occur" to "anywhere in the article", and using "return articles dated between 2010–2011". We then scanned through the first 2000 returned records (out of an estimated 18,800 hits) and found two more articles [9][14] that reported certain viewpoints on the topic. Nonetheless, in order to maintain our review protocol, we chose to review them separately in our full technical report [20] together with the other survey and viewpoint papers identified in the PVS. We did not use Google Scholar as the bibliographic database in our case study because it neither consistently indexed papers according to the publication years nor provided any option to search the abstracts. Searching only the keywords in the paper titles would miss many important references such as [S3][S4][S6][S8][S10][S14][S16][S19][S21][S24][S26][S29][S30][S32].

We further used our expert judgment to ensure that the PVS papers are of sufficient quality. The threats to validity of the case study can be found in our full technical report [20]. In particular, there is no golden standard as to what constitutes a search string to locate articles from various databases. Although the choice of search strings may affect the outcome of an MS, it would not affect the validation results of the feasibility of pursuing an MS by using the 5W+1H pattern in our case study.

3.4 Data analysis

In both phases of the case study, we critically and rigorously review each paper in the collections to extract and analyze the information on the six dimensions of the 5W+1H pattern. The results are detailed in our full technical report [20] while a concise summary is presented in Table 2, Table 3, Table 4, and Figure 2. In the next section, we share our experiences in applying the 5W+1H pattern to the MS in the case study.

4. Experiences in applying the 5W+1H pattern to a mapping study on CST interface

Here, we reflect on the direct mapping of the six dimensions of the 5W+1H pattern to summarize an MS, discuss the extension to hierarchical mapping and dimension integration, and suggest ways to enhance the cost-effectiveness of performing an MS.

4.1 Six dimensions to structure a mapping study

Our view is that researchers who are new to a particular research topic can greatly benefit by being informed of the representative researchers (*Who*) and important publication venues (*Where*) so that they can quickly assess the existing research progress (*When*) of the topic. It is not difficult to gather this basic information by collecting a set of published papers on the topic and extracting such basic data.

Moreover, researchers have to understand from the collected papers their research objectives (*Why*), research problems (*What*), proposed solutions, and relationships with other work in the same area (*How*). Simply browsing a paper from these three dimensions already helps the researcher appreciate the rationales of the research topic, relate different work, and trace their progress. However, this task is very time-consuming and it is nontrivial to obtain clear and consistent results. It is imperative to have several researchers working together to avoid individual bias and to resolve any inconsistencies among different judgments of the same paper.

The *Who* dimension in journalism seeks to identify the actors involved with news events. When studying a research topic, this dimension naturally suggests identifying the involved researchers and their research groups. We adopted this mapping in our case study of the MS of CST interface. However, we believe that the *Who* dimension may also guide us to explore more information than just the basics of individual researchers. This is because new research work is usually built from existing ones, whereas a news event may occur without necessarily bearing any relationship to other news events (unless it is a developing story that lasts for a period of time, such as a presidential election). Thus, we can trace the research relationships by viewing the references of a paper from the perspective of exploring relationships among different researchers. By doing so, given a set of papers, we can build a researcher citation relationship network to connect different researchers together. (Indeed, we have seen social or professional network sites [33] developed for similar purposes as well.) Different researchers in the network may work on various topics with different backgrounds. With the understanding of individual researchers and their relationships around a research topic, we may further study how these researchers identify research problems of the same topic from different perspectives. The understanding of citation relationships around a research topic may shed light on

how scholars are inspired about the same topic and consider how a new piece of work fits into the context of existing ones. We believe that inexperienced researchers can benefit a lot by learning the research methodologies of others.

The *Why* dimension in journalism is meant to reveal the motivations behind news events. In an MS, we have interpreted it to mean an investigation of the expected research objectives of each paper and the significance of the identified research challenges. The motivation to drive a new piece of research work often comes from noting or addressing the limitations of existing work. The same limitations of existing work may be addressed by different strategies, approaches, or solutions at different levels. Thus, we can group different pieces of existing work by their motivation and then compare one group with another in order to appreciate the best work in terms of the level of success in resolving the limitations. In our case study, we only explored the research objectives of the papers under the same identified research topics. We did not compare different individual pieces of work to explore their relationships and differences in terms of motivations. We believe that researchers would encounter this problem when their understanding of a new topic is insufficient. Conducting an MS of a topic, in fact, aims at improving the researchers' understanding of the topic. Thus, we recommend adopting an iterative process to drive the study of the *Why* dimension. Through improvements in the understanding of a topic after several iterations of an MS, researchers can obtain a clearer "motivation map" of the topic.

The *What* dimension in journalism usually aims at reporting the content of news events. In an MS, when reading a research paper, scholars are usually interested in finding answers to three questions: (1) *what* are the research problems that the paper identifies, (2) *what* is the innovative idea that the paper utilizes to address the identified problems, and (3) *what* are the possible limitations of the solutions proposed in the paper? The answers to these three questions, however, are not always explicitly presented in a paper. Researchers need to critically read a paper to search for the answers. This step often consumes the greatest effort. Moreover, to assure the validity of an MS, investigators need to collaborate with one another to resolve the possible differences in understanding the same paper.

The *Where* dimension in journalism mainly intends to report the geographical locations where news events happen. When studying a research topic, this dimension naturally suggests identifying the venues *where* the topic-related literature was published. Some people may argue that a good way of knowing the mainstream publication venues of a research topic is to ask the experienced researchers in the topic domain. From our own experience, most papers on an *emerging* topic, including even some of the influential papers, do not always appear in the mainstream publication venues commonly considered to be the best in the field. For example, the majority of papers reviewed in this study did not appear in top mainstream conferences or journals of the software engineering or services computing domain.

The *Where* dimension can also suggest that the investigator (as what we have done) generate a topic-specific structure to organize the research items (problems, solutions, limitations, and so on) identified by the *What* dimension. Such a structure provides new researchers with an overall understanding of the subtopics within the research topic under study.

The *When* dimension in journalism seeks to report the time when a news event occurs. In an MS, this dimension may suggest gathering data to demonstrate the popularity of a topic to the research community. For example, our case study adopted the number of papers published each year as an indicator of the degree of research attention from the community, but other metrics may also be used for this purpose. However, we argue that the *When* dimension alone may not reveal too much information about a research topic. The reason is that the time of publishing a research study just serves to tag the progress at a specific temporal phase. Hence, in our case study, we integrated the *When* and *How* dimensions to investigate *how* a paper evolved from previous work.

The *How* dimension in journalism describes the causality relationships among events in news reports. In an MS, this dimension can be taken to reveal the causality relationships among objects identified in the *What*, *Where*, and *When* dimensions. For example, our case study explored the citation relationships among the identified software testing topics, across different cloud service architectural layers, as well as among different years of publication. We believe that the *How* dimension can reveal more findings when viewing the causality relationships among different dimensions of the classification scheme. For example, we classified the collected papers using a two-dimensional scheme involving combinations of software testing topics and cloud service architectural layers, as well as along the temporal axis, as depicted in Figure 2. These dimensions and combinations can provide a useful context to enrich and deepen the understanding of relationships among papers.

4.2 Hierarchical mapping of the six dimensions

Section 4.1 above presents our understanding of directly mapping each dimension to structure RQs and report results of an MS. Such a direct mapping can generate a top-level research map of an investigated topic. However, due to insufficient topic-specific knowledge, the general research map can only provide researchers with an initial understanding rather than clearly identified research problems. From our experience in the case study, we think that the direct mapping can be further enhanced to provide more information for researchers to follow. For example, we identified 12 software testing topics in CST-interface research in Phase 1 of the case study. Software engineering researchers are well aware that there are plenty of testing topics, as software testing is a broad branch of the discipline. However, even with the limited number of identified testing topics, we still found it challenging to pinpoint the solid research problems that would directly motivate further study. It would be even more difficult for services computing researchers who are not familiar with software testing topics. Suppose that we are interested in two particular topics: concurrency testing and testing parallelization (Topics 3 and 7 in Table 4). We noted that the findings in this study could not provide sufficient understanding on the two topics. We analyzed the reason and found that the two topics were not cloud-specific but had been researched for many years on the testing of software executed on desktops and servers. Our case study only focused on exploring the two topics in the cloud domain and ignored the related progress in other domains. Thus, without an informed comparison with the research progress of the two topics from other domains, we could not precisely pinpoint the effect of the cloud computing domain on the two topics.

Therefore, we suggest hierarchically applying the 5W+1H pattern and iteratively investigating a topic. In the first iteration, we may identify some subproblems of the topic. In the next iteration, we may then set the selected subproblems as the objectives of the next-stage MS and repeat the same procedure.

A hierarchical mapping style may provide researchers with more knowledge when viewing a research topic from multiple domains. Take the integration testing topic (Topic 8 in Table 4) in our case study as an example. The first iteration of the direct mapping study identified papers that considered integration testing in the cloud context. We expect that by performing a second iteration of direct mapping to the integration testing topic, we would be able to obtain papers that treated integration testing from the perspective of other domains, such as web services. With two iterations, we could compare the perspectives of the web services domain with those of the cloud domain on the integration testing topic. This comparison may benefit our understanding of an existing topic from multiple application domains.

4.3 Completeness of report: Dimensions integration

The 5W+1H model is considered sufficient to completely report news, but we found it insufficient to completely and precisely report an MS of CST-interface research by simply mapping the 5W+1H pattern to organize the RQs and report results. This is because in order to adequately comprehend a piece of research work, including its values and limitations, the investigator has to simultaneously identify its motivation, problem, and solution, which are concerned with several dimensions rather than individual ones separately. Thus, it is essential to integrate some dimensions to review the research work.

In our case study, for instance, we tried to integrate the dimensions of *What*, *Why*, and *Where* together to review the research content of individual work. Initially, we simply studied the papers one by one to extract its research problem (*What*) and motivation (*Why*). In so doing, we soon found that we could not effectively relate each paper with other papers to comprehend the big picture. This limitation posed a challenge to us in synthesizing the common research motivations and issues. To address this challenge, we propose to supplement the 5W+1H pattern with topic-specific properties. In our case study, we adopted the generally accepted three-layer cloud service architectural model to refine the 5W+1H pattern. Since different classifications could lead to different styles in integrating multiple dimensions, investigators need to propose their own specific dimension integration mechanism to serve their research purposes.

4.4 Effort and cost effectiveness

Performing an MS requires great effort in each step of collecting and analyzing existing work on a topic. In our case study, with the input search string, we automatically retrieved almost up to 3000 records from the digital databases in the two phases. The first tedious task we did was to manually download each item-indexed paper and browse it to verify against the inclusion and exclusion criteria. The second time-consuming task was to critically and rigorously review each of the collected papers in the PVS to extract and analyze information for answering the research questions.

Based on our experiences, we have two suggestions to improve the cost-effectiveness in performing an MS. First, part of the paper identification process may be automated. For example, the first exclusion criterion (EC1) is to exclude a paper with fewer than 4

pages. The number of pages may be determined from the page numbers of the indexed papers. Thus, this step can largely be automatic.

The second suggestion is on the step of information extraction from papers and data analysis for answering the research questions. In our case study, four of the six dimensions under study are related to the metadata of the papers: the *Who* dimension is concerned with the author and affiliation data, the *Where* dimension is concerned with the publication venue data, whereas the *When* and *How* dimensions are concerned with the data of the publication dates and the papers' references. While we had collected these data by manually reading the papers in our case study, it is possible to extract these kinds of basic information automatically. On the other hand, some other information (such as the researchers' motivations in the *Why* dimension and the research problems in the *What* dimension) must rely on the investigators' manual effort in critically and carefully reviewing the papers. Take the two-dimension classification scheme in our case study as an example. To map the papers into the scheme correctly, all investigators individually classified each paper first and then gathered together to address any conflicts by face-to-face discussions. We found this procedure effective and relatively efficient in avoiding potential errors in producing the mapping results.

Staples and Niazi [41] suggested that defining narrow research questions is critical to controlling the effort of performing a SLR. However, defining narrow research questions itself requires nontrivial effort and good prior understanding of the topic. Once again it begs for the question of how to kick off the process by novices who are initially unfamiliar with the domain under study. Our proposal of applying the 5W+1H pattern is precisely one way out of such a dilemma.

5. Related work

In this section, we briefly review existing uses of the 5W+1H model in the software engineering and services computing domains as well as some recent studies on cloud software testing.

5.1 The use of 5W+1H model in software engineering and services computing

To enhance the rigor of the SLR process, Kitchenham et al. [27] propose to apply evidence-based concepts from medical research to the software engineering domain. Kitchenham and Charters [26] then define a general framework for an SLR on software engineering topics. Kitchenham and other colleagues [28] note that a manual search process might miss some relevant papers and, thus, propose to adopt an automated search process. She and her coauthors [6] also find that more guidelines are needed for an MS to be effective. Budgen et al. [6] and Petersen et al. [35], for example, have refined the MS process. However, the initial part of an MS process, which involves the selection of articles to start the exploratory search to formulate an initial impression on a topic, is still ill-defined [3][12]. Our work contributes to this part of the MS process.

The result of an automated search process of an SLR or MS is largely determined by the search string in various databases. There are other suggestions to paper searching, such as reference-based search strategies [40]. Webster and Watson [45] propose an incremental three-step paper searching approach called snowballing. This approach starts

with searching papers from some generally-accepted high quality publication venues (that is, journals or proceedings) as the first step. In the second step, it includes papers in the reference lists of those collected in the first step. In the third step, it searches the databases (such as ISI Web of Science) to find other papers that cite the papers collected in the previous two steps. The snowballing approach has been applied to an SLR of the software engineering domain [46] and an MS in the services computing domain [18].

Existing SLRs and MSs typically follow a similar reporting scheme in organizing their papers. However, SLR and MS in services computing and software engineering have only emerged for less than 10 years. It is not desirable to restrict oneself to the use of only one form of reporting scheme. Our paper proposes a 5W+1H pattern (see Table 2 and Table 3) as an alternative way to consolidate and report the results of an MS. To the best of our knowledge, there has been no existing work suggesting the application of the 5W+1H pattern to structure RQs and report results of an MS.

The 5W+1H model, though, has been used for other purposes in software engineering and services computing. Chung et al. [10] apply the model to the re-documentation of a given legacy system with UML visual models. They map the six dimensions of the model with topic-specific contexts as follows: the role of software developers (*Who*), the benefits of doing re-documentation (*Why*), the use of UML elements in various views by different roles (*What*), the different views, such as the use case view, of a legacy system (*Where*), different phases in the software development process (*When*), and the process of constructing the other dimension elements and building relationships (*How*).

Context-aware applications rely on the captured context information to maintain their performance. Existing context modeling techniques are specific to certain information (such as location), leading to a tight cohesion between contexts and applications. Jang et al. [19] use the 5W+1H model to build unified user-centric contextual information to be shared among several applications. The six dimensions would completely cover the complicated context. Yang et al. [49] use the model to build a conceptual modeling framework to analyze domain concepts and relationships from the six aspects. None of them has applied the model to conduct an SLR or MS in the field, nor did they formulate the structure in each dimension as what we have done in Table 2 and Table 3.

Literature reviews in the services computing domain are very popular. A number of excellent surveys were recently published, covering the quality of service assessment [29], service composition methods [32], execution simulation [40], and transaction control [42]. To the best of our knowledge, in spite of their excellent contributions, none of these surveys deals with the testing phase in the lifecycle of cloud applications and infrastructure, nor do they report their findings by following certain common patterns. The lack of a common way to summarize report findings shows the merit of the 5W+1H pattern that we have proposed in our work. For instance, our pattern requires a formulation of conjectures and then a validation of the conjectures instead of merely reporting what were observed but leaving the overall judgment on the reported observations to the readers.

The use of the 5W+1H pattern alleviates a problem of MSs conducted by inexperienced researchers. Some researchers may also find it difficult to formulate the inclusion and exclusion criteria after defining the search string. There are empirical studies reporting experiments to alleviate this difficulty. For instance, Skoglund and Runeson [39] propose to start an SLR (including an MS) with a set of “take-off” papers

and then follow the references of these papers to locate cited and citing articles and expand the set of papers iteratively. This independent approach further supports our comprehensive proposal to apply the 5W+1H pattern to MS. Incidentally, based on recent observations from four SLRs in an academic setting, Lavallée et al. [30] independently echo the proposal of an iterative approach for formulating research questions, searching and selecting the paper sets for review, as well as performing other tasks of the SLR to ensure the completeness and repeatability of the reviews conducted by novices. Our experiences, discussed in Section 4.2, likewise call for hierarchically applying the 5W+1H pattern to iteratively identify subproblems and objectives of the research topic under study and to deepen the investigator's understanding of the topic in multiple application domains.

5.2 Cloud software testing

In our case study, we have applied the 5W+1H pattern in surveying primary articles published between 2010 and 2012. It is natural that the research field has further evolved since then. In this section, we briefly review selected and representative work in cloud software testing published after the period under our case study. Like the majority of papers that have been summarized in the RQ4 of Table 2, the work reviewed here also focuses on addressing the testing challenge in the upper layer. These papers cite the work in the PS. As they were only published recently, we do not analyze their citations.

Yan et al. [48] propose WS-TaaS, which aims to address the service load testing problem (Topic 4) by building a SaaS application on top of PlanetLab. The research idea is to develop a new heuristic algorithm for test task scheduling while achieving geographical distribution diversity in terms of service invocation requests.

Portillo-Dominguez et al. [37] propose an automated load testing SaaS to test web applications interacting with an expert system. The idea is to periodically collect certain samples from the applications under test, and then feed these samples to the expert system instances to compute outputs. The work is interesting in that it no longer treats an application under test as a generic application. Rather, it specializes in one component as an expert system and uses this specialization to make the work different from a general-purpose load testing SaaS. To further ease the deployment activity, the same research group [44] proposes a domain-specific language to specify the deployment process and requirements as well as to generate installation scripts.

Batarseh et al. [4] propose CATCR, a test case reduction approach to test applications in the cloud by considering geographical context information. The idea is to study the local importance of each test case in its geographical site in the cloud and then select test cases for execution according to the local importance in each site. The paper does not explicitly state the cloud service architectural layer that CATCR is built on and applies to.

Gambi et al. [13] propose a testing methodology to automatically generate robustness test cases for detecting any violation of the elastic properties of software systems through a model-based approach. The idea is to construct an elastic model in the form of a labeled transition system, assess the model-based scaling behavior, and then refine a test case to breach such scaled behavior. The work only presents a conceptual overview of the idea.

We observe that efforts to organize specialized workshops or conference tracks on cloud software testing are gaining momentum. For instance, the first International

Workshop on Testing the Cloud (TTC 2013)[2] was held in July 2013. Moreover, in the series of International Symposia on Service-Oriented System Engineering (SOSE)[3], there are now sessions on cloud software testing. A full review of all the papers published in these venues would be beyond the scope of this paper.

We have also scanned Scopus with the keywords "cloud computing" and "software testing" that appeared in *Article Title*, *Abstract*, or *Keywords* in August 2014. However, there are still a very limited number of journal publications, indicating that there is much room for the field to progress further toward maturity. This observation is consistent with our finding on our PVS dataset presented in the case study.

6. Conclusion

It is challenging to perform mapping studies on a research topic that investigators are unfamiliar with. This paper has presented the first work that proposes to adopt a 5W+1H pattern to kick off the exploratory step in mapping studies. The 5W+1H (*Who*, *Why*, *What*, *Where*, *When*, and *How*) model has been widely used in journalism to report news. Based on the 5W+1H model, we have developed an architectural style and pattern through which the initial set of exploratory research questions can be systematically formulated in six coherent and complementary dimensions: researchers (*Who*), motivations and objectives (*Why*), research ideas, problems, and solutions (*What*), locations in the research map and publication venues (*Where*), publication dates and article citation immediacy (*When*), and relationships among individual studies (*How*).

To validate the feasibility of our proposal, we have conducted a mapping study on CST interface, that is, the intersection area between cloud computing and software testing. The two-phase case study investigated the state of CST interface published in a 3-year period. The process and results of the case study have provided evidence that our proposal indeed helps investigators kick off a mapping study on an unfamiliar topic.

Another major contribution of this work is the reporting of our first-hand experiences and reflective lessons learned from applying the 5W+1H pattern to systematically conduct a mapping study in an unfamiliar area. We have discussed the mapping of each dimension to the corresponding research contents, followed by extension to hierarchical and iterative applications of the pattern to deepen the understanding of a topic. To conclude, we postulate with substantiated evidence from our case study that the 5W+1H pattern can equip investigators with a generic framework to systematically study a new research topic at the initial exploratory phase.

References

[1] ACM Digital Library. http://dl.acm.org/.

[2] M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R.H. Katz, A. Konwinski, G. Lee, D.A. Patterson, A. Rabkin, I. Stoica, M. Zaharia, 2009. Above the clouds: A Berkeley view of cloud computing. Technical Report UCB/EECS-2009-28, Electrical Engineering and Computer Sciences, University of California at Berkeley, Berkeley, CA.

[2] http://issta2013.inf.usi.ch/workshop4/.

[3] http://sei.pku.edu.cn/conference/sose2013/.

[3] J. Bailey, D. Budgen, M. Turner, B.A. Kitchenham, O.P. Brereton, S. Linkman, 2007. Evidence relating to object-oriented software design: A survey. In: Proceedings of the 1st International Symposium on Empirical Software Engineering and Measurement (ESEM ’07). IEEE Computer Society, Los Alamitos, CA, pp. 482–484.

[4] F.A. Batarseh, A.J. Gonzalez, R. Knauf, 2013. Context-assisted test cases reduction for cloud validation. In: Modeling and Using Context, Lecture Notes in Computer Science, vol. 8175. Springer, Berlin, Germany, pp. 288–301.

[5] O.P. Brereton, B.A. Kitchenham, D. Budgen, M. Turner, M. Khalil, 2007. Lessons from applying the systematic literature review process within the software engineering domain. Journal of Systems and Software 80 (4), 571–583.

[6] D. Budgen, M. Turner, O.P. Brereton, B.A. Kitchenham, 2008. Using mapping studies in software engineering. In: Proceedings of the 20th Annual Workshop of the Psychology of Programming Interest Group (PPIG ’08). Lancaster, UK, pp. 195–204.

[7] W.K. Chan, L. Mei, Z. Zhang, 2009. Modeling and testing of cloud applications. In: Proceedings of the 2009 IEEE Asia-Pacific Services Computing Conference (APSCC ’09). IEEE Computer Society, Los Alamitos, CA, pp. 111–118.

[8] T.M. Chen, 2011. The cloud goes mainstream. Editor’s Note, IEEE Network 25 (4), 2–3.

[9] B. Chhabra, D. Verma, B. Taneja, 2010. Software engineering issues from the cloud application perspective. International Journal of Information Technology and Knowledge Management 2 (2), 669–673.

[10] S. Chung, D. Won, S.H. Baeg, S. Park, 2009. Service-oriented reverse reengineering: 5W1H model-driven re-documentation and candidate services identification. In: Proceedings of the 2009 IEEE International Conference on Service-Oriented Computing and Applications (SOCA ’09). IEEE Computer Society, Los Alamitos, CA, pp. 178–183.

[11] L. Ciortea, C. Zamfir, S. Bucur, V. Chipounov, G. Candea, 2010. Cloud9: A software testing service. ACM SIGOPS Operating Systems Review 43 (4), 5–10.

[12] P.A. da Mota Silveira Neto, I. do Carmo Machado, J.D. Mcgregor, E.S. de Almeida, S.R. de Lemos Meira, 2011. A systematic mapping study of software product lines testing. Information and Software Technology 53 (5), 407–423.

[13] A. Gambi, A. Filieri, S. Dustdar, 2013. Iterative test suites refinement for elastic computing systems. In: Proceedings of the 9th Joint Meeting on Foundations of Software Engineering (ESEC/FSE ’13). ACM, New York, NY, pp. 635–638.

[14] J. Gao, X. Bai, W.-T. Tsai, 2011. Cloud testing: Issues, challenges, needs and practice. Software Engineering: An International Journal 1 (1), 9–23.

[15] Google Scholar. http://scholar.google.com/.

[16] G. Hart, 1996. The five W’s: An old tool for the new task of task analysis. Technical Communication 43 (2), 139–145.

[17] IEEE Xplore Digital Library. http://ieeexplore.ieee.org/.

[18] P. Jamshidi, A. Ahmad, C. Pahl, 2013. Cloud migration research: A systematic review. IEEE Transactions on Cloud Computing 1 (2), 142–157.

[19] S. Jang, E.-J. Ko, W. Woo, 2005. Unified user-centric context: Who, where, when, what, how and why. In: Proceedings of the 1st International Workshop on Personalized Context Modeling and Management for UbiComp Applications (ubiPCMM ’05). Tokyo, Japan, pp. 26–34.

[20] C. Jia, Y. Cai, Y.T. Yu, T.H. Tse, 2015. 5W+1H pattern: A perspective of systematic mapping studies and a case study on cloud software testing. Technical Report, Department of Computer Science, City University of Hong Kong, Hong Kong.

[21] C. Jia, Y.T. Yu, 2013. Using the 5W+1H model in reporting systematic literature review: A case study on software testing for cloud computing. In: Proceedings of the 13th International Conference on Quality Software (QSIC ’13). IEEE Computer Society, Los Alamitos, CA, pp. 222–229.

[22] R. Kipling, 1902. Just So Stories. Macmillan, London, UK.

[23] B.A. Kitchenham, O.P. Brereton, D. Budgen, 2012. Mapping study completeness and reliability: A case study. In: Proceedings of the 16th International Conference on Evaluation and Assessment in Software Engineering (EASE '12). IET, London, UK, pp. 126–135.

[24] B.A. Kitchenham, O.P. Brereton, D. Budgen, M. Turner, J. Bailey, S. Linkman, 2009. Systematic literature reviews in software engineering: A systematic literature review. Information and Software Technology 51 (1), 7–15.

[25] B.A. Kitchenham, D. Budgen, O.P. Brereton, 2011. Using mapping studies as the basis for further research: A participant-observer case study. Information and Software Technology 53 (6), 638–651.

[26] B.A. Kitchenham, S. Charters, 2007. Guidelines for performing systematic literature reviews in software engineering. Technical Report EBSE 2007-001, Keele University and Durham University Joint Report.

[27] B.A. Kitchenham, T. Dybå, M. Jørgensen, 2004. Evidence-based software engineering. In: Proceedings of the 26th International Conference on Software Engineering (ICSE '04). IEEE Computer Society, Los Alamitos, CA, pp. 273–281.

[28] B.A. Kitchenham, R. Pretorius, D. Budgen, O.P. Brereton, M. Turner, M. Niazi, S. Linkman, 2010. Systematic literature reviews in software engineering: A tertiary study. Information and Software Technology 52 (8), 792–805.

[29] O. Kondratyeva, A. Cavalli, N. Kushik, N. Yevtushenko, 2013. Evaluating quality of web services: A short survey. In: Proceedings of the IEEE 20th International Conference on Web Services (ICWS '13). IEEE Computer Society, Los Alamitos, CA, pp. 587–594.

[30] M. Lavallée, P.-N. Robillard, R. Mirsalari, 2014. Performing systematic literature reviews with novices: An iterative approach. IEEE Transactions on Education 57 (3), 175–181.

[31] A. Lenk, M. Klems, J. Nimis, S. Tai, T. Sandholm, 2009. What's inside the cloud? An architectural map of the cloud landscape. In: Proceedings of the 2009 ICSE Workshop on Software Engineering Challenges of Cloud Computing (CLOUD '09). IEEE Computer Society, Los Alamitos, CA, pp. 23–31.

[32] R. Maigre, 2010. Survey of the tools for automating service composition. In: Proceedings of the 2010 IEEE International Conference on Web Services (ICWS '10). IEEE Computer Society, Los Alamitos, CA, pp. 628–629.

[33] Microsoft Academic Search. http://academic.research.microsoft.com/.

[34] Z. Pan, G.M. Kosicki, 1993. Framing analysis: An approach to news discourse. Political Communication 10 (1), 55–75.

[35] K. Petersen, R. Feldt, S. Mujtaba, M. Mattsson, 2008. Systematic mapping studies in software engineering. In: Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering (EASE '08). British Computer Society, Swindon, UK, pp. 68–77.

[36] M. Petticrew, H. Roberts, 2006. Systematic Reviews in the Social Sciences: A Practical Guide. Wiley-Blackwell, Malden, MA.

[37] A.O. Portillo-Dominguez, M. Wang, J. Murphy, D. Magoni, 2014. Automated WAIT for cloud-based application testing. In: Proceedings of the 2014 IEEE International Conference on Software Testing, Verification, and Validation Workshops (ICSTW '14). IEEE Computer Society, Los Alamitos, CA, pp. 370–375.

[38] Scopus. http://www.scopus.com/.

[39] M. Skoglund, P. Runeson, 2009. Reference-based search strategies in systematic reviews. In: Proceedings of the 13th International Conference on Evaluation and Assessment in Software Engineering (EASE '09). British Computer Society, Swindon, UK, pp. 31–40.

[40] M. Smit, E. Stroulia, 2013. Simulating service-oriented systems: A survey and the services-aware simulation framework. IEEE Transactions on Services Computing 6 (4), 443–456.

[41] M. Staples, M. Niazi, 2007. Experiences using systematic review guidelines. Journal of Systems and Software 80 (9), 1425–1437.

[42] C.-A. Sun, E. el Khoury, M. Aiello, 2011. Transaction management in service-oriented systems: Requirements and a proposal. IEEE Transactions on Services Computing 4 (2), 167–180.

[43] The Economic Impacts of Inadequate Infrastructure for Software Testing. Final Report, National Institute of Standards and Technology, Gaithersburg, MD. Available at http://www.nist.gov/director/planning/upload/report02-3.pdf.

[44] A. Thiery, T. Cerqueus, C. Thorpe, G. Sunyé, J. Murphy, 2014. A DSL for deployment and testing in the cloud. In: Proceedings of the IEEE 7th International Conference on Software Testing, Verification and Validation Workshops (ICSTW '14). IEEE Computer Society, Los Alamitos, CA, pp. 376–382.

[45] J. Webster, R.T. Watson, 2002. Analyzing the past to prepare for the future: Writing a literature review. MIS Quarterly 26 (2), xiii-xxiii.

[46] C. Wohlin, R. Prikladniki, 2013. Editorial: Systematic literature reviews in software engineering. Information and Software Technology 55 (6), 919–920.

[47] C. Wohlin, P. Runeson, P.A. da Mota Silveira Neto, E. Engström, I. do Carmo Machado, E.S. de Almeida, 2013. On the reliability of mapping studies in software engineering. Journal of Systems and Software 86 (10), 2594–2610.

[48] M. Yan, H. Sun, X. Liu, T. Deng, X. Wang, 2014. Delivering web service load testing as a service with a global cloud. Concurrency and Computation: Practice and Experience. doi: 10.1002/cpe.3246.

[49] L. Yang, Z. Hu, J. Long, T. Guo, 2011. 5W1H-based conceptual modeling framework for domain ontology and its application on STPO. In: Proceedings of the 7th International Conference on Semantics, Knowledge and Grids (SKG '11). IEEE Computer Society, Los Alamitos, CA, pp. 203–206.

[50] H. Zhang, M. Ali Babar, P. Tell, 2011. Identifying relevant studies in software engineering. Information and Software Technology 53 (6), 625–637.

Changjiang Jia is a PhD candidate in the Department of Computer Science at City University of Hong Kong. He received his BEng and MEng degrees from National University of Defense Technology, China. His research interests are program analysis, test sampling, concurrency bug detection and failure diagnosis. His research results have been reported in *IEEE Transactions on Parallel and Distributed Systems*, *IEEE Transactions on Services Computing*, *International Conference on Quality Software*, and *IEEE International Conference on Web Services*.

Yan Cai received the PhD degree from the Department of Computer Science at City University of Hong Kong. He is an associate research professor in the State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing, China. His current research interest is concurrency bug detection and reproduction in large-scale multithreaded and distributed systems. His research results have been reported in *IEEE Transactions on Software Engineering*, *IEEE Transactions on Parallel and Distributed Systems*, *IEEE Transactions on Services Computing*, *Software: Practice and Experience*, *International Conference on Software Engineering*, *ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, *IEEE International Conference on Web Services*, *International Symposium on Software Reliability Engineering*, and others.

Yuen Tak Yu received the PhD degree from the University of Melbourne, Australia. He is an associate professor in the Department of Computer Science at City University of Hong Kong. His research interests include software engineering, software testing, computers in education and e-commerce. His publications have appeared in scholarly journals, including *ACM Transactions on Software Engineering and Methodology*, *IEEE Transactions on Software Engineering*, *Journal of Systems and Software*, *Information and Software Technology*, *Software: Practice and Experience*, *Computers and Education*,

Information Research, and so on, as well as in leading international conferences such as *ICSE*, *ACM SIGSOFT/FSE*, *COMPSAC*, *ISSRE*, *ICCE*, and others. He was a past chairman of the IEEE Hong Kong Section Computer Society Chapter.

T.H. Tse received the PhD degree from the London School of Economics and was a visiting fellow at the University of Oxford. He is an honorary professor in computer science at The University of Hong Kong after retiring from the full professorship in July 2014. His current research interest is in program testing, debugging, and analysis. He is the steering committee co-chair of QRS and an editorial board member of the *Journal of Systems and Software*, *Software Testing, Verification and Reliability*, *Software: Practice and Experience*, and the *Journal of Universal Computer Science*. He also served on the search committee for the editor-in-chief of the *IEEE Transactions on Software Engineering* in 2013. He is a fellow of the British Computer Society, a fellow of the Institute for the Management of Information Systems, a fellow of the Institute of Mathematics and Its Applications, and a fellow of the Hong Kong Institution of Engineers. He was awarded an MBE by The Queen of the United Kingdom.