The Journal of Systems and Software 84 (2011) 415–427

Contents lists available at ScienceDirect

The Journal of Systems and Software

journal homepage: www.elsevier.com/locate/jss

A systematic literature review of software quality cost research

Lars M. Karg [a], Michael Grottke [a,*], Arne Beckhaus [b]

[a] University of Erlangen-Nuremberg, Lange Gasse 20, D-90403 Nuernberg, Germany
[b] University of Freiburg, Germany

ARTICLE INFO

Article history:
Received 26 October 2009
Received in revised form 4 November 2010
Accepted 18 November 2010
Available online 9 December 2010

Keywords:
Prevention appraisal failure cost scheme
Software development
Systematic literature review
Quality costs

ABSTRACT

Software quality costs have not received as much attention from the research community as other economic aspects of software development. Over the last three decades, a number of articles on this topic have appeared in a range of journals, but comprehensive overviews of this body of research are not available.

For the detailed review of software quality cost research presented in this article, we collect 87 articles published between 1980 and 2009 in 60 leading computing journals. We study the distribution of these articles across research disciplines and journals as well as over time. Moreover, we identify the predominant researchers in the software quality cost domain and the related research clusters. We also classify the articles according to three properties, namely, research topic, research scope, and research approach. This categorization enables us to identify aspects emphasized by previous research on software quality costs and to point out promising future research directions. Our review shows that prevention costs have gained the least attention, in spite of their big cost impact. It also reveals that only one article has targeted multiple companies. Further, we observe that many articles do not empirically validate their findings. This is especially true for those articles dealing with an entire firm.

1. Introduction

For decades, users of software solutions have been suffering from poor solution quality (Whittaker and Voas, 2002). Over the years, quality has emerged to be a key issue in software development (Prahalad and Krishnan, 1999). Software vendors have attempted to tackle this issue by adapting concepts from other engineering disciplines, such as manufacturing (Antony and Fergusson, 2004). There, approaches ranging from Total Quality Management over Six Sigma and Kaizen to Lean Production have led to significant gains in productivity and quality. To attain similar results in software development, many of these concepts have been adapted and tailored to its characteristics (Middleton and Sutton, 2005). In this quest for higher productivity and quality, the economics of software engineering are of particular interest (Boehm, 1981). While some economic aspects, such as software development effort estimation (Jorgensen and Shepperd, 2007) and software process improvement (Hansen et al., 2004), have frequently been discussed, others have received less attention. Indeed, little research has specifically been devoted to those costs which are "incurred in the pursuit of [software] quality or in performing quality-related activities" (Pressman, 2010, p. 407). This is remarkable, because software vendors typically spend 30–50% of their development budget on defect detection and correction (Ebert and Dumke, 2010).

In most engineering disciplines, literature studies summarizing the latest research results on quality costs are regularly published (e.g., Williams et al., 1999; Schiffauerova and Thomson, 2006). To the best of our knowledge, this is not the case in software engineering; no review prior to ours has in particular been devoted to software quality cost research. However, several studies published in the broader field of software quality and software economics have also covered some quality cost aspects. For instance, the survey of software quality assurance research by Rai et al. (1998) considers software quality costs among other economic aspects, and Jorgensen and Shepperd (2007) systematically review work on software development effort estimation including approaches applicable to software quality cost estimation.

This article tries to close this research gap. It is exclusively devoted to software quality cost research. Our objective is to systematically review and structure the existing body of research on software quality costs and to identify areas for future research. We analyze 87 articles published between 1980 and 2009 in 60 leading computing journals for answering eight research questions. These research questions are directed at the research domain in general as well as at specifics of the existing research, regarding the software quality cost categories examined, the scope of investigation, and the research approaches employed.

* Corresponding author. Tel.: +49 911 5302 276; fax: +49 911 5302 277.
E-mail address: Michael.Grottke@wiso.uni-erlangen.de (M. Grottke).

0164-1212/$ – see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.jss.2010.11.904

The main contributions of this systematic literature review are thus two-fold: (1) We systematically gather and discuss domain-relevant articles, covering 30 years and a large number of computing journals. (2) By answering our eight research questions, we identify aspects emphasized by prior research and areas that future work should address.

The remainder of this article is structured as follows: In Section 2, the eight research questions of our systematic literature review are presented. Next, Section 3 introduces the review method applied and the classification scheme used. The results of the review are then discussed in Section 4, answering the research questions formulated. The article closes with Section 5, which sums up our findings and suggests areas for future research.

2. Research questions

Conducting any systematic literature review needs the postulation of research questions, which drive the entire research methodology (Kitchenham and Charters, 2007). In accordance with prior studies, such as Jorgensen and Shepperd (2007), Beecham et al. (2008), and Kitchenham et al. (2009), we postulate the eight research questions discussed in the following sub-sections to investigate the software quality cost domain.

2.1. Historical development

As in other engineering disciplines (Dale, 2003), the understanding of software quality has gone through different phases proposing different approaches for coping with the challenge of low quality and high quality-related costs (Whittaker and Voas, 2002; Karg and Beckhaus, 2007). Nevertheless, software quality remains low, while quality-related costs are high. In recent decades, software engineering economics in general (Boehm, 1981; Biffl et al., 2006) and software quality costs in particular (RTI, 2002) have moved into the spotlight. These developments, together with the need to cope with the high quality-related costs, motivate the assumption that the research intensity in the software quality cost domain may have increased in recent years. By proposing the following research question (RQ), we try to verify this assumption:

RQ 1. How did research on software quality costs develop over time?

2.2. Research disciplines

According to Glass et al. (2004), the field of computing consists of three research disciplines: *computer science*, *information systems*, and *software engineering*. While systems/software, systems/software management and organizational concepts are primarily addressed by the disciplines information systems and software engineering, the discipline computer science aims at mathematical aspects. Since the quality cost concept has originated in engineering management (Dale, 2003), it can be assumed that research on software quality costs is most commonly conducted within the disciplines information systems and software engineering. To check this assumption and to reveal which discipline is the most active one, we postulate the following research question:

RQ 2. Which discipline does most frequently publish software quality cost research?

2.3. Relevant journals

Previous investigations have shown that research on software engineering economics and quality management is published in several journals (Rai et al., 1998; Jorgensen and Shepperd, 2007). Software quality costs form a sub-domain of these two research domains. Hence, it can be assumed that only in some of these journals research on software quality costs is published as well. Identifying these journals will provide a valuable reference to the research community. By answering the following research question, we provide a ranking of those journals:

RQ 3. Which journals do most frequently publish software quality cost research?

2.4. Predominant researchers

In most research domains, there is only a very small number of researchers who are highly active and thus shape the research domain (Jorgensen and Shepperd, 2007). Our goal is to identify these leading researchers and the research clusters they belong to. Knowing these researchers and the topics they work on helps to develop a better understanding of the research domain. We thus want to answer the following research question:

RQ 4. Who are the predominant researchers, and what are the related research clusters?

2.5. Research topics

Several topics and cost elements can be distinguished with regard to software quality costs. According to cost accounting theory, quality cost elements can be structured by different classification schemes (Horngren et al., 2008). For software development, the PAF (prevention, appraisal, and failure) cost scheme is the one most commonly applied (Galin, 2003; Karg and Beckhaus, 2007; Grottke and Graf, 2009; Pressman, 2010). The scheme distinguishes between three quality cost types related to corresponding activity types (Pressman, 2010):

- Prevention costs, i.e., costs for activities like quality planning and training which help to avoid future appraisal and failure costs;
- Appraisal costs, i.e., costs for appraisal activities like testing, control, and measurement; and
- Failure costs, i.e., costs for failure-related activities like rework, failure mode analysis, and corrective maintenance.

By classifying articles based on these cost types, we can provide an answer to the following research question:

RQ 5. Which cost types are the predominant topics of software quality cost research?

2.6. Research scopes

Quality management has a long history, and it has focused on different aspects and granularities (Whittaker and Voas, 2002; Yong and Wilkinson, 2002). The the same holds for research on software quality costs, implying that it can be carried out at different levels (Williams et al., 1999). Some work focuses on the costs of one particular quality assurance activity, while other work operates at a coarser granularity level, e.g., work addressing the costs of all failure-related activities in a company. We wish to find out at which granularities research is conducted most frequently and thus propose the following research question:

RQ 6. What are the primary research scopes?

2.7. Research approaches

Usually, several research approaches are used to explore a research domain (Ahire et al., 1995). However, the approaches predominantly used can have a strong influence on the findings of the research domain as well as on its development, which is strongly

related to its findings (Jorgensen and Shepperd, 2007). By studying the research approaches employed and by identifying the ones used most frequently, we can give recommendations on how to further develop the research methodology in the software quality cost domain. Therefore, we formulate the following research question:

RQ 7. What are the research approaches predominantly used?

2.8. Research topics, scopes, and approaches

In Sections 2.5–2.7, we have proposed three research questions, which independently address the three properties research topic, research scope, and research approach. This, however, does not take into account that there might be interdependencies between these properties. To investigate them, we postulate our eighth and final research question:

RQ 8. What are the interdependencies between the research topics, scopes, and approaches?

3. Method

For conducting our systematic literature review, we followed the guidelines proposed by Kitchenham and Charters (2007) and adjusted them to our research domain. This adjustment is in accordance with those made in the studies by Jorgensen and Shepperd (2007) and by Kitchenham et al. (2009). As we have already motivated the need for a systematic review in Section 1 and have postulated our research questions in Section 2, the following subsections document the further steps performed in our systematic literature review.

3.1. Journal selection

Software quality cost research is spread over many research disciplines and communities, each of which is using its own terminology. Identifying relevant articles by searching a database like IEEE Xplore with a set of key terms would thus have been unreliable. We therefore chose to carry out a manual search. Our decision is backed up by the findings by Jorgensen and Shepperd (2007). These findings suggest that in the field of software development cost estimation there is no standardized terminology on which a key-term-based search could rely.

To conduct a manual, journal-based search procedure, it is necessary to compile the set of journals to be scanned. Therefore, we gathered an initial set of journals following the approach used by Glass et al. (2004), adapting it to our research domain. This means that for *computer science* we applied the approach proposed in the study by Geist et al. (1996) to identify the relevant journals; for *information systems* we used the journals named by Mylonopoulos and Theoharakis (2001); and for *software engineering* we started with the journals mentioned by Tse et al. (2006). We extended this initial set by adding journals referenced in previous surveys and systematic literature reviews in the field of software economics and quality (such as Rai et al., 1998; Jorgensen and Shepperd, 2007) and journals from common rankings. This resulted in an initial set of more than 100 highly-ranked, peer-reviewed, leading English computing journals covering the three disciplines *computer science*, *information systems*, and *software engineering*.

For a first version of this article, written in 2009, the first two authors screened the initial set of journals and excluded those journals for which it seemed unlikely that they publish software quality cost research because they are either too technical or too application-specific. The decisions were made by reading the editorial notes of the journals and by randomly scanning some articles published. Most of the journals excluded belong to the disciplines *computer science* and *information systems*. In the first round, we thus selected a set of 45 journals.

The reviewers of the first version of this article requested us to ensure that our review comprehensively covers the software quality cost research published in journals. In the second round, when preparing the final version of our literature review, we therefore added 15 journals previously excluded to further reduce the probability of having missed any software-quality-cost-related article published in a leading computing journal. We thus compiled a set of 60 journals, listed in Appendix A. In this list, the titles of the 45 journals already included in the first version of the review are set in bold type.

3.2. Article extraction

In the first round, the first author of this review scanned each journal of the initial set of 45 journals for articles relevant to the software quality cost domain and published between 1980 and 2008. In the second round, he rescanned these 45 journals for articles published in 2009, and he scanned the additional 15 journals for articles published between 1980 and 2009. In accordance with Jorgensen and Shepperd (2007), for each journal his search for software-quality-cost-related articles was carried out issue-by-issue, by reading the title and abstract of each article published in this journal. In this process, the first author applied the article selection criteria presented in Section 3.3, while the third author was responsible for cross-validation. He drew a random sample of 5 journals and validated the articles included and excluded. The validation by the third author indicated that the selection criteria had been applied correctly—no mis-selection was identified.

By following this procedure, we identified 82 articles in the first round and 5 articles in the second round. 2 of these additional 5 articles were published in 2009. The other 3 articles are stemming from 2 journals not included in the first round, namely, the Australasian Journal of Information Systems and the IEICE Transactions on Information and Systems.

In total, we thus determined 87 articles published between 1980 and 2009. In Appendix A, they are listed next to the respective journals.

3.3. Article selection criteria

The main criterion for including a journal article is its topic. We only considered articles that explicitly address *software quality costs* and that were published in the journals identified as described in Section 3.1. This restriction led to the omission of those articles in the field of effort estimation which have no clear focus on test effort and quality cost estimation, and of those articles on software quality assurance techniques which do not primarily deal with cost aspects. Furthermore, we excluded generic and domain-independent cost accounting approaches that happen to be applicable to software quality cost calculation, such as Ittner (1999), as well as pure discussion/opinion articles.

3.4. Data collection and article classification

In both rounds, the first and third author independently read the full text of all articles and extracted the data necessary to answer the research questions postulated in Section 2. This involved the classification of all articles according to the scheme presented in Section 3.5.

The second author coordinated the data extraction and classification tasks of the first and the third author. When there was any disagreement regarding the classification according to the three research properties, the second author reclassified the article. Based on this third opinion, we discussed the classification issue

until we reached agreement. However, this was only necessary for less than 10% of all articles.

By following this procedure for the 82 articles of the first round and the additional 5 articles of the second round, we classified all 87 articles published between 1980 and 2009. The detailed results of this classification are shown in Appendix A–D.

3.5. Article classification scheme

We followed the approach by Glass et al. (2004), according to which each computing journal can be assigned to exactly one of the three **research disciplines** computer science, information systems, and software engineering; all articles published in a specific journal are thus assumed to belong to the research discipline attributed to this journal. In fact, for many of the journals included in our review Glass et al. (2004) have already provided a classification by research discipline. We compared the additional journals to those classified by Glass et al. (2004) and assigned them to a research discipline if at least one article relevant to our review has been published in the journal. This was, for instance, necessary for the journals CrossTalk, The Journal of Defense Software Engineering and Empirical Software Engineering. We discussed the classification of each of these journals until we reached agreement.

To classify the **research topic** of an article, we could have used the three cost types discussed in Section 2.5. For example, for a specific article both the prevention costs category and the failure costs category might apply. However, to simplify classification, we transformed the three non-exclusive categories into the following seven exclusive ones:

- Prevention costs only (*Prev-Costs*);
- Appraisal costs only (*Appr-Costs*);
- Failure costs only (*Fail-Costs*);
- Prevention and appraisal costs only (*PrevAppr-Costs*);
- Prevention and failure costs only (*PrevFail-Costs*);
- Appraisal and failure costs only (*ApprFail-Costs*); and
- Prevention, appraisal and failure costs (*PrevApprFail-Costs*).

These categories are complete and disjoint; that is, they form a partition of the set of all software quality cost research articles based on the cost types addressed. In the following, we will use the abbreviation given in brackets when referring to a specific category.

For examining the level addressed by each article and classifying its **research scope**, we employed the scheme suggested by Williams et al. (1999). It consists of the following four complete and exclusive categories:

- *Industry level*, which covers articles targeting multiple companies or governmental institutions employing a workforce of software engineers;
- *Company level*, which covers research dealing with an entire firm;
- *Project/product level*, which covers those articles targeting the whole verification and validation chain of a software release; and
- *Activity level*, which covers research on individual activities, for example a single quality assurance activity.

Finally, the classification of the **research approach** (i.e., the methodological orientation) uses the following eight complete but non-exclusive categories adapted from Jorgensen and Shepperd (2007):

- *Theory* includes articles presenting non-empirical research findings or evaluating the quality cost concept for software development.
- *Model* groups articles presenting software quality cost models grounded on different quality modeling approaches and covering different cost elements.
- *Estimation* (*method*) covers those articles presenting quality cost estimation approaches, such as test effort estimation.
- *Simulation* relates to articles using simulation as an evaluation approach.
- *Case study* covers articles studying a small number of cases (fewer than 10).
- *Empirical* relates to articles studying a large number of cases (10 or more).
- *Example* includes those articles using hypothetical numerical examples to illustrate a model or approach.
- *Others* groups articles using approaches not falling into any of the seven above-mentioned categories.

3.6. Analysis

The raw data collected (shown in Appendix A–D) already provides a first picture of software quality cost research. To answer the research questions postulated in Section 2, we aggregated and tabulated this data. Our results are presented and discussed in Section 4.

3.7. Potential limitations

Of course, our systematic review may have some limitations. One of them might be a *publication bias*. While we cannot fully exclude the possibility of such a bias, we believe that our systematic review process, based on the guidelines proposed by Kitchenham and Charters (2007), has lead us to a representative sample of journals. Although we may not have covered each and every journal publishing software quality cost research, we are confident that by including 60 leading computing journals in our study we have not missed any of the most important ones. This assumption is supported by the fact that only 2 of the 15 journals added in the second round of our studies contained any relevant articles.

A second limitation might concern our *classification approach*. While we derived the classification scheme used from previous reviews to ensure its robustness and reliability, the descriptions of the categories might be further improved. Moreover, some of the classification decisions could be subject to discussion. However, all three authors have experience in and knowledge of the software quality cost research domain, and have tried to make their judgments as objective as possible.

4. Results and discussion

In this section, we answer our research questions based on the results of our systematic literature review.

4.1. Historical development (RQ 1)

Fig. 1 depicts the distribution of the 87 articles over time. There is obviously quite some fluctuation, but some trends are also clearly visible. While one article dates back to 1980, research on software quality costs intensified from the mid-1980s until the mid-1990s. Since its peak in 1996 there has been a decline in the number of articles published per year. (In relative terms, the decline is more pronounced, because the total number of articles published per year in the 60 journals has increased.) An explanation for this decline might be that in recent times new research topics have moved into the spotlight. Consequently, researchers have started to focus on those topics and have moved away from quality cost research. However, there is evidence that software quality cost research has become an established research domain: In recent

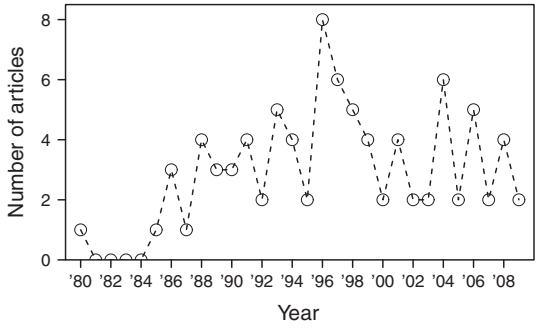

Fig. 1. Number of articles published per year.

years, the number of articles published per year seems to have stabilized.

We also investigated whether there is any link between the time of publication and the research topic, scope, and approach(es) chosen by an article, but we did not discover any significant dependence; it seems that the articles per category are randomly distributed over time. This is an interesting finding because one might have expected a temporal development from one category to another.

4.2. Research disciplines (RQ 2)

Glass et al. (2004) have studied how research articles in the general field of computing are distributed across the disciplines computer science, information systems, and software engineering. The left part of Fig. 2 illustrates their findings: 14% have been attributed to information systems, while the largest part (about two thirds of the articles) have been assigned to software engineering. For the reasons given in Section 2.2, it can be expected that the largest part of the research on software quality costs is carried out within these two disciplines.

Our classification of the 87 articles, shown in the right part of Fig. 2, reveals that 84% of them are related to either information systems or software engineering, marking a (modest) increase of 3 percentage points as compared with the combined proportions of the two disciplines according to Glass et al. (2004). However, at a proportion of 76% the software engineering discipline plays an even more important role for software quality cost research than for the field of computing in general. This phenomenon may be due to the specificity and the complexity of the software quality cost research domain: To focus on software quality costs, researchers have to be interested in the economic side of the software development process as well as in software quality.

Table 1
Journals reviewed and number of articles included.

Rank	Title of the journal	Articles
1	IEEE Trans. Softw. Eng.	13
2	IEEE Softw.	8
2	J. Syst. Softw.	8
4	CrossTalk, J. Defense Softw. Eng.	7
4	Empirical Softw. Eng.	7
6	J. Softw. Maint. [& Evol.]: Res. Pract.	5
7	IEEE Trans. Rel.	4
8	Commun. ACM	3
8	Softw. Quality J.	3
10	ACM Trans. Softw. Eng. Methodol.	2
10	Hewlett-Packard J.	2
10	IEEE Trans. Comput.	2
10	IEICE Trans. Inf. Syst.	2
10	Int. J. Syst. Sci.	2
10	Manage. Sci.	2
10	Quality Progress	2
17–31	Journals with one article each	15
32–60	Journals with zero articles each	0
	Total	87

4.3. Relevant journals (RQ 3)

The distribution of the 87 articles across journals is shown in Table 1. According to our data, software quality cost research is most frequently published in the IEEE Transactions on Software Engineering, in IEEE Software, and in the Journal of Systems and Software. These three journals alone account for one third of all 87 articles considered. 72 articles appeared in those 16 journals in which at least 2 articles on software quality cost research were published. Our study also shows that 31 of the considered journals published at least one relevant article, whereas 29 journals contained none. A possible explanation for these findings may be that the software quality cost domain is rather specific (cf. Section 4.2). Therefore, its research results are most appropriate for journals covering a wide spectrum of topics instead of niche-journals, which are not devoted to the topic and are themselves too specific to deal with it.

Due to the small number of articles published per journal, we did not investigate the question whether or not software quality cost research articles published in certain journals feature any specific characteristics.

4.4. Predominant researchers (RQ 4)

A total of 155 researchers are among the authors of our set of 87 articles on software quality costs. Table 2 lists those 12 authors who

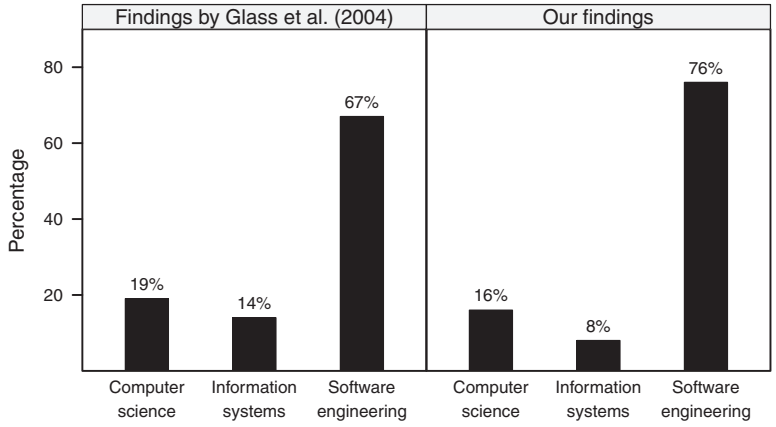

Fig. 2. Article distribution across research disciplines.

Table 2
Top 12 authors.

Rank	Name	Articles	First author	First year	Last year
1	Pham, H.	4	2	1996	2004
1	Yamada, S.	4	3	1985	1999
3	Huang, C.-Y.	3	2	2005	2008
3	Rothermel, G.	3	1	2004	2006
3	Weyuker, E. J.	3	3	1990	1999
6	Banker, R. D.	2	2	1993	1997
6	Elbaum, S.	2	1	2004	2004
6	Houston, D.	2	1	1998	1998
6	Kusumoto, S.	2	1	1992	2004
6	Osaki, S.	2	0	1985	1987
6	Slaughter, S. A.	2	1	1997	1998
6	Zhang, X.	2	1	1998	1999

(co-)authored at least two articles, ranked according to the number of articles they were involved in. Researchers with the same rank are listed in alphabetical order. The 12 authors (i.e., around 8% of all authors involved in the articles studied) account for 24 of the 87 articles (i.e., around 28%); their contribution thus forms a substantial part of the articles published in this research domain.

We further examined the presence of research clusters focusing on software quality costs. We define a research cluster as a set of at least two researchers who collaborated on at least one publication, with at least two relevant articles published by the research cluster. Fig. 3 shows the research clusters identified. The value behind '#' represents the total number of articles published by the respective research cluster. This number also includes articles published by a single author belonging to the cluster. The width of each edge connecting two authors indicates how frequently these authors have published joint work relevant to our review.

Comparing Fig. 3 with Table 2 reveals that almost all top 12 researchers publish their software-quality-cost-related articles in cooperation with other authors. The only exception we found is Weyuker, who is the sole author of three articles on Appr-Costs and Fail-Costs (Weyuker, 1990, 1996, 1999). Besides evaluating the cost efficiency of quality improvements and presenting success measures, she studies the costs of data flow testing.

The seven research clusters identified can be grouped and distinguished by the specific focus of their work on software quality costs:

Three clusters are concerned with software quality cost models and use software reliability growth models (SRGM) as the modeling foundation. While the largest SRGM-related cluster is centered around Yamada, the same number of articles has been contributed by the one grouped around Pham, which only consists of three authors; the third cluster forms around Huang. The quality cost models proposed by these three researchers and their co-authors aim at the quality-cost-optimal release time, in particular by including context factors like test effort (Yamada et al., 1986; Yamada and Osaki, 1987), imperfect debugging (Pham, 1996), and test effort and efficiency (Huang and Lyu, 2005; Huang, 2006; Lin and Huang, 2008). Since all three researchers can be assigned to the software reliability growth community, this finding indicates that this community may have a general interest in software quality cost topics.

The fourth and biggest cluster centers around Rothermel and Elbaum. Both researchers are well known for their interest in software quality assurance. Hence, it is not surprising that they discuss quality cost aspects particularly related to software quality assurance. In one article (Rothermel et al., 2004), they focus on the determination of cost-efficient regression test strategies under given context factors, while other work by the two researchers is devoted to the determination of the optimal number of test cases, as well as the cost-optimal prioritization of test cases. While some of the proposed approaches are test-technique-independent (Elbaum et al., 2004), others are meant for unit tests only (Do et al., 2006).

The fifth cluster is rather small, and it is driven by Houston. In two articles he and his co-authors discuss the applicability of the quality cost concept to software development (Houston and Keats, 1998; Krasner and Houston, 1998).

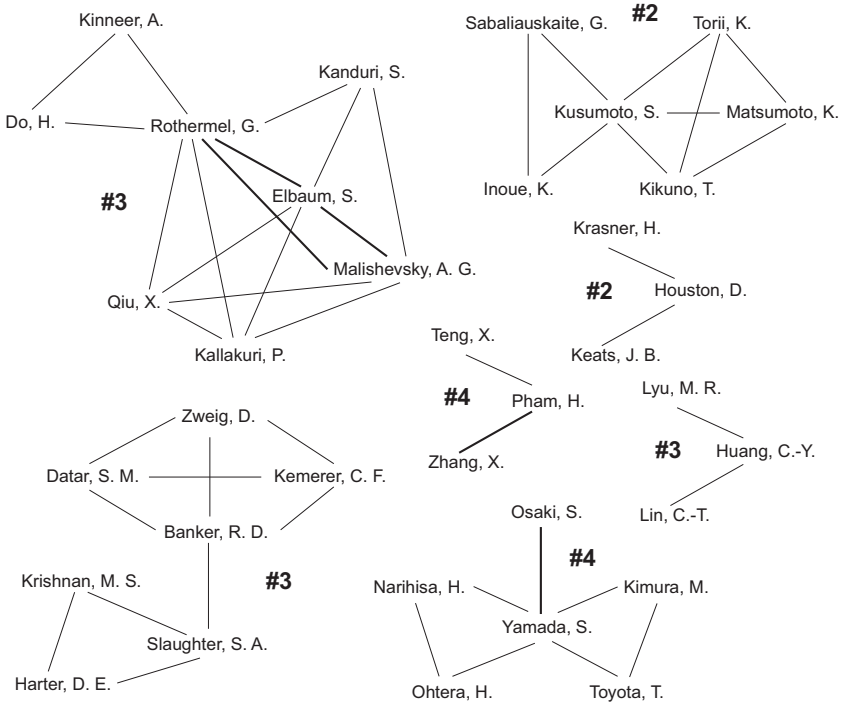

Fig. 3. Research clusters.

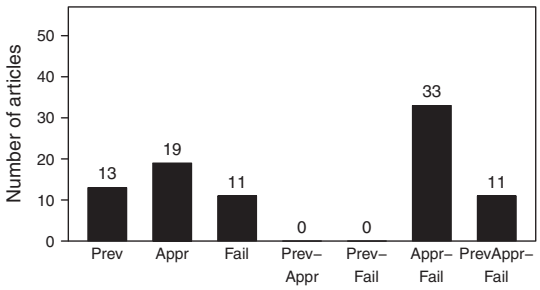

Fig. 4. Article distribution across research topics.

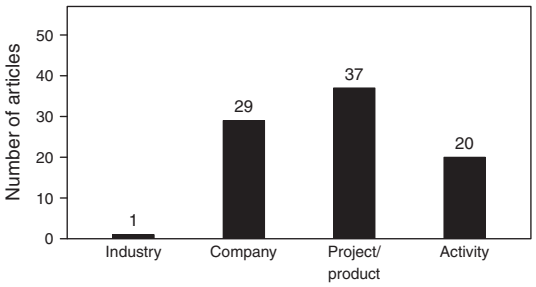

Fig. 5. Article distribution across research scopes.

The sixth cluster forms around Kusumoto. It is devoted to modeling and improving the costs of software inspections (Kusumoto et al., 1992; Sabaliauskaite et al., 2004).

The final cluster (the second-largest one we identified) centers around Banker and Slaughter. The researchers of this cluster study quality costs from an information-system-centric perspective. By doing so, they strongly focus on the engineering management aspect of software development. For instance, in two articles they empirically analyze factors influencing the distribution of corrective maintenance effort (Banker et al., 1993; Banker and Slaughter, 1997).

As the discussion shows, research on software quality costs is driven by several research clusters which aim at different aspects and use different research approaches. These clusters, together with single but well-established researchers such as Weyuker, help to advance research in the software quality cost domain.

4.5. Research topics (RQ 5)

Fig. 4 reveals that nearly half of all articles (namely, 43) are dealing with a single cost type. While the largest number of them fall into the Appr-Costs category, all three cost types have frequently been analyzed individually.

As for the 44 articles addressing multiple cost types, they are concentrated on two cost categories: 75% of them (33 articles) are dealing with ApprFail-Costs, and the remaining 11 articles are covering all three cost types. However, prevention costs have never been studied in combination with appraisal costs only (PrevAppr-Costs) or failure costs only (PrevFail-Costs).

Taking the perspective of different cost types in individual or cross-topical studies, we observe that 63 articles are devoted to appraisal costs (alone or in combination with other cost categories), and 55 articles are concerned with failure costs. In comparison, only 24 articles are in any way dealing with prevention costs. This finding may be explained as follows: While appraisal costs and failure costs are closely related to individual projects or products, it is more challenging to identify and assess prevention costs (as investments helping avoid future appraisal and failure costs).

4.6. Research scopes (RQ 6)

With respect to the research scope, Fig. 5 shows that the project/product level has been addressed most often, by 37 articles. A possible explanation is that since the beginnings of software engineering, researchers have investigated and attempted to predict project/product-related costs. This traditional view on software engineering organization from a project management perspective may hinder research on other scopes. However, today many companies (not only from the software sector) employ a constant workforce of software engineers. It is therefore of great interest to also evaluate organizational forms others than projects. The research community partly addresses this need by conducting studies on the activity level or the company level. Research at the industry level is probably most involved, especially if it is to deal with actual data from multiple organizations. This may be the reason why we merely detected a single article focusing on this research scope.

4.7. Research approaches (RQ 7)

Research approach consists of the largest number of categories, and unlike the other two properties analyzed its categories are non-exclusive. As Fig. 6 indicates, there are popular research approaches (such as model building) and niche approaches (such as simulation). Appendix D reveals that there is no overlap between the articles employing models and those dealing with theory. More than 75% of all articles (66 out of 87) have thus been devoted to model building or theory generation. In contrast, only 31 of them (i.e., less than 36%) validate their findings empirically (including 10 cases or more) or based on a smaller case study. The availability of actual quality cost data thus appears to be a major challenge. This deficiency may hinder methodologically sound studies aiming at a holistic understanding of software quality costs.

4.8. Research topics, scopes, and approaches (RQ 8)

Table 3 depicts the results of our joint analysis of all three properties (i.e., research topic, research scope, and research approach) and highlights the interdependencies between these properties. While the four levels of the research scope are represented by the four sub-tables, the research topics and the research approaches form the rows and the columns of these sub-tables. Note that the right-most column, giving the number of articles with a specific combination of scope and topic, does not contain row totals. Since the categories of research approach are non-exclusive, a single article may be counted under multiple approaches within a row.

As noted before, only one of the articles studied in this review is dealing with the industry level. A sample size of one is of course insufficient for drawing any conclusions. We therefore omit this level in the following discussion. For the other levels, Table 3 indi-

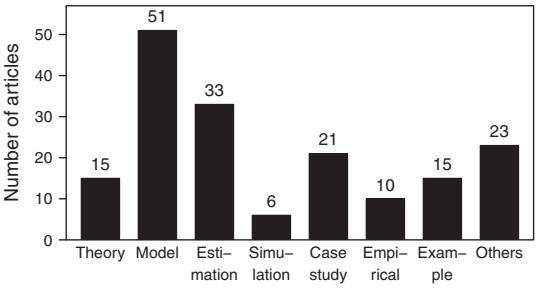

Fig. 6. Article distribution across research approaches.

Table 3
Joint analysis of all three research properties.

Scope	Topic	Approach								
		Theory	Model	Estimation	Simulation	Case study	Empirical	Example	Others	Total
Industry level	Prev-Costs	0	0	0	0	0	1	0	1	1
	Appr-Costs	0	0	0	0	0	0	0	0	0
	Fail-Costs	0	0	0	0	0	0	0	0	0
	PrevAppr-Costs	0	0	0	0	0	0	0	0	0
	PrevFail-Costs	0	0	0	0	0	0	0	0	0
	ApprFail-Costs	0	0	0	0	0	0	0	0	0
	PrevApprFail-Costs	0	0	0	0	0	0	0	0	0
	Total	0	0	0	0	0	1	0	1	1
Company level	Prev-Costs	2	1	1	0	3	1	1	7	12
	Appr-Costs	0	0	0	0	0	0	0	0	0
	Fail-Costs	1	2	0	0	0	2	0	0	3
	PrevAppr-Costs	0	0	0	0	0	0	0	0	0
	PrevFail-Costs	0	0	0	0	0	0	0	0	0
	ApprFail-Costs	1	2	0	0	0	1	0	0	3
	PrevApprFail-Costs	11	0	0	0	0	1	0	0	11
	Total	15	5	1	0	3	5	1	7	29
Project/product level	Prev-Costs	0	0	0	0	0	0	0	0	0
	Appr-Costs	0	3	3	0	1	0	2	1	4
	Fail-Costs	0	7	5	0	4	0	1	2	8
	PrevAppr-Costs	0	0	0	0	0	0	0	0	0
	PrevFail-Costs	0	0	0	0	0	0	0	0	0
	ApprFail-Costs	0	23	17	4	7	0	11	2	25
	PrevApprFail-Costs	0	0	0	0	0	0	0	0	0
	Total	0	33	25	4	12	0	14	5	37
Activity level	Prev-Costs	0	0	0	0	0	0	0	0	0
	Appr-Costs	0	10	5	2	2	3	0	7	15
	Fail-Costs	0	0	0	0	0	0	0	0	0
	PrevAppr-Costs	0	0	0	0	0	0	0	0	0
	PrevFail-Costs	0	0	0	0	0	0	0	0	0
	ApprFail-Costs	0	3	1	0	4	1	0	2	5
	PrevApprFail-Costs	0	0	0	0	0	0	0	0	0
	Total	0	13	6	2	6	4	0	9	20

cates that there is indeed a substantial association between the research scope and the topics studied as well as the approaches chosen. We discuss our findings level by level.

With respect to the research topics, the articles at the ***company level*** show the largest variety; as the scope shifts from the company to the activity level, the number of topics discussed is narrowed down from four to two. This is mainly due to the fact that (with the exception of the one article at the industry level) prevention costs, either alone or in combination with other costs types, are exclusively dealt with at the company level. Usually, prevention costs are long-term investments, which are difficult to allocate to a specific project/product or activity.

The topics specifically in focus at the company level are Prev- and PrevApprFail-Costs. Research on the former topic relies on a variety of approaches: 7 of the 12 articles investigating Prev-Costs use some *other* approach. An explanation for this finding is that the topic Prev-Costs covers a wide range of preventive activities, like process improvements (Dion, 1993) and software reuse (Mohagheghi and Conradi, 2007), which often require specific research approaches.

PrevApprFail-Costs, the second research topic in focus at the company level is always studied *theoretically*. For example, Knox (1993) analyzes the quality cost concept with respect to its adaptability to software development, while Webb and Patton (2008) examine its business value for software vendors. However, only 1 of the 11 articles on PrevApprFail-Costs provides any *empirical* data: Slaughter et al. (1998) empirically assess the business value of the quality cost concept. Although a better understanding of this value would provide helpful insights, there is an unfortunate lack of systematic research.

In fact, the lack of empirical work already identified based on Fig. 6 is specifically pronounced at the company level: In only 8 of the 29 articles (i.e., about 28%) the findings are validated by a *case study* or a more extended *empirical* analysis. Naturally, obtaining data related to an entire firm is more demanding than collecting data for an individual project or activity.

Table 3 shows that 33 of the 37 articles dealing with the ***project/product level*** are devoted to quality cost *models*. Further investigation of the individual articles reveals that the majority of them employ software reliability growth models as the modeling foundation, probably because these models can easily be extended by quality cost elements. This finding mirrors our identification of three research clusters driven by researchers from the software reliability growth community in Section 4.4.

The modeling foundation chosen endows 25 of 33 the quality cost models proposed with the ability to *estimate* costs. However, only 12 of these models rely on a *case-study*-based validation (consisting of less than 10 cases); not even one of them presents *empirical* data (including 10 or more cases). In comparison, 14 of the 33 model-related articles provide illustrative numerical *examples*. On the one hand, this finding might again be explained by the data availability challenge: Most models pre-

sented at the project/product level require data for specific input parameters which are difficult to gather, especially for a large number of projects or products. An example for such parameters are the costs of various test activities, which often vary across projects. On the other hand, the chosen modeling foundation gives rise to a second explanation of why illustrative *examples* are often used: In the software reliability growth community it is not uncommon to rely on numerical *examples* to demonstrate a model's performance; as our data reveal, the same approach has been adopted in the validation of quality cost models (cf. Pham, 2006).

With respect to the topic, 25 of the 37 articles at the project/product level are concerned with ApprFail-Costs. Most of the related models are used to study the cost-optimal release time of a software product under various constraints and covering different appraisal and failure cost elements. Examples for those cost elements and constraints are fault removal times and costs of risk and uncertainty (Zhang and Pham, 1998), as well as external failure and risk costs (Pham and Zhang, 1999). However, there are a few articles presenting extensions to software reliability growth models covering either appraisal costs only, or failure costs only. For example, Singpurwalla (1991) aims at finding the optimal time interval for testing and debugging under uncertainty, while Gutjahr (1995) presents a method for predicting software failure costs under the consideration of several reliability measures.

The articles at the final research scope—the ***activity level***—usually focus on Appr-Costs (15 out of 20 articles), while only 5 articles study both appraisal and failure costs. Most of these 5 articles on ApprFail-Costs are concerned with the costs of inspections (Bourgeois, 1996; McCann, 2001; O'Neill, 2003; Freimut et al., 2005). However, the costs of inspections are also studied in several of the articles related to Appr-Costs (Collofello and Woodfield, 1989; Grady and von Slack, 1994). Furthermore, the discussion of Appr-Costs focuses on the determination of cost-efficient strategies for regression tests (Rothermel et al., 2004) or system tests (Cangussu et al., 2002) as well as on the cost-optimal selection and prioritization of test cases (Brown et al., 1989; Elbaum et al., 2004; Do et al., 2006). Researchers thus predominantly study the costs of particular quality assurance techniques and related questions.

In 13 of the 20 articles at the activity level, a *model* is proposed. This research approach thus plays an important role, although less so than at the project/product level. However, in contrast to the project/product level, a much larger fraction of these models (10 out of 20) is validated *empirically* based on at least 10 cases, or by a smaller *case study*. Obviously, data availability is less of an issue when studying individual activities. Instead, *examples* are not employed at all. Also, it seems that these models are less applicable for cost *estimation* than those based on software reliability growth models.

5. Conclusions

The main goal of this systematic literature review on software quality costs was to structure existing work in this field and to guide researchers to promising future research directions.

Our results have revealed that software quality cost research is mostly published in software-engineering-related journals. While only the IEEE Transactions on Software Engineering have published more than ten articles, articles on software quality cost research have appeared in as many as 31 journals. We have seen that 12 authors have been involved in at least two of the articles examined. These authors account for a considerable fraction of all publications covered in this review. They and the seven related research clusters shape the research domain by studying software quality costs from different perspectives.

Regarding the content of the software quality cost studies, we proposed three properties (namely, research topic, research scope, and research approach) and categorized all identified articles. We found that appraisal and failure costs are often analyzed jointly. This may be due to the direct link between these kinds of costs: Failure costs tend to increase when less effort is spent on appraisal activities, and vice versa. There is no such direct interdependency with prevention costs, which are related to long-term investments like process improvement initiatives. Therefore, prevention costs can easily be analyzed separately from the other cost types.

We also found that software quality cost research has primarily been carried out by means of model building and theory generation. While the community has thus developed a sound understanding of the research domain's structure, empirical validation is often lacking. Only about a third of the analyzed articles presents a case study or more extensive empirical results. This appears to be insufficient for software quality cost research, which strongly relies on quantitative data to generate new findings. There is thus a need for novel approaches to gather quality cost data, as well as stronger cooperation between industry and research to make such data available.

Further, our classification of all articles has unveiled some interesting dependencies between the three properties. For example, we observed a link between the research scope and the software quality cost categories studied. While at the company level, prevention costs are sometimes investigated, the more specific project/product and activity levels concentrate on appraisal and failure costs. This finding can be explained by the fact that prevention activities typically occur in form of process improvements, which cannot be assigned to a single project.

Overall, prevention costs have received the least attention, although the highest quality cost savings can be achieved by avoiding defects in the first place via investments into preventive activities. Consequently, it seems to be suggestive to put more focus on this cost type. The same holds for research on the industry level. We are only aware of one article that gathers industry-wide data and benchmarks software companies by their quality costs.

Regarding software quality cost modeling, our review provides a mixed picture: While many models have been proposed at the project/product level and the activity level, there is no article suggesting a comprehensive model at the company level (or at the industry level). Only such comprehensive models including prevention, appraisal and failure costs might give a holistic view on software quality costs as well as insights into the right balance between these three cost types.

Appendix A. Journals and articles included

Our review includes the following 60 leading computing journals and the software quality cost research articles published therein between 1980 and 2009. The journals that were already included in the first version of the review are set in bold type. For those journals containing at least one relevant article, we also provide a classification into the three computing-related research disciplines computer science (CS), information systems (IS), and software engineering (SE). This classification is shown in brackets following the journal title.

1. ACM Transactions on Information Systems
2. **ACM Transactions on Software Engineering and Methodology** (SE): Rothermel et al. (2004) and Weyuker (1996)

3. Advances in Engineering Software
4. **Annals of Software Engineering**
5. **AT&T Technology Journal** (CS): Pettijohn (1986)
6. Australasian Journal of Information Systems (IS): Hollingsworth et al. (1999)
7. **Automated Software Engineering**
8. **Communications of the ACM** (SE): Arthur (1997), Banker et al. (1993) and Slaughter et al. (1998)
9. Communications of the Association of Information Systems
10. **Computer Journal**
11. **CrossTalk, The Journal of Defense Software Engineering** (SE): Bourgeois (1996), Brodman and Johnson (1996), Krasner and Houston (1998), McCann (2001), O'Neill (2003), Spiewak and McRitchie (2008) and Webb and Patton (2008)
12. **Datamation** (SE): Rivard and Kaiser (1989)
13. Decision Support Systems
14. Embedded Systems Programming
15. **Empirical Software Engineering** (SE): Do et al. (2006), Ellims et al. (2006), Hewett and Kijsanayothin (2009), Jones and Tabberer (1993), Khoshgoftaar et al. (2001), Laitenberger (2001) and Mohagheghi and Conradi (2007)
16. **European Journal of Information Systems**
17. **European Journal of Operations Research** (CS): Yamada and Osaki (1987)
18. **European Management Journal**
19. **Hewlett-Packard Journal** (CS): Franz and Shih (1994) and Ward (1991)
20. **HP Digital Technical Journal** (CS): Knox (1993)
21. **I&O (Information and Organization)**
22. **IBM Systems Journal**
23. **IEEE Computer** (CS): Jones (1996)
24. **IEEE Journal on Selected Areas in Communications** (CS): Mandeville (1990)
25. **IEEE Software** (SE): Diaz and Sligo (1997), Dion (1993), Ehrlich et al. (1993), Grady and von Slack (1994), Lim (1994), Sherer (1991), Simmons (1996) and van Solingen (2004)
26. **IEEE Transactions on Computers** (CS): Pham and Zhang (1999) and Teng and Pham (2004)
27. **IEEE Transactions on Engineering Management** (SE): Hullocker (1986)
28. **IEEE Transactions on Reliability** (CS): Hou et al. (1996), Huang and Lyu (2005), Yamada et al. (1986) and Yamada and Osaki (1985)
29. **IEEE Transactions on Software Engineering** (SE): Biffl and Halling (2003), Binkley (1997), Boehm and Papaccio (1988), Brown et al. (1989), Cangussu et al. (2002), Chavez (2000), Freimut et al. (2005), Gutjahr (1995), King et al. (2000), Porter et al. (1997), Singpurwalla (1991), Song et al. (2006) and Weyuker (1990)
30. IEICE Transactions on Information and Systems (IS): Kusumoto et al. (1992) and Sabaliauskaite et al. (2004)
31. **Industrial Management and Data Systems**
32. **Information and Management**
33. **Information and Software Technology** (IS): Calzolari et al. (2001)
34. Information Science Journal
35. **Information Systems Journal**
36. **Information Systems Research**
37. **Information Technology and Management**
38. **International Journal of Information Management**
39. International Journal of Project Management
40. **International Journal of Software Engineering and Knowledge Engineering**
41. **International Journal of Systems Science** (SE): Pham (1996) and Zhang and Pham (1998)
42. Journal of Computer and System Sciences
43. Journal of Information Technology
44. **Journal of Management Information Systems**
45. **Journal of Software Maintenance: Research and Practice** (until 2000)/**Journal of Software Maintenance and Evolution: Research and Practice** (since 2001) (SE): Granja-Alvarez and Barranco-Garcia (1997), Hsia et al. (1998), Leach (1996), Schach (1994) and Sneed (1991)
46. Journal of Strategic Information Systems
47. **Journal of Systems and Software** (SE): Collofello and Woodfield (1989), Engel and Last (2007), Huang (2006), Leung (1992), Lin and Huang (2008), Okumoto and Goel (1980), Westland (2002) and Weyuker (1999)
48. Journal of the Association of Information Systems
49. **Management Information Systems Quarterly** (IS): Abdel-Hamid (1988)
50. **Management Science** (IS): Arora et al. (2006) and Banker and Slaughter (1997)
51. Programming and Computer Software
52. **Quality Engineering** (SE): Houston and Keats (1998)
53. **Quality Progress** (SE): Daughtrey (1988) and Stewart (1988)
54. **Reliability Engineering and System Safety** (CS): Kimura et al. (1999)
55. Scandinavian Journal of Information Systems
56. **Software Engineering Journal** (SE): Wohlin and Koerner (1990)
57. **Software Process: Improvement and Practice** (SE): Deissenboeck and Pizka (2008)
58. **Software Quality Journal** (SE): Elbaum et al. (2004), Issa et al. (2009) and van Megen and Meyerhoff (1995)
59. **Software Quality Professional** (SE): Galin (2004)
60. **Software Testing Verification and Reliability**

Appendix B. Classification by research topic

For all categories of the research topic (defined in Section 3.5), the following list shows the articles to which the respective category applies:

- Prev-Costs: Arthur (1997), Brodman and Johnson (1996), Diaz and Sligo (1997), Dion (1993), Houston and Keats (1998), Jones (1996), Khoshgoftaar et al. (2001), Lim (1994), Mohagheghi and Conradi (2007), Rivard and Kaiser (1989), Simmons (1996), Sneed (1991) and van Solingen (2004)
- Appr-Costs: Biffl and Halling (2003), Binkley (1997), Brown et al. (1989), Cangussu et al. (2002), Collofello and Woodfield (1989), Do et al. (2006), Elbaum et al. (2004), Grady and von Slack (1994), Issa et al. (2009), King et al. (2000), Kusumoto et al. (1992), Laitenberger (2001), Lin and Huang (2008), Porter et al. (1997), Rothermel et al. (2004), Sabaliauskaite et al. (2004), Singpurwalla (1991), Weyuker (1990) and Weyuker (1999)
- Fail-Costs: Banker et al. (1993), Banker and Slaughter (1997), Granja-Alvarez and Barranco-Garcia (1997), Gutjahr (1995), Hewett and Kijsanayothin (2009), Hsia et al. (1998), Leach (1996), Schach (1994), Song et al. (2006), Stewart (1988) and Ward (1991)
- PrevAppr-Costs: *none*
- PrevFail-Costs: *none*
- ApprFail-Costs: Abdel-Hamid (1988), Arora et al. (2006), Boehm and Papaccio (1988), Bourgeois (1996), Calzolari et al. (2001), Chavez (2000), Deissenboeck and Pizka (2008), Ehrlich et al. (1993), Ellims et al. (2006), Engel and Last (2007), Franz and Shih (1994), Freimut et al. (2005), Hou et al. (1996), Huang (2006), Huang and Lyu (2005), Kimura et al. (1999), Leung (1992), McCann (2001), Okumoto and Goel (1980), O'Neill (2003), Pham (1996), Pham and Zhang (1999), Sherer (1991), Spiewak

and McRitchie (2008), Teng and Pham (2004), van Megen and Meyerhoff (1995), Westland (2002), Weyuker (1996), Wohlin and Koerner (1990), Yamada et al. (1986), Yamada and Osaki (1985), Yamada and Osaki (1987) and Zhang and Pham (1998)
- PrevApprFail-Costs: Daughtrey (1988), Galin (2004), Hollingsworth et al. (1999), Hullocker (1986), Jones and Tabberer (1993), Knox (1993), Krasner and Houston (1998), Mandeville (1990), Pettijohn (1986), Slaughter et al. (1998); and Webb and Patton (2008)

Appendix C. Classification by research scope

For all categories of the research scope (defined in Section 3.5), the following list shows the articles to which the respective category applies:

- Industry level: Brodman and Johnson (1996)
- Company level: Arora et al. (2006), Arthur (1997), Banker et al. (1993), Banker and Slaughter (1997), Boehm and Papaccio (1988), Daughtrey (1988), Diaz and Sligo (1997), Dion (1993), Galin (2004), Hollingsworth et al. (1999), Houston and Keats (1998), Hullocker (1986), Jones and Tabberer (1993), Jones (1996), Khoshgoftaar et al. (2001), Knox (1993), Krasner and Houston (1998), Lim (1994), Mandeville (1990), Mohagheghi and Conradi (2007), Pettijohn (1986), Rivard and Kaiser (1989), Simmons (1996), Slaughter et al. (1998), Sneed (1991), Stewart (1988), van Solingen (2004), Webb and Patton (2008) and Westland (2002)
- Project/product level: Abdel-Hamid (1988), Calzolari et al. (2001), Chavez (2000), Deissenboeck and Pizka (2008), Ehrlich et al. (1993), Engel and Last (2007), Franz and Shih (1994), Granja-Alvarez and Barranco-Garcia (1997), Gutjahr (1995), Hewett and Kijsanayothin (2009), Hou et al. (1996), Hsia et al. (1998), Huang (2006), Huang and Lyu (2005), Issa et al. (2009), Kimura et al. (1999), Leach (1996), Leung (1992), Lin and Huang (2008), Okumoto and Goel (1980), Pham (1996), Pham and Zhang (1999), Schach (1994), Sherer (1991), Singpurwalla (1991), Song et al. (2006), Spiewak and McRitchie (2008), Teng and Pham (2004), van Megen and Meyerhoff (1995), Ward (1991), Weyuker (1996), Weyuker (1999), Wohlin and Koerner (1990), Yamada et al. (1986), Yamada and Osaki (1985), Yamada and Osaki (1987) and Zhang and Pham (1998)
- Activity level: Biffl and Halling (2003), Binkley (1997), Bourgeois (1996), Brown et al. (1989), Cangussu et al. (2002), Collofello and Woodfield (1989), Do et al. (2006), Elbaum et al. (2004), Ellims et al. (2006), Freimut et al. (2005), Grady and von Slack (1994), King et al. (2000), Kusumoto et al. (1992), Laitenberger (2001), McCann (2001), O'Neill (2003), Porter et al. (1997), Rothermel et al. (2004), Sabaliauskaite et al. (2004) and Weyuker (1990)

Appendix D. Classification by research approach

For all categories of the research approach (defined in Section 3.5), the following list shows the articles to which the respective category applies:

- Theory: Boehm and Papaccio (1988), Daughtrey (1988), Galin (2004), Hollingsworth et al. (1999), Houston and Keats (1998), Hullocker (1986), Jones and Tabberer (1993), Knox (1993), Krasner and Houston (1998), Mandeville (1990), Pettijohn (1986), Simmons (1996), Slaughter et al. (1998), Stewart (1988) and Webb and Patton (2008)
- Model: Abdel-Hamid (1988), Arora et al. (2006), Banker et al. (1993), Banker and Slaughter (1997), Biffl and Halling (2003), Brown et al. (1989), Calzolari et al. (2001), Cangussu et al. (2002), Chavez (2000), Collofello and Woodfield (1989), Deissenboeck and Pizka (2008), Do et al. (2006), Ehrlich et al. (1993), Elbaum et al. (2004), Engel and Last (2007), Franz and Shih (1994), Freimut et al. (2005), Granja-Alvarez and Barranco-Garcia (1997), Gutjahr (1995), Hewett and Kijsanayothin (2009), Hou et al. (1996), Hsia et al. (1998), Huang (2006), Huang and Lyu (2005), Issa et al. (2009), Khoshgoftaar et al. (2001), Kimura et al. (1999), Kusumoto et al. (1992), Leach (1996), Leung (1992), Lin and Huang (2008), McCann (2001), Okumoto and Goel (1980), O'Neill (2003), Pham (1996), Pham and Zhang (1999), Rothermel et al. (2004), Sabaliauskaite et al. (2004), Schach (1994), Sherer (1991), Singpurwalla (1991), Song et al. (2006), Teng and Pham (2004), Westland (2002), Weyuker (1990), Weyuker (1996), Wohlin and Koerner (1990), Yamada et al. (1986), Yamada and Osaki (1985), Yamada and Osaki (1987) and Zhang and Pham (1998)
- Estimation: Brown et al. (1989), Cangussu et al. (2002), Deissenboeck and Pizka (2008), Ehrlich et al. (1993), Engel and Last (2007), Freimut et al. (2005), Grady and von Slack (1994), Granja-Alvarez and Barranco-Garcia (1997), Gutjahr (1995), Hewett and Kijsanayothin (2009), Hou et al. (1996), Hsia et al. (1998), Huang (2006), Huang and Lyu (2005), Issa et al. (2009), Kimura et al. (1999), Kusumoto et al. (1992), Leach (1996), Leung (1992), Lin and Huang (2008), Okumoto and Goel (1980), Pham (1996), Pham and Zhang (1999), Sabaliauskaite et al. (2004), Singpurwalla (1991), Sneed (1991), Song et al. (2006), Teng and Pham (2004), Wohlin and Koerner (1990), Yamada et al. (1986), Yamada and Osaki (1985), Yamada and Osaki (1987) and Zhang and Pham (1998)
- Simulation: Abdel-Hamid (1988), Brown et al. (1989), Cangussu et al. (2002), Chavez (2000), Engel and Last (2007) and Sherer (1991)
- Case study: Abdel-Hamid (1988), Biffl and Halling (2003), Ehrlich et al. (1993), Ellims et al. (2006), Franz and Shih (1994), Freimut et al. (2005), Granja-Alvarez and Barranco-Garcia (1997), Hewett and Kijsanayothin (2009), Issa et al. (2009), Jones (1996), King et al. (2000), Lim (1994), McCann (2001), O'Neill (2003), Schach (1994), Spiewak and McRitchie (2008), van Megen and Meyerhoff (1995), van Solingen (2004), Ward (1991), Wohlin and Koerner (1990) and Zhang and Pham (1998)
- Empirical: Banker et al. (1993), Banker and Slaughter (1997), Bourgeois (1996), Brodman and Johnson (1996), Diaz and Sligo (1997), Do et al. (2006), Elbaum et al. (2004), Rothermel et al. (2004), Slaughter et al. (1998) and Westland (2002)
- Example: Deissenboeck and Pizka (2008), Gutjahr (1995), Huang (2006), Huang and Lyu (2005), Khoshgoftaar et al. (2001), Kimura et al. (1999), Leung (1992), Lin and Huang (2008), Pham (1996), Pham and Zhang (1999), Singpurwalla (1991), Teng and Pham (2004), Weyuker (1996), Yamada et al. (1986) and Yamada and Osaki (1985)
- Others: Arthur (1997), Binkley (1997), Bourgeois (1996), Brodman and Johnson (1996), Diaz and Sligo (1997), Dion (1993), Ellims et al. (2006), Grady and von Slack (1994), Jones (1996), King et al. (2000), Laitenberger (2001), Leach (1996), Mohagheghi and Conradi (2007), Porter et al. (1997), Rivard and Kaiser (1989), Rothermel et al. (2004), Sabaliauskaite et al. (2004), Song et al. (2006), Spiewak and McRitchie (2008), van Megen and Meyerhoff (1995), van Solingen (2004), Weyuker (1990) and Weyuker (1999)

References

Abdel-Hamid, T.K., 1988. The economics of software quality assurance: a simulation-based case study. MIS Q. 12 (3), 395–411.
Ahire, J., Landeros, R., Golhar, D., 1995. Total quality management: a literature review and an agenda for future research. Prod. Operations Manage. 4 (3), 277–306.
Antony, J., Fergusson, C., 2004. Six sigma in the software industry: results from a pilot study. Manage. Audit. J. 19, 1025–1032.

Arora, A., Caulkins, J.P., Telang, R., 2006. Research note: Sell first, fix later: impact of patching on software quality. Manage. Sci. 52 (3), 465–471.
Arthur, L.J., 1997. Quantum improvements in software system quality. Commun. ACM 40 (6), 46–52.
Banker, R.D., Datar, S.M., Kemerer, C.F., Zweig, D., 1993. Software complexity and maintenance costs. Commun. ACM 36 (11), 81–94.
Banker, R.D., Slaughter, S.A., 1997. A field study of scale economies in software maintenance. Manage. Sci. 43 (12), 1709–1725.
Beecham, S., Baddoo, N., Hall, T., Robinson, H., Sharp, H., 2008. Motivation in software engineering: a systematic literature review. Inf. Softw. Technol. 50 (9–10), 860–878.
Biffl, S., Aurum, A., Boehm, B., Erdogmus, H., Grünbacher, P. (Eds.), 2006. Value-Based Software Engineering. Springer, Berlin.
Biffl, S., Halling, M., 2003. Investigating the defect detection effectiveness and cost benefit of nominal inspection teams. IEEE Trans. Softw. Eng. 29 (5), 385–397.
Binkley, D., 1997. Semantics guided regression test cost reduction. IEEE Trans. Softw. Eng. 23 (8), 498–516.
Boehm, B.W., 1981. Software Engineering Economics. Prentice-Hall, Englewood Cliffs.
Boehm, B.W., Papaccio, P.N., 1988. Understanding and controlling software costs. IEEE Trans. Softw. Eng. 14 (10), 1462–1477.
Bourgeois, K.V., 1996. Process insights from a large-scale software inspection data analysis. CrossTalk, J. Defense Softw. Eng. 9 (10), 17–23.
Brodman, J., Johnson, D., 1996. Return on investment from software process improvement as measured by U.S. industry. CrossTalk, J. Defense Softw. Eng. 9 (4), 23–28.
Brown, D.B., Maghsoodloo, S., Deason, W.H., 1989. A cost model for determining the optimal number of software test cases. IEEE Trans. Softw. Eng. 15 (2), 218–221.
Calzolari, F., Tonella, P., Antoniol, G., 2001. Maintenance and testing effort modeled by linear and nonlinear dynamic systems. Inf. Softw. Technol. 43 (8), 477–486.
Cangussu, J.W., DeCarlo, R.A., Mathur, A.P., 2002. A formal model of the software test process. IEEE Trans. Softw. Eng. 28 (8), 782–796.
Chavez, T., 2000. A decision-analytic stopping rule for validation of commercial software systems. IEEE Trans. Softw. Eng. 26 (9), 907–918.
Collofello, J.S., Woodfield, S.N., 1989. Evaluating the effectiveness of reliability-assurance techniques. J. Syst. Softw. 9 (3), 191–195.
Dale, B.G., 2003. Managing Quality, 4th edition. Wiley-Blackwell, Oxford, UK.
Daughtrey, T., 1988. The search for software quality. Quality Progress 21 (11), 29–31.
Deissenboeck, F., Pizka, M., 2008. Probabilistic analysis of process economics. Softw. Process: Improve. Pract. 13 (1), 5–17.
Diaz, M., Sligo, J., 1997. How software process improvement helped Motorola. IEEE Softw. 14 (5), 75–81.
Dion, R., 1993. Process improvement and the corporate balance sheet. IEEE Softw. 10 (4), 28–35.
Do, H., Rothermel, G., Kinneer, A., 2006. Prioritizing JUnit test cases: an empirical assessment and cost-benefits analysis. Empirical Softw. Eng. 11 (1), 33–70.
Ebert, C., Dumke, R., 2010. Software Measurement: Establish–Extract–Evaluate–Execute. Springer, Berlin.
Ehrlich, W., Prasanna, B., Stampfel, J., Wu, J., 1993. Determining the cost of a stop-testing decision. IEEE Softw. 10 (2), 33–42.
Elbaum, S., Rothermel, G., Kanduri, S., Malishevsky, A.G., 2004. Selecting a cost-effective test case prioritization technique. Softw. Quality J. 12 (3), 185–210.
Ellims, M., Bridges, J., Ince, D.C., 2006. The economics of unit testing. Empirical Softw. Eng. 11 (1), 5–31.
Engel, A., Last, M., 2007. Modeling software testing costs and risks using fuzzy logic paradigm. J. Syst. Softw. 80 (6), 817–835.
Franz, L.A., Shih, J.C., 1994. Estimating the value of inspections and early testing for software projects. Hewlett-Packard J. 45 (6), 60–67.
Freimut, B., Briand, L.C., Vollei, F., 2005. Determining inspection cost-effectiveness by combining project data and expert opinion. IEEE Trans. Softw. Eng. 31 (12), 1074–1092.
Galin, D., 2003. Software Quality Assurance. Pearson, Harlow.
Galin, D., 2004. Toward an inclusive model for the costs of software quality. Softw. Quality Professional 6 (4), 25–31.
Geist, R., Chetuparambil, M., Hedetniemi, S., Turner, A.J., 1996. Computing research programs in the U.S. Commun. ACM 39 (12), 96–99.
Glass, R.L., Ramesh, V., Vessey, I., 2004. An analysis of research in computing disciplines. Commun. ACM 47 (6), 89–94.
Grady, R.B., von Slack, T., 1994. Key lessons in achieving widespread inspection use. IEEE Softw. 11 (4), 46–57.
Granja-Alvarez, J.C., Barranco-Garcia, M.J., 1997. A method for estimating maintenance cost in a software project: A case study. J. Softw. Maint.: Res. Pract. 9 (3), 161–175.
Grottke, M., Graf, C., 2009. Modeling and predicting software failure costs. In: Proc. 33rd Annual IEEE International Computer Software and Applications Conference ,. IEEE Computer Society, Los Alamitos, pp. 180–189.
Gutjahr, W.J., 1995. Optimal test distributions for software failure cost estimation. IEEE Trans. Softw. Eng. 21 (3), 219–228.
Hansen, B., Rose, J., Tjornehoj, G., 2004. Prescription, description, reflection: the shape of the software process improvement field. Int. J. Inf. Manage. 24 (6), 457–472.
Hewett, R., Kijsanayothin, P., 2009. On modeling software defect repair time. Empirical Softw. Eng. 14 (2), 165–186.
Hollingsworth, I.P., Keogh, W., Atkins, M.H., 1999. Applying quality costs in a software development environment. Aust. J. Inf. Syst. 6 (2), 64–75.
Horngren, C.T., Foster, G., Datar, S.M., Rajan, M., Ittner, C., 2008. Cost Accounting—A Managerial Emphasis, 13th edition. Prentice-Hall, Englewood Cliffs.
Hou, R.-H., Kuo, S.-Y., Chang, Y.-P., 1996. Needed resources for software module test using the hyper-geometric software reliability growth model. IEEE Trans. Rel. 45 (4), 451–459.
Houston, D., Keats, J.B., 1998. Cost of software quality: a means of promoting software process improvement. Quality Eng. 10 (3), 563–573.
Hsia, P., Hsu, C.-T., Kung, D.C., Byrne, E.J., 1998. Incremental delivery reduces maintenance cost: a COCOMO-based study. J. Softw. Maint.: Res. Pract. 10 (4), 225–247.
Huang, C.-Y., 2006. Cost-reliability-optimal release policy for software reliability models incorporating improvements in testing efficiency. J. Syst. Softw. 77 (2), 139–155.
Huang, C.-Y., Lyu, M.R., 2005. Optimal release time for software systems considering cost, testing-effort, and test efficiency. IEEE Trans. Rel. 54 (4), 583–591.
Hullocker, C.P., 1986. Finding the cost of software quality. IEEE Trans. Eng. Manage. 33 (4), 223–228.
Issa, A.A., Abu Rub, F.A., Thabata, F.F., 2009. Using test case patterns to estimate software development and quality management cost. Softw. Quality J. 17 (3), 263–281.
Ittner, C.D., 1999. Activity-based costing concepts for quality improvement. Eur. Manage. J. 17 (5), 492–500.
Jones, A., Tabberer, M., 1993. Process quality costing adapted to software development. Empirical Softw. Eng. 2 (3), 199–208.
Jones, C., 1996. The economics of software process improvement. IEEE Comp. 29 (1), 95–97.
Jorgensen, M., Shepperd, M., 2007. A systematic review of software development cost estimation studies. IEEE Trans. Softw. Eng. 33 (1), 33–53.
Karg, L.M., Beckhaus, A., 2007. Modelling software quality costs by adapting established methodologies of mature industries. In: Proc. 2007 IEEE International Conference on Industrial Engineering and Engineering Management ,. IEEE, Piscataway, pp. 267–271.
Khoshgoftaar, T.M., Allen, E.B., Jones, W.D., Hudepohl, J.P., 2001. Cost-benefit analysis of software quality models. Empirical Softw. Eng. 9 (1), 9–30.
Kimura, M., Toyota, T., Yamada, S., 1999. Economic analysis of software release problems with warranty cost and reliability requirement. Reliability Eng. Syst. Saf. 66 (1), 49–55.
King, S., Hammond, J., Chapman, R., Pryor, A., 2000. Is proof more cost-effective than testing? IEEE Trans. Softw. Eng. 26 (8), 675–686.
Kitchenham, B., Brereton, O.P., Budgen, D., Turner, M., Bailey, J., Linkman, S., 2009. Systematic literature reviews in software engineering—a systematic literature review. Inf. Softw. Technol. 51 (1), 7–15.
Kitchenham, B., Charters, S., 2007. Guidelines for performing systematic literature reviews in software engineering. Tech. Rep. EBSE 2007-001, Keele University and Durham University Joint Report.
Knox, S.T., 1993. Modeling the cost of software quality. Digital Tech. J. 5 (4), 9–17.
Krasner, H., Houston, D., 1998. Using the cost of quality approach for software. CrossTalk, J. Defense Softw. Eng. 11 (11), 6–11.
Kusumoto, S., Matsumoto, K., Kikuno, T., Torii, K., 1992. A new metric for cost effectiveness of software reviews. IEICE Trans. Inf. Syst. E75-D (5), 674–680.
Laitenberger, O., 2001. Cost-effective detection of software defects through perspective-based inspections. Empirical Softw. Eng. 6 (1), 81–84.
Leach, R.J., 1996. Methods of measuring software reuse for the prediction of maintenance effort. J. Softw. Maint.: Res. Pract. 8 (5), 309–320.
Leung, Y.-W., 1992. Optimum software release time with a given cost budget. J. Syst. Softw. 17 (3), 233–242.
Lim, W.C., 1994. Effects of reuse on quality, productivity, and economics. IEEE Softw. 11 (5), 23–30.
Lin, C.-T., Huang, C.-Y., 2008. Enhancing and measuring the predictive capabilities of testing-effort dependent software reliability models. J. Syst. Softw. 81 (6), 1025–1038.
Mandeville, W., 1990. Software cost of quality. IEEE J. Selected Areas Commun. 8 (2), 315–318.
McCann, R.T., 2001. How much code inspection is enough? CrossTalk, J. Defense Softw. Eng. 14 (7), 9–12.
Middleton, P., Sutton, J., 2005. Lean Software Strategies. Productivity Press.
Mohagheghi, P., Conradi, R., 2007. Quality, productivity and economic benefits of software reuse: a review of industrial studies. Empirical Softw. Eng. 12 (5), 472–516.
Mylonopoulos, N.A., Theoharakis, V., 2001. On site: global perceptions of IS journals. Commun. ACM 44 (9), 29–33.
Okumoto, K., Goel, A., 1980. Optimal release time for software systems based on reliability and cost criteria. J. Syst. Softw. 1 (4), 315–318.
O'Neill, D., 2003. Determining return on investment using software inspections. CrossTalk, J. Defense Softw. Eng. 16 (3), 16–21.
Pettijohn, C.L., 1986. Achieving quality in the development process. AT&T Tech. J. 65 (2), 85–93.
Pham, H., 1996. A software cost model with imperfect debugging, random life cycle and penalty cost. Int J. Syst. Sci. 27 (5), 455–463.
Pham, H., 2006. System Software Reliability. Springer, Heidelberg.
Pham, H., Zhang, X., 1999. A software cost model with warranty and risk costs. IEEE Trans. Comput. 48 (1), 71–75.
Porter, A.A., Siy, H.P., Toman, C.A., Votta, L.G., 1997. An experiment to assess the cost-benefits of code inspections in large scale software development. IEEE Trans. Softw. Eng. 23 (6), 329–346.
Prahalad, C.K., Krishnan, M.S., 1999. The new meaning of quality in the information age. Harvard Bus. Rev. 77 (5), 109–118.

Pressman, R.S., 2010. Software Engineering: A Practitioner's Approach, 7th edition. McGraw-Hill, New York.
Rai, A., Song, H., Troutt, M., 1998. Software quality assurance: an analytical survey and research prioritization. J. Syst. Softw. 40 (1), 67–83.
Rivard, E., Kaiser, K., 1989. The benefits of quality IS. Datamation 35 (1), 53–58.
Rothermel, G., Elbaum, S., Malishevsky, A.G., Kallakuri, P., Qiu, X., 2004. On test suite composition and cost-effective regression testing. ACM Trans. Softw. Eng. Methodol. 13 (3), 277–331.
RTI, 2002. The economic impacts of inadequte infrastructure for software testing. Planning report 02-3, National Institute of Standards and Technology, Gaithersburg, MD.
Sabaliauskaite, G., Kusumoto, S., Inoue, K., 2004. Extended metrics to evaluate cost effectiveness of software inspections. IEICE Trans. Inf. Syst. E87-D (2), 475–480.
Schach, S.R., 1994. The economic impact of software reuse on maintenance. J. Softw. Maint.: Res. Pract. 6 (4), 185–196.
Schiffauerova, A., Thomson, V., 2006. A review of research on cost of quality models and best practices. Int. J. Quality Reliability Manage. 23 (6), 647–669.
Sherer, S.A., 1991. A cost-effective approach to testing. IEEE Softw. 8 (2), 34–40.
Simmons, P., 1996. Quality outcomes: determining business value. IEEE Softw. 13 (1), 25–32.
Singpurwalla, N.D., 1991. Determining an optimal time interval for testing and debugging software. IEEE Trans. Softw. Eng. 17 (4), 313–319.
Slaughter, S.A., Harter, D.E., Krishnan, M.S., 1998. Evaluating the cost of software quality. Commun. ACM 41 (8), 67–73.
Sneed, H.M., 1991. Economics of software re-engineering. J. Softw. Maint.: Res. Pract. 3 (3), 163–182.
Song, Q., Shepperd, M., Cartwright, M., Mair, C., 2006. Software defect association mining and defect correction effort prediction. IEEE Trans. Softw. Eng. 32 (2), 69–82.
Spiewak, R., McRitchie, K., 2008. Using software quality methods to reduce cost and prevent defects. CrossTalk, J. Defense Softw. Eng. 21 (12), 23–27.
Stewart, N., 1988. Software error costs. Quality Progress 21 (11), 48–49.
Teng, X., Pham, H., 2004. A software cost model for quantifying the gain with considerations of random field environments. IEEE Trans. Comput. 53 (3), 380–384.
Tse, T.H., Chen, T.Y., Glass, R.L., 2006. An assessment of systems and software engineering scholars and institutions (2000–2004). J. Syst. Softw. 79 (6), 816–819.
van Megen, R., Meyerhoff, D.B., 1995. Costs and benefits of early defect detection: experiences from developing client server and host applications. Softw. Quality J. 4 (4), 247–256.
van Solingen, R., 2004. Measuring the ROI of software process improvement. IEEE Softw. 21 (3), 32–38.
Ward, W.T., 1991. Calculating the real cost of software defects. Hewlett-Packard J. 42 (5), 55–58.
Webb, G., Patton, N., 2008. Quality and cost—it's not either/or: making the case with cost of quality. CrossTalk, J. Defense Softw. Eng. 21 (11), 23–28.
Westland, J.C., 2002. The cost of errors in software development: evidence from industry. J. Syst. Softw. 62 (1), 1–9.
Weyuker, E.J., 1990. The cost of data flow testing: an empirical study. IEEE Trans. Softw. Eng. 16 (2), 121–128.
Weyuker, E.J., 1996. Using failure cost information for testing and reliability assessment. ACM Trans. Softw. Eng. Methodol. 5 (2), 87–98.
Weyuker, E.J., 1999. Evaluation techniques for improving the quality of very large software systems in a cost-effective way. J. Syst. Softw. 47 (2–3), 97–103.
Whittaker, J.A., Voas, J.M., Nov/Dec 2002. 50 years of software: key principles for quality. IT Pro, 28–35.
Williams, A., Wiele, A., Dale, B., 1999. Quality costing: a management review. Int. J. Manage. Rev. 1 (4), 441–460.
Wohlin, C., Koerner, U., 1990. Software faults: spreading, detection and costs. Softw. Eng. J. 5 (1), 33–42.
Yamada, S., Ohtera, H., Narihisa, H., 1986. Software reliability growth models with testing effort. IEEE Trans. Rel. 35 (1), 19–23.
Yamada, S., Osaki, S., 1985. Cost-reliability optimal release policies for software systems. IEEE Trans. Rel. 34 (4), 422–424.
Yamada, S., Osaki, S., 1987. Optimal software release policies with simultaneous cost and reliability requirements. Eur. J. OR 31 (1), 46–51.
Yong, J., Wilkinson, A., 2002. The long and winding road: the evolution of quality management. Total Quality Manage. 13 (1), 101–121.
Zhang, X., Pham, H., 1998. A software cost model with error removal times and risk costs. Int J. Syst. Sci. 29 (4), 435–442.

Lars M. Karg holds a Diplom degree in Information Systems from the Darmstadt University of Technology and a Ph.D. from the University of Erlangen-Nuremberg. His research interests are directed to information systems development, software engineering economics, and quality cost accounting. In his research career, he has (co-)authored several research papers presented at conferences including COMPSAC, IEEM, and HICSS.

Michael Grottke holds an M.A. in Economics from Wayne State University and a Diplom degree in Business Administration as well as a Ph.D. from the University of Erlangen-Nuremberg. Between 2004 and 2007, he was a Research Associate and Assistant Research Professor in the Department of Electrical and Computer Engineering at Duke University. In 2010, he received the Habilitation degree from the University of Erlangen-Nuremberg. His research interests include software performance and dependability, software engineering economics, and stochastic modeling.

Arne Beckhaus holds a Diplom degree in Information Systems from the Darmstadt University of Technology and a Ph.D. from the University of Freiburg. His research interests are focused on software engineering economics and organization of information systems development. He has (co-)authored diverse research publications published in, e.g., Issues in Information Systems and the conference proceedings of HICSS and COMPSAC.