A Systematic Mapping Study on the Combination of Static and Dynamic Quality Assurance Techniques

Frank Elberzhager[a], Jürgen Münch[b], Vi Tran Ngoc Nha[c]

a Fraunhofer Institute for Experimental Software Engineering (IESE), Fraunhofer Platz 1, 67663 Kaiserslautern, Germany
b University of Helsinki, P.O. Box 68, Gustaf Hällströmin katu 2b, 00014 Helsinki, Finland
c University of Kaiserslautern, Gottlieb-Daimler Str., 67663 Kaiserslautern, Germany

Abstract

Context: A lot of different quality assurance techniques exist to ensure high quality products. However, most often they are applied in isolation. A systematic combination of different static and dynamic quality assurance techniques promises to exploit synergy effects, such as higher defect detection rates or reduced quality assurance costs. However, a systematic overview of such combinations and reported evidence about achieving synergy effects with such kinds of combinations is missing.
Objective: The main goal of this article is the classification and thematic analysis of existing approaches that combine different static and dynamic quality assurance technique, including reported effects, characteristics, and constraints. The result is an overview of existing approaches and a suitable basis for identifying future research directions.
Method: A systematic mapping study was performed by two researchers, focusing on four databases with an initial result set of 2498 articles, covering articles published between 1985 and 2010.
Results: In total, 51 articles were selected and classified according to multiple criteria. The two main dimensions of a combination are integration (i.e., the output of one quality assurance technique is used for the second one) and compilation (i.e., different quality assurance techniques are applied to ensure a common goal, but in isolation). The combination of static and dynamic analyses is one of the most common approaches and usually conducted in an integrated manner. With respect to the combination of inspection and testing techniques, this is done more often in a compiled way than in an integrated way.
Conclusion: The results show an increased interest in this topic in recent years, especially with respect to the integration of static and dynamic analyses. Inspection and testing techniques are currently mostly performed in an isolated manner. The integration of inspection and testing techniques is a promising research direction for the exploitation of additional synergy effects.

Keywords:
Combination, dynamic quality assurance, inspection, static quality assurance, systematic mapping study, testing

1. Introduction

Nowadays, software and software-intensive systems can be found all around us. Since they are growing both in size and complexity, developing high-quality software is becoming more challenging and expensive. In order to achieve the desired time, cost, and quality goals, the development approach, including the quality assurance (QA) activities, has to be optimized. According to Burnstein [49], in this article, QA activities are understood as all kinds of analytical activities conducted during software development with the intention of finding and

removing defects. One important strategy in this direction is a systematic combination of existing QA techniques in order to obtain certain synergy effects, such as reduced costs or higher effectiveness.
Various kinds of QA techniques and methods to ensure software quality exist that involve preventing defects or detecting existing defects. Those QA activities that focus on detecting existing defects are also called verification and validation activities, which is defined by the *IEEE Standard Glossary of Software Engineering Terminology* as "the process of determining whether the requirements for a system or component are complete and correct, the products of each development phase fulfill the requirements or conditions imposed by the previous phase, and the final system or component complies with specified requirements" [37].
With respect to software verification and validation activities, static and dynamic QA techniques can be distinguished. Static QA techniques (e.g., inspections, reviews, walkthroughs, or static analyses such as program slicing) do not need models or code to be executed, but rather examine artifacts such as requirements documents, design models, or code without running them. In contrast, dynamic QA techniques (e.g., equivalence partitioning, boundary value analysis, control-flow based testing techniques, or dynamic analyses such as program profiling) need to execute programs or program parts.
The use of both static and dynamic QA activities is well known to software engineering practitioners. However, usually there is no systematic combination of such activities to exploit further benefits.
Today, a large number of well-established static and dynamic QA techniques exist, such as various inspection and testing techniques [48][49][56]. However, the effort for applying these techniques sometimes consumes more than 50 percent of the overall development effort, especially for conducting testing activities [64][34][55]. Thus, one objective is often to improve the efficiency of testing and to reduce the overall QA effort. Besides this, further desirable goals include improving overall effectiveness (i.e., finding as many defects as possible – especially critical ones – before distributing software), planning and controlling QA activities, and improving the overall quality.
In the past, a lot of research has been performed to develop and improve a variety of static and dynamic QA techniques. Juristo et al. [52] examined 25 years of empirical studies with respect to a large number of different testing techniques, classified them, and summarized the main findings. They conclude that the current testing knowledge is very limited. With respect to software inspections, Aurum et al. [58] examined software inspection processes published during the 25 years since inspection as a QA technique was first published by Fagan in 1976 [24]. They identified different inspection processes and support for the inspection, such as reading techniques, tools, and support for deciding to perform a re-inspection. In conclusion, Aurum et al. [58] stated that the identified studies contribute to the evolution of software inspections, but many research questions remain open. Another examination of software inspection research, covering the period between 1991 and 2005, was performed by Kollanus and Koskinen [60]. They classified the identified articles into a technical view (e.g., reading techniques, effectiveness factors), a management view (e.g., inspection impact on development process), and other topics (e.g., defect estimation, inspection tools). The two authors concluded that much research has been performed with respect to software inspections, but that empirical knowledge remained low.
One fundamental observation with respect to research on inspection and testing techniques, which are two of the best-established static, respectively dynamic, QA techniques, is that most often, this research is done to improve inspections or testing themselves. In contrast, some studies compare different inspection and testing techniques [23][73], which often resulted in the conclusion to apply them in combination [2][59]. Other studies calculated effectiveness values when applying them in combination to demonstrate the benefit of a joint

application [21][6]. However, except for suggestions to apply both, no concrete process or additional advices is usually provided.
Conventional software engineering, particularly standard lifecycles (such as the waterfall and "V" models), have emphasized static methods during early (pre-code) development phases and dynamic development during later (post-code) development phases. However, from our point of view, combining different static and dynamic QA techniques, such as inspections and testing, is a promising way to improve QA and to cope with problems such as high QA costs. The connection between static and dynamic QA seems intuitively clear and obvious, but in practice it is often lost or obscured. The result is poorly prioritized and often redundant QA effort. It is perfectly possible that combining static and dynamic quality assurance techniques may have been used in practice already, because the underlying reasoning is grounded on well-known software engineering practices. However, even in this case, it is questionable whether existing approaches actually rely on explicit, well-grounded and evaluated approaches instead of merely being based on common sense and unsystematic procedures.
The main objective of this systematic mapping study is to obtain a profound overview of existing approaches that combine static and dynamic QA techniques, and of their goals (e.g., reduction of effort, improvement of defect detection). To the best of our knowledge, no systematic mapping study of the combination of static and dynamic QA techniques exists. One survey [73] summarizes some defect detection studies and compares inspection and testing techniques. However, this study has a narrower scope, as it only considers inspection and testing (and not static and dynamic quality assurance activities in general), and the survey did not explicitly follow procedures for a systematic mapping study, which raises questions regarding its completeness. Aurum et al. [58], for example, mentioned ten open research questions in their conclusion. One of these is about the relationship between inspections and software testing and the best way these techniques might complement each other. Thus, the results of this mapping study may be a substantial starting point for comprehensive systematic literature reviews and future research in the area of combining static and dynamic QA techniques.
The article is structured as follows. Section 2 presents the research methodology pursued to perform the systematic mapping study. The results are shown and described in Section 3. Section 4 comprises a discussion of the results and their implications. Section 5 concludes the article and presents directions for future work.

2. Research methodology

According to Petersen et al. [61], a systematic mapping study is considered as a kind of secondary study that reviews articles related to a specific research topic, and which aims at providing a classification, conducting a thematic analysis, or presenting publication channels. A number of research questions must be defined in order to obtain these objectives in a systematic manner. Consequently, presenting an overview of a certain research area and identifying research gaps are the main goals of a structured mapping study. Our main motivation for this systematic mapping study can be summarized as (i) giving an overview of combined QA approaches and (ii) establishing promising future research directions with respect to the given research area.
Unlike a systematic literature review [32], which is another kind of secondary study and aims at answering specific research questions by identifying, analyzing, and interpreting all relevant evidence, such questions are usually not answered by a systematic mapping study because the identified articles are not analyzed in enough detail [61]. However, based on an initially defined process for performing systematic mapping studies [61], best practices defined by Kitchenham and Charters [32], such as using a protocol during the mapping process or considering quality criteria when evaluating the identified articles, are considered in addition in order to exploit additional benefits.

Petersen et al. [61] suggest five process steps when performing a systematic mapping study, which can be grouped into three main phases, namely planning, conduction, and reporting the review (which is consistent with systematic literature reviews [32]). In the planning phase, the research questions are developed in order to define the review scope. In addition, we developed a protocol for the entire process of the mapping study. The purpose of this review protocol is to support researchers in avoiding bias in conducting the review. In the conduction phase, all activities are performed according to the designed protocol, including searching and selecting primary studies based on screening the articles found. Afterwards, the necessary data is extracted and synthesized. Finally, in the reporting phase, the findings of the systematic mapping study are used to answer the research questions. The results of the review are prepared in an appropriate format considering the intended audience and research community. The research questions and the main steps pursued by the authors when performing the systematic mapping study are described in detail next.

2.1 Research questions

The overall objective of the systematic mapping study is to **identify approaches that combine static and dynamic quality assurance techniques**. This high-level objective was divided into six concrete research questions (RQ) in order to obtain a more detailed and comprehensive view on the topic. Besides the creation of a basis regarding combined static and dynamic QA approaches, the publication channel and the publication year were gathered. Furthermore, the objectives and evidences were analyzed. Finally, concrete QA techniques that are combined and input data that are needed to apply the QA techniques were extracted. Table 1 shows the six corresponding questions together with a rationale for each one.

No.	Research question	Rationale
RQ1	What are existing approaches that combine static and dynamic quality assurance techniques and how can they be classified?	The first research question defines the basis of this systematic mapping study and provides an overview of the existing approaches that combine static and dynamic quality assurance techniques.
RQ2	In which sources and in which years were approaches regarding the combination of static and dynamic quality assurance techniques published?	The second research question indicates whether there are specific publication channels and when effort regarding this research area was made.
RQ3	Is any kind of evidence presented with respect to the combination of quality assurance techniques and if so, which kind of evidence is given?	The third research question shows whether the approaches were empirically evaluated or whether just initial ideas are presented. This information was used to evaluate the maturity of the approaches.
RQ4	What are the objectives of combined quality assurance approaches?	The fourth research question provides detailed information what the purpose of each approach is and what is addressed and improved when applying a combined approach.
RQ5	Which static and dynamic quality assurance techniques are used in combined quality assurance approaches?	The fifth research question presents the concrete static and dynamic QA techniques that are combined.
RQ6	Which input is used for static and dynamic quality assurance techniques in combined quality assurance approaches?	The sixth research question gives information about the data or information needed to apply the combined approach, with respect to both static and dynamic QA techniques.

Table 1: Research questions and their rationals

2.2 Source selection and definition of search strategy

The search for primary studies was based on the following four reference databases: Inspec, Compendex, IEEE Xplore, and ACM Digital Library. The reason for choosing these libraries was that these are some of the most relevant sources in software engineering. The first two databases, Inspec and Compendex, are comprehensive databases containing millions of

publications, especially in the engineering and computer science domains. Moreover, these two databases are accessed by using the Engineering Village interface [29], which is considered to be user-friendly and provides advanced search features. In order not to miss any important publication, IEEE Xplore and ACM Digital Library, which are not covered by Inspec and Compendex, were also used to identify relevant literature.
References of identified articles were scanned, but basically it turned out that they were either not relevant (e.g., description of single quality assurance technique, reference used to motivate the topic, reference used to define certain terms) or were already included in, respectively summarized by, included articles. Furthermore, including articles based on reference lists would have had no influence on the derived classification. Therefore, a decision was made not to include articles based on reference lists, which also made the final result set of articles more traceable.
Kitchenham and Charters [32] recommend using the so-called PICO (population, intervention, comparison, and outcome) criteria for deriving the search string. Petersen et al. [61] share this recommendation and mention that the search term should be driven by the research questions. The PICO criteria were taken into account when we formulated our search term. Furthermore, the search string was determined systematically based on keywords derived from the overall research objective, which resulted in four attributes: *static QA techniques*, *dynamic QA techniques*, *software domain* and *combination*. The term "software", for example, belongs to the population criteria, but no concrete term was selected for the outcome aspect in order not to restrict the result set, i.e., all existing outcomes regarding combined approaches are of interest in this mapping study.
One constraint when defining a search string is that the result set is of manageable size, but still has the maximum possible coverage. Therefore, some additional synonyms and relevant terms that are most common for each attribute were selected and added to the search string. For example, "static analysis", respectively "dynamic analysis", was added (which is consistent with definitions given in [37]). Furthermore, the term "quality assurance" was taken, as this is used very commonly by different authors in this area (e.g., [48][49]) and IEEE Standards.
However, terms that have no clear meaning or whose scope is too broad, were not included. For example, quality control could be a reasonable term. Nevertheless, (i) the IEEE Standard Glossary of Software Engineering Terminology [37] states that the term has no standardized meaning, (ii) Jones [57], for examples, mentions that "quality control involves defect prevention and defect removal activities", whereas the mapping study only focuses on defect removal activities (i.e., analytical quality assurance activities), (iii) quality control is sometimes associated with evaluating software instead of finding and removing defects, and (iv) static or dynamic quality control is no common term. Other examples of potential search terms are "verification" and "validation". However, the IEEE Standard Glossary of Software Engineering Terminology [37] provides only general definitions and does not provide any concretization with respect to certain quality assurance techniques. Consequently, the terms are used inconsistently (e.g., verification and validation are sometimes considered only as testing activities; another common understanding is that verification comprises every quality assurance activity prior to acceptance testing).
Boolean operators were used to form the proper search string. The resulting search string was: *(inspection or review or "static analysis" or "static quality assurance") AND (test* or "dynamic quality assurance" or "dynamic analysis") AND software AND (combin* or integrat* or synergy or "trade off")*, and was applied to check keyword, title, and abstract fields within the corresponding databases.

2.3 Study selection

The aim of the selection process was to identify those articles that are most relevant for the objective of this mapping study. Overall, 2498 articles were considered. First, the review team, consisting of two researchers, retrieved a total of 2012 articles, with 929 articles from Inspec and 1083 articles from Compendex. The reviewers utilized the reference management tool Zotero [74] to handle resources such as bibliography generation, categorization of the articles, storage of downloaded full-text articles, and sorting of articles in alphabetical order to identify duplicates.
The search engine Engineering Village provided support for discarding all non-English articles by title (i.e., articles were removed that did not have an English title or English abstract). Of 2012 articles, 309 duplicated ones were removed (phase 0). The remaining 1703 articles were chosen as a basis for the next steps, namely, the performance of inclusion and exclusion. In general, the main inclusion criteria for articles were that these *articles describe a combination of static and dynamic QA techniques, describe evidence with respect to such a combination, or present an overview of existing approaches that combine static and dynamic QA techniques.*
Phase 1 comprised inclusion and exclusion based on article title and publication year. 1597 articles were found to be irrelevant, resulting in 106 articles included for the next steps. The titles of all articles were reviewed by the two researchers independently, with an agreement of more than 95% (either both researchers decided to include or to exclude an article). The resulting Cohen’s Kappa value is about 0.65, which indicates substantial agreement [65]. Articles that were judged differently based on the title were discussed by the two researchers until an agreement was found. The main exclusion criteria based on the title were complete conference proceedings, articles related to different domains (e.g., articles related to hardware, microprocessors, or electronics), and articles covering only static or dynamic QA techniques. Finally, articles published earlier than 1985 were excluded. The main motivation for this was twofold: first, new static and dynamic QA techniques were still evolving and thus, articles published later than 1985 are more relevant with respect to a combination of QA techniques. Second, approaches before 1985 presented merely initial insights and the conclusions drawn are not necessarily valid when referring to today’s combined quality assurance.
Of 106 chosen articles, 46 articles were discarded and 60 articles were selected based on the abstract in phase 2. The most common exclusion criteria were articles that do describe a combined approach but not with respect to quality assurance purposes (e.g., the goal of combination was to improve the design process) and articles related to different domains. Finally, articles covering only static or dynamic QA techniques or combinations of only static or only dynamic QA techniques were excluded.
Finally, phase 3 – inclusion and exclusion – was done based on full texts. 60 remaining articles from phase 2 were reviewed and 48 articles were chosen by applying six quality criteria to check their suitability. 12 articles were discarded due to their short content (e.g., presentation outline) or non-English full text (i.e., it happened that the title and the abstract of an article were written in English (and thus not discarded in phase 0), but the full text was in a foreign language). Of these 48 articles, 47 primary studies were identified and one secondary study [73]. In order not to miss any relevant article in the current mapping study, the articles considered in the above-mentioned secondary study (i.e., survey) were checked against those included in this mapping study. Another four articles could be added in this way, resulting in a final set of 51 primary studies.
In order to assure that no important article was missing, a cross-check was performed by applying the same search process using ACM Digital Library and IEEE Xplore. Unfortunately, both databases did not allow using the complete search string as defined. Therefore, slightly adapted and simplified search strings were used instead according to the provided ‘and’ and ‘or’ conjunctions of the search interfaces. 263 articles were retrieved from

IEEE Xplore and 223 from ACM Digital Library. Duplicates were removed and inclusion and exclusion criteria were also applied to those articles, i.e., the same phases were run through for analyzing the retrieved articles. However, those articles identified as relevant had already been included during the previous selection steps (i.e., the set of relevant articles found by IEEE Xplore and ACM Digital Library was only a subset of the relevant articles found by Compendex and Inspec). Consequently, the number of selected articles (i.e., primary studies [32]) remained 51. Figure 1 gives an overview of the search process.

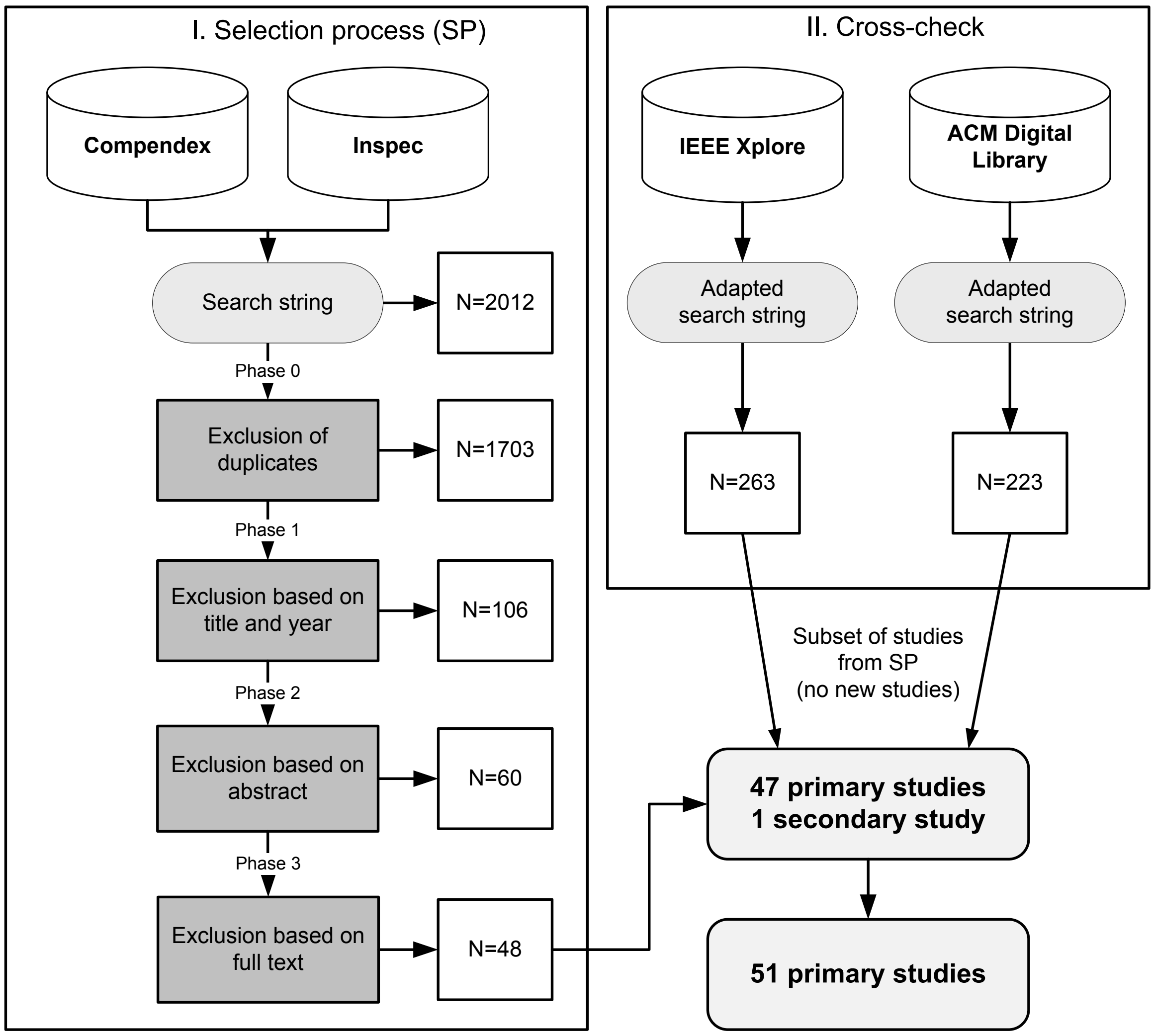

Figure 1: Search process for the selection of primary studies

2.4 Data extraction and synthesis

A data extraction form was developed to gather all relevant data from the identified articles. Besides general information (e.g., title and author), data with respect to each research question were extracted. This mainly includes a description of the approach (RQ1), the publication year and channel (RQ2), evaluation results (RQ3), goals of the combined approaches (RQ4), QA techniques used (RQ5), and necessary input for QA techniques (RQ6). Overall, the extraction form comprises 19 fields to be completed for each article. The articles included in the results were checked according to the following quality criteria:

- The research goals are clearly described.
- The approach is explained sufficiently.
- Context and environmental factors are clearly presented.
- Input and output data for using the approach are explicitly mentioned.

- The performance of the approach is clearly stated.
- Evidence of the approach is documented.

For creating an overview of existing combination approaches, the first two quality criteria are most important. They could be answered positively, resulting in the fact that all 51 articles were included in the final result set. However, some of the remaining quality criteria could not be checked for each article because the corresponding information was not contained, which shows that some articles lack certain kinds of information (e.g., in case no evaluation was presented, no such data could be extracted). Nevertheless, this did not affect the possibility of answering the corresponding research questions, but is reflected in the concrete results (e.g., if no evidence was presented for an approach, this was stated in the corresponding research question and presents an overview of the ratio of evaluated and not-evaluated approaches). Consequently, it was decided to exclude none of the 51 articles based on the six stated, respectively analyzed, criteria. Finally, the data extracted from the articles was synthesized to answer the six research questions.

3. Results

The results of the systematic mapping study are presented next, ordered by the six research questions.

3.1 RQ1: What are existing approaches that combine static and dynamic quality assurance techniques and how can they be classified?

The 51 identified articles were analyzed and necessary data were extracted using a data extraction form. Many different terms are used in the articles when the combination of static and dynamic techniques is described (e.g., collaboration, combination, composition, synergy, integration). However, two major kinds of combination of static and dynamic QA techniques could be identified: **compilation** and **integration**.

In this regard, compilation means that different static and dynamic QA techniques are applied for a common goal but in isolation, without using input from one technique for the second one. 26 articles were found that were classified according to this category. Three sub-categories were identified. The first sub-category comprises the compilation of static and dynamic analyses, i.e., articles explicitly mentioning static and dynamic analyses used in the described combined approach were put into this sub-category. Second, approaches explicitly combining inspection (i.e., static QA) and testing (i.e., dynamic QA) techniques in a compiled manner comprise the second sub-category. Finally, other compilations of static and dynamic QA techniques represent the third sub-category.

In contrast, integration means that output from one QA technique is used as input for the second QA technique. Overall, 23 articles were put into this category. Two sub-categories were distinguished: first, the integration of static and dynamic analyses, and second, the integration of inspection and testing techniques.

Besides those two main categories, articles that focus on any other aspect of combination (e.g., support for selecting different QA techniques) were classified as misc, which covers two articles. Figure 2 presents the classification of the combinations of static and dynamic QA techniques, which is discussed in the following in detail.

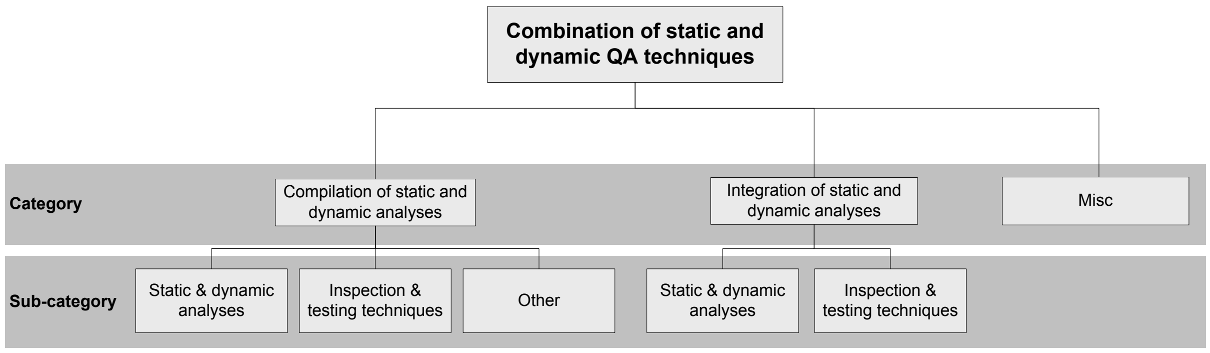
Combination of static and dynamic QA techniques
Category
Compilation of static and dynamic analyses
Integration of static and dynamic analyses
Misc
Sub-category
Static & dynamic analyses
Inspection & testing techniques
Other
Static & dynamic analyses
Inspection & testing techniques

Figure 2: Classification of the combination of static and dynamic QA techniques

3.1.1 Compilation

In the compilation category, three sub-categories were determined. In the first group, static and dynamic analyses are combined into novel approaches, focusing on thread escape analysis [35], atomicity analysis [33] protocol analysis [28], vulnerability analysis [42], concurrent program analysis [20], or on defects in general [5][66]. All approaches are tool-supported, either by tools (e.g., the open source tool ESC/Java2 [5]) or by proprietary tool prototypes (most of the approaches fall into this area, and tool prototypes are usually not made available).
The second sub-category compares different inspection and testing techniques and discusses advantages and disadvantages of both techniques. This is often substantiated by empirical studies and experiments that investigated which technique is superior to the other. However, in most cases the suggestion is made to combine inspection and testing techniques. So et al. [7] compared six different inspection and testing techniques, but most often, two or three techniques are compared to each other [8][9][31][63][6][10][25][23]. Four identified articles first performed inspections and one or more testing activities afterwards and compared the overall effect when inspection and testing techniques are applied in sequence [11][59][62][30].
The last sub-category describes other combinations. Inspection and testing techniques are combined with formal specifications, walkthroughs, or bug-finding tools [22][44]. Furthermore, comprehensive quality assurance processes from industrial environments are described, which comprises several inspection and testing techniques, requirements analyses, static analyses, walkthroughs, simulations, and tools [1][40][72]. Finally, Chen et al. [3] describe a view-based approach and combine this with inspection and testing techniques.

3.1.2 Integration

With respect to integration approaches, two sub-categories were defined. The integration of static and dynamic analyses encompasses most approaches. With respect to the order of application of static and dynamic analyses, most often static analysis is applied first, followed by dynamic analysis. However, some approaches use alternatives, for instance, dynamic analysis is used first and static analysis afterwards [51], static and dynamic analyses are performed in an iterative way [68], or dynamic analysis is used first, followed by static analysis and another dynamic analysis [26]. Static and dynamic analysis approaches grouped in this category focus on several vulnerability analyses [4][19][70][36][14][53], concurrent program analyses [68], defects in aspect-orientated programs [13], or on defects in general [43][16][18][50][17][15][26][51][67][14]. One approach additionally supports debugging [27]. A lot of different tools and algorithms support these analyses.
The second sub-category groups articles that integrate inspection and testing approaches. Three articles describe approaches that use different scenario-based reading techniques during the inspection to derive test cases that can be used during testing [12][38][45]. Furthermore, an approach is given where automatically generated test code is inspected [39]. Finally, Liu

presents how executing paths from testing can guide inspectors in checking the tested paths as well as additional ones for defects [41].

3.1.3 Misc

Two more articles could not be classified with respect to the first two groups, resulting in a "misc" category for those articles. Klaes et al. [46] propose a model that supports different QA management tasks, and various QA techniques (and even the combination of those) can be improved, planned, or controlled. Finally, Strooper and Wojcicki [54] present an approach that supports the selection of different QA techniques.

3.1.4 Summary

Based on 51 articles identified regarding the topic, compilation and integration were the two main approaches for the combination of static and dynamic QA techniques. Both categories comprise the combination of static and dynamic analyses and the combination of inspection and testing techniques. In addition, a mix of various static and dynamic QA techniques could also be identified for the compilation group. Finally, two more articles, categorized as misc approaches, were identified that could additionally support the combination of static and dynamic QA techniques. Figure 3 shows the number of articles, respectively approaches, per category.

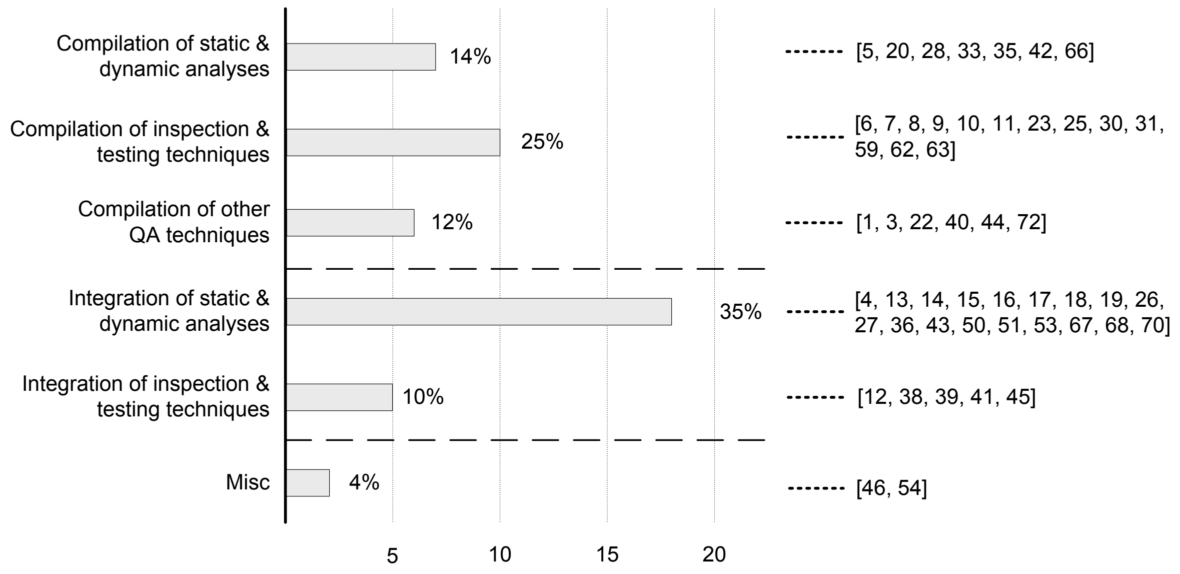

Figure 3: Number of articles per category and references

3.2 RQ2: In which sources and in which years were approaches regarding the combination of static and dynamic quality assurance techniques published?

3.2.1 Publication sources

Overall, four different publication channels could be identified. More than 80% (40 articles) of the articles were presented at conferences (26 articles), symposiums (10 articles), and workshops (4 articles). The remaining 17% (8 articles) of the articles were published in journals. Table 2 lists all sources, the different publication channels, and the number of articles per source. One observation is that a variety of different publication channels exist for publishing combined quality assurance approaches. Moreover, there is no source where more than 3 articles are published with respect to this research area. Overall, 45 concrete sources were identified. One conclusion is that the research topic covers different software areas and therefore, different publication channels seem to make sense. However, it also shows that the topic tends to be heterogeneous and that a lot of different, very small research communities exist that cover this topic.

Source	Publication channel	No.	%
International Conference on Software Engineering	Conference	3	5.9
Annual IEEE International Computer Software and Applications Conference	Conference	2	3.9
IEEE Software	Journal	2	3.9
International Symposium on Empirical Software Engineering and Measurement	Symposium	2	3.9
International Symposium on Software Reliability Engineering	Symposium	2	3.9
ACM Conference on Object-oriented Programming Systems Languages and Applications	Conference	1	2.0
Aerospace Software Engineering for Advanced Systems Architectures	Conference	1	2.0
Annual Computer Security Applications Conference	Conference	1	2.0
Annual Hawaii International Conference on System Sciences	Conference	1	2.0
Confederated International Conferences, CoopIS, DOA, IS, and ODBASE (OTM)	Conference	1	2.0
Euromicro Conference on Software Engineering and Advanced Applications	Conference	1	2.0
European Software Engineering Conference	Conference	1	2.0
European Software Engineering Conference and Symposium on the Foundations of Software Engineering	Conference	1	2.0
International Conference on Agile Processes in Software Engineering and Extreme Programming	Conference	1	2.0
International Conference on Biomedical Engineering and Informatics	Conference	1	2.0
International Conference on Computer Science and Software Engineering	Conference	1	2.0
International Conference on Engineering of Complex Computer Systems	Conference	1	2.0
International Conference on Formal Engineering Methods	Conference	1	2.0
International Conference on Fundamental Approaches to Software Engineering	Conference	1	2.0
International Conference on Quality Software	Conference	1	2.0
International Conference on Software Engineering and Knowledge Engineering	Conference	1	2.0
International Conference on Software Maintenance	Conference	1	2.0
International Conference on Testing of Communicating Systems	Conference	1	2.0
International Conference on Tests and Proofs	Conference	1	2.0
International Conference on Tools and Algorithms for the Construction and Analysis of System	Conference	1	2.0
International Conference on Product Focused Software Development and Process Improvement	Conference	1	2.0
ACM Transactions on Software Engineering and Methodology	Journal	1	2.0
Empirical Methods and Studies in Software Engineering	Journal	1	2.0
Hewlett-Packard Journal	Journal	1	2.0
IEEE Transactions on Software Engineering	Journal	1	2.0
Information and Software Technology	Journal	1	2.0
Information Systems Control Journal	Journal	1	2.0
Journal of Defense Software Engineering	Journal	1	2.0
Software Testing, Verification and Reliability	Journal	1	2.0
Annual ACM Symposium on Applied Computing	Symposium	1	2.0

IEEE Symposium on Security and Privacy	Symposium	1	2.0
International Software Metrics Symposium	Symposium	1	2.0
International Symposium on Empirical Software Engineering	Symposium	1	2.0
International Symposium on Web Site Evolution	Symposium	1	2.0
International Workshop on Abstract Interpretation of Object-oriented Languages	Workshop	1	2.0
International Workshop on Rapid Integration of Software Engineering Techniques	Workshop	1	2.0
NASA Software Engineering Workshop	Workshop	1	2.0
Workshop on Parallel and Distributed Systems: Testing, Analysis, and Debugging	Workshop	1	2.0
Workshop on Program Analysis for Software Tools and Engineering	Workshop	1	2.0

Table 2: Publication sources of identified articles

3.2.2 Publication years

Figure 4 presents the number of articles published per year from 1985 to 2010, which was the timescale for included articles. When analyzing the number of articles that were published in 5-year intervals, only between one and six articles were found to have been published in these timeframes until 2004. This shows that the focus regarding a combination of static and dynamic QA techniques was rather low during this time. However, between 2005 and 2009, 32 articles were published, which is twice as many as were published in the 20 years before. Consequently, this research topic has gained an increased attention during the past five years and seems to be a promising research area for the future. The decrease in 2010 can be explained by the point in time when this mapping study was performed and probably does not reflect the real number of articles published in 2010.

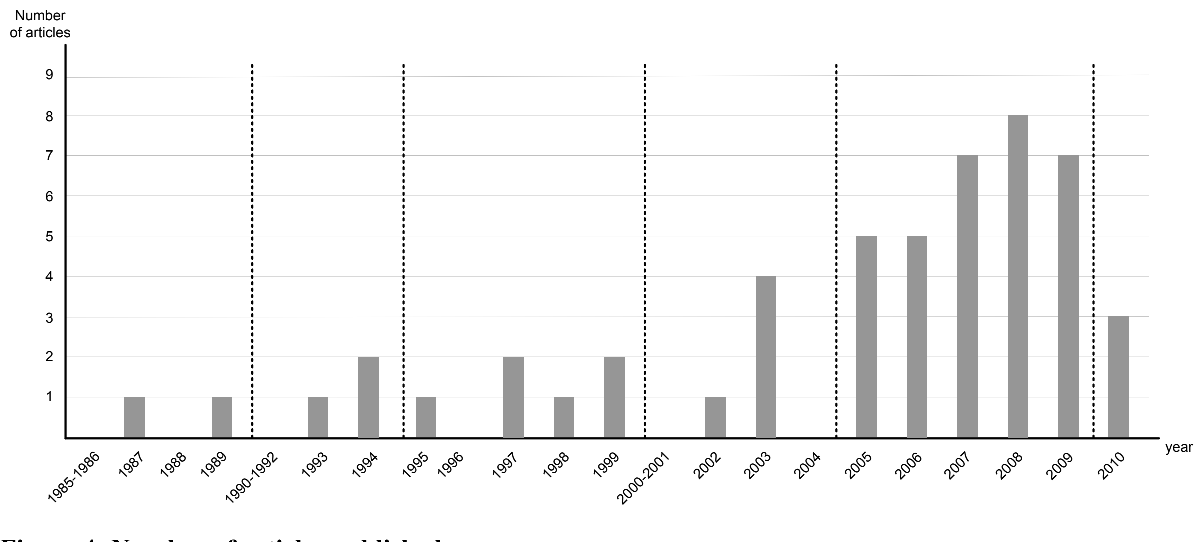

Figure 4: Number of articles published per year

3.2.3 Summary

A great diversity could be found when analyzing the publication channels of the 51 articles, i.e., a large number of different conferences, workshops, symposiums, and journals are used to publish combined approaches. No concrete source was used more than three times to publish articles on combined approaches. With respect to the publication year, a tremendous increase of published articles started in 2005. More than 30 articles were published in the years 2005 to 2009, which is twice as many compared to the 20 years before 2005. Hence, the interest in this topic has received much more attention in the last few years.

3.3 RQ3: Is any kind of evidence presented with respect to the combination of quality assurance techniques and if so, which kind of evidence is given?

3.3.1 Evaluated approaches versus non-evaluated approaches

In order to be able to answer research question three, a second kind of classification of the identified articles was prepared (Figure 5).

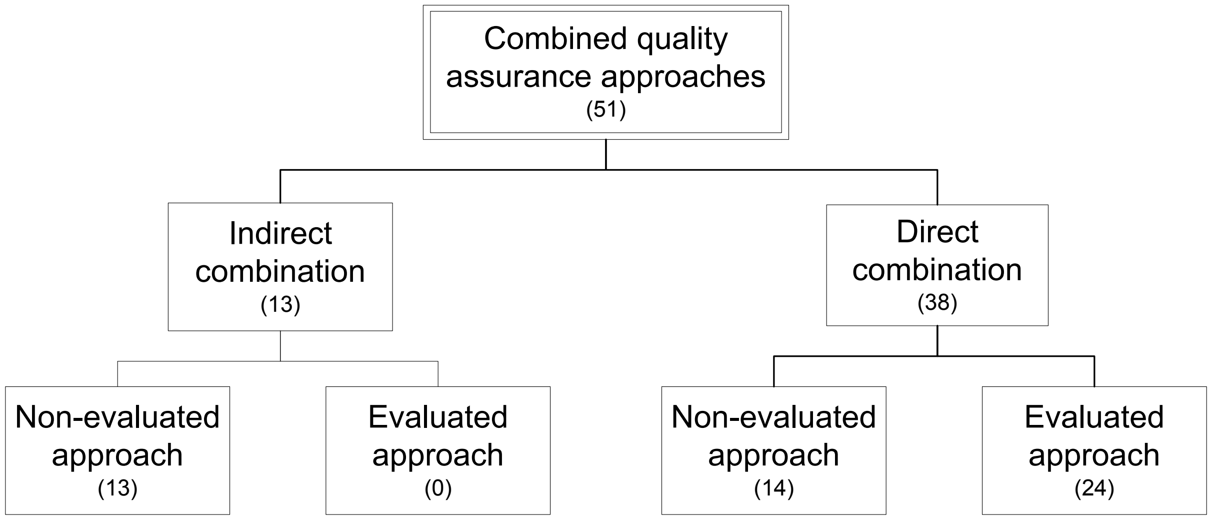
Combined quality assurance approaches (51)
Indirect combination (13)
Direct combination (38)
Non-evaluated approach (13)
Evaluated approach (0)
Non-evaluated approach (14)
Evaluated approach (24)

Figure 5: Number of articles that provide evidence, respectively provide no evidence

First, articles were grouped into one of the following two categories: **indirect combination** or **direct combination**. Indirect combination means that the approach supports the selection of different static and dynamic QA techniques or the article describes or analyzes different static and dynamic QA techniques. Based on this, the suggestion is made to combine these techniques. For example, certain inspection and testing techniques are described and compared with each other. Based on observations, such as inspection and testing techniques showing differences in effectiveness, the conclusion is made that it is most effective to combine them in a compiled manner. However, how this should be done and what the concrete benefit (e.g., in terms of effectiveness, efficiency, found defect types) of such a combination could be is neither described nor evaluated explicitly. Therefore, besides theoretical suggestions that more defects can be found when applying different static and dynamic QA techniques, no concrete evaluation is done regarding the combined application or additional synergy effects resulting from the combination.

In contrast, direct combination means that the combination of a static and a dynamic QA technique is explicitly described. Two possible kinds of articles grouped in this category exist. First, the approach only explains how the combination could be done, either in a compiled or an integrated manner. In this case, no evaluation is presented. Second, beside the description of the combined approach, an evaluation is presented, either in a quantitative or qualitative way. Three different kinds of empirical studies were identified: experiments comprise about 60 percent (14 articles), case studies about 30 percent (8 articles), and experiences about 10 percent (2 articles).

Finally, Figure 6 shows the distribution of evaluated and non-evaluated combined approaches over the past 25 years. Again, the numbers of articles per year with respect to the combination categories "indirect" and "direct" are shown. The first approaches found only gave some indirect ideas for combinations or proposed a concrete combination without giving any evidence. The first evaluations were given with respect to a combination of different inspection and testing techniques in 1997 and 1998. After 2004, the number of proposed approaches increased and about half of the proposed combined approaches per year were also evaluated. One possible explanation may be the source of the publication. Most of the articles identified from before 2005 were not published in journals or at conferences that require evaluations. This explanation also means that since 2004, the evaluation of combined approaches is of higher interest, which resulted in more evaluations.

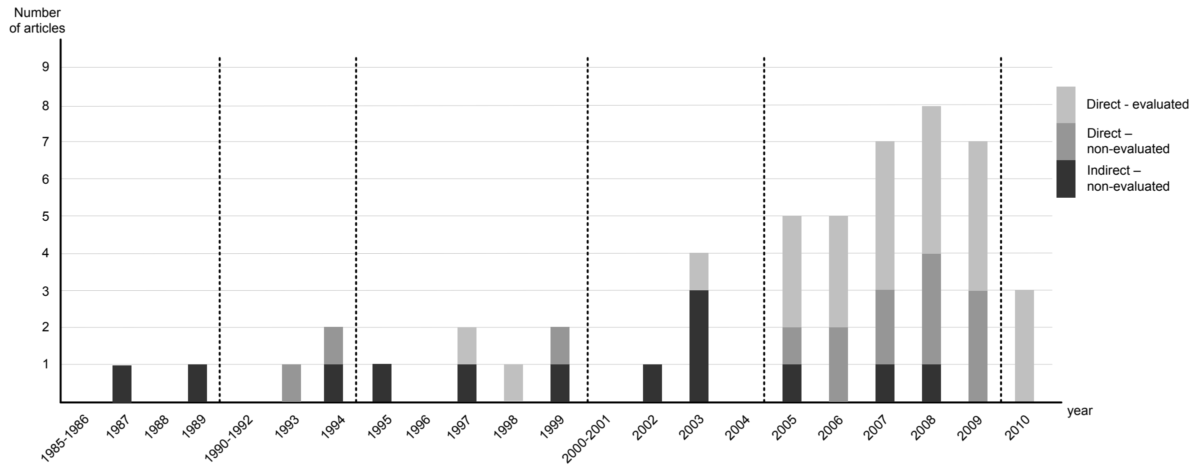

Figure 6: Numbers of evaluated and non-evaluated approaches per year

3.3.2 Summary

50% of the identified articles present evidence regarding combined approaches. The remaining articles describe just ideas on how a combination could be done or describe a combined approach concretely without giving any evidence. To some extent, this can be explained by the fact that many combined approaches have emerged in the past five years and thus, evaluation still has to be done. Furthermore, considering the publication channels, it may be that articles from relevant conferences and journals were not found and thus, certain evaluations were not incorporated into this analysis.

3.4 RQ4: What are the objectives of combined quality assurance approaches?

For answering research question four, only those articles were analyzed that were categorized as “direct combination” (see RQ3 in Section 3.3). The main reason is that the remaining ten articles, classified as “indirect combination”, mainly focus on comparing inspection and testing techniques, comparing inspection and testing techniques with tools, comparing different tools, or supporting the selection and managing of different QA techniques. However, no concrete combined approaches are presented except for some initial suggestions. The objectives of these suggestions remain rather generic, such as general improvement of quality, or are not mentioned explicitly. Therefore, those ten articles were excluded in the analysis of RQ4.

Consequently, a total of 38 articles were taken into account for answering this research question. First, objectives that target the quality of the corresponding QA process were considered. These objectives are either to improve the QA process by applying a combined approach or to introduce a combined approach in a new environment. Figure 7 presents an overview of the identified categories, the number of articles per category (absolute and relative), and the corresponding references. With respect to improvement, which is the most common objective, three different aspects were distinguished: improvement of coverage (13 articles), of effectiveness (23 articles), and of efficiency (11 articles). A lot of articles were classified into two or three categories, depending on which objectives should be achieved and which objectives were investigated with respect to the described approaches. Coverage was seen in two ways; first, as a measure of the proportion of a program being executed and tested by running a test suite (e.g., statement or branch coverage), and second, as coverage of

requirements or use-cases by executing test cases. Effectiveness is the ratio of total number of identified defects and total number of existing defects in a QA artifact. Finally, efficiency or cost effectiveness refers to the number of defects found per period of time. About 80 percent of the corresponding approaches aim at achieving improvement objectives. In general, different approaches were found that describe how the improvements are achieved. For instance, by combining complementary static and dynamic QA techniques, different kinds of defects are found and consequently, effectiveness is improved. Another idea is that the output of a static analysis points the dynamic analysis to certain parts that can improve effectiveness and efficiency. However, the inspection results are not used to focus testing activities.
Eight more articles were classified into feasibility studies aimed at investigating whether a combined approach is able to detect defects in new environments, such as aspect-oriented software systems or web-based applications.
A second kind of objective of combined approaches are the defect types they address in order to achieve certain product quality objectives such as reliability, security, or safety. 23 articles do not mention a certain defect type, but the described approaches focus on finding defects in general, i.e., their objective is to improve the general reliability of the software product. 13 articles explicitly mention a concrete defect type that the combined approach focuses on, with security defects being the most common group. Finally, three more articles are classified as misc, covering not the quality of the final product, but the quality of intermediate artifacts such as design or test code. Figure 8 summarizes the articles with respect to the product quality objectives.

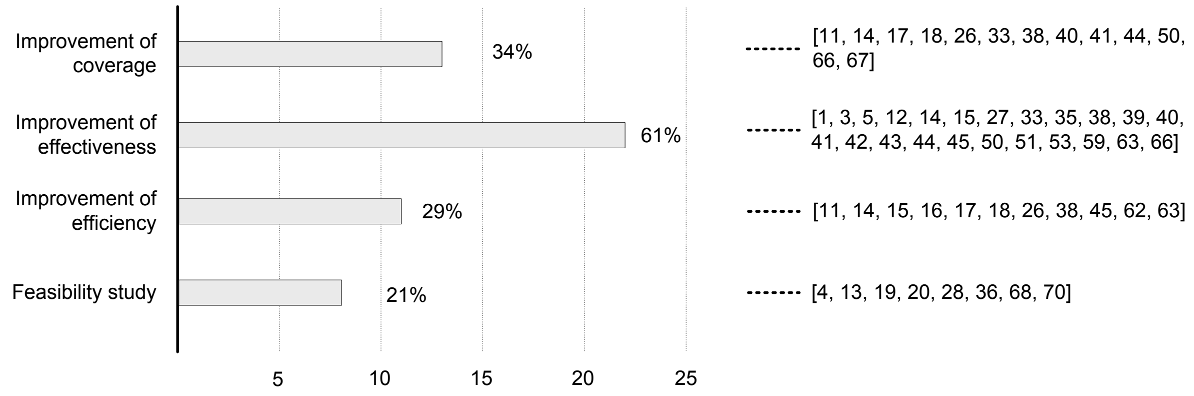

Figure 7: Numbers of articles with respect to quality assurance process objectives

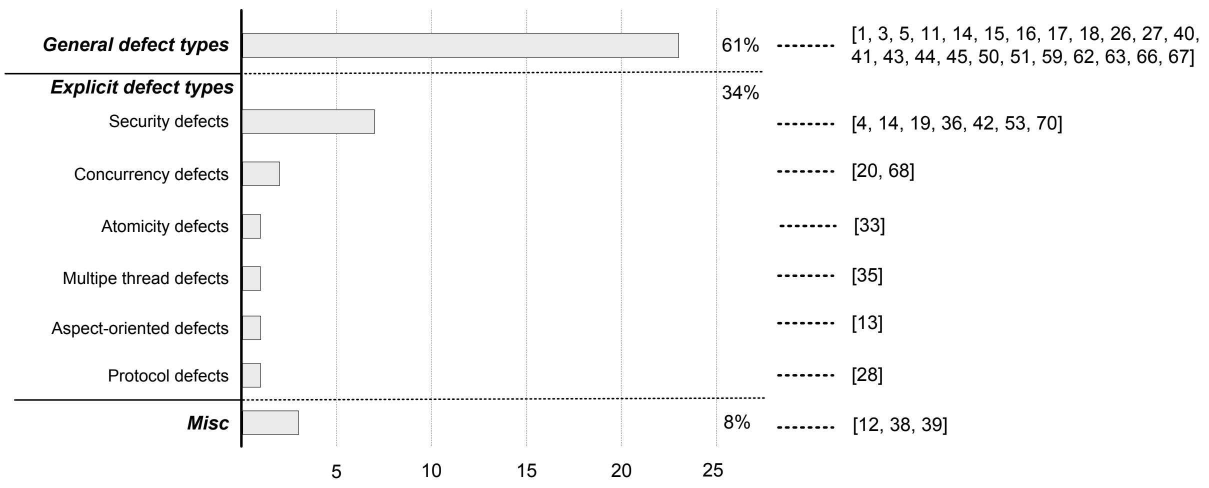

Figure 8: Numbers of articles with respect to defect types addressed by the combined approaches

In summary, two kinds of objectives could be extracted from the identified articles. On the one hand, there are QA process quality objectives, where an improvement of effectiveness is the main goal to be achieved when applying a combined approach, followed by coverage and efficiency improvement. This could be achieved by using input from a static analysis to focus the dynamic analysis or by deriving test cases for testing during an inspection, for instance. However, inspection results are not used to focus testing activities or to prioritize certain parts of the system for testing. In most of the articles, it is stated that a combination is worthwhile and resulted in additional benefits with respect to the objectives. However, concrete improvement numbers and results from evaluations are hardly presented.
On the other hand, product quality objectives were identified. More than 60% of the approaches do not focus on a certain defect type and thus, concentrate on improving reliability objectives. Furthermore, around one third of the approaches focus on certain defect types and thus, on certain product quality objectives. The remaining three articles focus on intermediate product quality aspects.

3.5 RQ5: Which static and dynamic quality assurance techniques are used in combined quality assurance approaches?

As already seen in the classification of static and dynamic QA techniques, two main types of combinations were identified, which is also reflected in the results of RQ5.
First, approaches that combine inspection and testing techniques were analyzed. The main results are shown in Figure 9. Usually no concrete inspection technique is mentioned explicitly (e.g., Fagan inspection, review, peer desk). Some articles mention the reading technique used, for instance, checklist-based reading, perspective-based reading, or usage-based reading. However, instead of mentioning the concrete inspection technique, the phase an inspection is applied to (respectively the artifact that is checked) is mentioned, such as requirements, design, or source code. With respect to testing techniques, structural (i.e., white-box) and functional (i.e., black-box) testing techniques can be distinguished. Most often, functional testing techniques are compared or combined with different inspection techniques. Most often, source code inspections were compared with structural or functional testing techniques. Furthermore, holistic approaches were identified that combine structural, functional, and other testing techniques with different inspection techniques, but not necessarily all of them. Finally, one article describing produced test code which is then inspected, one article that compares testing and inspection with tools, and one article that compares design inspections with testing in general were found and classified as miscellaneous. An overview is given in Figure 9.

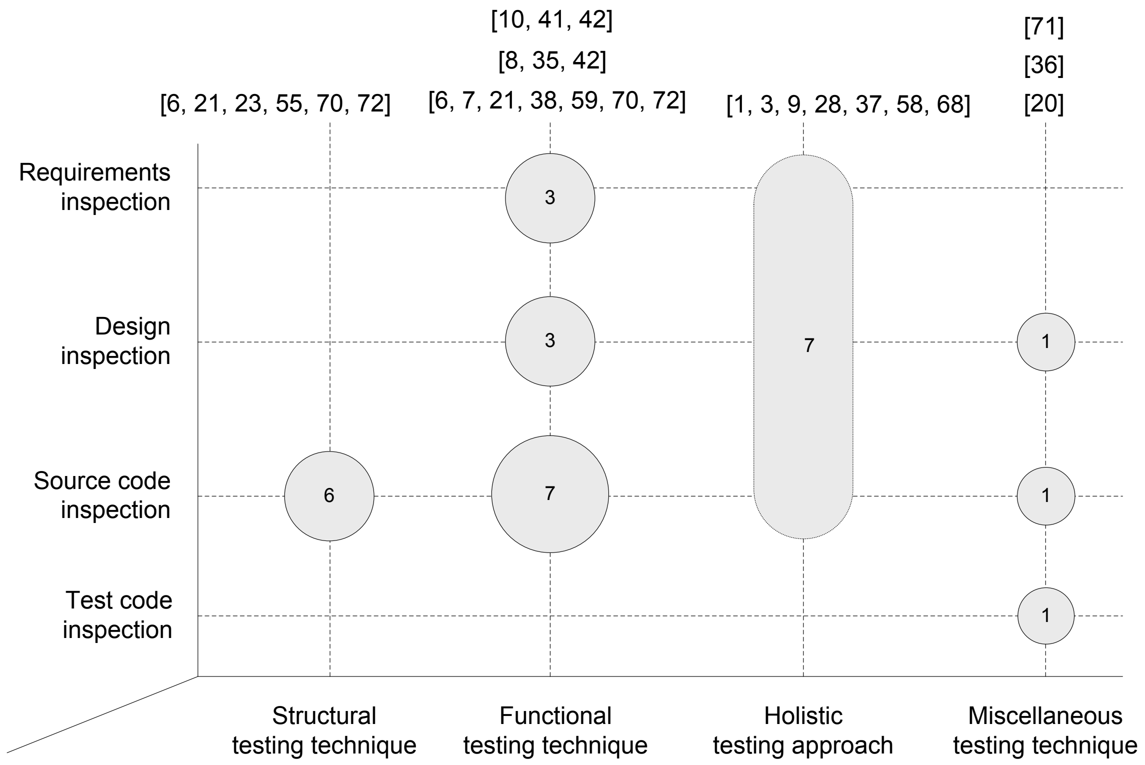

Figure 9: Overview of inspection and testing techniques that are applied in combination

Second, approaches that combine static and dynamic analysis are analyzed with respect to the concrete techniques used. Table 3 summarizes techniques applied in the combined approaches as mentioned in the corresponding articles. The main observation is that most approaches use very specialized techniques to address the corresponding objectives, domains, artifacts, and restrictions. Regarding the description of the concrete techniques, the level of detail differs, i.e., some approaches present very detailed information about the techniques applied while other approaches only cover the techniques in general.

No.	Used techniques	Ref.
1.	Static and dynamic analyses including a static vulnerability revealer, a goal-path-oriented system, and a dynamic vulnerability detector	[4]
2.	Static and dynamic analysis using a static checker, a runtime assertion checker, and a specification-based unit test generator	[5]
3.	Static analysis generating test sequences based on dynamic interactions and dynamic analysis verifying the execution of the selected sequences	[13]
4.	Static program analysis by symbolic execution, testing and runtime instrumentation, model checking focusing on systematic state-space exploration and automated constraint solving	[14]
5.	Static analysis performing theorem proving techniques, deriving specific error conditions using a constraint solver, and deriving test cases	[15]
6.	Static and dynamic analysis using generic algorithms	[16]
7.	Static analysis using model-checking and dynamic analysis using model-based testing	[17]
8.	Static analysis and dynamic analysis using structural program testing	[18]
9.	Static analysis focusing on native methods, reflection, and multi-threaded code followed by dynamic analysis of identified parts	[19]
10.	Static analysis of an abstract model using compositional reachability analysis and dynamic analysis of source code with temporal analysis of	[20]

	runtime traces	
11.	Dynamic invariant detection, static path exploration, and automatic test-case generation	[26]
12.	Static change impact analysis, dynamic test execution information extraction and three phase delta debugging algorithm	[27]
13.	Static analysis checking protocol invariants and runtime analysis	[28]
14.	Static analysis checking synchronizations and accesses to shared variables and dynamic analysis of run-time values using a monitor	[33]
15.	Static and dynamic thread escape analysis using static summary trees and a monitor	[35]
16.	Dynamic analysis through instrumentation of source code to extract the control flow of a program, followed by a static analysis using slicing and constraint bound checking	[36]
17.	Static analysis using a constraint solver and dynamic analysis	[42]
18.	Web application static and dynamic analysis	[43]
19.	Static analysis using symbolic analysis and automated theorem proving, followed by dynamic analysis	[50]
20.	Static and dynamic analysis	[51]
21.	Sanitization-aware static analysis and testing sanitization routines	[53]
22.	Static analysis and dynamic analysis (i.e., testing) in general	[66]
23.	Simulation, symbolic execution, theorem prover, and static analysis	[67]
24.	Static analysis using value schedule generator and additional components (e.g., cfg, mhp), dynamic analysis using a model checker	[68]
25.	Static and dynamic analysis using taint analysis and genetic algorithms	[70]

Table 3: Overview of static and dynamic analyses that are applied in combination

The two articles classified as misc are not listed here because they only support the QA selection, but do not present concrete QA techniques. In summary, a lot of different static and dynamic QA techniques are combined, both as inspection and testing combinations and as combinations of static and dynamic analyses. This shows the high potential of combining different QA techniques.

3.6 RQ6: Which input is used for static and dynamic quality assurance techniques in combined quality assurance approaches?

For answering research question six, only those articles were analyzed that were categorized as “direct combination” (see RQ3 in Section 3.3) and that provide clear information on which input is necessary instead of giving only ideas or suggestions. Consequently, 38 articles were considered.

With respect to input for static QA techniques, source code is mostly used due to the fact that both inspections and general static analyses can use source code. Further kinds of input are requirements (including formal specifications), design, test code, respectively test path information, or additional ones such as binary code or byte code. Figure 10 presents an overview.

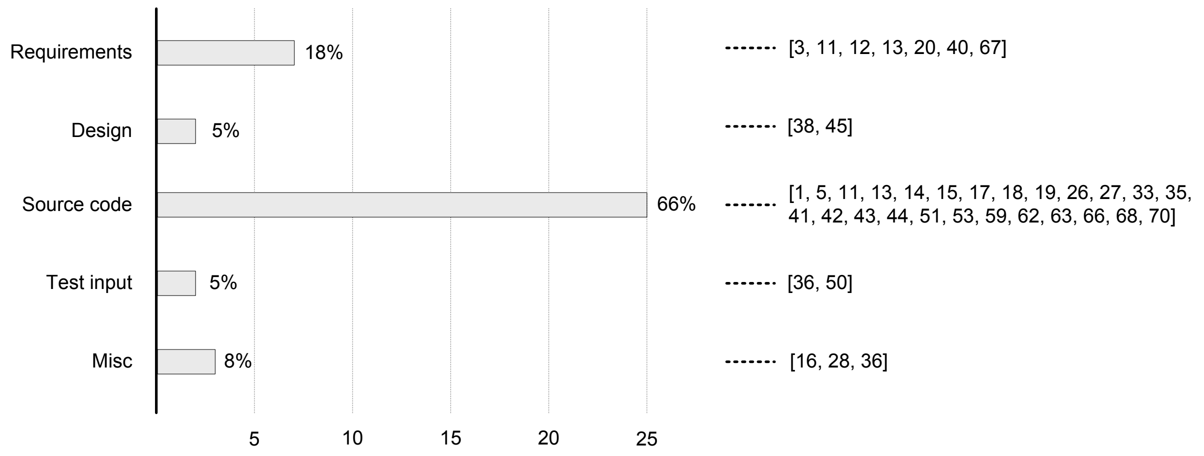

Figure 10: Input for static QA techniques

Regarding the input for dynamic QA techniques, usually only source code is necessary. In addition, output from static QA techniques is sometimes considered, for instance, for focusing the dynamic QA technique (e.g., based on a flow graph [19]).
In summary, in combined approaches, source code is mostly used as input for static and dynamic QA techniques. With respect to static QA techniques, some additional kinds of input exist, such as requirements or design. Furthermore, the output from static QA techniques is sometimes considered as input for the dynamic QA technique.

3.7 Threats to validity

A common fact with respect to the result of a systematic mapping study is that it is affected by the researchers conducting the review, by the databases selected, by the selected search term and the chosen timeframe. Therefore, four different kinds of threats to validity are discussed next.
Conclusion validity: In order to present results that are reasonable and can be reproduced by other research groups, detailed information regarding the concrete search term and the databases used is presented. Furthermore, inclusion and exclusion criteria are presented. However, the decision on which articles to include in the final result set also depends on the concrete researchers conducting the systematic mapping study. The exclusion/inclusion steps were conducted by two researchers independently. A high level of agreement was achieved (the result of Cohen's Kappa value was about 0.65), which reduces this threat to validity. If the researchers disagreed on including or excluding an article, this was discussed until agreement was obtained. Though a replication of this systematic mapping study might lead to a slightly different set of primary studies, the main conclusions drawn from the identified set of articles might not differ much.
Construct validity: In order to obtain as complete a set of primary studies covering the given research topic as possible, the search term was derived systematically. Different terms for static and dynamic quality assurance techniques were determined, which is also true for alternative terms for the term combination. However, such a list might not be complete and additional or alternative terms might have affected the number of articles found. Moreover, only two criteria (population, intervention) of the recommended PICO structure were determined to be relevant in our context, and thus considered.
Internal validity: The analysis of articles found during the cross-check was performed by one researcher only, which may have resulted in missing articles. However, the initial selection of articles based on the stated qualitative criteria was performed by two researchers in order to find the final result set. A high conformance value was achieved, which indicates a similar understanding of relevance, and thus, the threat may be reduced.

External validity: The results of the systematic mapping study were considered with respect to approaches in the software domain. Thus, the classification presented and the conclusions drawn are only valid in the given context. Furthermore, if reference lists contained in identified articles had been investigated, additional articles might have been found. However, it is expected that most such articles describe, for example, other aspects of the already found approaches, additional evaluations of the approach, or the approach applied in a different context. Therefore, this classification can serve as a starting point. Additional articles and approaches that are identified in the future can be categorized accordingly. Due to the systematic procedure followed during the mapping study, the general conclusions with respect to the defined research questions may be true independently of this threat.

4. Discussion and implications

4.1 General findings

Based on the results of the systematic mapping study, one main observation with respect to the combination of static and dynamic QA techniques is that this research area has gained increased attention since 2005. Some articles published before suggested combining, for instance, inspection and testing techniques in order to improve overall quality assurance. Nevertheless, concrete combination approaches were mainly published during the last few years.

The combination of static and dynamic analyses is one of the most common approaches. Lots of different tools or algorithms are presented for different environments, which shows the heterogeneity of combining static and dynamic analyses. Moreover, this kind of combination is usually conducted in an integrated manner.

With respect to the combination of inspection and testing techniques, which is another common approach for combining static and dynamic QA techniques, this is done more often in a compiled way than in an integrated way. Different articles suggest applying both QA techniques in order to find more defects, but no information is given on the expected synergy effects. The approaches integrating inspection and testing techniques mainly support test case derivation. However, inspections are not used to focus testing activities, which might lead to additional benefits regarding the overall quality assurance process.

Other combinations were also identified. They usually propose using a multitude of available static and dynamic QA techniques in a compiled way to improve defect detection. Finally, some approaches were found that could not be classified with respect to the categories of integration and compilation. They mainly support the combination, for instance, by presenting advice on how to select an appropriate set of QA techniques. However, supporting approaches were not the major focus of this mapping study and probably more such approaches exist.

Different objectives were in focus when combining static and dynamic QA techniques, most often the improvement of effectiveness, efficiency, or coverage criteria. Furthermore, some combined approaches were investigated with respect to their feasibility in new environments, showing that the combined approaches are able to address the challenges of new environments.

Finally, the conclusion of the secondary study identified [73], namely, that no clear winner regarding inspection and testing exists, could be confirmed. Additional effort was invested to combine and integrate different static and dynamic quality assurance techniques. Furthermore, effectiveness and efficiency are still an important objective when those quality assurance techniques are applied. However, while the identified secondary study [73] only focused on comparing inspection and testing techniques, our systematic mapping study has (i) a much broader scope, covering additional static and dynamic quality assurance activities, and (ii) focuses not only on articles comparing these activities, but also on approaches explicitly combining static and dynamic activities.

4.2 Implications for practitioners

From a practitioner's point of view, the results presented in this study provide an overview of existing approaches to combining static and dynamic QA techniques and of the objectives that are achieved with the existing combinations. Furthermore, experiences when applying the approaches were presented and can give an overview of the maturity of the approaches. Finally, if practitioners want to exploit a combined approach in order to improve their quality assurance, they can choose one based on these results. However, in this case, it is important to consider the concrete context and adaptations may be necessary.

4.3 Implications for researchers

As mentioned in Section 2, the objectives of this review were to give an overview of existing combination approaches in order to identify existing gaps and to establish future research directions. First of all, new combinations may be defined in order to improve overall quality assurance. Furthermore, new contexts have to be considered with combined approaches (e.g., new domains, new applications, and new development environments) and must probably be adapted to them. Third, missing evidence regarding existing approaches may lead to more empirical studies, which includes comparisons of combined approaches and approaches not combining quality assurance techniques. Finally, new purposes for combined approaches can be determined, e.g., more precise prediction of quality.

5. Conclusion

In this article, the results of a systematic mapping study regarding the combination of static and dynamic quality assurance techniques were presented. Six detailed research questions were answered. The main results are as follows:

- RQ1. 51 identified articles were mainly classified into two groups of combination approaches, namely compilation and integration approaches. Both groups contain the same number of articles.
- RQ2. The topic has gained increasing interest since 2005, with some articles describing mainly general combination ideas before. A large number of different publication sources exist.
- RQ3. Of the 51 articles, almost 50% present evidence regarding the approaches proposed. The remaining 53% either present some indirect ideas for a combination or present a concrete combination approach, but without presenting any evidence.
- RQ4. With respect to QA process quality objectives, the approaches focus on the improvement of effectiveness, coverage, and efficiency. In addition, performing feasibility studies of combined approaches in order to investigate combined approaches in new environments is another objective. With respect to ensuring product quality objectives, about 50% of the approaches do not focus on a certain defect type, while about one third of the approaches explicitly do focus on certain defect types.
- RQ5. A variety of different static and dynamic analyses and inspection and testing techniques are used in the combined approaches, which demonstrates the great heterogeneity and different environmental requirements that the QA techniques have to face.
- RQ6. Most often, the combination of static and dynamic QA techniques is applied on the code level. Static QA techniques can also be applied to requirements, design, or test code. With respect to the input for dynamic QA techniques, source code is used most often, enhanced sometimes by output from static QA techniques.

From our point of view, the major result of this systematic mapping study is the identification and classification of existing approaches that combine static and dynamic QA techniques.

This may support practitioners in selecting worthwhile combinations and serve as a basis for researchers in terms of promising future research directions. Besides the general future research directions as mentioned in Section 4, we are most interested in a close combination of inspection and testing techniques, i.e., an integration of inspection and testing techniques, in order to exploit additional synergy effects. One concrete idea is to define an approach that, based on inspection results, supports the focusing of testing activities. This could be done by prioritizing system parts (e.g., code classes) that are expected to contain more defects or by prioritizing defect types that are particularly relevant. Such an initial approach was already presented by Elberzhager et al. [47][69][71] and showed promising results with respect to effort reduction. However, additional questions emerged and a refinement of this approach and additional evaluation are planned.

Acknowledgments
Parts of this work have been funded by the Stiftung Rheinland-Pfalz für Innovation project "Qualitäts-KIT" (grant: 925). We would like to thank Sonnhild Namingha for proofreading.

References

[1] T.F. Chang, A. Danylyzsn, S. Norimatsu, J. Rivera, D. Shepard, A. Lattanze, J. Tomayko, "Continuous verification" in mission critical software development, Proceedings of the Thirtieth Hawaii International Conference on System Sciences, 1997, pp. 273-284.

[2] A. Endres, D. Rombach, A Handbook of Software and Systems Engineering, Addison Wesley, 2003.

[3] Y. Chen, S. Liu, W.E Wong, A method combining review and testing for verifying software systems, Proceedings of the 2008 International Conference on BioMedical Engineering and Informatics, 2008, pp. 827-831.

[4] A. Hanna, H.Z. Ling, X. Yang, M. Debbabi, A synergy between static and dynamic analysis for the detection of software security vulnerabilities. Proceedings of the Confederated International Conferences, CoopIS, DOA, IS, and ODBASE 2009 on On the Move to Meaningful Internet Systems: Part II, 2009, pp. 815-832.

[5] D.M. Zimmerman, J.R. Kiniry, A verification-centric software development process for Java. Proceedings of the 9th International Conference on Quality Software, 2009, pp. 76-85.

[6] M. Roper, M. Wood, J. Miller, An empirical evaluation of defect detection techniques, Information and Software Technology 39 (11) (1997) pp. 763-775.

[7] S.S. So, S.D. Cha, T.J. Shimeall, Y.R. Kwon, An empirical evaluation of six methods to detect faults in software, Software Testing, Verification and Reliability 12 (3) (2002) pp. 155-171.

[8] E. Kamsties, C.M. Lott, An empirical evaluation of three defect-detection techniques, 5th European Software Engineering Conference, 1995, pp. 362-383.

[9] R. Conradi, A.S. Marjara, B. Skatevik, An empirical study of inspection and testing data at Ericsson, Norway, 24th NASA Software Engineering Workshop, 1999, pp. 1-7.

[10] C. Andersson, T. Thelin, P. Runeson, N. Dzamashvili, An experimental evaluation of inspection and testing for detection of design faults. Proceedings of the 2003 International Symposium on Empirical Software Engineering, 2003, pp. 174-184.

[11] T. Berling, T. Thelin, An industrial case study of the verification and validation activities. Proceedings of the 9th International Symposium on Software Metrics, 2003, pp. 226-238.

[12] T.Y. Chen, P.L. Poon, S.F. Tang, T.H. Tse, Y.T. Yu, Applying testing to requirements inspection for software quality assurance. Information Systems Control Journal 6 (2006) pp. 50-56.

[13] P. Massicotte, L. Badri, M. Badri, Aspects-classes integration testing strategy: an incremental approach, Rapid Integration of Software Engineering Techniques 3943 (2006) pp. 158-173.

[14] P. Godefroid, P. de Halleux, A.V. Nori, S.K. Rajamani, W. Schulte, N. Tillmann, M.Y. Levin, Automating software testing using program analysis. IEEE Software 25 (5) (2008) pp. 30-37.

[15] C. Csallner, Y. Smaragdakis, Check 'n' crash: combining static checking and testing, Proceedings of the 27th International Conference on Software engineering, 2005, pp. 422-431.

[16] C. Artho, A. Biere, Combined static and dynamic analysis, Electronic Notes in Theoretical Computer Science 131 (2005) pp. 3-14.

[17] J. Chen, H. Zhou, S.D. Bruda, Combining model checking and testing for software analysis, Proceedings of the 2008 International Conference on Computer Science and Software Engineering, 2008, pp. 206-209.

[18] O. Chebaro, N. Kosmatov, A. Giorgetti, J. Julliand, Combining static analysis and test generation for C program debugging. Tests and Proofs 6143 (2010) pp. 94-100.

[19] P. Centonze, R. Flynn, M. Pistoia, Combining static and dynamic analysis for automatic identification of precise access-control policies, Proceedings of the 23rd Annual Computer Security Applications Conference, 2007, pp. 292-303.

[20] F.D. Anger, R.V. Rodriguez, M. Young, Combining static and dynamic analysis of concurrent programs, Proceedings of the International Conference on Software Maintenance, 1994, pp. 89-98.

[21] M. Wood, M. Roper, A. Brooks, J. Miller, Comparing and combining software defect detection techniques - a replicated empirical study, 6th European Software Engineering Conference, 1997, pp. 262-277.

[22] S. Wagner, J. Jürjens, C. Koller, P. Trischberger, Comparing bug finding tools with reviews and test, Proceedings of the 17th International Conference on Testing of Communicating Systems, 2005, pp. 40-55.

[23] V.R. Basili, R.W. Selby, Comparing the effectiveness of software testing strategies, IEEE Transactions on Software Engineering 13 (12) (1987) pp. 1278-1296.

[24] M.E. Fagan, Design and code inspections to reduce errors in program development. IBM Systems Journal 15 (3) (1976) pp. 182-211.

[25] P. Runeson, A. Andrews, Detection or isolation of defects? An experimental comparison of unit testing and code inspection, Proceedings of the 14th International Symposium on Software Reliability Engineering, 2003, pp. 3-13.

[26] C. Csallner, Y. Smaragdakis, T. Xie, DSD-Crasher: A hybrid analysis tool for bug finding, ACM Transactions on Software Engineering and Methodology 17 (2) (2008), pp. 1-37.

[27] S. Zhang, Y. Lin, Z. Gu, J. Zhao, Effective identification of failure-inducing changes: a hybrid approach. Proceedings of the 8th ACM SIGPLAN-SIGSOFT workshop on Program analysis for software tools and engineering, 2008, pp. 77-83.

[28] M. Gopinathan, S.K. Rajamani, Enforcing object protocols by combining static and runtime analysis. Proceedings of the 23rd ACM SIGPLAN conference on Object-oriented programming systems languages and applications, 2008, pp. 245-260.

[29] Engineering Village. <http://www.engineeringvillage2.org>.

[30] L. Franz, J. Shih, Estimating the value of inspections and early testing for software projects, Hewlett-Packard Journal 45 (1994), pp. 60-67.

[31] N. Juristo, S. Vegas, Functional testing, structural testing, and code reading: what fault type do they each detect? Empirical Methods and Studies in Software Engineering, 2003, pp. 208–232.

[32] B.A. Kitchenham, S. Charters, Guidelines for Performing Systematic Literature Reviews in Software Engineering, Technical Report EBSE-2007-01, Keele University and University of Durham, 2007.

[33] Q. Chen, L. Wang, Z. Yang, S.D. Stoller, HAVE: detecting atomicity violations via integrated dynamic and static analysis, Proceedings of the 12th International Conference on Fundamental Approaches to Software Engineering: Held as Part of the Joint European Conferences on Theory and Practice of Software, 2009, pp. 425-439.

[34] Health, Social, and Economic Research, The economic impacts of inadequate infrastructure for software testing, National Institute of Standards and Technology, 2002.

[35] Q. Chen, L. Wang, Z. Yang, HEAT: an integrated static and dynamic approach for thread escape analysis. Proceedings of the 33rd Annual IEEE International Computer Software and Applications Conference, 2009, pp. 142-147.

[36] P.D. Kumar, A. Nema, R. Kumar, Hybrid analysis of executables to detect security vulnerabilities: security vulnerabilities. Proceedings of the 2nd India software engineering conference, 2009, pp. 141-148.

[37] IEEE Standard 610.12-1990. IEEE Standard Glossary of Software Engineering Terminology, 1990.

[38] D. Winkler, B. Riedl, S. Biffl, Improvement of design specifications with inspection and testing, Proceedings of the 31st Euromicro Conference on Software Engineering and Advanced Applications, 2005, pp. 222-231.

[39] F. Lanubile, T. Mallardo, Inspecting automated test code: a preliminary study, Proceedings of the 8th international conference on Agile processes in software engineering and extreme programming, 2007, pp. 115-122.

[40] N. Ward, Integrated formal verification and validation of safety critical software. Proceedings of the Aerospace Software Engineering for Advanced Systems Architectures Conference, 1993, pp. 10-13.

[41] S. Liu, Integrating specification-based review and testing for detecting errors in programs. Proceedings of the 9th international conference on formal methods and software engineering, 2007, pp. 136-150.

[42] A. Aggarwal, P. Jalote, Integrating static and dynamic analysis for detecting vulnerabilities, 30th Annual International Conference on Computer Software and Applications, 2006, pp. 343-350.

[43] G.A.D. Lucca, M.D. Penta, Integrating static and dynamic analysis to improve the comprehension of existing web applications, Proceedings of the 7th IEEE International Symposium on Web Site Evolution, 2005, pp. 87-94.

[44] S. Liu, T. Tamai, S. Nakajima, Integration of formal specification, review, and testing for software component quality assurance. Proceedings of the 2009 ACM symposium on Applied Computing, 2009, pp. 415-421.

[45] D. Winkler, S. Biffl, K. Faderl, Investigating the temporal behavior of defect detection in software inspection and inspection-based testing. Product-Focused Software Process Improvement 6156 (2010), pp. 17-31.

[46] M. Klaes, F. Elberzhager, H. Nakao, Managing software quality through a hybrid defect content and effectiveness model. Proceedings of the 2nd ACM-IEEE international symposium on Empirical software engineering and measurement, 2008, pp. 321-323.

[47] F. Elberzhager, J. Muench, D. Rombach, B. Freimut, Optimizing cost and quality by integrating inspection and test processes, International Conference on Software and Systems Process, pp. 3-12, 2011.

[48] K.E Wiegers, Peer Reviews in Software, Addison-Wesley, 2002.

[49] I. Burnstein, Practical Software Testing, Springer, 2002.

[50] P. Joshi, K. Sen, M. Shlimovich, Predictive testing: amplifying the effectiveness of software testing, Proceedings of the 6th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on the foundations of software engineering, 2007, pp. 561-564.

[51] P. Jalote, V. Vangala, T. Singh, P. Jain, Program partitioning: a framework for combining static and dynamic analysis. Proceedings of the 2006 international workshop on Dynamic systems analysis, 2006, pp. 11-16.

[52] N. Juristo, A.M. Moreno, S. Vegas, Reviewing 25 years of testing technique experiments, Empirical Software Engineering 9 (1-2) (2004) pp. 7-44.

[53] D. Balzarotti, M. Cova, V. Felmetsger, N. Jovanovic, E. Kirda, C. Kruegel, G. Vigna, Saner: composing static and dynamic analysis to validate sanitization in web applications. Proceedings of the 2008 IEEE Symposium on Security and Privacy, 2008, pp. 387-401.

[54] P. Strooper, M. Wojcicki, Selecting V&V technology combinations: how to pick a winner? Proceedings of the 12th International Conference on Engineering Complex Computer Systems, 2007, pp. 87-96.

[55] R. Pressman, Software engineering: a practitioner's approach, 5th ed., McGraw-Hill, London, 2000.

[56] T. Gilb, D. Graham, Software Inspections, Addison-Wesley, 1993.

[57] C. Jones, Software project management practices: failure versus success, Crosstalk - the journal of defense software engineering 17 (10) (2004) pp. 5-9.

[58] A. Aurum, H. Petersson, C. Wohlin, State-of-the-art: software inspections after 25 years, Software Testing, Verification and Reliability 12 (3) (2002) pp. 133-154.

[59] O. Laitenberger, Studying the effects of code inspection and structural testing on software quality, Proceedings of the 9th International Symposium on Software Reliability Engineering, 1998, pp. 237-246.

[60] S. Kollanus, J. Koskinen, Survey of software inspection research: 1991-2005, Computer Science and Information Systems reports, Working Papers WP-40, University of Jyväskylä, Finland, 2007.

[61] K. Petersen, R. Feldt, S. Mujtaba, M. Mattsson, Systematic mapping studies in software engineering, Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering, 2008, pp. 1-10.

[62] F. Iturbe, Systematic testing and reviewing. Proceedings of the 11th International Conference on Software Engineering and Knowledge Engineering, 1999, pp. 295-299.

[63] A. Gupta, P. Jalote, Test inspected unit or inspect unit tested code? Proceedings of the 1st International Symposium on Empirical Software Engineering and Measurement, 2007, pp. 51-60.

[64] M.J. Harrold, Testing: A roadmap, International Conference on Software Engineering, The Future of Software Engineering, 2000, pp. 61-72.

[65] J.R. Landis, G.G. Koch, The measurement of observer agreement for categorical data. Biometrics 33 (1) (1977), pp. 159-174.

[66] P. Anderson, The use and limitations of static-analysis tools to improve software quality. CrossTalk: The Journal of Defense Software Engineering 21 (6) (2008) pp. 18-21.

[67] A.V. Nori, S.K. Rajamani, S. Tetali, A.V. Thakur, The Yogi project: software property checking via static analysis and testing. Proceedings of the 15th International Conference on Tools and Algorithms for the Construction and Analysis of Systems: Held as Part of the Joint European Conferences on Theory and Practice of Software, 2009, pp. 178-181.

[68] J. Chen, S. MacDonald, Towards a better collaboration of static and dynamic analyses for testing concurrent programs. Proceedings of the 6th workshop on Parallel and distributed systems: testing, analysis, and debugging, 2008, pp. 1-9.

[69] F. Elberzhager, R. Eschbach, Towards reduction of test effort: predicting defect-prone code classes and expected defect types based on inspection results, 36th Euromicro Software Engineering and Advanced Application, Proceedings of the Work in Progress Session, 2010.

[70] A. Avancini, M. Ceccato, Towards security testing with taint analysis and genetic algorithms, Proceedings of the 2010 ICSE Workshop on Software Engineering for Secure Systems, 2010, pp. 65-71.

[71] F. Elberzhager, R. Eschbach, J. Muench, Using Inspection Results for Prioritizing Test Activities, 21st International Symposium on Software Reliability Engineering, Supplemental Proceedings, 2010, pp. 263-272, available: <http://inspection.iese.fhg.de/?p=documents>.

[72] E. Duke, V&V of flight and mission-critical software, IEEE Software 6 (3) (1989) pp. 39-45.

[73] P. Runeson, C. Andersson, T. Thelin, A. Andrews, T. Berling, What do we know about defect detection methods? IEEE Software 23 (3) (2006) pp. 82-90.

[74] Zotero tool <http://www.zotero.org/>.