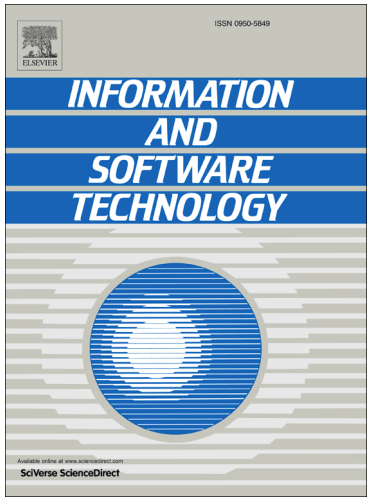
Accepted Manuscript



Knowledge Management Initiatives in Software Testing: a Mapping Study

Érica Ferreira de Souza, Ricardo de Almeida Falbo, Nandamudi L. Vijaykumar

PII: S0950-5849(14)00133-5
DOI: http://dx.doi.org/10.1016/j.infsof.2014.05.016
Reference: INFSOF 5473

To appear in: *Information and Software Technology*

Received Date: 7 December 2013
Revised Date: 21 May 2014
Accepted Date: 22 May 2014

Please cite this article as: r.F.d. Souza, R.d.A. Falbo, N.L. Vijaykumar, Knowledge Management Initiatives in Software Testing: a Mapping Study, *Information and Software Technology* (2014), doi: http://dx.doi.org/10.1016/j.infsof.2014.05.016

Knowledge Management Initiatives in Software Testing: a Mapping Study

Érica Ferreira de Souza[a], Ricardo de Almeida Falbo[b], Nandamudi L. Vijaykumar[a]

[a]*National Institute for Space Research (INPE), Av. dos Astronautas, 1758, 12227-010 São José dos Campos, SP, Brazil*

[b]*Federal University of Espírito Santo (UFES), Av. Fernando Ferrari, 514, 29075-910 Vitória, ES, Brazil*

erica.souza@lac.inpe.br, falbo@inf.ufes.br, vijay.nl@inpe.br

ABSTRACT

Context: Software testing is a knowledge intensive process, and, thus, Knowledge Management (KM) principles and techniques should be applied to manage software testing knowledge.

Objective: This study conducts a survey on existing research on KM initiatives in software testing, in order to identify the state of the art in the area as well as the future research. Aspects such as purposes, types of knowledge, technologies and research type are investigated.

Method: The mapping study was performed by searching seven electronic databases. We considered studies published until December 2013. The initial resulting set was comprised of 562 studies. From this set, a total of 13 studies were selected. For these 13, we performed snowballing and direct search to publications of researchers and research groups that accomplished these studies.

Results: From the mapping study, we identified 15 studies addressing KM initiatives in software testing that have been reviewed in order to extract relevant information on a set of research questions.

Conclusions: Although only a few studies were found that addressed KM initiatives in software testing, the mapping shows an increasing interest in the topic in the recent years. Reuse of test cases is the perspective that has received more attention. From the KM point of view, most of the studies discuss aspects related to providing automated support for managing testing knowledge by means of a KM system. Moreover, as a main conclusion, the results show that KM is pointed out as an important strategy for increasing test effectiveness, as well as for improving the selection and application of suited techniques, methods and test cases. On the other hand, inadequacy of existing KM systems appears as the most cited problem related to applying KM in software testing.

Keywords: Software Testing, Knowledge Management, Mapping Study

1. Introduction

Software development is an error prone task. To achieve quality software products, it is essential to perform Verification & Validation (V&V) activities throughout the software development process. Verification and Validation (V&V) activities intend to ensure, respectively, that a software product is being built in conformance with its specification, and that it satisfies its intended use and the user needs [12]. V&V activities can be static or dynamic. Static V&V activities are typically done by means of technical reviews and inspections, and they do not require code execution. Dynamic V&V activities, in turn, involve code execution, and are done by means of testing [10, 25]. Thus, Software Testing consists of dynamic V&V of the behavior of a program on a finite set of test cases, against the expected behavior [12].

Advances in technology and the emergence of increasingly complex and critical applications require using testing strategies, in order to achieve high quality and reliability software products [3]. Currently, software testing is considered as a process consisting of activities, techniques, resources and tools [25, 26]. During software testing, a significant amount of information is generated. In fact, software testing is a knowledge intensive process, and thus it is important to provide computerized support for tasks of acquiring, processing, analyzing and disseminating testing knowledge for reuse [3]. In this context, testing knowledge should be captured and represented in an affordable and manageable way, and therefore, could make use of principles of knowledge management.

According to O´Leary [29], Knowledge Management (KM) formally manages knowledge resources in order to facilitate access and reuse, typically by using advanced Information Technology (IT), playing a major supporting role in KM. IT-supported KM solutions are built around an organizational structure that integrates informal, semiformal, and formal knowledge to facilitate its access, sharing, and reuse [34]. The main goal of KM is to promote knowledge storage and sharing, as well as the emergence of new knowledge [30].

The Software Engineering community has recognized the need for managing knowledge and that it could learn much from the KM community [11]. Software development is a quickly changing, knowledge-intensive business, involving many people working in different phases and activities [24]. Knowledge in Software Engineering is diverse and organizations have problems in identifying its content, location, and use. An improved use of this knowledge is the basic motivation and driver for KM in Software Engineering. As a consequence, KM in Software Engineering has been subject of deeper analyses, such as those conducted by Rus and Lindvall [24], and by Bjørnson and Dingsøyr [11].

As a sub-area of Software Engineering, Software Testing also presents the same features. Knowledge can be applied to different testing tasks and purposes [15]. Knowledge on application domain and on testing techniques, as well as personal experiences, can be used to guide test case design, and to recognize failures. In an exploratory approach for software testing, where test cases are not defined in advance in an established test plan, but are dynamically designed, executed, and modified, tester knowledge is crucial. In this case, knowledge together with the observed actual behavior of the tested system can be used to create new, better tests during exploratory testing [15].

Given the great importance of knowledge for software testing, and the potential benefits of managing this knowledge, this paper aims to identify the state of the art on KM initiatives in Software Testing, by means of a mapping study. A mapping study provides a broad overview of a research area in order to determine whether there is research evidence on a particular topic.

Results of a mapping study help identifying gaps in order to suggest future research and provide a direction to appropriately position new research activities [18, 19, 32].

The mapping study presented in this paper is an extension of an initial study we performed to identify publications discussing principles of KM applied to software testing, which can be referred to in [35]. In this initial study, we searched studies published until January 2013, and investigated the following aspects: study distribution over the years, purposes of employing KM in software testing, types of knowledge items typically managed in the context of software testing, supporting technologies used, and benefits and problems reported on implementing KM initiatives in software testing. In the extension presented in this paper, we include studies published until December 2013, and we change the search string to incorporate other terms that are also related to KM, leading to new relevant studies. Moreover, new studies were selected by means of snowballing the primary study references, as well as by directly searching publications from researchers and research groups of the studies previously selected. Snowballing is a process that checks if the selected studies cite other relevant studies, retrieve those studies, and continue this process until no more relevant studies are found [13]. Finally, we also enlarge the scope of our investigation, by considering other aspects not previously addressed in the initial study, namely: source of the publication, research focus from the testing perspective, research focus from the KM perspective, and type of the research performed.

The remainder of this paper is organized as follows. Section 2 presents a brief overview of KM and software testing, as well as related research. Section 3 discusses the research method applied to perform the mapping study. Results are presented in Section 4. Section 5 discusses the results, their implications, and limitations. Finally, Section 6 concludes the paper and presents directions for future work.

2. Background

In this section, we briefly present the main concepts related to the topics addressed in this paper, namely: Knowledge Management and Software Testing. Moreover, we briefly discuss related research, i.e. secondary studies that are related to these topics.

2.1. Knowledge Management (KM)

Knowledge is one of the most valuable assets for most organizations. There are two main types of knowledge [27]: tacit and explicit. Nonaka and Takeuchi [27], use the tacit-explicit distinction to differentiate unarticulated and articulated stocks of knowledge [37]. Tacit knowledge is the subjective and experience-based knowledge that cannot be documented, and typically remains only in people's minds. This type of knowledge depends on personal experience and involves intangible factors such as beliefs, perspectives, values and intuition [27]. Tacit knowledge covers knowledge that is unarticulated and associated to the senses, movement skills, physical experiences, intuition, or implicit rules of thumb. Even if we try hard, this type of knowledge cannot be fully articulated [28]. Explicit knowledge, in turn, represents the objective and rational knowledge that can be documented, and, thus can be accessed by multiple individuals [27]. Explicit knowledge can be uttered and captured in drawings and writing, and can be easily used and shared. The concept of “knowledge conversion” explains how tacit and explicit knowledge interact along a continuum [28].

In the context of software testing, KM can be used to capture knowledge and experience generated during the testing process. However, usually, this knowledge has been stored on paper or in people's minds. When a problem arises, we look for experts across our working environment, relying on people we know, or we look for documents. Unfortunately, paper has limited accessibility and it is difficult to update [29]. On the other hand, in a large organization, it can be difficult to locate who knows a certain subject, and knowledge in people's minds (tacit knowledge) is lost when individuals leave the organization. Therefore, testing knowledge has to be systematically collected, stored in an organizational repository, and shared across the organization. In other words, KM is, indeed, necessary.

KM can be viewed as the development and leveraging of organizational knowledge to increase organization's value [40]. Organizational knowledge creation aims at making available and amplifying knowledge created by individuals as well as crystallizing and connecting it to an organization's knowledge system [28]. KM entails formally managing knowledge resources in order to facilitate access and reuse of knowledge, typically by using advanced information technology [29].

Information technology plays a major supporting role in KM [34]. A wide range of technologies have been used in the development of KM systems, such as databases, data mining, intranets and internet, intelligent information retrieval, intelligent agents, case-based reasoning, yellow pages, ontologies, visualization models, and groupware [21, 29].

2.2 Software Testing

Software Testing consists of dynamic verification and validation of the behavior of a program on a finite set of test cases, against the expected behavior. Software Testing should be supported by a well defined and controlled testing process [7]. Testing process consists of several activities, typically including [2, 6, 25, 26]: Test Planning, Test Case Design, Test Implementation, Test Execution and Test Result Analysis. First, testing should be planned. Key aspects of test planning include, among others, defining the test environment for the project, planning and scheduling testing activities, and planning for possible undesirable outcomes. Test planning is documented in a Test Plan. Test case design aims at designing the test cases to be run. Test cases are documented, and then implemented. During test execution, test cases are run, producing results, which are then analyzed to determine whether test cases have been passed or failed.

Testing activities are performed at different levels. Unit testing focuses on testing each program unit or component, isolated from the other components in the system. Integration testing takes place when such units are put together, aiming at ensuring that the interfaces among the components are defined and properly handled. Finally, system testing regards the behavior of the entire system. It is also considered appropriate for comparing the system behavior to the non-functional requirements established for it [2, 6, 25, 26, 33].

Software testing must reveal as many defects as possible. There are many testing techniques providing systematic guidelines for designing test cases, with the goal of making testing efforts more efficient and effective [33]. The principle underlying such techniques is to be as systematic as possible to identify a representative set of program behaviors [1]. Testing techniques can be classified, among others, as [7]: White-box Testing Techniques, which are based on information about how the software has been designed and coded; Black-box Testing Techniques, which generate test cases relying only on the input/output behavior, without the aid of the code that is under test; Defect-based Testing Techniques, which aim at revealing

categories of likely or predefined faults; Model-based Testing Techniques, which are based on models, such as Statecharts and Finite State Machines.

Advances in technology and the emergence of increasingly critical applications made the activities of software testing more complex and extremely necessary. During the testing process, a variety of knowledge is necessary, ranging from knowledge regarding the application domain to knowledge relative to testing techniques, making software testing a knowledge intensive process. In fact, tester's knowledge and experience is indispensable [4, 31], especially when experience-based testing techniques, such as exploratory testing, are applied [4, 15].

According to Beer and Ramler [4], although a wide range of methods and techniques offer guidance for systematic and rigorous testing, it is the tester's knowledge that ultimately plays an essential role in successfully applying these methods and techniques in the context of a project. The most notable activity that relies heavily on experience is test case design. For this activity, both domain knowledge and testing experience are a prerequisite to develop useful test cases. Similar to the development of new test cases, the selection of regression test cases depends mainly on the experience of the testers, as well as test automation, since automating test cases requires comprehensive experience with the automation frameworks and tools [4]. Testers are also capable of recognizing different types of software failures based on their experience [15].

Testers gain experience from diverse sources. According to [4], the most often mentioned sources involve testing previous versions of the software system, involvement in analyzing and fixing defects, taking part in development and maintenance, and working in the domain using similar software systems. However, much of the knowledge involved in software testing remains tacit [15]. Since, experience plays a key role in testing, management of past experience can help to effectively tailor methods and techniques to new projects [3]. So, it is necessary to provide computerized support for tasks of acquiring, processing, analyzing and disseminating testing knowledge for reuse [3].

2.3 Related Work

In this paper we present a secondary study, i.e. a study that is based on analyzing research papers (referred to as primary studies) [19]. Our study is a mapping study, i.e. a secondary study that aims to identify and classify all research related to a broad software engineering topic [19]. Mapping studies are intended to provide an overview of a topic area and identify whether there are sub-topics where more primary studies are needed [19].

Before accomplishing the secondary study presented in this paper, we performed a tertiary study looking for secondary studies investigating the state of the art in KM initiatives in software testing. In this tertiary study, we used the search string shown in Table 1, which was applied in three metadata fields (title, abstract and keywords).

Table 1
Search Terms of the Tertiary Study on KM in Software Testing

Areas	**Search Terms**
Software Testing	*"software testing", "software test"*
KM	*"knowledge management", "knowledge reuse"*
Review	*"systematic literature review", "systematic review", "systematic mapping", "mapping study", "systematic literature mapping"*
Search string: *("software testing" OR "software test") AND ("knowledge management") AND ("systematic literature review" OR "systematic review" OR "systematic mapping" OR "mapping study" OR "systematic literature mapping").*	

The search string was applied in the following electronic databases: *IEEE Xplore, ACM Digital Library, SpringerLink, Scopus, SicenceDirect, Compendex* and *ISI of Knowledge*. Nevertheless, no publication was returned. As we did not find any secondary study that addressed KM in Software Testing, we decided to investigate secondary studies that deal with KM and Software Testing separately.

For the tertiary study that looks for secondary studies in Software Testing, we used the search string shown in Table 2:

Table 2
Search Terms of the Tertiary Study on Software Testing

Areas	Search Terms
Software Testing	*"software testing", "software test"*
Review	*"systematic literature review", "systematic review", "systematic mapping", "mapping study", "systematic literature mapping"*
Search string: *("software testing" OR "software test") AND ("systematic literature review" OR "systematic review" OR "systematic mapping" OR "mapping study" OR "systematic literature mapping").*	

The same seven electronic databases were searched, returning 149 results. After eliminating duplications and applying the selection criteria, we reached 28 papers presenting secondary studies on software testing. 16 are SLRs, while 12 are mapping studies. Different areas related to software testing have been investigated by means of secondary studies. From the 28 secondary studies analyzed, we grouped them in the following categories: (i) Testing of specific software types (13 studies), highlighting Software Product Line Testing (4 studies) and Testing of Software Oriented Architecture and Web Services (3 studies); (ii) Testing Techniques (8 studies), highlighting Search-Based Software Testing (3 studies); (iii) Testing and Software Process (3 studies); (iv) Test Case Prioritization (3 studies); and (v) Others (3 studies), including Testing Automation, Alignment of Software Testing with Requirements, and Test effort reduction. It is worthwhile to point out that we classified 2 studies in more than one category, since they focus on a specific testing technique applied to a specific software type (unit testing approaches for web services, and mutation testing for aspect-oriented programming).

Regarding the tertiary study looking for secondary studies in Knowledge Management, we used the search string shown in Table 3.

Table 3
Search Terms of the Tertiary Study on KM

Areas	Search Terms
KM	*"knowledge management"*
Review	*"systematic literature review", "systematic review", "systematic mapping", "mapping study", "systematic literature mapping"*
Search string: *("knowledge management") AND ("systematic literature review" OR "systematic review" OR "systematic mapping" OR "mapping study" OR "systematic literature mapping").*	

The same seven electronic databases were searched, and 239 elements were returned. Duplications were eliminated and search criteria were applied. This resulted in 23 papers presenting secondary studies on KM. 22 are SLRs, while only one paper presents a mapping study. From the 23 secondary studies analyzed, we grouped them into the following categories: (i) General Aspects of KM (10 studies), including KM diffusion, KM in organizations, and relationships between KM and other related study areas, such as corporate culture, leadership, innovation, social media, competition and cooperation; (ii) Software Engineering/Software Development (7 studies); (iii) Health (3 studies), (iv) Ontologies and KM (2 studies); and (v) Emergency Management (1 study).

Based on the results from these two investigations by employing tertiary studies, one can say that there is a great diversity of secondary studies in Software Testing and in KM. However, as far as the investigations were concerned, no mapping study or SLR combining these two areas was found. It is also interesting to note that the 51 secondary studies that were analyzed have been published since 2008.

3. Research method

The research method for the mapping study presented in this paper is defined based on the guidelines given by Kitchenham and Charters [18] which involves three main phases: (i) **Planning**: refers to the pre-review activities, and aims at establishing a review protocol defining the research questions, inclusion and exclusion criteria, sources of studies, search string, and mapping procedures; (ii) **Conducting**: searches and selects the studies, in order to extract and synthesize data from them; (iii) **Reporting**: final phase that aims at writing up the results and circulating them to potentially interested parties. In this phase the findings of the systematic mapping study are used to answer the research questions.

In the conduction phase, as suggested by Kitchenham and Charters [18], in addition to the searches in the databases, snowballing from reference lists of selected studies was also applied, in order to identify additional relevant studies through the reference lists of the papers found using the search strings [13]. Furthermore, as suggested by Kitchenham et al. [19], direct searches for works developed by important researchers and research groups were performed. These researchers/research groups were identified from the previously selected papers (retrieved by the searches in the databases, as well as from snowballing). In this way, we tried to overcome the limitation of using a specific set of electronic databases.

In this section we discuss the main steps we performed for the mapping study. Section 3.1 presents the research questions. Section 3.2 discusses the study selection, while Section 3.3 discusses data extraction and synthesis. Section 3.4 presents the classification scheme we have adopted. Finally, limitations of this mapping are presented in Section 3.5.

3.1 Research questions

The goal of this mapping study is to depict a general view of the current status of the research regarding KM applied to software testing. Table 4 presents the research questions that this mapping study aims to answer, as well as the rationale for considering them.

Table 4
Research questions and their rationales

No.	Research question	Rationale
RQ1	When and where have the studies been published?	The topic of this mapping study seems to be broad and new. This research question aims at giving an understanding on whether there are specific publication sources for these studies, and when they have been published.
RQ2	From the software testing perspective, what aspects have been focused in the research?	Investigates which aspects of software testing have been the subject of KM initiatives. This information can help us to identify which aspects of software testing have gained more attention when applying KM in software testing.
RQ3	From the KM perspective, which topics have been focused?	Similarly to the previous, this research question investigates the KM sub-topics that have been more explored when applying KM to software testing.
RQ4	What types of research have been done?	As pointed out by Wieringa et al. [38] and Petersen et al. [32], different types of research have been presented in scientific papers (e.g., solution proposal, validation research, experience papers, opinion paper and evaluation research). This research question investigates which type of the research is reported in each selected study. This is an important question, since it can be used to evaluate the current maturity stage of the area.
RQ5	What are the problems reported by software organizations related to knowledge about software testing?	Provides an overview of the main problems reported by organizations related to the lack of knowledge about software testing. The goal of this question is to point out the problems that have motivated the research in this field.
RQ6	What are the purposes of employing KM in software testing?	This question complements the previous one, looking for the purposes declared in the studies for managing software testing knowledge. This is important to point out why such studies have been accomplished.
RQ7	What are the types of knowledge items typically managed in the context of software testing?	Investigates the types of knowledge items that have been managed in software testing. This information is important, since it gives a roadmap to define which types of knowledge have been considered more important in software testing.
RQ8	What are the technologies used to provide KM in software testing?	Highlights the main technologies currently used to provide KM in software testing. This is useful for researchers and practitioners that intend to accomplish new initiatives of KM in software testing, as well as to guide future research towards new technologies in order to fill the existing gaps.
RQ9	What are the main conclusions reported regarding applying KM in software testing?	Compiles the main conclusions reported on the studies regarding KM in software testing. This information is useful for evaluating the actual benefits of the current studies of KM in software testing, as well as to point out problems that remain and, thus, need to be subject of further research.

3.2 Study Selection

For retrieving the studies, we performed a selection process in which, among others, the following aspects were addressed: (i) terms and search string definition; (ii) source selection for searching; (iii) inclusion and exclusion criteria definition; and (iv) how to store data. These aspects are discussed in the following, as well as how we assessed the study selection.

a) Terms and Search String

The search string considers two areas - Software Testing and KM (Table 5) - and it was applied in three metadata fields: title, abstract and keywords. The search string went through syntactic adaptations according to particularities of each source.

Table 5
Search Terms of the Mapping Study on KM in Software Testing

Areas	**Search Terms**
Software Testing	*"software testing", "software test", "software project", "test", "testing"*
KM	*"knowledge management", "knowledge reuse", "knowledge sharing", knowledge transfer"*
Search string: *("software testing" OR "software test" OR ("software project" AND ("test" OR "testing")))* AND *("knowledge management" OR "knowledge reuse" OR "knowledge sharing" OR "knowledge transfer")*	

b) Sources

The search was performed in the following seven electronic databases that we considered the most relevant from the third studies we accomplished previously (see Subsection 2.3):

IEEE Xplore (*http://ieeexplore.ieee.org*)
ACM Digital Library (*http://dl.acm.org*)
SpringerLink (*http://www.springerlink.com*)
Scopus (*http://www.scopus.com*)
Science Direct (*http://www.sciencedirect.com*)
Compendex (*http://www.engineeringvillage2.org*)
ISI of Knowledge (*http://www.isiknowledge.com*)

c) Inclusion and Exclusion Criteria

The selection criteria are organized in one inclusion criterion (IC) and five exclusion criteria (EC). The inclusion criterion is: (IC1) The study discusses a KM initiative in software testing. The exclusion criteria are: (EC1) The study does not have an abstract; (EC2) The study is just published as an abstract; (EC3) The study is not written in English; (EC4) The study is an older version of other study already considered; and (EC5) The study is not a primary study, such as editorials, summaries of keynotes, workshops, and tutorials.

d) Data storage

The publications returned in the searching phase were cataloged and stored appropriately. A data extraction form was developed to gather all relevant data from the identified studies (e.g., id and bibliographic reference). This catalog helped us in the classification and analysis procedures.

e) Assessment

Before conducting the mapping, we tested its protocol. This test was conducted in order to verify its feasibility and adequacy, based on a pre-selected set of studies considered relevant to our investigation. In particular, in order to elaborate the search string, the set of search terms were devised in an iterative fashion, i.e. we started with an initial set of terms and iteratively improved this set until all relevant pre-selected studies were found. The review process was conducted by one of the authors and the other two carried out its validation. They analyzed all (100%) the studies, using two different samples.

3.3 Data extraction and synthesis

In the search process, we considered the studies published until December 2013. Searches were conducted for the last time in March 2014. As a result of searching the selected sources, a total of 562 publications were returned, out of which 77 from **IEEE Xplore**, 86 from **Compendex**, 95 from **Scopus**, 4 from **Science Direct**, 19 from **ACM Digital Library**, 270 from **SpringerLink**, and 11 from **Thomson Reuters Web of Knowledge**. Then we followed a selection process comprising five stages, as shown in Figure 1.

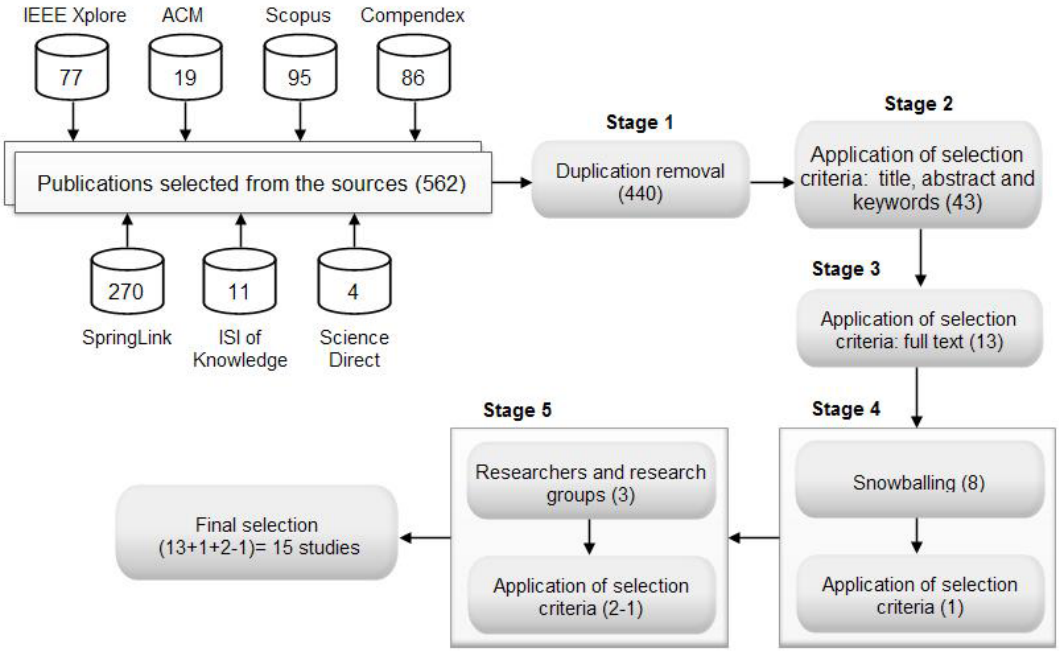


Fig. 1. Search and selection process

In the 1st stage, we eliminated duplications (publications that appear in more than one source), achieving **440** publications (reduction of approximately 22%). In the 2nd stage, we applied the selection criteria (inclusion and exclusion criteria) over title, abstract and keywords, resulting in **43** papers (reduction of approximately 90%). 9 papers were eliminated by EC1 (The study does not have an abstract); 2 by EC3 (The study is not written in English); 23 by EC5 (The study is not a primary study); and 363 for **not** satisfying IC1 (The study discusses a KM initiative in software testing). It is important to emphasize that, at this stage, we only excluded studies that were clearly unrelated to the subject, as suggested by Kitchenham and Charters [18]. In case of doubt, the paper was taken to the next stage. In the 3rd stage, the selection criteria were applied considering the full text, resulting in a set of **13** studies (reduction of approximately 70%). 2

papers were eliminated by EC4 (The study is an older version of another study already considered); and 28 papers were eliminated for **not** satisfying IC1 (The study discusses a KM initiative in software testing).

Over these 13 studies considered relevant, we moved to the 4th stage to perform snowballing, which resulted in 8 papers. After applying the selection criteria over title, abstract and keywords, 3 papers remained (reduction of 62.5% over the 8 papers selected by snowballing). For these 3 papers, the selection criteria were applied considering the full text, and only 1 paper remained (reduction of approximately 66.7% over the 3 previously selected papers). Criterion EC3 eliminated 4 papers (The study is not written in English); 1 paper was eliminated by EC5 (The study is not a primary study); and 2 papers for **not** satisfying IC1 (The study discusses a KM initiative in software testing).

Finally, from the 14 papers selected until then, in the 5th stage, we looked for publications authored by the researchers and research groups involved in these studies. To conduct that, we searched for their personal pages, their entries in The DBLP Computer Science Bibliography, as well as other publications authored by them in the digital libraries that we used as sources for this mapping. 3 papers were selected from the same research group. From these 3 papers, 1 of them was eliminated by EC4 (The study is an older version of other study already considered). Moreover, one of the papers of the resulting set of the 3rd stage was also eliminated by EC4, since one of the papers selected from the research group were newer and more complete.

As a final result, we got to 15 studies to be analyzed (12 from the sources, 1 from snowballing, and 2 from direct search to researchers and research groups). Table 6 summarizes the stages and their results. It shows the progressive reduction of the number of studies throughout the selection process. It is worth pointing out that the reduction percentage computed in the 4th and 5th stage refers to the papers introduced by snowballing and direct search to researchers and research groups, respectively.

Table 6
Results from the selection stages

Stage	Applied Criteria	Analyzed Content	Initial Number of Studies	Final N. of Studies	Reduction (%)
1st	Duplicate Removal	Title, abstract and keywords	562	440	21.7%
2nd	IC1, EC1, EC3 and EC5	Title, abstract and keywords	440	43	90.2%
3rd	IC1 and EC4	Full Text	43	13	69.8%
4th(a)	Snowballing, EC3 and EC5	Title, abstract and keywords	8 (added by snowballing)	3 (added by snowballing)	62.5%
4th(b)	Snowballing, IC1	Full Text	3 (added by snowballing)	1 (added by snowballing)	66.7%
5th	Research Group, EC4	Full Text	3 (added by research groups)	2 (added by research groups) - 1 (sources)	33.3%
Final Result			562 (sources) + 8 (snowballing) + 3 (research groups) = 573	13 (sources) + 1 (snowballing) + 2 (research groups) - 1 (sources) = 15	97.4%

Table 7 presents the bibliographic reference of the selected studies plus an identifier (#id) for each paper. Throughout the remainder of this paper, we use these identifiers to refer to the corresponding publication.

Table 7

Selected Studies

ID	Bibliographic Reference
#1	Y. Liu, J. Wu, X. Liu, G. Gu, Investigation of Knowledge Management Methods in Software Testing Process, in: International Conference on Information Technology and Computer Science, Kiev, Ukraine, 2009, 90-94.
#2	O. K. Wei, T. M. Ying, Knowledge Management Approach in Mobile Software System Testing, in: International Conference on Industrial Engineering and Engineering Management, Singapore, 2007, pp. 2120-2123.
#3	L. Xu-Xiang, Z. Wen-Ning, The PDCA-based software testing improvement framework, in: International Conference on Apperceiving Computing and Intelligence Analysis (ICACIA), Chengdu, China, 2010, pp.490-494.
#4	R. Abdullah, Z. D. Eri, A. M. Talib, A Model of Knowledge Management System in Managing Knowledge of Software Testing Environment, in: 5th Malaysian Conference in Software Engineering (MySEC), Johor Bahru, Malaysia, 2011, pp. 229-233.
#5	X. Li, W. Zhang, Ontology-based Testing Platform for Reusing, in: International Conference on Internet Computing for Science and Engineering, Henan, China, 2012, pp.86-89.
#6	A. Desai, S. Shah, Knowledge Management and Software Testing. In: International Conference and Workshop on Emerging Trends in Technology (ICWET), Mumbai, India, 2011, pp. 767-770.
#7	J. Andrade, J. Ares, M. Martínez, J. Pazos, S. Rodríguez, J. Romera, S. Suárez, An architectural model for software testing lesson learned systems, Information and Software Technology, 55 (2013), 18-34.
#8	K. Nogeste, D.H.T. Walker, Using knowledge management to revise software-testing processes. Journal of Workplace Learning, 18 (2006), Issue 1, 6-27.
#9	W. Janjic, C. Atkinson, Utilizing software reuse experience for automated test recommendation, in: Proceedings of the 8th International Workshop on Automation of Software Test (AST), San Francisco, USA, 2013, pp. 100-106.
#10	C. Kerkhof, J. Ende, I. Bogenrieder, Knowledge Management in the Professional Organization: A Model with Application to CMG Software Testing, Knowledge and Process Management, 10 (2003), Issue 2, 77-84.
#11	S. Vegas, V.R. Basili, A Characterization Schema for Software Testing Techniques, Empirical Software Engineering, 10 (2005), Issue 4, 437-466.
#12	S. Vegas, N. Juristo, V.R. Basili, Packaging experiences for improving testing technique selection, The Journal of Systems and Software, 79 (2006), Issue 11, 1606-1618.
#13	D. Larsson, H. Bertilsson, R. Feldt, Challenges and Solutions in Test Staff Relocations within a Software Consultancy Company. First International Conference on Software Testing, Verification, and Validation, Lillehammer, Norway, 2008, pp. 423 - 431.
#14	R. Abdullah, Towards Developing Software Testing as a Service (Staas) Model in Cloud Computing: a Case of Collaborative Knowledge Management System, In: Proceedings of the WSEAS International Conference on Software Engineering, Parallel and Distributed Systems, Cambridge, UK, 2012, pp. 22 - 24.
#15	I. Douglas, Testing Object Management (TOM): A Prototype for Usability Knowledge Management in Global Software, Second International Conference on Usability and Internationalization, Beijing, China, Lecture Notes in Computer Science, vol. 4559, 2007, pp. 297 - 305.

3.4 Classification scheme

For conducting a systematic mapping, a classification scheme needs to be defined [32]. We considered different facets, one for each research question. The exception is the last question (RQ9), for which we collected unstructured data without a predefined classification. We only looked at the main findings found as benefits and problems related with the implementation of KM in software testing reported by the selected studies. The categories comprising the other facets were defined following two approaches: (i) based on categories already considered in the literature; and (ii) taking the selected studies into account. Following, the categories of these facets are presented.

Research focus from the software testing perspective (RQ2): Studies on KM in software testing have focused on different aspects of software testing. Based on the selected studies, we considered six main categories, described below. Note that, according to our classification scheme, one study can span more the one research focus regarding the testing perspective.

- **Testing Process:** the focus is on managing knowledge in the context of a given testing process.
- **Test Case:** the focus is on managing knowledge about test cases, for supporting, e.g., test case reuse.
- **Testing Phase:** the study discusses the application of KM in a specific testing phase, such as unit testing, system testing, regression testing etc.
- **Testing Technique:** the focus is on managing knowledge about testing techniques, aiming at helping testers to select better suited testing technique for designing test cases.
- **Third Party Testing:** the study discusses KM applied to situations in which testing is accomplished by a third party (e.g. outsourcing).
- **General:** this category is used to classify those papers that discuss KM in software testing in general, without focusing in any specific aspect of software testing.

Research focus from the KM perspective (RQ3): Similarly to the previous, studies on KM in software testing also focus on different aspects of KM. Based on the selected studies, we considered the categories described below. Again, a study can span more the one research focus regarding the KM perspective.

- **Knowledge Management Model:** the study discusses a model for managing knowledge, considering knowledge processes, and, eventually some aspects of it, such as knowledge carriers.
- **Knowledge Representation:** the study discusses aspects related to how to represent testing knowledge.
- **Knowledge Packing:** the study goes beyond aspects related to knowledge representation, focusing on how to pack it. Studies classified in this category are also classified in the previous.
- **Knowledge Capturing:** the study addresses aspects related to how to acquire and store testing.

- **Knowledge Elicitation:** the study goes beyond aspects related to knowledge capturing, discussing also ways to elicit knowledge from experts. Studies classified in this category are also classified in the Knowledge Capturing category.
- **Knowledge Retrieval:** the study addresses aspects concerning retrieval of testing knowledge. In this case, the user is responsible for searching knowledge items.
- **Knowledge Dissemination:** regards pro-actively disseminating testing knowledge.
- **Knowledge Evolution:** the study approaches aspects related to the evolution of the testing knowledge already stored, such as evaluation and maintenance.
- **Knowledge Management Systems (KMS):** the study discusses aspects related to providing automated support for managing testing knowledge by means of a system. Studies classified in this category may be describing an actual system, as well as an architectural model or general features of a KMS.

Research type (RQ4): This facet captures the research approach used in the studies. It is general and independent from a specific focus area, and thus we adopted an existing classification: the one proposed by Wieringa et al. [38], and revisited by Petersen et al. [32] to become more general. Based on the selected studies, we disregarded some of its categories, since none of the selected studies was classified in those categories. The categories used in this mapping are summarized below. As pointed out by Wieringa et al. [38], studies can span more than one category, although some combinations are unlikely.

- **Solution Proposal:** In this research approach, the study proposes a solution for a problem and argues for its relevance, without a full-blown validation. The solution must be novel, or at least a significant improvement of an existing one. A proof-of-concept may be offered by means of a small example, a sound argument, or by some other means.
- **Validation Research:** In this research approach, the study investigates the properties of a proposed solution that has not yet been implemented in practice. It may have already been proposed elsewhere by the author or by someone else. The investigation uses a thorough, methodologically sound research setup. Possible research methods include, among others, experiments, prototyping and simulation.
- **Evaluation Research**: In this type of research, the study discusses the implementation of a technique in practice, and what are the consequences of the implementation in terms of benefits and drawbacks (implementation evaluation). The novelty of the technique is not necessarily a contribution of the study. However, if no industry cooperation or real world project is mentioned, then the study cannot be considered an evaluation research [32].

Reported Problems (RQ5): The categories for this facet are based on the main problems related to knowledge about software testing reported in the selected studies. We have identified five main categories of problems, namely: (i) Barriers in transferring testing knowledge; (ii) Loss of testing knowledge; (iii) Low reuse rate of testing knowledge; (iv) Testing knowledge is not properly shared; and (v) Testing knowledge is not properly considered for planning the testing process (including human resource allocation to testing activities). Studies can span more than one category.

Purposes (RQ6): In this facet, we wanted to learn the organizations' purposes, when employing KM in software testing. We have identified five main categories of such purposes: (i) Reuse of knowledge related to software testing, (ii) Support for decision making, (iii) Cost reduction, (iv) Competitive advantages, and (v) Organizational learning. Studies can span more than one category.

Types of knowledge (RQ7): This facet concerns the types of knowledge that have been dealt with in the research. In this case, we adopt the classification, proposed by Nonaka and Takeuchi [27], that distinguishes between tacit and explicit knowledge. Moreover, for explicit knowledge, based on the selected studies, we identified the types of knowledge that have been managed, into the following categories: Software Artifacts (essentially test cases), Classification Schemes (for testing techniques), Lessons Learned (about software testing), and Knowledge Packages. Since some studies did not specify a specific type of knowledge, in such cases we considered that they fall in the General category. Studies can span more than one category.

Technologies used (RQ8): This facet discusses the technologies that have been used or proposed to support managing software testing knowledge. Based on the selected studies, we consider the following categories: (i) Ontologies, (ii) Yellow Pages (or Knowledge Maps), (iii) Agents, (iv) Recommendation Systems, (v) Data warehouse, and (vi) Conventional Technologies. This last category mentions IT conventional technologies, such as databases, intranets, and Internet. Studies can span more than one category, but can also fit any, if the study does not explicitly approach this issue.

3.5 Limitations of this mapping

It is important to clarify that the mapping presented in this paper has some limitations. The study selection and data extraction steps were initially performed by just one of the authors, and thus some subjectivity could have been embedded. In order to reduce this subjectivity, the other two authors performed these same steps over the entire sample, so that all studies were analyzed by at least two researchers. The results of each reviewer were then compared in order to detect possible bias. An analysis of degree of conformance was performed to measure the level of agreement between the results obtained from the reviewers in the selection process. So, we calculated the kappa coefficient [20], which is a statistical measure of inter-rater agreement for qualitative (categorical) items. It is considered to be a more robust measure than simple percent agreement calculation, since it takes into account the agreement occurring by chance. Kappa maximum value is 1, representing total agreement. Values close to and below 0 indicate no agreement. For calculating the kappa coefficient, we considered the result set from Stage 1 (duplication removal), which contains 439 papers. So, in our case, the overall kappa coefficient obtained was 0.75, whose interpretation, according to Landis and Koch [20], is to have a substantial agreement.

Terminological problems in the search strings may have led to miss some primary studies. In order to minimize these problems, we performed previous simulations in the selected databases. We decide not to search any specific conference proceedings, journals, or the grey literature (technical reports and works in progress). Thus, we have just considered studies indexed by the selected electronic databases, and those obtained from by snowballing and direct search for works from researchers/research groups. The exclusion of other sources makes the review more repeatable, but possibly some valuable studies may have been left out of our analysis.

As pointed by Wohlin et al. [39], although the ambition of a mapping study is to summarize all relevant research in an area, different sets of papers will be obtained given a number of decisions taken. Researchers conducting secondary studies in general have to make a lot of decisions and exercise a lot of judgment. The decisions taken by researchers and the judgments exercised influences the outcome both in terms of which papers are found and what the researchers conclude from their secondary studies [39]. Moreover, Wohlin et al. [39] point out several difficulties in performing secondary studies, such as those related to the inclusion/exclusion criteria, how the area actually is defined, classification schemes, and influence from the research questions. In our case, we clearly experienced some of these difficulties.

Regarding the factor related to how the area actually is defined, this factor posed difficulties in defining the search string. Initially, we considered only studies that mention the term "knowledge management". However, is it enough for characterizing the studies in the area? The final version of our research string considers also knowledge sharing, reuse and transfer, trying to broadly cover the area.

Once we retrieved the papers, we had to decide if the studies meet our selection criteria. Some studies were subject of intense debate. For instance, we decided to eliminate two studies from the same research group ([16, 36]), although they were cleared related to KM in software testing. We read the full papers, and concluded that they do not actually regard "KM initiatives in software testing" (our inclusion criterion). The first paper [16] presents an empirical study investigating the relationship between schedules and knowledge transfer in software testing. Results from this study indicate that increased knowledge transfer between testing and earlier phases of software development is associated with testing schedule over-runs. A qualitative case study was conducted to interpret these results, and the authors conclude that this relationship can be explained with the size and complexity of software, knowledge management issues, and customer involvement. However, the paper does not discuss the application of KM in software testing. The second paper [36] presents an exploratory study to understand the practice of software testing, and to derive hypotheses on testing organizations and KM. The authors report that the results of their study can be used in developing testing organizations and their knowledge management strategies. However, again, the work does not discuss the application of KM in software testing. Both works fail to answer most of our research questions, and thus, we decided to exclude them from the mapping study. Thus, to some extension, our judgment was influenced by the research questions.

Finally, regarding classification schemes and how studies fit the corresponding categories, there is also much judgment involved. For instance, we adopted the classification scheme for research type proposed by Wieringa et al. [38]. However, as pointed by Wohlin et al. [39] this classification scheme was not developed for secondary studies in general. It was originally proposed for use in requirements engineering, and maybe it has to be revised to fit other contexts too, or there may be a need to provide more details and examples together with the classification scheme to ensure that the interpretation becomes more coherent. Moreover, classification is a judgment and, depending on the researchers' background and expertise, it may result in a certain classification bias, for example, favoring one type of research type over another.

4. Results

We performed the mapping study according to the steps described in Section 3. In this section, we present the results for each of the research questions defined in Section 3.1. To answer the questions, we used a form with the id of the paper, its bibliographic reference, and including the facets of the classification schema aforementioned. This form was used to extract the answers for each research question.

4.1. Classification by publication year and source (RQ1)

In order to offer a general view of the efforts in the area of KM in Software Testing, a distribution of the 15 selected papers over the years is shown in Fig. 2. As this figure suggests, KM initiatives in software testing is recent, starting basically in 2003. Since then, the research in the area has been relatively stable.

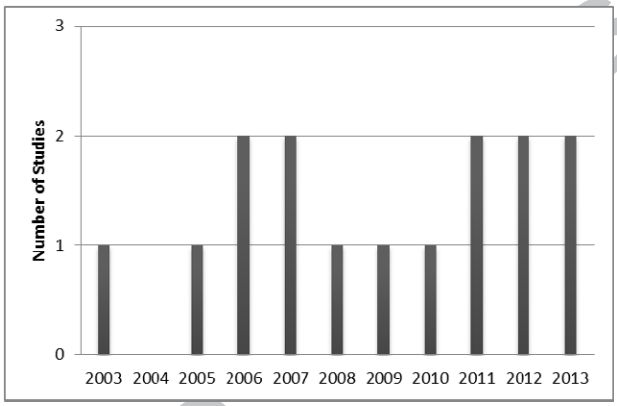


Fig. 2. Distribution of the selected studies over the years

The selected studies were published in three main vehicles: Journals, Conferences and Workshops. Conferences have been the main forum for presenting KM in software testing, encompassing 60.0% (9 studies in 15). Journals are the publication forum of 33.3% (5 out of 15). Finally, only one was presented in a Workshop (6.7%). It is worth noting that the 15 papers are published in 15 different venues, meaning that there is no "home" for this type of research yet. Table 8 presents the publication sources of the selected studies, their types, and their (#id). It is worth pointing out that 15 different publication sources were identified, one for each study, showing that currently there is no well-established forum for discussing the topic. Publications vehicles in areas such as Information Technology, Software Engineering and Knowledge Management seem to be more receptive for presenting studies in the area.

Table 8
Publication Sources

Publication Source	Type	#ID
International Workshop on Automation of Software Test (AST)	Workshop	#9
International Conference on Information Technology and Computer Science (ITCS)	Conference	#1
Industrial Engineering and Engineering Management	Conference	#2
Apperceiving Computing and Intelligence Analysis (ICACIA)	Conference	#3
Malaysian Conference in Software Engineering (MySEC)	Conference	#4
International Conference on Internet Computing for Science and Engineering (ICICSE)	Conference	#5
International Conference and Workshop on Emerging Trends in Technology (ICWET)	Conference	#6
International Conference on Software Testing, Verification, and Validation	Conference	#13
WSEAS International Conference on Software Engineering, Parallel and Distributed Systems	Conference	#14
International Conference on Usability and Internationalization	Conference	#15
Information and Software Technology	Journal	#7
Journal of Workplace Learning	Journal	#8
Knowledge and Process Management	Journal	#10
Empirical Software Engineering: An International Journal	Journal	#11
Journal of Systems and Software	Journal	#12

4.2. Research focus from the software testing perspective (RQ2)

Table 9 shows the distribution of the studies according to the research focus from the software testing perspective. Most of the papers address a specific aspect of software testing (10 out of 15 – 66.7%) for providing KM facilities. Test case reuse (4), knowledge related to a specific test phase or activity (4), and third party testing (3) are aspects that have also received attention. In particular, test case reuse has been the main focus of the recent research. We can also notice that there has been some focus on a specific testing phase. For instance, in #8, Nogeste and Walker (2006) proposes a KM-based regression testing process; in #9, Janjic and Atkinson (2013) discuss KM-based support for reusing class and unit test cases; in #15, Douglas (2007) focus on usability testing. On the other hand, 5 studies (33.3%) discuss KM applied to software testing in general, i.e. without focusing on any specific aspect of software testing.

Table 9
Research focus from the software testing perspective along the years

Research Focus	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	Total (%)
Test Case					#15				#6	#5	#9	4 (26.7%)
Test Phase				#8	#2, #15						#9	4 (26.7%)
Third Party Testing					#2	#13		#3				3 (20.0%)
Testing Process				#8				#3				2 (13.3%)
Testing Technique			#11	#12								2 (13.3%)
General	#10						#1		#4	#14	#7	5 (33.3%)

4.3. Research focus from the KM perspective (RQ3)

With respect to KM, as Table 10 shows, the great majority of the studies discussed aspects related to the KM process as a whole (KM Model) or focusing on one of its activities: knowledge representation (9), capture (6), and retrieval (5). Moreover, most of the papers (73.3%) describe KM systems to provide KM-based support for software testing. The study performed by Andrade et al. (2013) (#7) is the one that covers the largest number of KM activities.

Table 10
Distribution over research focus regarding the KM perspective

Research Focus	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	Total (%)
KM Systems (KMS)				#8	#2, #15		#1	#3	#4, #6	#5, #14	#7, #9	11 (73.3%)
Knowledge Representation			#11	#12	#2, #15	#13	#1			#5, #14	#7	9 (60.0%)
KM Model	#10			#8					#4, #6	#5	#7	6 (40.0%)
Knowledge Capturing			#11	#8, #12	#2		#1				#7	6 (40.0%)
Knowledge Retrieval			#11		#2		#1				#7, #9	5 (33.3%)
Knowledge Dissemination					#15					#14	#7, #9	4 (26.7%)
Knowledge Elicitation					#2						#7	2 (13.3%)
Knowledge Packing				#12								1 (6.7%)
Knowledge Evolution										#5		1 (6.7%)

4.4. Research Type (RQ4)

Table 11 presents the distribution according to the research types. Considering the categories defined by Wieringa et al. [38], research in KM initiatives in software testing is dominated by Solution Proposals. All the 15 studies propose some solution for KM in software testing. In addition to presenting a solution proposal, most of them also discuss some sort of evaluation: 7 studies also report an Evaluation Research (46.3%), while 3 studies report a Validation Research (20.0%). The 7 studies with an Evaluation Research discuss a practical implementation, and its consequences. There is a great interest in implementing the proposals in practice and especially in studies of real cases. Five studies (33.3%) present only a solution proposal. It is worthwhile to point out that some of these five studies discuss some sort of evaluation, namely #13 and #14. However, the evaluations performed are too simple for us to classify as Validation or Evaluation Research.

Table 11
Distribution over research type

Research type	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	Total (%)
Solution Proposal	#10		#11	#8, #12	#2, #15	#13	#1	#3	#4, #6	#5, #14	#7, #9	15 (100%)
Evaluation Research	#10			#8, #12				#3	#6	#5	#7	7 (46.3%)
Validation Research			#11				#1				#9	3 (20.0%)

4.5. Reported problems (RQ5)

Table 12 shows the distribution over the years considering the problems reported by software organizations related to knowledge about software testing. We regard these problems as a motivation to perform research in KM to software testing. We can notice that “Barriers in Testing Knowledge Transfer” and “Low Reuse Rate Knowledge” have the largest representativeness (9 out of 15, corresponding to 60%). Regarding reuse, software testing, in general, can involve reusing modules, test cases, components, and experiences. However,

testing teams, generally, do not reuse or take advantage on the knowledge acquired or the experience gained. Therefore, the same mistakes are repeated even though there are individuals in the organization with the knowledge and experience required to prevent this [3]. With respect to "Barriers in Knowledge Transfer", it stands out because transfer of organizational knowledge can be quite difficult to achieve. This occurs also because most of the knowledge in organizations is tacit, that is, derived from experience, and it becomes difficult to articulate.

Table 12
Distribution of related problems (motivation)

Problems	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	Total (%)
Barriers in testing knowledge transfer				#8	#2	#13	#1	#3	#4, #6	#5	#7	9 (60.0%)
Low reuse rate of testing knowledge	#10		#11	#12			#1		#4, #6	#5	#7, #9	9 (60.0%)
Problems related to testing knowledge sharing	#10				#15	#13	#1		#4, #6	#14	#7	8 (53.3%)
Loss of testing knowledge	#10				#2		#1		#6	#5		5 (33.3%)
Testing knowledge not considered in test planning				#8			#1		#6	#5		4 (26.7%)

4.6. Purposes to employ KM in software testing (RQ6)

Table 13 presents the distribution over the years considering the organizations' purposes in managing software testing knowledge. We can notice that "Knowledge Reuse" (11 studies – 73.3%) and "Organizational Learning" (8 studies – 53.3%) have the largest representativeness. We should highlight that some purposes that were identified are strongly related. For instance, lessons learned are means to promote both knowledge reuse and organizational learning. Thus, studies that reported that one of the purposes of applying KM in software testing is registering and disseminating lessons learned (7 studies – 46.6%) were considered in both categories. Knowledge reuse, in turn, helps increasing test effectiveness and thus leads to competitive advantages and cost reduction.

Table 13
Distribution of purposes

Purposes	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	Total (%)
Knowledge Reuse				#8	#2, #15	#13	#1	#3	#4, #6	#5, #14	#7	11 (73.3%)
Organizational Learning				#8, #12		#13		#3	#4	#14	#7, #9	8 (53.3%)
Competitive Advantages	#10			#8			#1	#3		#5	#7	6 (40.0%)
Decision Making			#11		#2				#4, #6		#7	5 (33.3%)
Cost Reduction					#2			#3			#7	3 (20.0%)

4.7. Types of knowledge items being managed (RQ7)

As already mentioned, we adopt the classification proposed by Nonaka and Takeuchi [27] that distinguishes between tacit and explicit knowledge. In the 15 selected studies, both are considered. Tacit knowledge is discussed in 10 studies (66.7%). They mention that it can be acquired from discussions, experiences from project members, questionnaires, expert network,

personalized training and communications. On the other hand, all the 15 studies (100%) have discussed explicit knowledge.

Moreover, for explicit knowledge, based on the 15 selected studies, we identify the types of knowledge that have been managed. Table 14 presents the distribution over the years. 40% of studies do not specify a specific type of knowledge. In such cases, we considered that they fall into the General category. Out of all the identified types, Software Artifacts are the most commonly cited category (6 out 15 studies, corresponding to 40%). Moreover, Test Cases are the software artifacts managed in all the studies classified in this category. According to Li and Zhang [22], the quality of test cases has a direct and significant impact on the software testing quality, and hence in software quality. Thus, reusing test cases seems to be effective and can promote test case design efficiency.

Table 14
Distribution of explicit knowledge

Explicit knowledge	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	Total (%)
Software Artifacts				#8, #15		#13			#6	#5	#9	6 (40.0%)
Lessons Learned						#13				#14	#7	3 (20.0%)
Classification Schema			#11	#12								2 (13.3%)
Knowledge Packages				#12								1 (6.7%)
General	#10				#1		#2	#3	#4, #6			6 (40.0%)

4.8. Technologies used (RQ8)

The technologies used to implement KM in software testing are shown in Table 15. We found 6 different technologies in this context. It is worthwhile to point out that 4 studies do not address this issue (#3, #11, #12, #13), representing 26.7%. In #3, Xu-Xiang and Wen-Nin (2010) propose a process, but they do not discuss technologies; Vegas and Basili (2005) (#11) and Vegas et al. (2006) (#12) propose a classification scheme, but they used it without any automated support, and without discussing aspects related to technology; in #13, Larsson et al. (2008) propose templates for two types of documents, to register knowledge regarding testing and lessons learned, respectively.

The great majority of the studies use conventional technologies (9 in 15 studies, corresponding to 60%). This category concerns IT conventional technologies, such as databases, intranets, and Internet. Yellow Pages (or Knowledge Maps) are discussed in 20% of the studies (3 in 15). Organizations can extract testing knowledge and create the testing knowledge base and then establish the reusable test knowledge repository. This repository may be a data warehouse, as cited by Li and Zhang (2012) (#5), and Desai and Shah (2011) (#6) (13.3% of the studies). Moreover, the representation of the knowledge structure can be in accordance with an ontology representation (#1, #5) (13.3%).

Table 15
Distribution of technologies used

Technologies	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	Total (%)
Conventional Technologies	#10			#8	#2, #15				#4, #6	#5, #14	#7	9 (60.0%)
Yellow Pages (or Knowledge Maps)							#1			#5	#7	3 (20.0%)
Ontologies							#1			#5		2 (13.3%)
Data Warehouse									#6	#5		2 (13.3%)
Recommendation Systems											#7, #9	2 (13.3%)
Agents											#9	1 (6.7%)
Study does not address this issue			#11	#12		#13		#3				4 (26.7%)

4.9. Benefits and Problems reported on the implementation of KM initiatives in software testing

As we can see from this mapping, there are many benefits of implementing KM in organizations for managing software testing knowledge:

A. **Test effectiveness increase.** Knowledge and experience about the domain and the system under test are essential for increasing test effectiveness. This helps testers to decide on which techniques to use, and to select test cases, among others.

B. **Selection and application of suited techniques, methods and test cases.** Experience plays a key role in testing, and managing past experience helps to effectively tailor the techniques and methods to the ongoing project. Some of these techniques, such as White-box Testing Techniques, Black-box Testing Techniques, Defect-based Testing Techniques, and even more Exploratory Testing, depend on the knowledge, experience and intuition of the tester.

C. **Competitive advantages.** In organizations, KM is now seen as a strategic factor and knowledge is also recognized as one of the main sources of cost savings and competitive advantage. The ability to transfer best practices in the organization is a means to build competitive advantage through the appropriation from scarce knowledge.

D. **Cost reduction.** In the testing context, cost is strongly related with time. Tester's experience is crucial for designing test cases and regression test selection. A good selection of test cases minimizes not only costs but also time.

Table 16 shows how the main benefits aforementioned are perceived in the selected studies. Two of these benefits highlight, being cited by over half the studies: "Test Increase effectiveness", which is cited in 10 of the 15 selected studies (66.7%); and "Selection and application of suited techniques, methods and test cases", which is cited in 7 of the 15 selected studies (46.7%).

Table 16
Distribution of the identified benefits

Benefits	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	Total (%)
A.			#11, #12	#8	#2, #15			#3	#4, #6	#5	#9	10 (66.7%)
B.			#11, #12	#8					#6	#5	#7, #9	7 (46.7%)
C.	#10			#8			#1	#3		#5		5 (33.3%)
D.			#11, #12		#2							3 (20.0%)

Although KM in software testing brings many benefits, there are also problems:

A. **KM systems are not appropriate yet:** There are many difficulties in implementing knowledge acquisition, coding, storage and searching functionalities effectively in KM systems, because it involves all the problems mentioned above, such as how to represent knowledge, and time and interest of the employees.

B. **Employees are normally reluctant to share their knowledge:** Experiences are grasped by only a few people and do not become public knowledge. This fact disables knowledge transfer in testing.

C. **Increased workload:** Shortage of time is a potential risk to incorporate the principles of KM in software testing, because knowledge sharing can imply in increasing the employee workload and costs.

Table 17 shows how the main problems aforementioned are perceived in the selected studies. The inadequacy of existing KM systems appears as the most cited problem (40%). It is important to highlight that not all studies mention problems related to managing testing knowledge.

Table 17
Distribution of problems

Problems	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	Total (%)
A.	#10				#2, #15			#3	#6	#5		6 (40.0%)
B.					#15		#1		#6		#7	4 (26.7%)
C.			#11						#6	#13	#7	4 (26.7%)

5. Discussion

Software has become more and more widespread and indispensable in critical and complex application domains, making testing processes increasingly important and complex. Testing is a knowledge-intensive process, and in this context, sharing knowledge and experience can provide several benefits However, studies have reported that the greatest problem related to managing software testing knowledge is the lack of knowledge reuse within the organizations. Knowledge is retained with a single individual and therefore becomes more difficult to raise this knowledge to the organizational level. Even when some KM strategy is applied, it is not always feasible to achieve organizational learning, because the employees are reluctant to share their knowledge as they feel that retaining this knowledge is an advantage over their colleagues [3]. All these problems are pointed out as purposes for applying KM in software testing, in particular knowledge reuse.

Out of the 15 studies analyzed in this mapping, KM in software testing has focused on different aspects. Reuse of test cases is a perspective that has received more attention. Software testing in general involves reuse of modules and test cases. Reusing knowledge already present in existing test cases has a potential to significantly reduce software development costs and time consumption [9, 14].

With respect to the KM perspective, the great majority of the studies discuss aspects related to implementing a KM process as a whole or focusing on one of its activities. Most of the studies discuss aspects related to providing automated support for managing testing knowledge by means of a system. These studies present a Knowledge Management System (KMS) as an actual system, as a prototype or as an architectural model.

Considering the research type performed by the 15 selected papers, solution proposal is the most prominent research type employed. All the 15 studies present a solution proposal. Moreover, 7 studies discuss the implementation of an approach in practice. There is a significant interest in implementing the proposals in practice and especially in real world scenarios.

With respect to the types of knowledge items managed, both tacit and explicit have been considered important. Most of the studies identify that tacit knowledge is more difficult to acquire, as part of personal experiences by the members of the test team. According to Nonaka and Takeuche [27], as expected, tacit knowledge really is hard to be acquired, and it requires good strategies to acquire and process this knowledge; however, it is more valuable. In papers #4, #7, #10, for instance, although dealing with the two types of knowledge (tacit and explicit), the authors also emphasize the importance to capture the tacit knowledge and treat it.

During the mapping it was possible to infer that much of the explicit knowledge was related to the reuse of test cases. According to Liu et al. [23], more detailed information on test cases can provide a greater learning. As test cases evolve in applications, they may be changed for a variety of reasons. Thus an efficient and effective KM process can help in evaluating the impact and in conducting changes in the test cases.

With respect to the technologies used to implement KM, Knowledge Maps or Yellow Pages seem to have good results where testers want to find the right experts for helping them. A Knowledge Map contains information about experiences that an employee possesses. It is a stock catalogue about knowledge [23]. In #1, Liu et al. (2009), for instance, create a KM model whose one of its main components is a Knowledge Map repository, considered the kernel of the system. The system identifies, by means of statistics, the staff with knowledge, and that will improve the culture of knowledge-sharing in the enterprise. In #5, Li and Zhang (2012) present a Knowledge Management Model and one of the elements of this model is also a knowledge map.

On the other hand, Data Warehouse, Recommendation System and Ontologies are also discussed. In special, ontologies minimize ambiguity and imprecision in interpreting shared information [17]. Ontologies are considered a key enabling technology for KM [5, 17]. They provide a shared and common understanding of a domain that can be communicated between people and application systems. Their use offers an opportunity for improving KM capabilities in large organizations [8]. In ontology-based KM systems, ontologies are mainly used [1]: (i) to support knowledge search, retrieval, and personalization; (ii) to serve as basis for knowledge gathering, integration, and organization; and (iii) to support knowledge visualization. More specifically, domain ontologies are used to structure the content area of documents and providing background knowledge for inferences.

Three studies (#1, #5, and #7) discuss ontologies in some aspect. However, only two (#1 and #5) indeed, use ontology. This seems to be a problem, since, as pointed out by Staab et al. [34], ontologies are capable of binding KM activities together, allowing a content-oriented view of KM. One possible explanation for this low number of studies addressing ontology-based KM initiatives in software testing is the fact that developing an ontology is a hard task, especially in complex domains, as is the case of software testing.

6. Conclusions

According to Kitchenham et al. [19], a mapping study gives an idea, in the early phases, of shortcomings in existing evidence, which becomes a basis for future studies. In this paper, we presented a systematic mapping on KM initiatives in software testing. Nine research questions were defined and addressed investigating the following facets: (i) distribution of the selected studies over the years; (ii) research focus from the software testing perspective; (iii) research focus from the KM perspective; (iv) research type; (v) reported problems; (vi) purposes to employ KM in software testing; (vii) types of knowledge typically managed in the context of software testing; (viii) technologies used in KM in software testing; (ix) main conclusions (benefits and problems) reported on the implementation of KM.

KM in software testing has shown to be a very promising research area, since KM helps in handling knowledge within the organization in several respects, as shown by this systematic mapping. The main contribution of this work is on making evident some aspects associated to the employment of KM in software testing, in order to drive future research in this area. In this context, we highlight the following conclusions: (i) KM in software testing is a recent research; (ii) the major problem in organizations are low reuse rate of knowledge and barriers in knowledge transfer; (iii) reuse of testing knowledge is the main purpose of applying KM in software testing; (iv) there is a great concern with explicit knowledge, in particular, test cases artifacts, although tacit knowledge has been also recognized as a very useful knowledge item; (v) advanced technologies used to provide KM in software testing include recommendation systems, ontologies and yellow pages (knowledge maps).

ACKNOWLEDGMENTS

The first author acknowledges FAPESP (Process: 2010/20557-1) for the financial grant. The second author acknowledges FAPES (Process Number 52272362/11) and CNPq (Process Number 485368/2013-7) for the financial grant.

References

[1] A. Abecker, L. van Elst, Ontologies for Knowledge Management, International Handbooks on Information Systems, Springer-Verlag, 2004, pp. 435-454.

[2] A. Abran, P. Bourque, R. Dupuis,W.M. James, Guide to the Software Engineering Body of Knowledge – SWEBOK, NJ, USA: IEEE Press, 2004.

[3] J. Andrade, J. Ares, M. Martínez, J. Pazos, S. Rodríguez, J. Romera, S. Suárez, An architectural model for software testing lesson learned systems, Information and Software Technology, 55 (2013), 18-34.

[4] A. Beer, R. Ramler, The Role of Experience in Software Testing Practice, in: Software Engineering and Advanced Applications (SEAA), 2008, pp. 258-265

[5] V. R. Benjamins, D. Fensel, A. G. Pérez, Knowledge Management through Ontologies, in: Proc. of the 2nd International Conference on Practical Aspects of Knowledge Management, Switzerland, 1998, pp. 5-12.

[6] R. Black, J. L. Mitchell, Advanced Software Testing: guide to the ISTQB advanced certification as an advanced technical test analyst, USA:Oreilly & Assoc, 2008.

[7] I. Burnstein, Practical Software Testing: a process-oriented approach, third ed., New York: Springer Professional Computing, 2003.

[8] J. Davies, D. Fensel, F. Van Harlemen, Towards the Semantic Web: Ontology-driven Knowledge Management, John Wiley & Sons, 2003.

[9] Desai, S. Shah, Knowledge Management and Software Testing. In: International Conference and Workshop on Emerging Trends in Technology (ICWET), Mumbai, India, 2011, pp. 767-770.

[10] F. Elberzhager, J. Münch, V. T. N. Nha, A systematic mapping study on the combination of static and dynamic quality assurance techniques. Information and Software Technology, 54 (2012), 1-15.

[11] F.O. Bjørnson, T. Dingsøyr, Knowledge management in software engineering: A systematic review of studied concepts, findings and research methods used, Information and Software Technology, 50 (2008), pp. 1055-1068.

[12] IEEE Std 1012-2004: IEEE Standard for Software Verification and Validation, New York, NY, USA, 2004, pp. 120.

[13] S. Jalali, C. Wohlin, Systematic literature studies: database searches vs. backward snowballing, in: international symposium on Empirical software engineering and measurement, 2012, pp. 29-38.

[14] W. Janjic, C. Atkinson, Utilizing software reuse experience for automated test recommendation, in: Proceedings of the 8th International Workshop on Automation of Software Test (AST), San Francisco, USA, 2013, pp. 100-106.

[15] J. Itkonen, M. V. Mäntylä,, C. Lassenius, The Role of the Tester's Knowledge in Exploratory Software Testing, IEEE Trans. on Software Engineering. 39(2013), pp. 707-724.

[16] K. Karhu, O. Taipale, K.i Smolander, Investigating the relationship between schedules and knowledge transfer in software testing, Information & Software Technology 51(3), 663-677, (2009)

[17] H. M. Kim, Developing Ontologies to Enable Knowledge Management: Integrating Business Process and Data Driven Approaches, AAAI Workshop on Bringing Knowledge to Business Processes, 2000, pp.20-22.

[18] B. Kitchenham, S. Charters, Guidelines for performing Systematic Literature Reviews in Software Engineering, School of Computer Science and Mathematics Keele University and Department of Computer Science University of Durham, UK, v. 2.3, 2007.

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

[20] JR. Landis, GG. Koch, The measurement of observer agreement for categorical data. Biometrics, 33(1977), pp. 159-174.

[21] S. Liao, Knowledge management technologies and applications-literature, Expert Systems with Applications, 25 (2003), 155–164.

[22] X. Li, W. Zhang, Ontology-based Testing Platform for Reusing, in: International Conference on Internet Computing for Science and Engineering, Henan, China, 2012, pp.86-89.

[23] Y. Liu, J. Wu, X. Liu, G. Gu, Investigation of Knowledge Management Methods in Software Testing Process, in: International Conference on Information Technology and Computer Science, Kiev, Ukraine, 2009, 90-94.

[24] R. Loana, M. Lindvall, Knowledge Management in Software Engineering, IEEE Software, 2002, pp. 26-38.

[25] A. P. Mathur, Foundations of Software Testing. fifth ed., Delhi, India: Dorling Kindersley, Pearson Education in South Asia, 2012.

[26] G. J. Myers, The Art of Software Testing, second ed., Canada: John Wiley and Sons, 2004.

[27] I. Nonaka, H. Takeuchi, The Knowledge-Creating Company. How Japanese Companies Create the Dynamics of Innovation, USA:Oxford University Press, 1997.

[28] I. Nonaka, G. Krogh, Tacit Knowledge and Knowledge Conversion: Controversy and Advancement in Organizational Knowledge Creation Theory, Organization Science, 20(2009), n. 3, 63-652.

[29] D. E. O'Leary, Enterprise Knowledge Management. Journal Computer, 31(1998), 54-6.

[30] D. E. O'Leary, R. Studer. Knowledge management: An interdisciplinary approach. Intelligent Systems,16 (2001), 24-25.

[31] Pak-Lok Poon, T. H. Tse, Sau-Fun Tang, Fei-Ching Kuo, Contributions of tester experience and a checklist guideline to the identification of categories and choices for software testing, Software Quality Journal 19(2011), pp. 141-163.

[32] K. Petersen, R. Feldt, S. Mujtaba, M. Mattsson, Systematic mapping studies in software engineering, in: Proceedings of the 12th international conference on Evaluation and Assessment in Software Engineering, 2008, pp.68-77.

[33] S.L. Pfleeger, J.M. Atlee, Software Engineering: Theory and Practice, fourth ed., Prentice Hall, 2009.

[34] S. Staab, R. Studer, H. P. Schurr, Y. Sure, Knowledge Processes and Ontologies, IEEE Intelligent Systems, 16 (2001), 26-34.

[35] E. F. Souza, R. A. Falbo, N.L. Vijaykumar, Knowledge management applied to software testing: A systematic mapping, in: The 25th International Conference on Software Engineering and Knowledge Engineering (SEKE 2013), Boston, USA, 2013, pp. 562-567.

[36] O. Taipale, K. Karhu, K. Smolander, Observing Software Testing Practice from the Viewpoint of Organizations and Knowledge Management, in: Empirical Software Engineering and Measurement (ESEM), Madrid, 2007, 21-30.

[37] I. Tuomi, Data Is More Than Knowledge: Implications of the Reversed Knowledge Hierarchy for Knowledge Management and Organizational Memory, in: Proceedings of the 32nd Hawaii International Conference on System Sciences, 1999, pp. 1-12.

[38] R. Wieringa, N. Maiden, N. Mead, C. Rolland, Requirements engineering paper classification and evaluation criteria: a proposal and a discussion, Requirements Engineering, 11(2006), 102–107.

[39] C. Wohlin, P. Runeson, P. A. M. S.Neto, E.Engström, I. C. Machado, E. S. Almeida, On the reliability of mapping studies in software engineering, The Journal of Systems and Software, 86 (2013) 2594-2610.

[40] M. H. Zack, M. Serino, Knowledge Management and Collaboration Technologies, in: Knowledge, Groupware and the Internet, Butterworth-Heinemann, 2000, pp. 303-315.