The Journal of Systems and Software 86 (2013) 951–969

Contents lists available at SciVerse ScienceDirect

The Journal of Systems and Software

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

Evaluation studies of software testing research in Brazil and in the world: A survey of two premier software engineering conferences

Otávio Augusto Lazzarini Lemos[a,*], Fabiano Cutigi Ferrari[b], Marcelo Medeiros Eler[c], José Carlos Maldonado[c], Paulo Cesar Masiero[c]

[a] Departamento de Ciência e Tecnologia, Universidade Federal de São Paulo (UNIFESP), São José dos Campos, SP, Brazil
[b] Departamento de Computação, Universidade Federal de São Carlos (UFSCar), São Carlos, SP, Brazil
[c] Departamento de Sistemas de Computação, Universidade de São Paulo (ICMC/USP), São Carlos, SP, Brazil

ARTICLE INFO

Article history:
Received 21 January 2012
Received in revised form 19 October 2012
Accepted 19 November 2012
Available online 11 December 2012

Keywords:
Software testing
Evaluation studies
Software testing research in Brazil

ABSTRACT

This paper reports on a historical perspective of the evaluation studies present in software testing research published in the Brazilian Symposium on Software Engineering (SBES) in comparison to the International Conference on Software Engineering (ICSE). The survey characterizes the software testing-related papers published in the 25-year history of SBES, investigates the types of evaluation presented in these publications, and how the rate of evaluations has evolved over the years. A similar analysis within the same period is made for ICSE, allowing for a comparison between the national and international scenario. Results show that the rate of papers that present evaluation studies in SBES has significantly increased over the years. However, among the papers that described some kind of evaluation, only around 20% performed more rigorous evaluations (*i.e.* case studies, quasi experiments, or controlled experiments). Such percentage is low when compared to ICSE, which presented 40% of papers with more rigorous evaluations within the same period. Nevertheless, we noticed that both venues still lack the publication of research reporting controlled experiments: only a single paper in each conference presented this type of evaluation.

1. Introduction

There is a common knowledge that science in general develops through theory and experimentation. Theory tries to define the nature and causes of a problem, while experimentation may confirm or refute such definitions. On the other hand, experimentation may also find new phenomena that can be explained through a theory (Feitelson, 2007). Therefore, the application of experiments and other types of evaluation is considered essential to the development of any scientific field.

Several computer science researchers feel that our field has not yet insisted enough on experimentation (*e.g.* Freeman, 2008). This is also true with respect to the software engineering (SE) community. Victor Basili is one of the voices that have been insisting on a more mature field of SE, where rigorous evaluation is part of any developed research. In fact, although the origins of SE can be dated back to the famous 1968 NATO conference, it cannot be said to have become an empirical science until the 1970s, with the advent of Basili's work (Boehm et al., 2005). Nevertheless, even though there has long been a positive encouragement on the application of adequate evaluation in SE, a 2005 survey showed that only 1.9% of the work published on prestigious SE venues until that date have applied controlled experiments (Sjoberg et al., 2005).

The Brazilian Symposium on Software Engineering (SBES, from the Portuguese acronym) is the premier Brazilian SE conference. SBES has been held annually since 1987, summing up 25 editions to date. Lately, the conference has been gathering nearly 500 people, including researchers, students, and practitioners working on the field (Garcia, 2011). If the Brazilian symposium is to grow into a respectable community, we must also take into account how research work published in SBES is being evaluated.

Software testing is a very important and prominent SE subfield and several papers published in the SBES history fall into this category. Since cost constraints and high effectiveness goals are common within software testing, every novel approach has to be adequately evaluated according to these characteristics to be deemed useful or not. In fact, experimentation and other types of evaluation is an essential part of research in software testing (Andrews et al., 2005). For instance, at many times one is interested in comparing the fault-detection effectiveness of testing criteria used to derive test cases. In this case experimentation is a handy tool to obtain evidence about this question (Andrews et al., 2005).

* Corresponding author.
E-mail addresses: otavio.lemos@unifesp.br (O.A.L. Lemos), fabiano@dc.ufscar.br (F.C. Ferrari), mareler@icmc.usp.br (M.M. Eler), jcmaldon@icmc.usp.br (J.C. Maldonado), masiero@icmc.usp.br (P.C. Masiero).

0164-1212/$ – see front matter © 2012 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.jss.2012.11.040

In this paper we present a survey that serves as an initial assessment of the dissemination of adequate evaluation of software testing research in SBES. We have analyzed the 25 available SBES proceedings and characterized the evaluation studies presented in papers related to software testing. We categorized the papers within the software testing field, collected information about authors and affiliations, and classified the presented evaluations, looking into its evolution along the symposium's history. To put this data into a global context, we have also analyzed the proceedings of the International Conference on Software Engineering (ICSE). The ICSE survey included proceedings of the same period of the SBES study – 1987–2011 –, to be able to compare results.

Our assessment shows that the number of evaluations presented in SBES papers is increasing significantly along the years. However, our data also shows that there is still room for improvement in this area, specially with respect to the rigor of the conducted studies. In particular, we notice a lack of publications reporting controlled or quasi experiments involving software testing research. With respect to the ICSE analysis, our data shows that the international conference is more mature in this sense, since it presents 40% of testing papers reporting more rigorous studies. However, we noticed that both venues still lack the publication of research reporting controlled experiments: only a single paper in each conference presented this type of evaluation.

The analysis presented in this paper is important as a self-assessment of both Brazilian and international software testing communities, with respect to research evaluation. It also serves as a guide into how researchers that want to publish papers in prestigious venues should conduct and present studies that assess their work. The remainder of this paper is structured as follows. Section 2 presents basic concepts about evaluation in SE and software testing and Section 3 describes the research method used to select and classify the papers in the survey and other characteristics of our study. Section 4 presents the results of our survey and Section 5 discusses such results. Finally, Section 6 presents related work and Section 7 concludes the paper.

2. Background

There are many types of evaluation studies that can be applied to software engineering research. Based on previous empirical software engineering literature, Zannier et al. (2006) classifies them into the following: *Controlled experiment*, *Quasi experiment*, *Case study*, *Exploratory case study*, *Experience report*, *Meta-analysis*, *Example application*, *Survey*, and *Discussion*. Each of these types has different characteristics and level of rigor. In this paper we use the same classification to characterize evaluation studies in software testing research published in SBES and ICSE. In the following we synthesize Zannier et al.'s classification.

Controlled experiments apply random assignment of treatments to subjects, contain large sample sizes – generally > 10 subjects –, formulate hypotheses, select an independent variable, and sometimes apply random sampling[1]; while *Quasi experiments* are controlled experiments with one or more of its characteristics missing. *Controlled experiments* also usually define research questions, which are later answered based on the reached conclusions. In software testing, research questions are usually formulated based on a comparison between different approaches, with respect to their effectiveness or application effort, for instance. To give an example, the *Controlled experiment* reported by Rothermel et al. (2000) formulates the following research question: "*Do programmers who use our testing methodology create test suites that are more effective in terms of du-adequacy than programmers who use an ad hoc approach?*"

From a statistical point of view, a simple observation of the means or medians from sample observations is not enough to infer about the actual populations. For instance, while comparing two testing approaches, the fact that on average one gives better results than the other according to some metric might only be a coincidence caused by random sampling. Therefore, *Controlled experiments* also usually apply statistical hypothesis tests, to check whether the observed differences are in fact significant. A common statistical tool used in such experiments is the *analysis of variance* (ANOVA), which supports testing whether or not the means of several groups are all equal. The ANOVA is a generalization of the *t*-test, which is also commonly used when there are only two groups being compared. In fact, the two *Controlled experiments* found in our survey (Rothermel et al., 2000; Campanha et al., 2010) apply the ANOVA for their experimental analyses. Some *Quasi experiments* included in our study also apply statistical tests (*e.g.* Kim and Porter, 2002).

With respect to the differences between *Controlled* and *Quasi* experiments, a common missing characteristics of a *Quasi experiment* report is failing to formally define hypotheses and to test them later. One such example included in our survey is the study reported by Wong et al. (1995), published in ICSE 1995. Other studies fail to be categorized as controlled experiments because the sample size is small. For example, the study reported by Graves et al. (1998) in ICSE 1998 included only 6 small subject programs, and therefore was categorized as a *Quasi experiment* according to our classification.

Case studies state a research question and unit(s) of analysis, report a logic link between data and propositions, provide criteria for interpreting findings, and are performed in real-world scenarios; while *Exploratory case studies* are case studies with one or more of its characteristics missing. A common missing characteristic of *Exploratory case studies* is failing to be performed in a real-world setting. There are several studies classified in our survey as exploratory case studies due to the absence of such characteristic. One such example is the study reported by Deng et al. (2005) in ICSE 2005. The proposed approach was analyzed on example applications provided by the JDBC tutorial and other publicly available benchmarks, and thus not in a real-world scenario.

Experience reports are retrospective reports with no propositions, do not necessarily contain answers to how or why some findings were attained, and often include lessons learned. *Meta-Analyses* analyze a body of similar studies to reach a common result; *Example applications* only describe an application to assist the definition of the approaches (these are commonly alleged as "evaluations" or "validations" of the study). *Surveys* collect answers to structured or unstructured questionnaires given to participants; while *Discussions* provide qualitative, textual, and opinion-related evaluation.

It is important to notice that, in general, SE is regarded as a discipline that needs to improve on the use of experiments and more rigorous forms of evaluation. However, as reported by Zannier et al. (2006), the community seems to be evolving significantly with this respect over the years. For instance, over the lifetime of the International Conference on Software Engineering (ICSE) until 2006, there was a significant increase in the number of papers with an evaluation component. If this is true with respect to the international community, an important question is whether it also holds to more

[1] In the original classification by Zannier et al. (2006), controlled experiments necessarily apply random sampling. In our classification we have relaxed such characteristic, because it is usually hard to contemplate it on software engineering experiments. Availability sampling is frequently applied, using, for instance, open source projects that are more readily available. We have also defined a *large* sample size more loosely, since it may vary depending on the study context. For instance, three very large software projects may be considered an adequate sample size for experimenting with a software testing approach.

local communities such as SBES, with respect to more specific fields such as Software Testing.

2.1. Software testing research and evaluation

Software testing can be defined as the execution of a program against test cases with the intent of revealing faults (Myers et al., 2004). The different testing techniques are defined based on the artifact used to derive test cases. Functional – or black-box – testing derives test cases from the specification or description of a program; structural – or white-box – testing derives test cases from implementations; fault-based testing derives test cases from fault models based on common mistakes committed by programmers; and model-based testing derives test cases from system specification models. To deem a software system *correct*, one could test every possible element of the system's input domain and check whether the output is consistent with the expected output. However, even for simple programs this is usually infeasible, because the input domains tend to be very large (imagine, for instance, the input space of a compiler system) (Myers et al., 2004). Therefore, a large portion of testing research focus on proposing ways to select meaningful subsets of test cases to enhance the chance of revealing faults. Based on the categories of testing techniques described above, several testing selection criteria were proposed (Mathur, 2007).

Besides testing techniques and criteria, there are many other aspects involved in the testing activity. For instance, in general, it is too expensive to test programs manually; therefore, software testing usually relies on tools to automate the test case generation, execution, and results gathering. After faults are revealed while testing the programs, they must be localized and fixed. This activity is usually not included under the *software testing* activity, being called *debugging*. Since it is closely related to testing, we decided to include papers concerned with it in our survey. Other topics that are important to software testing and were included are the following: *fault-injection*,[2] which consists in intentionally introducing known failures into the system during its execution to evaluate if the system is robust enough to recover without crashing (Hsueh et al., 1997); *regression testing*, which consists in selectively retesting a system to verify whether modifications have not caused unwanted effects (IEEE, 1990); and *testing strategy*, which consists in the way by which test case design methodologies are combined to provide an effective testing activity (Myers et al., 2004).

Different types of software testing research reclaim different types of evaluation. Some proposals might be easier to evaluate, while others might require more work. For instance, evaluating the effectiveness of a testing criterion might require the use of real applications, a large pool of test cases, and random selection of tests not to introduce bias in the test case generation (for instance, as done by Lai et al. (2008)). In other cases, it might require only the simulation of an algorithm with different configurations. For instance, in the case of some approaches to automated test case generation, evaluation may consists only in running an implementation with different configurations and comparing the outcomes, which is an experiment easier to configure than a test criteria study. Moreover, experiments that require human subjects are also harder to setup than evaluations that can be completely automated. In any case, evaluation studies are very important for software testing, because we need approaches that are at the same time effective but also feasible. A researcher can only have evidence that a testing approach is useful or not only when it is adequately measured with respect to effectiveness and effort factors.

[2] Usually related to the system's *fault tolerance*.

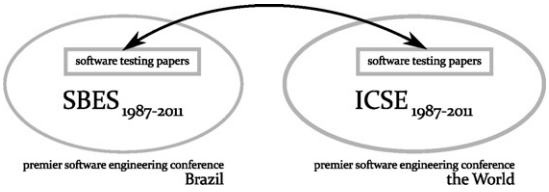


Fig. 1. The scope of our survey.

3. Study setup

3.1. Research goals

Our main goal is to investigate the dissemination of software testing evaluations in SBES and compare them to the ICSE scenario. We assess the increase in performed evaluations in terms of the percentage of papers with an evaluation component over the total number of published papers in a year. We define a paper as containing an evaluation component when it presents at least a study involving subjects – humans, programs, or specifications – and not only a single example application (studies that do not fall into the category of "*Example application*" as defined in Section 2). Note that we have not applied statistical analyses because we are dealing with the entire population of software testing papers accepted at each event. Statistical tests make sense when the researcher wants to verify whether results on a given sample may generalize to the whole population. In that sense, the analyses made here are factual and not statistical.

A complementary goal of this survey is to characterize the software testing community that publishes papers both in SBES and ICSE. We do this by analyzing authors, schools, and topics involved in the selected publications.

The scope of our survey is depicted in Fig. 1. The questions we address are targeted at the software testing papers published both in SBES and ICSE within the period of 1987–2011, the history of SBES. Comparisons are made between the Brazilian and international scenarios, restricted to the communities of the target conferences.

3.2. Paper selection

The selection of the papers analyzed in this study was based on the proceedings of the 25 SBES and ICSE editions, from 1987 to 2011. For SBES, the papers published from 1987 to 1998, in 2000 and in 2003 are available only in printed format. Papers published from 1999 to 2008 (apart from 2000 and 2003) are also available online.[3] As of 2009, the SBES proceedings are also available at the IEEE Digital Library. [4,5,6] On the other hand, all papers published in ICSE from 1988 to 2011 are available at the ACM and IEEE Digital Libraries.[7,8] The 1987 proceedings are available at the ACM Digital Library.[9]

The paper selection process was inspired by a process for running systematic mapping studies (Petersen et al., 2008). The first three authors of this paper performed the paper selection

[3] http://www.lbd.dcc.ufmg.br:8080/bdbcomp/servlet/PesquisaEvento?evento=sbes – accessed 19.10.12.
[4] http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=5336057 – accessed 19.10.12.
[5] http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=5628346 – accessed 19.10.12.
[6] http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=6063704 – accessed 19.10.12.
[7] http://dl.acm.org/event.cfm?id=RE228 – accessed 19.10.12.
[8] http://ieeexplore.ieee.org/xpl/conhome.jsp?punumber=1000691 – accessed 19.10.12.
[9] http://dl.acm.org/citation.cfm?id=800054 – accessed 19.10.12.

iteratively. In the first iteration we used an inclusion criterion which defines that relevant papers must be related to software testing. Firstly, we performed a preliminary analysis of the papers published in sessions related to *Verification, Validation and Testing* (VV&T) of the main tracks of each SBES and ICSE proceedings. In the next step, we searched for papers related to software testing in the remaining parts of the proceedings, since some testing papers were allocated to other sessions (*e.g.* a paper on testing aspect-oriented programs can be allocated to the AOP session). Note that we have considered neither SBES nor ICSE satellite events such as Tool Sessions and collocated workshops, given that our main goal was to evaluate the conferences' main tracks as a vehicle to disseminate testing-related research that performs some kind of evaluation.

The second iteration was carried out by the same authors, hereafter called the *reviewers*. The identified papers were distributed amongst reviewers so that they could read the title, abstract and introduction aiming at identifying the papers that contained an evaluation component. In the third iteration the reviewers performed further analysis of the papers identified in the previous iteration to exclude "false positives" (*e.g.* papers that addressed bug fixing – *i.e.* maintenance – or other organizational matters). At this point, for each paper we collected relevant information into tables. Extracted details included authors' names, affiliations, testing approach addressed by the paper and, when applicable, the type and attributes of the reported evaluation. The next section presents some details about the classification schema we applied for the selected papers.

3.3. Paper classification

Since there are many elements involved in software testing, there are also several types of testing-related publications. Some of them focus on the proposal of a testing criterion, others focus on automating some aspect of the testing activity. There are also papers that evaluate testing criteria or varied testing strategies. Therefore, in a survey like the one reported herein, the large range of topics covered by software testing papers requires the adoption of some classification system that enables us to categorize the publications.

We classified the testing-related papers published in SBES and ICSE according to two dimensions: *Technique* and *Type*. The first addresses the main testing-related technique investigated in a paper. Examples are white-box testing and automated test case generation. The *Type* dimension characterizes a paper according to its nature. While a paper may propose a software testing approach such as a novel family of criteria, another may be concerned with evaluating such family of criteria with respect to its efficacy and effectiveness.

The categories related to *Technique* are the following:

A:	Automated test case generation
B:	Black-box (functional) testing
D:	Debugging
F:	Fault-based testing
I:	Fault Injection and fault tolerance
M:	Model-based testing
R:	Regression testing
S:	Testing Strategy
W:	White-box (structural) testing

The categories related to *Type* are the following:

A:	Approach proposal
E:	Evaluation, when the paper evaluates some aspect of software testing
T:	Tool, when the paper describes some testing tool or testing infrastructure implementation

As we will see in the next section, in some cases a paper can be classified into two categories of *Technique*. For example, a paper that describes an approach for deriving functional test cases based on the system's models is classified as **B** and **M**. Nevertheless, as far as possible we tried to assign a single category to each paper, according to the best related technique.

With respect to *Type*, we classified the papers according to their main contribution. For instance, in some cases a paper may propose a testing approach and at the same time evaluate it by means of an experiment. However, since the main contribution of the paper is the approach itself, we would classify such publication as an *approach proposal* paper, and not an *evaluation* paper.

4. Results and analysis

In this section we present the data gathered in our survey. We analyzed all available SBES proceedings from 1987 to 2011, and all ICSE proceedings of the same period. We then performed the selection process mentioned in the previous section. Firstly, we selected papers related to software testing; and secondly, we identified the ones that contain an evaluation component. Among the papers with an evaluation component, we then identified the ones that presented more rigorous evaluation studies.

4.1. Selected papers

From the available SBES proceedings, we selected 60 papers[10] that report on studies related to software testing. From the ICSE proceedings of the same period, we selected 111 software testing papers. Tables 11–16, located in Appendix A, present all papers and information about each. For each paper we present the year of publication, the title, the authors and their affiliation, the related testing technique and type, the evaluation type according to Zannier's classification (Zannier et al., 2006) (or n/a in the absence of an evaluation component), whether or not the paper includes an evaluation component according to the classification schema detailed in Section 3.1, the type of subjects evaluated in the paper (if any), and the number of citations to the paper gathered from Google Scholar.[11] Such information is used to characterize the software testing community that publishes in the conferences (Sections 4.2 and 4.3), and to analyze the evolution of software testing evaluation studies published in SBES and ICSE along the years (Sections 4.4 and 4.5). We also provide a broader view with respect of the relevance of software testing papers in each venue (Section 4.6).

4.2. Characterizing the community: authors

In this section we present the results of our survey with respect to the characterization of the community that has published software testing papers in SBES and ICSE in the period of 1987–2011. With respect to scholars, there are 89 authors that appear in the SBES software testing publications, and 278 in ICSE. Table 1 presents the top 15 ranked authors in SBES, and Table 2 present the top 16 ranked authors for ICSE.[12]

The tables include only authors with at least three software testing papers presented at the conferences' editions. To realize how the same authors evaluated their studies, the same tables include

[10] We selected 55 papers in our original study (Lemos et al., 2011). For this paper, a revision and an update of the dataset resulted in the inclusion of five papers: two of them were published in 1998, one in 1999 and two in 2011.

[11] All citation numbers included in this paper were gathered in 16.10.12 at http://scholar.google.com.

[12] Sometimes we list the top 15 authors/institutions, other times the top 16. This was done because there were ties in the top 15 list, which forced us to include an additional author or institution. For instance, in the case of ICSE authors, we list the top 16 to include all authors that have published three or more papers in the conference.

Table 1
Top 15 authors publishing software testing research in SBES (1987–2011).

Author	# Papers	# Evals	h-Index
J. C. Maldonado	33	15	24
M. Jino	13	5	11
P. C. Masiero	11	4	17
M. E. Delamaro	7	2	17
S. R. Vergilio	6	1	11
S. C. P. F. Fabbri	5	2	7
O. A. L. Lemos	5	2	8
A. S. Simão	5	2	6
A. M. R. Vincenzi	5	3	11
A. M. A. Price	5	0	3
S. R. S. Sousa	4	2	7
A. M. Crespo	3	3	4
A. Pasquini	3	3	9
E. Martins	3	2	12
M. L. Chaim	3	0	6

Table 2
Top 16 authors publishing software testing research in ICSE (1987–2011).

Author	# Papers	# Evals	h-Index
G. Rothermel	11	10	51
M. J. Harrold	6	4	57
A. Bertolino	5	1	25
M. M. Burnett	4	4	21
S. Elbaum	4	4	29
Y. Labiche	4	4	29
A. Orso	4	4	31
A. Porter	4	4	36
L. C. Briand	3	3	49
W. K. Chan	3	3	5
J. A. Clause	3	3	10
J. M. Kim	3	3	6
D. Leon	3	2	11
A. Podgurski	3	2	26
D. S. Rosenblum	3	2	34
M. L. Soffa	3	2	41

the number of papers that present an evaluation component ("*# evals*" column). As a publication impact analysis of the researchers, we have also added their h-index calculated according to Google Scholar.

Note that the h-index generated by Google Scholar is not completely precise (Jacso, 2009). One of the issues while generating the indexes is the occurrence of homonyms among researchers. To deal with this problem we have manually inspected the papers to check whether they were in fact published by the author. Papers that were published by homonyms were excluded in the analysis. Although the figures were cross-checked among the authors, a manual process can always incur in inconsistency. In general, we can see that the h-indexes vary much, and although ICSE top scholars obviously have higher figures, there are some Brazilian scholars with publication impact comparable to international researchers.

4.3. Characterizing the community: institutions

There are 30 institutions involved in the SBES papers included on our survey, and 109 in the ICSE papers. Table 3 presents the top 16 ranked institutions publishing in SBES, and Table 4 present the same data for ICSE. Similar to the data for the authors, the tables also show the number of papers that present an evaluation component. Note that for SBES, we show institutions that appear at least in two software testing papers. For ICSE, institutions with three or more papers are listed.

Table 3
Top 16 institutions publishing software testing research in SBES (1987–2011).

Institution	# Papers	# Evals
University of São Paulo (ICMC) (São Carlos – Brazi l)	38	17
State University of Campinas (FEEC) (Campinas – Brazil)	13	4
Federal University of Rio Grande do Sul (Porto Alegre – Brazil)	8	1
Federal University of São Carlos (São Carlos – Brazil)	5	2
Federal University of Paraná (Curitiba – Brazil)	4	1
Federal University of Campina Grande (Campina Grande – Brazil)	3	2
State University of Campinas (IC) (Campinas – Brazil)	3	2
State University of Maringá (Maringá – Brazil)	3	2
Federal University of Pernambuco (Recife – Brazil)	2	2
National Institute for Space Research (São José dos Campos – Brazil)	2	2
Purdue University (West Lafayette – USA)	2	2
Centro Universitário Eurípedes de Marília (Marília – Brazil)	2	1
Federal University of Technology – Paraná (Campo Mourão – Brazil)	2	1
Pontifical Catholic University of Rio Grande do Sul (Porto Alegre – Brazil)	2	1
State University of Ponta Grossa (Ponta Grossa – Brazil)	2	1
University of São Paulo (IFSC) (São Carlos – Brazi l)	2	0

Table 4
Top 15 institutions publishing software testing research in ICSE (1987–2011).

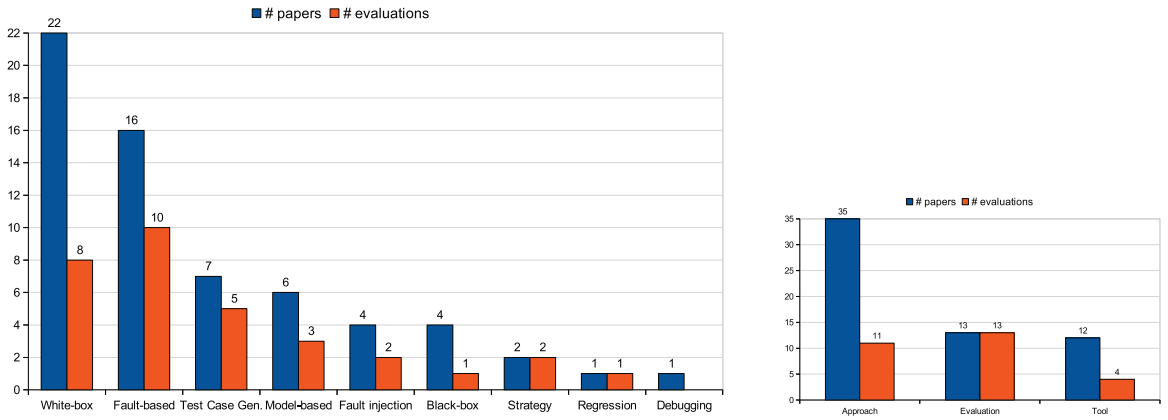
Institution	# Papers	# Evals
Oregon State University (Corvallis – USA)	10	10
Georgia Institute of Technology (Atlanta – USA)	6	6
University of Nebraska (Lincoln – USA)	6	6
University of Maryland (College Park – USA)	5	5
CNR (Pisa – Italy)	5	1
University of California (Berkeley – USA)	4	4
Ohio State University (Columbus – USA)	4	3
Case Western Reserve University (Cleveland – USA)	4	2
Carleton University (Ottawa – Canada)	3	3
City University of Hong Kong (Hong Kong – China)	3	3
Microsoft Research (Redmond – USA)	3	3
North Carolina State University (Raleigh – USA)	3	3
Purdue University (West Lafayette – USA)	3	2
University of Pittsburgh (Pittsburgh – USA)	3	2
Clemson University (Clemson – USA)	3	1

4.4. Characterizing the research topics

With respect to the covered topics and types of software testing papers published in SBES and ICSE, Figs. 2 and 3 present charts with the data for each axis of our classification system. Note that the top 4 covered topics in SBES were *White-box testing*, *Fault-based testing*, *Test case generation* and *Model-based testing*. In ICSE, *White-box testing*, *Test case generation*, *Testing strategy* and *Model-based testing* were the top 4 most investigate topics.

By analyzing Figs. 2 and 3, we can observe an overlapping of research interests; considering the full period (*i.e.* 1987–2011), 3 out of 4 topics are amongst the most investigated in both conferences. They are *White-box testing*, *Test case generation* and *Model-based testing*. Furthermore, with respect with the nature of the paper (*i.e. Approach proposal*, *Evaluation* or *Tool*), papers that describe approaches represent the great majority in our dataset. In total, approach proposal is the main topic of 58% (35 out of 60) of SBES papers and 65% (72 out of 111) of ICSE papers.

Our dataset shows the widely investigated topics within the software testing community that publishes in SBES and ICSE, and indicates topics that have been less covered. With respect to type, note that there are many more papers proposing approaches, and less focused on evaluations and tools. This also indicates a publication gap of experimentation papers, which are very important



(a) Papers by Technique (b) Papers by Type

Fig. 2. Charts of the covered topics and types of software testing papers in SBES (1987–2011).

in this field. Section 5 brings additional discussion regarding this issue.

4.5. Characterizing the evaluation studies

With respect to the evaluation studies present in the SBES surveyed papers, 35 out of 60 papers performed some kind of evaluation according to Zannier et al.'s classification (Zannier et al., 2006) (see "*Type of Eval. According to Zannier et al.*" column of Tables 11–16 in Appendix A). From this subset of 35 papers, 29 of them were categorized either as *Experiments*, *Quasi experiments*, *Case studies* or *Exploratory case studies*. Note that these 29 papers are marked with an "Y" in the "*Eval*" column of Tables 11–16. This means that approximately 50% of the whole set of SBES analyzed papers contained an evaluation component (*i.e.* not only an application example).

With respect to ICSE, we were able to classify 98 out of 111 papers according to Zannier et al.'s classification (Zannier et al., 2006). From them, 83 include evaluations characterized either as *Experiment*, *Quasi experiment*, *Case study* or *Exploratory case study*. Therefore, approximately 75% of the software testing papers published at ICSE from 1987 to 2011 present an evaluation component.

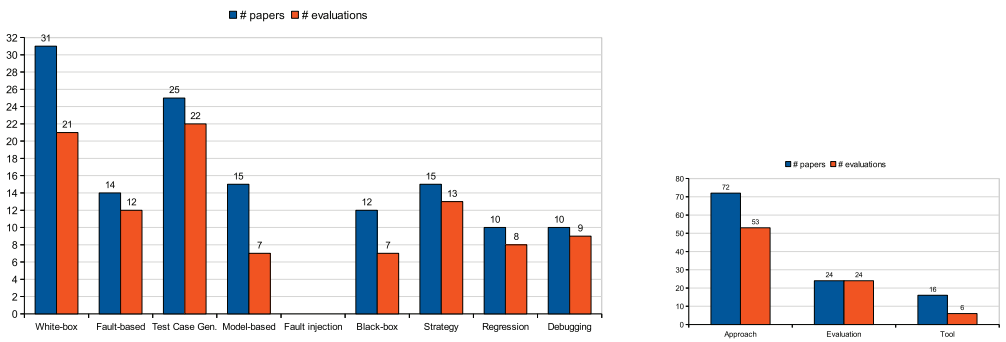
To show how the numbers of papers with evaluation studies have evolved over the addressed period in SBES and ICSE, we analyze the paper data aggregated per triennium. We did this because we noticed that an annual analysis would present too much variability. Table 5 shows the number of SBES papers that presented evaluation studies over the total number of published papers for each triennium. Table 6 shows the same type of data for ICSE. We covered all triennia from 1987 to 2011. As shown in Tables 11 and 12 (Appendix A), SBES editions 1991 and 1996 did not include any software testing paper. Note that we added the year 2011 to the last triennium for both conferences to avoid the need for a new group formed by a single year.

Table 5
Evolution of the evaluation studies in SBES (1987–2011).

Triennium	Eval. rate
1987–1989	0.00
1990–1992	0.00
1993–1995	0.28
1996–1998	0.30
1999–2001	0.58
2002–2004	0.56
2005–2007	0.64
2008–2011	0.82

Table 6
Evolution of the evaluation studies in ICSE (1987–2011).

Triennium	Eval. rate
1987–1989	0.67
1990–1992	0.67
1993–1995	0.52
1996–1998	0.48
1999–2001	0.67
2002–2004	0.77
2005–2007	1.00
2008–2011	1.00



(a) Papers by Technique (b) Papers by Type

Fig. 3. Charts of the covered topics and types of software testing papers in ICSE (1987–2011).

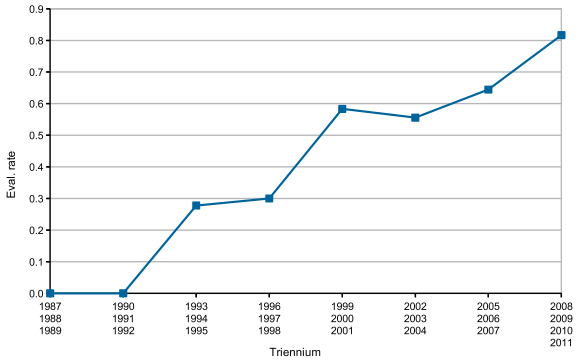


Fig. 4. Chart of the growth rate of papers with evaluation components per triennium in SBES (1987–2011).

The numbers presented in Tables 5 and 6 are graphically represented in Figs. 4 and 5. We draw lines between the data points only to provide an idea of the growth rate between periods. Note that the number of papers that present evaluation studies in both conferences have significantly increased over the triennia. In SBES, there is a noticeable upward trend, except for the 2002–2004 triennium, which is an interesting outlier. In ICSE, similar upward trend can also be noticed, having as outliers the period from 1993 to 1998. Section 5 provides an in-depth analysis of these numbers.

Fig. 6 and 7 present charts for SBES and ICSE with the distribution of evaluation studies among the categories we analyzed. Note that the mass majority of the SBES papers applied *Exploratory case studies* (24 papers, *i.e.* approximately 83%), while only 2 papers presented *Case studies* and other 2 *Quasi experiments*, and only 1 paper

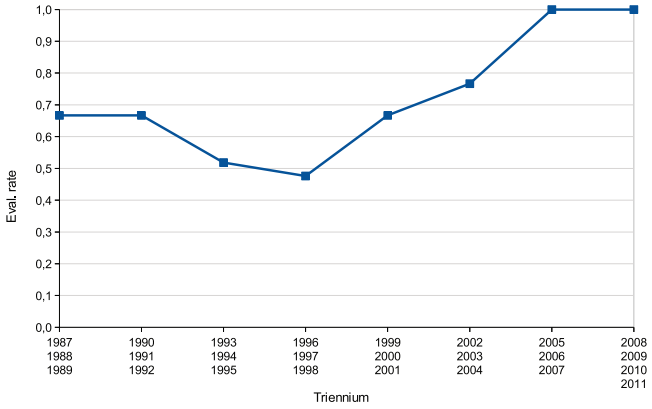


Fig. 5. Chart of the growth rate of papers with evaluation components per triennium in ICSE (1987–2011).

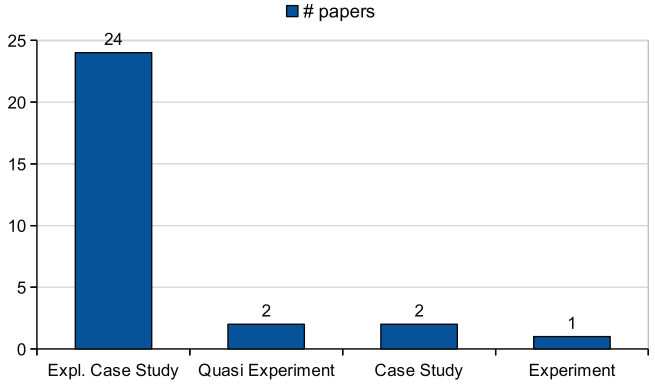


Fig. 6. Distribution of papers per evaluation type in SBES.

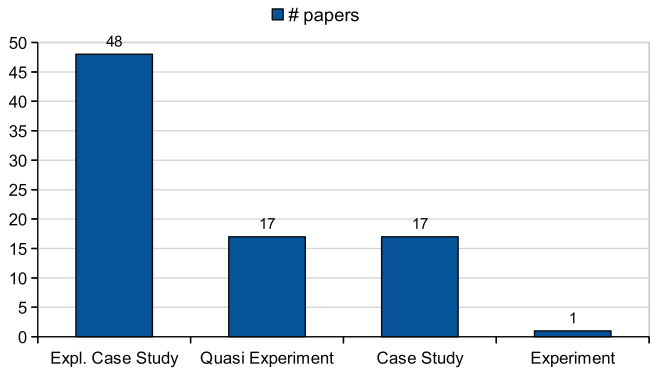


Fig. 7. Distribution of papers per evaluation type in ICSE.

presented a *Controlled experiment*. This shows that while SBES has promoted the increase in application of evaluation studies along the years, the rigor of these studies have not been strong. Similarly to the SBES-related data, most evaluations presented in ICSE papers are characterized as *Exploratory Case Studies*. In total, they represent nearly 58% (48 out of 83 papers). *Quasi experiments* and *Case studies* each represents approximately 20% of the evaluations, while only a single *Controlled experiment* related to software testing has been reported in the last 25 years. Section 5 discusses related issues in more detail.

4.5.1. Types of subjects

In both conferences, papers that report some kind of evaluation have mostly used *programs* as subjects for this purpose. These numbers are presented in Table 7. *Programs* represent 86% of the subject type in SBES and 92% in ICSE. Tables 11–16 in Appendix A show the types of subjects used in the studies (see the "*Type of subjects*" column).

To realize how each type of evaluation has been performed along the analyzed period, we distributed the SBES and ICSE papers that include an evaluation component over the triennia. Fig. 8 depicts such distribution. The most noticeable point in the graph regards the steep rise in the number of *Case Studies* reported in ICSE, which suggests that researchers are becoming more concerned about proving their theories in the industrial context. Besides this, we can observe that the number of *Exploratory Case Studies* reported in both conferences has increased significantly. Finally, there is also a growth in the number of quasi-experiments reported in ICSE in the last three triennia, even though there was a decrease in the last triennium. In general, these results corroborate our previous observations regarding the evaluation rate evolution.

4.6. Characterizing the relevance of software testing research

As a last analysis of the results, Table 8 summarizes the total number of papers published in ICSE and SBES by triennium, and the rate of software testing papers in the respective periods for each

Table 7
Types of subjects used in the evaluations in SBES and ICSE (1987–2011).

Subject type	# Papers	
	ICSE	SBES
Programs	77	25
Spreadsheets	3	0
People	2	1
Algorithms	1	0
Databases	1	0
Models	1	3
Testing criteria	1	0

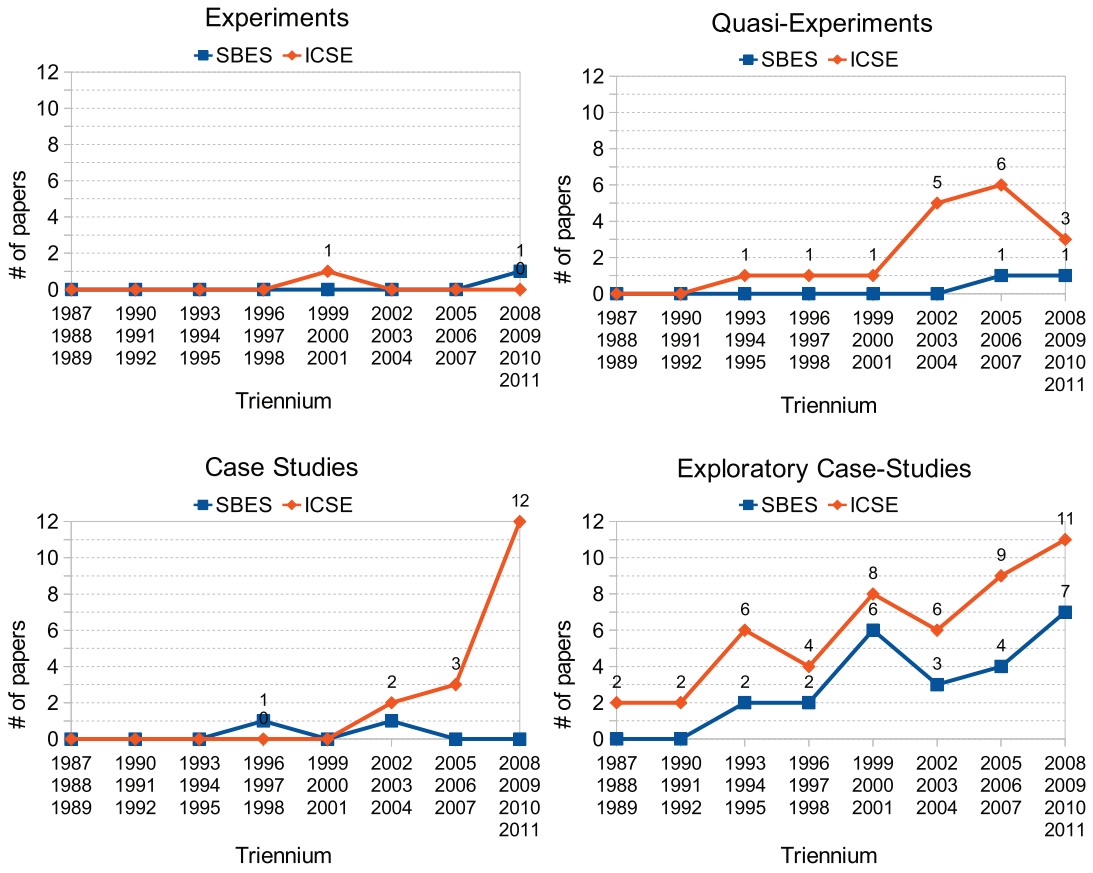


Fig. 8. Types of evaluation presented in both conferences per triennium (1987–2011).

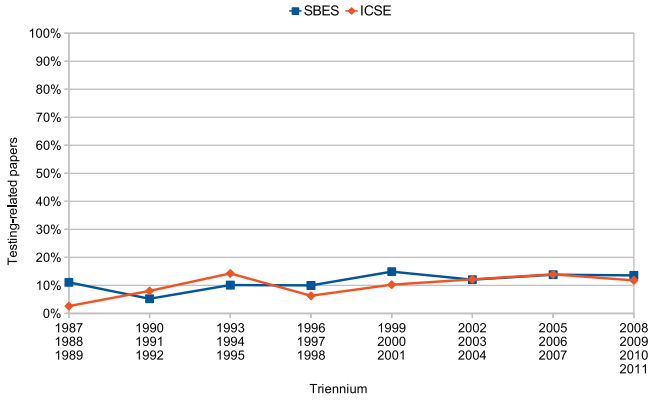


Fig. 9. Rate of software testing papers published in ICSE and SBES by triennium.

venue. As we can observe, although ICSE has generally accepted almost twice as many papers as SBES in the analyzed period, the rate of software testing papers is similar. In total, ICSE proceedings include 10% of software testing papers (111 out of 1092 papers), while SBES proceedings include 11% (60 out of 522 papers). Fig. 9 depicts the evolution of the rate for each venue.

We highlight that this similarity between the rates of software testing papers in both conferences supports the unbiased analysis of the results presented in the previous sections. It represents evidence on the level of importance given by both venues to the testing field. Noticeable differences in the percentages might otherwise lead one to question if the software testing research was more relevant to one particular event than to the other.

5. Discussion

5.1. The evolution of evaluations in software testing papers

An interesting analysis to be conducted is the observation of the evolution in the rate of software testing papers that present evaluation components published along the SBES lifetime, and also in ICSE.[13] For SBES, looking at Table 5 and Fig. 4, we can see that no evaluations were presented in the first and second periods, but in the third period, 28% of the papers had an evaluation component. In the last period, 82% of the papers presented evaluation components. To obtain the observed growth rate of the total period, we can use the following equation: $OR = (LT - FT/FT)$, where OR is the Observed Rate, FT is the First Triennium to present evaluations (1993–1995), and LT is the Last Triennium plus the 2011 proceedings (2008–2011). This analysis shows that there was a 193% increase in the rate of evaluated papers along all the period, a very significant evolution in terms of evaluation.

The average growth rate (AGR) in a given period i to j, that is, the rate by which a variable changes if varying by a constant rate, is given by the following equation: $AGR_{i,j} = \sqrt[j-i]{X_j/X_i} - 1$. For SBES, the AGR observed between triennia (i_3) can be computed by the following equation (derived from the previous): $i_3 = (1 + OR)^{1/5} - 1$, since there are six analyzed data points – excluding the first two, which are zero – and, therefore, five intervals. Assigning 1.93 to OR, we reach the average growth of 23.98% in the rate of evaluated

[13] We have used rate instead of absolute numbers because the number of testing papers accepted at each edition of the events varies. The rate allows us to see the proportion of software testing papers with evaluation over the total number of accepted papers on the subject.

Table 8
Rate of software testing papers published in ICSE and SBES by triennium.

Triennium	ICSE			SBES		
	# Papers	# Testing papers	Rate (%)	# Papers	# Testing papers	Rate (%)
1987–1989	114	3	3	45	5	11
1990–1992	87	7	8	57	3	5
1993–1995	105	15	14	79	8	10
1996–1998	143	9	6	70	7	10
1999–2001	146	15	10	67	10	15
2002–2004	148	18	12	58	7	12
2005–2007	129	18	14	65	9	14
2008–2011	220	26	12	81	11	14
Total	1092	111	10	522	60	11

papers within subsequent triennia. These numbers suggest that, if no particular changes occur in the field, in the next triennium every – or close to every – software testing paper published in SBES will contain an evaluation component.

With respect to ICSE, we can see from Table 6 and Fig. 5 that since 1987 there were software testing papers with evaluation components being published in the conference. The observed growth rate for ICSE in the whole period was 49.25%, much less dramatic then in SBES. The average growth observed between triennia is calculated by the equation $i_3 = (1 + \mathrm{OR})^{1/7} - 1$, because we have eight data points at this time. Assigning 0.4925 to OR, we reach the average growth of 5.88%. However, an interesting point to observe is that since 2005 all software testing papers published in the conference contain an evaluation component. This shows that the international conference is more mature than SBES in this sense, and that the national conference seems to be following in the same direction.

As observed in Section 4, there was an interesting outlier occurring in the 2002–2004 triennium in SBES. We believe this outlier can be explained by an increase in the awareness of the need for more serious evaluations in those years, which must have impacted in the number of evaluation studies. In fact, in 2005 an international survey of controlled experiments in software engineering (Sjoberg et al., 2005) showed that 2000 was the year with the highest number of papers describing experiments, both in absolute and relative numbers. Another interesting fact occurring in the same period is the creation of the International Symposium on Empirical Software Engineering (ISESE), later renamed to Empirical Software Engineering and Measurement (ESEM). According to the official website, the first symposium took place in 2002, maybe motivated by the growing awareness of the need for serious evaluation in Software Engineering.[14]

Moreover, if we look at the ICSE data, we can see that from the 1987–1989 to the 1996–1998 triennia there was a decrease in the rate of papers with evaluation components. Then, starting exactly in the triennium that includes the year 2000 (1999–2001), we can see a subsequent increase in this rate, culminating in the 2005–2007 triennium, when all software testing papers started containing evaluations. The similarity perceived here with respect to software testing research shows a connection between the SBES community with the international Software Engineering community. It also indicates that both national and international communities are considering evaluation studies as a requirement for paper acceptance.

5.2. The evaluations according to the technique and type dimensions

With respect to the distribution of papers according to the discussed testing technique in SBES, Fig. 2(a) reveals that there is a gap between the number of publications related to the most frequent topic and the number of evaluations: only 8 out of 22 *White-box testing* papers, *i.e.* 36%, reported some kind of evaluation. Other topics presented a better correlation between the number of publications and evaluations: *Strategy* (100%), *Regression testing* (100%), and *Test case generation* (71%), for example. In ICSE, these figures are different: 68% of *White-box testing* papers – also the most prominent topic here – reported evaluations. The smallest correlation is for *Model-based testing*, where 46% of papers included some kind of evaluation.

Let us narrow the analyzed period to the last four triennia – *i.e.* 1999–2011 –, which represents the period when the evaluation rates are higher than 50% in both conferences (see Figs. 4 and 5). In this period, we can observe minor changes in the most investigated topics (and how they have been evaluated) in SBES. *White-box testing*, *Fault-based testing* and *Test case generation* are still the top-3 addressed topics, in the same order of the full period (*i.e.* 1987–2011). However, the rate of papers that report some kind of evaluation between 1999 and 2011 for these topics are 62%, 67% and 100%, respectively, while these numbers are 37%, 63% and 71% if we consider the full period.

In ICSE, the scenario is slightly different. The top-3 investigated topics from 1999 to 2011 are *Test case generation*, *White-box testing* and *Model-based testing*, in this order. Considering the full period, the top-3 topics are the same, however the order differs: *White-box testing*, *Test case generation*, and *Strategy*/*Model-based testing*. From 1999 to 2011 in ICSE, the rate of papers that include an evaluation component are 91% for *Test case generation*, 94% for *White-box testing*, and 58% for *Model-based testing*. Comparing to the full period, we have 88% for *Test case generation*, 68% for *White-box testing*, and 47% for *Model-based testing*.

All these figures are summarized in Figs. 10 and 11. Generally, the analysis of this narrowed period – *i.e.* the last four triennia – corroborates the observed trend in regard to the increasing number of testing papers that report some kind of evaluation. On the other

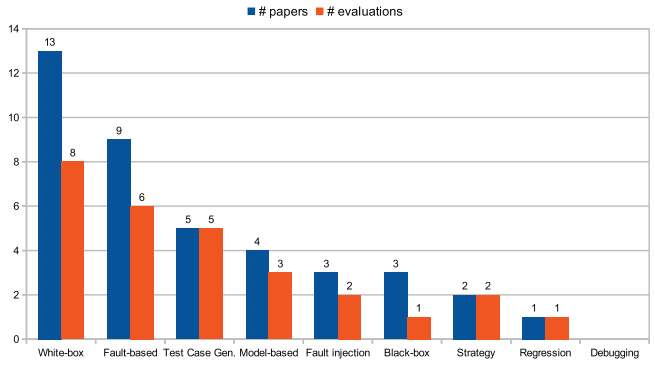


Fig. 10. Distribution of papers per topic in SBES in the last four triennia (1999–2011).

[14] http://www.esem-conferences.org/history.php – accessed 19.10.12.

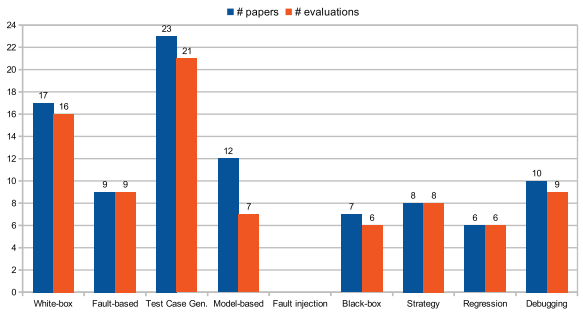


Fig. 11. Distribution of papers per topic in ICSE in the last four triennia (1999–2011).

hand, the most investigated topics by the research community who publishes in SBES and ICSE have not shown significant changes.

Regarding the *Type*-related classification, Fig. 2(b) showed that novel approaches published in SBES are hardly ever evaluated in the same paper: only 11 out of 35 papers, *i.e.* 31%, reported some kind of evaluation. This is an evidence that several approaches have been proposed in SBES but there are not a great concern with their evaluation. Papers describing some testing tool or related infrastructure implementation have also resulted in a low correlation between publications and evaluations: 33%. In ICSE, Fig. 3(b) shows that 53 out of 73 papers presenting novel approaches, *i.e.* 74%, also included evaluations. However, papers describing tools also had low evaluation correlation: 37%. Obviously, all papers aiming at evaluating some aspect of software testing in both conferences present an evaluation study. Nevertheless, we decided to keep the column *Evaluation* in Figs. 2(b) and 3(b) for the sake of completeness.

Again narrowing the analyzed period to the last four triennia – *i.e.* 1999–2011–, we can observe that, more recently, researchers who have addressed novel testing approaches in SBES have become more concerned with evaluation issues. This can be observed in the chart shown on the left-hand side of Fig. 12 (bars labeled with *Approach*). Around 59% of these papers reported some kind of evaluation between 1999 and 2011, which represents an increase of almost 100% if we consider the full period (*i.e.* 1987–2011).

With respect to ICSE, there is also an increase in this context. From 1999 to 2011, 90% of the novel approaches were somehow evaluated in the same paper they were proposed, against 74% if we consider the full period. This can be observed in the right-hand chart of Fig. 12.

The charts of Fig. 12 also reveal that all evaluations included in papers that focus on tools and infrastructure were published in the last four triennia (see Figs. 2(b) and 3(b) to crosscheck this information). Note that this holds for both venues, thus indicating that there is a greater concern in evaluating what is very often used to support the evaluations, that is, the tools themselves.

5.3. The rigor of the evaluations

With respect to the level of evaluation in the software testing studies published in SBES and ICSE, we believe that more rigorous evaluations could have been applied in some cases. For instance, the only two software testing *Controlled experiments* reported in SBES (Campanha et al., 2010) and ICSE (Rothermel et al., 2000) were in the fault-based and white-box testing domains, two popular software testing subjects. This is an evidence that such type of evaluation could also have been applied to other testing approaches in the same domains published in SBES and ICSE.

On the other hand, we must consider that it is hard to report a rigorous study in the same conference paper that presents an approach, because of the limited space available in conference publications. For instance, the approach evaluated in the ICSE study mentioned above was published in 1998 (Rothermel et al., 2000), two years prior to the experiment publication; and the SBES 2010 controlled experiment reported a study on well-established mutation testing approaches. Therefore it is important to note that the two most rigorous evaluations found in our survey focus on the experiment itself, not on the proposal of a testing approach.

Another related factor that might explain the low frequency of rigorous studies found in our survey is that authors might leave extended evaluations of their approaches for archival publications (*e.g.* journal papers), where there are less space constraints. For example, Rountev et al. (2003) published a paper in ICSE 2003 containing an exploratory case study, according to our classification. The same study was later extended to a quasi experiment, and reported in an IEEE Transactions on Software Engineering journal paper (Rountev et al., 2004). However, this should not justify the small number of *Controlled* and *Quasi experiments* found in our survey: the existence of two *Controlled experiments* and 19 *Quasi experiments* published in the two venues shows that these types of studies are feasible for different testing techniques.

Table 9
Impact in number of citations of the controlled and quasi experiments published in ICSE.

Paper title	Citations
Effect of test set minimization on fault detection effectiveness	257
An empirical study of regression test selection techniques	270
An empirical study of regression test application frequency	54
A history-based test prioritization technique for regression testing in resource constrained environments	151
The impact of test suite granularity on the cost-effectiveness of regression testing	52
Automated test case generation for spreadsheets	54
Improving web application testing with user session data	167
Improving test suites via operational abstraction	146
A framework of greedy methods for constructing interaction test suites	70
Demand-driven structural testing with dynamic instrumentation	48
Is mutation an appropriate tool for testing experiments?	323
An empirical study of fault localization for end-user programmers	32
An empirical evaluation of test case filtering techniques based on exercising complex information flows	37
Feedback-directed random test generation	242
Testing pervasive software in the presence of context inconsistency resolution services	28
The effect of program and model structure on MC/DC test adequacy coverage	25
Maintaining and evolving GUI-directed test scripts	34
WYSIWYT testing in the spreadsheet paradigm: an empirical evaluation	81
Average	115
Standard deviation	98

5.4. The impact of software testing papers

With respect to the impact of the software testing papers that contain evaluations published in ICSE, Table 9 shows the number of citations to each paper that reports *Quasi experiments* and the *Controlled experiment* (last paper shown in the table) found in our survey. We have gathered such information from Google Scholar and selected these papers because they report more rigorous studies. We included only ICSE in this analysis because in SBES proceedings only three papers describe these kinds of studies.

By looking into this data, the high standard deviation indicates that the variability is very high; that is, there seems to be no correlation between the rigor of evaluations and the number of citations. Therefore, we cannot make any fair comparisons of the impact of these papers with other papers published in ICSE or other venues. However, only to provide a basis for citation magnitude, we have looked into the number of citations of the last six ICSE papers that

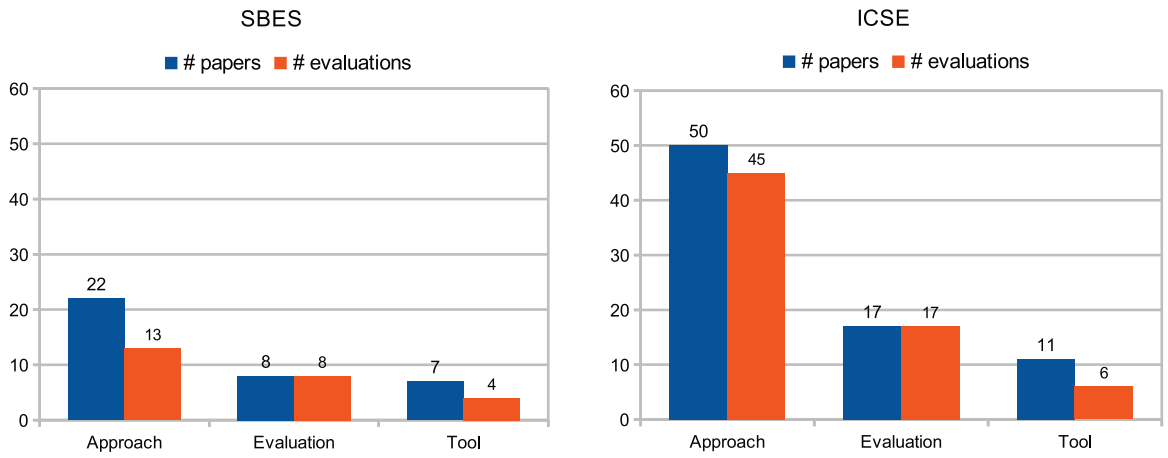


Fig. 12. Distribution of papers per type in SBES (a) and ICSE (b) in the last four triennia (1999–2011).

Table 10
Impact in number of citations of the last six papers elected as ICSE most influential papers.

Paper title	Citations	**Award year**
Analysis and testing of Web applications	394	2011
Bandera: extracting finite-state models from Java source code	1143	2010
A case study of open source software development: the Apache server	435	2010
N degrees of separation: multi-dimensional separation of concerns	1489	2009
Architecture-based runtime software evolution	623	2008
Designing distributed applications with mobile code paradigms	385	2007
Average	745	
Standard deviation	464	

have received the most influential paper award[15] (see Table 10). Again the standard deviation is high, also indicating that influence is not measured only by citations. In any case, the average number of citations for this group of papers is 745, much higher than the average citations to the papers in Table 9.

Although we believe it is unfair to compare such groups of papers, we can make a remark with respect to these numbers: one of the papers shown in Table 9 – the 11th paper in list – discusses the use of mutation as a tool for testing experiments, that is, an important topic from a software testing evaluation perspective. The number of citations to this paper – 323 in total – is close to the number of citations to the most influential paper of 1997 (last one in Table 10). This fact indicates that such topic is considered important for researchers in the field.

6. Related work

To the best of our knowledge, our paper is the first to analyze the evolution of evaluation studies of software testing research, both in a Brazilian context and in an international context. Other authors have reported surveys focusing on other aspects of software testing research, empirical software engineering, or software engineering research in Brazil and SBES.

Sjoberg et al. (2005), for instance, reports a survey on how controlled experiments in software engineering have been conducted and the extent to which relevant information was reported until 2005. Differently from our survey, however, the study focuses only on controlled experiments and the venues analyzed include top conferences and journals of the field. Moreover, the authors target the topic of software engineering in general, not software testing. An interesting correlation between our work and theirs was discussed in Section 5.

Zannier et al. (2006) conducted a study to analyze the successfulness of empirical studies published by ICSE. A difference from our analysis is that Zannier et al. covered all software engineering topics and not only software testing. Moreover, the reported study is empirical, that is, includes a sample of papers and try to draw conclusions for the whole population. Since our study focuses on software testing – a narrower scope – we were able to analyze all papers, and not only a sample. Therefore our conclusions do not rely on statistical tests, they are based on the entire population of published papers.

Juristo et al. (2006, 2009) reported surveys about the body of empirical findings related to software testing. They have compiled publications that evaluate software testing techniques from different aspects, and aggregated the empirical results. Our study is different from theirs in the sense that we analyze the evolution of evaluation studies of software testing research along the years, and not the derived results themselves.

Durelli et al. (2011) performed a systematic mapping study with the aim of characterizing software testing-related research published in SBES. Differently from our goals, Durelli et al.'s main objective was to analyze the most investigated topics within the software testing research field. Researchers' productivity levels in terms of number of papers and citations, and authorship networks are also analyzed. Furthermore, the authors point out current demands and research directions concerning software testing. Note that Durelli et al. surveyed both the main track of SBES (*i.e.* research papers) and Tools Section proceedings. Consequently, their dataset partially overlaps ours. The classification schema they used includes four categories: *Solution proposal*, *Evaluation research*, *Validation research* and *Opinion*. The first can be mapped to the *Approach* category within the *Type* dimension we used in this paper, whereas the next two can be both mapped to *Evaluation*. There is no representative for *Opinion* in our classification schema (more details in Section 3).

Delamaro et al. (2011) presented a historical perspective of the contributions of the Brazilian research community on software testing with respect to two techniques: structural-based testing and fault-based testing (more specifically, mutation testing). In particular, they describe the contributions – and the associated impact – of two Brazilian research groups located in two institutions: the University of São Paulo (ICMC) (São Carlos - Brazil) and the State University of Campinas (FEEC) (Campinas – Brazil).

[15] http://www.icse-conferences.org/mostinfluential.html – accessed 19.10.12.

Note that these two universities are the top-2 institutions that have most published testing-related papers in SBES, according to the results we presented in Section 4 (Table 3). Note also that Durelli et al.'s dataset is not limited only to SBES papers. Instead, the authors analyze the contributions of these two institutions locally (*i.e.* in Brazil) and in the international context. While Durelli et al.'s main objective was the analysis of two specific research groups, we aimed at providing a broader perspective focusing on how software testing researchers have been evaluating their work.

7. Conclusion

This paper presented a survey with a historical perspective on the application of evaluation studies in software testing papers published in SBES and ICSE, the premier Brazilian and international conferences on Software Engineering. We have analyzed publications in the 25-year history of SBES, and publications by ICSE in the same period. Our data shows that the national community has significantly improved in this subject, with a noticeable increase in the rate of evaluated testing-related publications. However, comparing to the international context, we still have to grow: since 2006 all papers published by ICSE contain an evaluation component. In SBES, in the last four years (*i.e.* 2008–2011), 82% of the software testing papers presented evaluations. With respect to the rigor of the performed evaluations, there is still room to improve in both scenarios: both SBES and ICSE presented each only a single software testing-related paper that reports on a controlled experiment results.

Our survey also provides other interesting insights. For instance, in SBES we found out that publications about *Test case generation* approaches were one of the most frequent to present an evaluation component (71%), and only 36% of papers on *White-box testing* – the dominant testing topic in SBES – have evaluated their proposals. This is consistent with the difference in difficulty in applying experiments for research work on those topics commented in Section 2.

Another interesting result that showed up in our data was an outlier with respect to papers containing evaluations: the SBES proceedings of the 2002–2004 triennium presented an uncommon increase in the rate of evaluated papers compared to the antecedent triennium. This result is consistent with three international events: (1) a 2005 international survey of software engineering controlled experiments, which showed that 2000 was the year with the highest number of reported experiments in the analyzed period (1993–2002) (Sjoberg et al., 2005); (2) a sudden increase in the rate of evaluated papers in ICSE starting in the triennium that includes the year 2000; and (3) the creation of the International Symposium on Empirical Software Engineering (ISESE; later renamed to Empirical Software Engineering and Measurement – ESEM) in the year 2002, around the same referred period. This indicates the increase in awareness for the need of serious evaluations, and also a close connection between the national and international software testing communities.

Appendix A. Tables with selected papers

See Tables 11–16.

Table 11
Testing-related papers published in the SBES proceedings (1/2).

Year	#	Title*	Tech	Type	Authors	Affiliations	Type of Eval. According to Zannier et al.	Eval	Type of Subjects	Citations (Google Scholar)
1987	1	(t) Visualizing the Control Flow of Programs	W	A	A. M. Price; F. Garcia; C. Purper	Federal University of Rio Grande do Sul (Porto Alegre - Brazil)	n/a	N	n/a	5
	2	(t) Controlled Execution of Programs	D	T	J. R. V. da Silva; D. L. Segalin; R. Vieira; P. A. Azevedo;	Federal University of Rio Grande do Sul (Porto Alegre - Brazil)	n/a	N	n/a	2
	3	(t) PROTESTE: Design of a Tool for Program Testing	B	T	A. M. Price; C. Purper; F. Garcia	Federal University of Rio Grande do Sul (Porto Alegre - Brazil)	n/a	N	n/a	0
1988	4	(t) Test Case Selection based on Data Flow through the Potential-Uses Criteria	W	A	J. C. Maldonado; M. L. Chaim; M. Jino;	University of São Paulo (ICMC) (São Carlos – Brazil) State University of Campinas (FEEC) (Campinas – Brazil)	n/a	N	n/a	0
1989	5	(t) Modeling and Determining Potential DU-Paths through Data Flow Analysis	W	A	M. L. Chaim; J. C. Maldonado; M. Jino	State University of Campinas (FEEC) (Campinas – Brazil) University of São Paulo (ICMC) (São Carlos – Brazil)	n/a	N	n/a	0
1990	6	(t) Environment to Support Structural Testing of Programs	W	T	A. M. A. Price; A. F. Zorzo	Federal University of Rio Grande do Sul (Porto Alegre - Brazil)	n/a	N	n/a	0
1992	7	(t) Unfeasible Paths in the Testing Activity Automation	W	A	S. R. Vergilio; J. C. Maldonado; M. Jino	State University of Campinas (FEEC) (Campinas – Brazil) University of São Paulo (ICMC) (São Carlos – Brazil)	Example Application	N	n/a	6
	8	(t) Potential-Uses Criteria: Analyzing the Application of a Benchmark	W	A	J. C. Maldonado; S. R. Vergilio; M. L. Chaim; M. Jino	University of São Paulo (ICMC) (São Carlos – Brazil) State University of Campinas (FEEC) (Campinas – Brazil)	Example Application	N	n/a	0
1993	9	(t) A Strategy for Generating Test Data	A	A	S. R. Vergilio; J. C. Maldonado; M. Jino	Federal University of Paraná (Curitiba – Brazil) University of São Paulo (ICMC) (São Carlos – Brazil) State University of Campinas (FEEC) (Campinas – Brazil)	Example Application	N	n/a	8
	10	Evaluation of the Cost of Alternate Mutation Strategies	F	E	A. P. Mathur; W. E. Wong	Purdue University (West Lafayette – USA)	Exploratory Case Study	Y	Programs	17
1994	11	(t) Applying Mutant Analysis to the Validation of Petri Net-based Specifications	F	A	S. C. P. F. Fabbri; J. C. Maldonado; P. C. Masiero; M. E. Delamaro	Federal University of São Carlos (São Carlos – Brazil) University of São Paulo (ICMC) (São Carlos – Brazil) University of São Paulo (IFSC) (São Carlos – Brazil)	Example Application	N	n/a	0
	12	Constrained Mutation in C Programs	F	E	W. E. Wong; J. C. Maldonado; M. E. Delamaro; A. P. Mathur	Hughes Network Systems University of São Paulo (ICMC) (São Carlos – Brazil) Purdue University (West Lafayette – USA)	Exploratory Case Study	Y	Programs	36
	13	(t) Unfeasible Paths in Integration Testing: Characterization, Estimation and Determination	W	A	S. R. Vergilio; J. C. Maldonado; M. Jino	Federal University of Paraná (Curitiba – Brazil) University of São Paulo (ICMC) (São Carlos – Brazil) State University of Campinas (FEEC) (Campinas – Brazil)	n/a	N	n/a	0
1995	14	(t) Test Data Generation: A Strategy that Preserves Criteria Hierarchy	W	A	S. R. Vergilio; J. C. Maldonado; M. Jino	Federal University of Paraná (Curitiba – Brazil) University of São Paulo (ICMC) (São Carlos – Brazil) State University of Campinas (FEEC) (Campinas – Brazil)	n/a	N	n/a	0
	15	(t) Integrating Fault Injection and Formal Testing in the Validation of Fault Tolerance	I	A	E. Martins	State University of Campinas (IC) (Campinas – Brazil)	n/a	N	n/a	0
	16	A G-Net Based Environment for Logical and Timing Analysis of Software Systems	M	T	A. Perkusich; J. C. A. Figueiredo	Federal University of Paraíba (João Pessoa – Brazil)	n/a	N	-	1
1997	17	(t) Potential-Uses Criteria Coverage and Software Reliability	W	E	A. N. Crespo; A. Pasquini; M. Jino; J. C. Maldonado	State University of Campinas (FEEC) (Campinas – Brazil) University of São Paulo (ICMC) (São Carlos – Brazil)	Case Study	Y	Programs	0
	18	(t) Strategy for Test Data Generation based on Symbolic and Dynamic Program Analysis	A	A	J. S. Herbert; A. M. A. Price	Federal University of Rio Grande do Sul (Porto Alegre - Brazil)	n/a	N	n/a	2
	19	(t) Integration Testing: Design of Operators for the Interface Mutation Criterion	F	A	M. E. Delamaro; J. C. Maldonado	University of São Paulo (IFSC) (São Carlos – Brazil) University of São Paulo (ICMC) (São Carlos – Brazil)	n/a	N	n/a	0
	20	(t) Applying the Mutant Analysis Criterion to the Validation of Statecharts-based Specifications	F	A	S. C. P. F. Fabbri; J. C. Maldonado; P. C. Masiero	Federal University of São Carlos (São Carlos – Brazil) University of São Paulo (ICMC) (São Carlos – Brazil)	Example Application	N	n/a	3
	21	(t) Evaluating the Impact of Test Set Minimization on the Cost and Efficacy of the Mutant Analysis Criterion	F	E	S. R. S. Souza; J. C. Maldonado	State University of Ponta Grossa (Ponta Grossa – Brazil) University of São Paulo (ICMC) (São Carlos – Brazil)	Exploratory Case Study	Y	Programs	0
1998	22	(t) A System to Support the Testing of Object-Oriented Programs based on a Reflexive Approach	M	T	I. M. Pinto; A. M. A. Price	Federal University of Rio Grande do Sul (Porto Alegre – Brazil)	Example Application	N	n/a	6
	23	(t) A Contribution for Determining a Sufficient Mutant Operator Set for Testing C Programs	F	E	E. F. Barbosa; A. M. R. Vincenzi; J. C. Maldonado	University of São Paulo (ICMC) (São Carlos – Brazil)	Exploratory Case Study	Y	Programs	0
1999	24	(t) Automatic Data Generation and Non-Executabiity Handling in Structural Software Testing	A, W	A	P. M. S. Bueno; M. Jino	State University of Campinas (FEEC) (Campinas – Brazil)	Exploratory Case Study	Y	Programs	4
	25	(t) A Study of the Cost Evaluation of Applying Mutant Analysis to the Validation of Finite State Machines	F	E	R. A. Carvalho; S. C. P. F. Fabbri; J. C. Maldonado	Federal University of São Carlos (São Carlos – Brazil) University of São Paulo (ICMC) (São Carlos – Brazil)	Exploratory Case Study	Y	Models	1
	26	(t) Interface Sufficient Operators: A Case Study	F	E	A. M. R. Vincenzi; J. C. Maldonado; E. F. Barbosa; M. E. Delamaro	University of São Paulo (ICMC) (São Carlos – Brazil) State University of Maringá (Maringá – Brazil)	Exploratory Case Study	Y	Programs	2
	27	Data Flow Based Integration Testing	W	A	P. Vilela; M. Jino; J. C. Maldonado	Telcordia Technologies Inc. State University of Campinas (FEEC) (Campinas – Brazil) University of São Paulo (ICMC) (São Carlos – Brazil)	n/a	N	n/a	6
2000	28	A Binomial Software Reliability Model Based on Coverage of Structural Testing Criteria	W	A	A. N. Crespo; M. Jino; A. Pasquini; J. C. Maldonado	São Francisco University (Campinas – Brazil) ENEA (Rome – Italy) State University of Campinas (FEEC) (Campinas – Brazil) University of São Paulo (ICMC) (São Carlos – Brazil)	Exploratory Case Study	Y	Programs	3
	29	Proteum-RS/PN: A Tool to Support Edition, Simulation and Validation of Petri Nets based on Mutation	F	T	A. S. Simão; J. C. Maldonado; S. C. P. F. Fabbri	University of São Paulo (ICMC) (São Carlos – Brazil)	Exploratory Case Study	Y	Models	13
	30	(t) Structural Software Testing: An Approach for Relational Database Applications	W	A	E. S. Spoto; M. Jino; J. C. Maldonado	State University of Maringá (Maringá – Brazil) State University of Campinas (FEEC) (Campinas – Brazil) University of São Paulo (ICMC) (São Carlos – Brazil)	Exploratory Case Study	Y	Programs	2

Legend for Tech: A: Test case generation. B: Functional (black-box) testing. D: Debugging. F: Fault-based testing. S: Testing strategy. I: Fault injection/tolerance. M: Model-based testing. R: Regression testing. W: Structural (white-box) testing. Legend for Type: A: Approach proposal. E: Evaluation. T: Tool and infrastructure.
[a]Paper titles marked with a (t) were translated from Portuguese to English.

Table 12
Testing-related papers published in the SBES proceedings (2/2).

Year	#	Title*	Tech	Type	Authors	Affiliations	Type of Eval. According to Zannier et al.	Eval	Type of Subjects	Citations (Google Scholar)
2001	31	Mudel: A Language and a System for Describing and Generating Mutants	F	T	A. S. Simão; J. C. Maldonado	University of São Paulo (ICMC) (São Carlos – Brazil)	n/a	N	n/a	11
	32	(t) FCCE: A Testing Criteria Family for the Validation of Systems Specified in Estelle	W	A	S. R. S. Souza; J. C. Maldonado; S. C. P. F. Fabbri	State University of Ponta Grossa (Ponta Grossa – Brazil) University of São Paulo (ICMC) (São Carlos – Brazil) Federal University of São Carlos (São Carlos – Brazil)	n/a	N	n/a	0
	33	Mutant Operators for Testing Concurrent Java Programs	F	A	M. E. Delamaro; M. Pezzè; A. M. R. Vincenzi; J. C. Maldonado	State University of Maringá (Maringá – Brazil) University of Milan – Bicocca (Milano – Italy) University of São Paulo (ICMC) (São Carlos – Brazil)	n/a	N	n/a	15
2002	34	Selection and Evaluation of Test Data Sets Based on Genetic Programming	F	A	M. C. F. P. Emer; S. R. Vergilio	Federal University of Paraná (Curitiba – Brazil)	Exploratory Case Study	Y	Programs	0
	35	(t) Tests and Code Generation for Web Systems	A	T	E. Aranha; P. Borba	Federal University of Pernambuco (Recife – Brazil)	Case Study	Y	Programs	0
2003	36	(t) A Method for Functional Testing for the Verification of Components	B	A	C. M. Farias; P. D. L. Machado	Federal University of Campina Grande (Campina Grande – Brazil)	n/a	N	n/a	10
	37	A Family of Coverage Testing Criteria for Coloured Petri Nets	F	A	A. S. Simão; S. R. S. Souza; J. C. Maldonado	University of São Paulo (ICMC) (São Carlos – Brazil)	n/a	N	n/a	2
2004	38	(t) Unit Testing of Aspect-Oriented Programs	W	A	O. A. L. Lemos; A. M. R. Vincenzi; J. C. Maldonado; P. C. Masiero	University of São Paulo (ICMC) (São Carlos – Brazil) Centro Universitário Eurípedes de Marília (Marília – Brazil)	Exploratory Case Study	Y	Programs	6
	39	(t) Reuse in the Software Testing Activity to Reduce VV&T Cost and Effort in the Development and Re-engineering of Software	R	T	M. I. Cagnin; J. C. Maldonado; A. Chan; R. Penteado; F. Germano	University of São Paulo (ICMC) (São Carlos – Brazil) Federal University of São Carlos (São Carlos – Brazil)	Exploratory Case Study	Y	People	4
	40	(t) A Methodology for the Verification of Partial Systems Modeled with Object-Based Graph Grammar	M	A	F. L. Dotti; F. Pasini; O. M. Santos	Pontifical Catholic University of Rio Grande do Sul (Porto Alegre – Brazil)	n/a	N	n/a	0
2005	41	(t) Distributed Environment of Communication Fault Injection for Testing of Network Java Applications	I	T	J. Gerchman; G. Jacques-Silva; R. J. Drebes; T. S. Weber	Federal University of Rio Grande do Sul (Porto Alegre - Brazil)	n/a	N	n/a	8
	42	Automatic test data generation for path testing using a new stochastic algorithm	A	E	B. T. Abreu; E. Martins; F. L. Sousa	State University of Campinas (IC) (Campinas – Brazil) National Institute for Space Research (São José dos Campos – Brazil)	Exploratory Case Study	Y	Programs	8
	43	(t) An Aspect-based Tool for Functional Testing of Java Programs	B	T	A. D. Rocha; A. S. Simão; J. C. Maldonado; P. C. Masiero	University of São Paulo (ICMC) (São Carlos – Brazil)	n/a	N	n/a	5
2006	44	(t) Automatic Generation of Test Drivers and Stubs for JUnit based on U2TP Specifications	A, M	T	L. Biasi; K. Becker	Pontifical Catholic University of Rio Grande do Sul (Porto Alegre – Brazil)	Exploratory Case Study	Y	Programs	4
2007	45	Static Analysis of Java Bytecode for Domain-Specific Software Testing	W	A	M. E. Delamaro; P. A. Nardi; O. A. L. Lemos; P. C. Masiero; E. S. Spoto; J. C. Maldonado; A. M. R. Vicenzi	Centro Universitário Eurípedes de Marília (Marília – Brazil) University of São Paulo (ICMC) (São Carlos – Brazil) Catholic University of Santos (Santos – Brazil)	n/a	N	n/a	4
	46	Generalized Extremal Optimization: A Competitive Algorithm for Test Data Generation	A	E	B. T. Abreu; E. Martins; F. L. Souza	State University of Campinas (IC) (Campinas – Brazil) National Institute for Space Research (São José dos Campos – Brazil)	Exploratory Case Study	Y	Programs	8
	47	Experimental Evaluation of Coverage Criteria for FSM-Based Testing	M	E	A. S. Simão; A. Petrenko; J. C. Maldonado	University of São Paulo (ICMC) (São Carlos – Brazil) Centre de Recherche Informatique de Montreal (CRIM)	Quasi Experiment	Y	Models	4
	48	Pairwise Structural Testing of Object and Aspect-Oriented Java Programs	W	A	I. G. Franchin; O. A. L. Lemos; P. C. Masiero	University of São Paulo (ICMC) (São Carlos – Brazil)	n/a	N	n/a	7
	49	Integration Testing of Aspect-Oriented Programs: A Characterization Study to Evaluate how to Minimize the Number of Stubs	S	A	R. Ré; P. C. Masiero	Federal University of Technology - Paraná (Campo Mourão – Brazil) University of São Paulo (ICMC) (São Carlos – Brazil)	Exploratory Case Study	Y	Programs	8
2008	50	(t) Using Similarity Functions to Reduce Test Suites in Strategies for Model-based Testing	M, B	A	E. G. Cartaxo; P. D. L. Machado; F. G. Oliveira Neto; J. F. S. Ouriques	Federal University of Campina Grande (Campina Grande – Brazil)	Exploratory Case Study	Y	Programs	0
	51	(t) Generation of Faultloads for Testing Campaigns with Fault Injection from UML Testing Models	I	A	J. Gerchman; C. Menegotto; T. S. Weber	Federal University of Rio Grande do Sul (Porto Alegre - Brazil)	Exploratory Case Study	Y	Programs	0
	52	Obtaining Trustworthy Test Results in Multi-threaded Systems	I	A	A. Dantas; M. Gaudencio; F. Brasileiro; W. Cirne	Federal University of Campina Grande (Campina Grande – Brazil)	Exploratory Case Study	Y	Programs	8
	53	Integration Testing of Aspect-Oriented Programs: a Structural Pointcut-Based Approach	W	A	O. A. L. Lemos; P. C. Masiero	University of São Paulo (ICMC) (São Carlos – Brazil)	Example Application	N	n/a	4
	54	(t) A Catalog of Stubs to Support the Integration Testing of Aspect-Oriented Programs	S	E	R. Ré; A. L. S. Domingues; P. C. Masiero	Federal University of Technology - Paraná (Campo Mourão – Brazil) University of São Paulo (ICMC) (São Carlos – Brazil)	Exploratory Case Study	N	n/a	0
2009	55	Applying Code Coverage Approach to an Infinite Failure Software Reliability Mode	W	A	A. N. Crespo; A. Pasquini; M. Jino; J. C. Maldonado	Deep Blue (Rome – Italy) University of São Paulo (ICMC) (São Carlos – Brazil) State University of Campinas (FEEC) (Campinas – Brazil) Renato Archer Research Center (Campinas – Brazil)	Quasi Experiment	Y	Programs	2
2010	56	Characterising Faults in Aspect-Oriented Programs: Towards Filling the Gap between Theory and Practice	F	E	F. C. Ferrari; O. A. L. Lemos; R. Burrows; A. F. Garcia; J. C. Maldonado	University of São Paulo (ICMC) (São Carlos – Brazil) Lancaster University (Lancaster – UK) Pontifical Catholic University of Rio de Janeiro (Rio de Janeiro – Brazil) Federal University of São Paulo (São José dos Campos – Brazil)	Exploratory Case Study	Y	Programs	4
	57	Built-in structural testing of web services	W	A	M. M. Eler; M. E. Delamaro; J. C. Maldonado; P. C. Masiero	University of São Paulo (ICMC) (São Carlos – Brazil)	n/a	N	n/a	6
	58	(t) Mutation Testing in Procedural and Object-Oriented Paradigms: An Evaluation of Data Structure Programs	F	E	D. N. Campanha; S. R. S. Sousa; J. C. Maldonado	University of São Paulo (ICMC) (São Carlos – Brazil)	Experiment	Y	Programs	0
2011	59	Agile Testing of Exceptional Behavior	W	A	R. di Bernardo; R. Sales; F. Castor Filho; R. Coelho; N. Cacho; S. Soares	Federal University of Pernambuco (Recife – Brazil) Federal University of Rio Grande do Norte (Natal – Brazil)	Exploratory Case Study	Y	Programs	1
	60	Contextual Integration Testing of Object-Oriented and Aspect-Oriented Programs: A Structural Approach for Java and AspectJ	W	A	B. B. P. Cafeo; P. C. Masiero	University of São Paulo (ICMC) (São Carlos – Brazil)	Exploratory Case Study	Y	Programs	0

Legend for Tech: A: Test case generation. B: Functional (black-box) testing. D: Debugging. F: Fault-based testing. S: Testing strategy. I: Fault injection/tolerance. M: Model-based testing. R: Regression testing. W: Structural (white-box) testing. Legend for Type: A: Approach proposal. E: Evaluation. T: Tool and infrastructure.
[a]Paper titles marked with a (t) were translated from Portuguese to English.

Table 13
Testing-related papers published in the ICSE proceedings from 1987 to 2011 (1/4).

Year	#	Title	Tech	Type	Authors	Affiliations	Type of Eval. According to Zannier et al.	Eval	Type of Subjects	Citations (Google Scholar)
1987	1	Modeling software failures and reliability growth during system testing	S	E	W. K. Ehrlich; T. J. Emerson	AT&T Bell Laboratories (Piscataway – USA) AT&T Bell Laboratories (Warren – USA)	Exploratory Case Study	Y	Programs	11
1988	2	Modeling mutation on a vector processor	F	T	A. P. Mathur; E. W. Krauser	Purdue University (West Lafayette – USA)	n/a	N	n/a	21
1989	3	An Error Complexity Model For Software Reliability Measurement	F, S	A	Y. Nakagawa; S. Hanata	NTT Software Laboratories (Tokyo – Japan)	Exploratory Case Study	Y	Programs	13
1990	4	Application of software reliability modeling to product quality and test process	S	A	W. K. Ehrlich; J. P. Stampfel; J. R. Wu	AT&T Bell Laboratories (Holmdel – USA)	Exploratory Case Study	Y	Programs	42
1991	5	Parameter value computation by least square method and evaluation of software availability and reliability at service-operation by the hyper-geometric distribution software reliability growth model (HGDM)	S	A	R. Jacoby; Y. Tohma	Tokyo Institute of Technology (Tokyo – Japan)	Exploratory Case Study	Y	Programs	18
1992	6	Incremental Testing Of Object-Oriented Class Structures	S	A	M. J. Harrold; J. D. McGregor; K. J. Fitzpatrick	Clemson University (Clemson – USA)	Example Application	N	n/a	239
	7	Testing For Linear Errors In Nonlinear Computer Programs	W	A	F. H. Afifi; S. J. Zeil; L. J. White	Case Western Reserve University (Cleveland – USA) Old Dominion University (Norfolk – USA)	Example Application	N	n/a	12
	8	Towards A Method Of Programming With Assertions	B, W	T	D. S. Rosenblum	AT&T Bell Laboratories (Murray Hill – USA)	n/a	N	n/a	93
	9	Specification-based Test Oracles For Reactive Systems	M	A	D. J. Richardson; S. L. Aha; T. O. O'Malley	University of California (Irvine – USA)	Example Application	N	n/a	265
	10	Validating Real-time Systems By History-checking TRIO Specifications	M	A	M. Felder; A. Morzenti;	Politecnico di Milano (Milano – Italy)	n/a	N	n/a	63
1993	11	An analytical comparison of the fault-detecting ability of data flow testing techniques	W	E	P. G. Frankl; E. J. Weyuker	Polytechnic University (New York – USA) New York University (New York – USA)	n/a	Y	Testing criteria	21
	12	Test templates: a specification-based testing framework	M	A	P. A. Stocks; D. A. Carrington	The University of Queensland (Queensland – Australia)	n/a	N	n/a	79
	13	Coverage measurement experience during function test	B, W	T	P. Piwowarski; M. Ohba; J. Caruso	IBM Corporation	Exploratory Case Study	Y	Programs	128
	14	Predicate-based test generation for computer programs	F	A	K. -C. Tai	North Carolina State University (Raleigh – USA)	Exploratory Case Study	Y	Programs	35
	15	Modeling software for accurate data flow representation	W	A	H. Ural; B. Yang ;	University of Ottawa (Ottawa – Canada) Bell-Northern Research Ltd. (Ottawa – Canada)	Example Application	N	n/a	12
	16	Exploring dataflow testing of arrays	W	T	D. Hamlet; B. Gifford; B. Nikolik	Portland State University (Portland – USA)	n/a	N	n/a	25
	17	Dynamic mutation testing in integrated regression analysis	F, R	A	J. Laski; W. Szermer; P. Luczycki	Oakland University (Rochester – USA)	Example Application	N	n/a	6
	18	An experimental evaluation of selective mutation	F	E	A. J. Offutt; G. Rothermel; C. Zapf	George Mason University (Fairfax – USA) Clemson University (Clemson - USA)	Exploratory Case Study	Y	Programs	132
	19	Experimental evaluation of a fuzzy-set based measure of software correctness using program mutation	S	E	F. B. Bastani; G. DiMarco; A. Pasquini;	University of Houston (Houston – USA) ENEA (Roma – Italy)	Exploratory Case Study	Y	Programs	14
1994	20	Experiments of the effectiveness of dataflow- and control-flow-based test adequacy criteria	W	E	M. Hutchins; H. Foster; T. Goradia; T. Ostrand	Siemens Corporate Research, Inc. (Princeton – USA)	Exploratory Case Study	Y	Programs	575
	21	A framework for evaluating regression test selection techniques	R	A	G. Rothermel; M. J. Harrold	Clemson University (Clemson – USA)	n/a	N	n/a	51
	22	TESTTUBE: a system for selective regression testing	R	T	Y. -F. Chen; D. S. Rosenblum; K. -P. Vo	AT&T Bell Laboratories (Murray Hill – USA)	Exploratory Case Study	Y	Programs	249
1995	23	Effect of Test Set Minimization on Fault Detection Effectiveness	W	E	W. E. Wong; J. R. Horgan; S. London; A. P. Mathur	Bell Communications Research (Morristown - USA) Purdue University (West Lafayette – USA)	Quasi-Experiment	Y	Programs	257
	24	Testing Real-Time Constraints in a Process Algebraic Setting	W, B	A	D. Clarke; I. Lee	University of Pennsylvania (Philadelphia – USA)	Example Application	N	n/a	26
	25	Using Testability Measures for Dependability Assessment	S	A	A. Bertolino; L. Strigini	CNR (Pisa – Italy)	n/a	N	n/a	14
1996	26	Assertion-Oriented Automated Test Data Generation	A	A	B. Korel; A. M. Al-Yami	Illinois Institute of Technology (Chicago – USA)	Exploratory Case Study	Y	Programs	75
	27	A Specification-Based Adaptive Test Case Generation Strategy for Open Operating System Standards	A	A	A. Watanabe; K. Sakamura	University of Tokyo (Tokyo – Japan)	Exploratory Case Study	Y	Programs, Models	7
	28	Reducing and Estimating the Cost of Test Coverage Criteria	W	A	M. Marré; A. Bertolino	University of Buenos Aires (Buenos Aires – Argentina) CNR (Pisa - Italy)	n/a	N	n/a	21
	29	A Reliability Model Combining Representative and Directed Testing	W, B	A	B. Mitchell; S. J. Zeil	Old Dominion University (Norfolk – USA)	n/a	N	n/a	15
	30	An Exact Array Reference Analysis for Data Flow Testing	W	A	I. Forgács	Hungarian Academy of Sciences (Budapesn – Hungary)	Example Application	N	n/a	7
	31	A Demand-Driven Analyzer for Data Flow Testing at the Integration Level	W	A	E. Duesterwald; R. Gupta; M. L. Soffa	University of Pittsburgh (Pittsburgh – USA)	Exploratory Case Study	Y	Programs	23

Legend for Tech: A: Test case generation. B: Functional (black-box) testing. D: Debugging. F: Fault-based testing. S: Testing strategy. I: Fault injection/tolerance. M: Model-based testing. R: Regression testing. W: Structural (white-box) testing. Legend for Type: A: Approach proposal. E: Evaluation. T: Tool and infrastructure.

Table 14
Testing-related papers published in the ICSE proceedings from 1987 to 2011 (2/4).

Year	#	Title	Tech	Type	Authors	Affiliations	Type of Eval. According to Zannier et al.	Eval	Type of Subjects	Citations (Google Scholar)
1997	32	A Theory of Probabilistic Functional Testing	B	A	G. Bernot; L. Bouaziz; P. L. Gall	University of Évry (Évry - France) CERMICS ENPC (Noisy le Grand - France)	Example Application	N	n/a	14
1998	33	An Empirical Study of Regression Test Selection Techniques	R	E	T. L. Graves; M. J. Harrold; J. -M. Kim; A. Porter; G. Rothermel	Bell Laboratories (Naperville – USA) Ohio State University (Columbus – USA) University of Maryland (College Park – USA) Oregon State University (Corvallis – USA)	Quasi-Experiment	Y	Programs	270
	34	What You See Is What You Test: A Methodology for Testing Form-Based Visual programs	W	A	G. Rothermel; L. Li; C. DuPuis; M. M. Burnett	Oregon State University (Corvallis – USA)	Exploratory Case Study	Y	Programs	94
1999	35	Coca: An automated Debugger for C	D	A, T	M. Ducassé	IRISA (Rennes – France)	Example Application	N	n/a	79
	36	Using a Goal-Driven Approach to Generate Test Cases for GUIs	A	A	A. M. Memon; M. E. Pollack; M. L. Soffa	University of Pittsburgh (Pittsburgh – USA)	Example Application	N	n/a	91
	37	Lutess: A Specification-Driven Testing Environment for Synchronous Software	A	A	L. du Bousquet; F. Ouabdesselam; J. -L. Richier; N. Zuanon	LSR-IMAG (St-Martin-d'Hères – France)	Exploratory Case Study	Y	Programs	73
	38	Residual Test Coverage Monitoring	W	A	C. Pavlopoulou; M. Young	Purdue University (West Lafayette – USA) University of Oregon (Eugene – USA)	Exploratory Case Study	Y	Programs	109
	39	Model-Based Testing in Practice	A, M	E	S. R. Dalal; A. Jain; N. Karunanithi; J. M. Leaton; C. M. Lott; G. C. Patton; B. M. Horowitz	Bell Communications Research (Morristown – USA) Bell Communications Research (Piscataway – USA)	Exploratory Case Study	Y	Programs	313
2000	40	Multivariate visualization in observation-based testing	W	A	D. Leon; A. Podgurski; L. J. White	Case Western Reserve University (Cleveland – USA)	Example Application	N	n/a	48
	41	An empirical study of regression test application frequency	R	E	J. -M. Kim; A. Porter; G. Rothermel	University of Maryland (College Park – USA) Oregon State University (Corvallis – USA)	Quasi-Experiment	Y	Programs	54
	42	Testing levels for object-oriented software	S	A	Y. Labiche; P. Thévenod-Fosse; H. Waeselynck; M. -H. Durand	LAAS-CNRS (Toulouse – France) Airbus (Toulouse – France)	Exploratory Case Study	Y	Programs	61
	43	Deriving test plans from architectural descriptions	M	A	A. Bertolino; F. Corradini; P. Inverardi; H. Muccini	CNR (Pisa – Italy) University of L'Aquila (L'Aquila – Italy)	Example Application	N	n/a	68
	44	WYSIWYT testing in the spreadsheet paradigm: an empirical evaluation	W	E	K. J. Rothermel; C. R. Cook; M. M. Burnett; J. Schonfeld; T. R. G. Green; G. Rothermel	University of Leeds (Leeds – UK) Oregon State University (Corvallis – USA)	Experiment	Y	Programs, People	81
2001	45	Analysis and Testing of Web Applications	W	A	F. Ricca; P. Tonella;	ITC-irst (Trento – Italy)	Exploratory Case Study	Y	Programs	394
	46	The Specification and Testing of Quantified Progress Properties in Distributed Systems	M	T	P. Krishnamurthy; P. A. G. Sivilotti	Ohio State University (Columbus – USA)	Example Application	N	n/a	6
	47	An Explorative Journey from Architectural Tests Definition downto Code Tests Execution	M	E	A. Bertolino; P. Inverardi; H. Muccini	CNR (Pisa – Italy) University of L'Aquila (L'Aquila – Italy)	Exploratory Case Study	Y	Programs	31
	48	Incorporating Varying Test Costs and Fault Severities into Test Case Prioritization	S	A	S. Elbaum; A. G. Malishevsky; G. Rothermel	University of Nebraska (Lincoln – USA) Oregon State University (Corvallis – USA)	Exploratory Case Study	Y	Programs	145
	49	Finding Failures by Cluster Analysis of Execution Profiles	S	A	W. Dickinson; D. Leon; A. Podgurski	Case Western Reserve University (Cleveland – USA)	Exploratory Case Study	Y	Programs	168
2002	50	A history-based test prioritization technique for regression testing in resource constrained environments	R	A	J. -M. Kim; A. Porter	University of Maryland (College Park – USA)	Quasi-Experiment	Y	Programs	151
	51	The impact of test suite granularity on the cost-effectiveness of regression testing	R	E	G. Rothermel; S. Elbaum; A. Malishevsky; P. Kallakuri; B. Davia	Oregon State University (Corvallis – USA) University of Nebraska (Lincoln – USA)	Quasi-Experiment	Y	Programs	52
	52	Automated test case generation for spreadsheets	A	A	M. Fisher; M. Cao; G. Rothermel; C. R. Cook; M. M. Burnett	Oregon State University (Corvallis – USA)	Quasi-Experiment	Y	Spreadsheets	54
	53	An empirical evaluation of fault-proneness models	S	E	G. Denaro; M. Pezzè	Politecnico di Milano (Milano – Italy) University of Milan – Bicocca (Milano – Italy)	Exploratory Case Study	Y	Programs	92
	54	Tracking down software bugs using automatic anomaly detection	D	T	S. Hangal; M. S. Lam	Sun Microsystems India Pvt. Ltd. (Bangalore – India) Stanford University (Stanford – USA)	Exploratory Case Study	Y	Programs	453
2003	55	Constructing Test Suites for Interaction Testing	S	A	M. B. Cohen; P. B. Gibbons; W. B. Mugridge; C. J. Colbourn	University of Auckland (Auckland – New Zealand) Arizona State University (Tempe – USA)	Exploratory case study	Y	Programs	200
	56	Improving Web Application Testing with User Session Data	A	A	S. Elbaum; S. Karre; G. Rothermel	University of Nebraska (Lincoln – USA) Oregon State University (Corvallis – USA)	Quasi-experiment	Y	People	167
	57	Improving Test Suites via Operational Abstraction	A	A	M. Harder; J. Mellen; M. D. Ernst	Massachusetts Institute of Technology (Cambridge – USA)	Quasi-experiment	Y	Programs	146
	58	Cadena: An Integrated Development, Analysis, and Verification Environment for Component-based Systems	M	T	J. Hatcliff; X. Deng; M. B. Dwyer; G. Jung; V. P. Ranganath	Kansas State University (Manhattan – USA)	n/a	N	n/a	202
	59	Fragment Class Analysis for Testing of Polymorphism in Java Software	W	A	A. Rountev; A. Milanova; B. G. Ryder	Ohio State University (Columbus – USA) Rutgers University (Piscataway – USA)	Exploratory case study	Y	Programs	81
	60	A Framework for Component Deployment Testing	B	T	A. Bertolino; A. Polini	CNR (Pisa – Italy)	n/a	N	n/a	44
	61	Data Flow Testing as Model Checking	A, M. W	A	H. S. Hong; S. D. Cha; I. Lee; O. Sokolsky; H. Ural	Korea Advanced Institute of Science and Technology (Daejeon – South Korea) University of Pennsylvania (Philadelphia – USA) University of Ottawa (Ottawa – Canada)	Example Application	N	n/a	95
	62	Modular Verification of Software Components in C	M	T	S. Chaki; E. Clarke; A. Groce; S. Jha; H. Veith	Carnegie Mellon University (Pittsburgh – USA) University of Wisconsin (Madison - USA) Vienna University of Technology (Vienna – Austria)	Example Application	N	n/a	456

Legend for Tech: A: Test case generation. B: Functional (black-box) testing. D: Debugging. F: Fault-based testing. S: Testing strategy. I: Fault injection/tolerance. M: Model-based testing. R: Regression testing. W: Structural (white-box) testing. Legend for Type: A: Approach proposal. E: Evaluation. T: Tool and infrastructure.

Table 15
Testing-related papers published in the ICSE proceedings from 1987 to 2011 (3/4).

Year	#	Title	Tech	Type	Authors	Affiliations	Type of Eval. According to Zannier et al.	Eval	Type of Subjects	Citations (Google Scholar)
2004	63	Using Simulation to Empirically Investigate Test Coverage Criteria Based on Statechart	M	E	L. C. Briand; Y. Labiche; Y. Wang	Carleton University (Ottawa – Canada)	Exploratory case study	Y	Programs	55
	64	Automated Generation of Test Programs From Closed Specifications of Classes and Test Cases	A	T	W. K. Leow; S. C. Khoo; Y. Sun	National University of Singapore (Queenstown – Singapore)	n/a	N	n/a	29
	65	Bi-Criteria Models for All-Uses Test Suite Reduction	W, R	A	J. Black; E. Melachrinoudis; D. Kaeli	Northeastern University (Boston – USA)	Case study	Y	Programs	55
	66	Generating Tests from Counterexamples	A, M	A	D. Beyer; A. J. Chlipala; T. A. Henzinger; R. Jhala; R. Majumdar	University of California (Berkeley – USA) University of California (Los Angeles – USA)	Exploratory case study	Y	Programs	152
	67	Automated Support for Development, Maintenance, and Testing in the Presence of Implicit Control Flow	W	A	S. Sinha; A. Orso; M. J. Harrold	Georgia Institute of Technology (Atlanta – USA)	Case study	Y	Programs	28
2005	68	Testing Database Transactions with AGENDA	A	T	Y. Deng; P. G. Frankl; D. Chays	Polytechnic University (New York – USA) Adelphi University (New York – USA)	Exploratory case study	Y	Programs	53
	69	A Framework of Greedy Methods for Constructing Interaction Test Suites	S	T	R. C. Bryce; C. J. Colbourn; M. B. Cohen	Arizona State University (Tempe – USA) University of Nebraska (Lincoln – USA)	Quasi-experiment	Y	Algorithms	70
	70	Demand-Driven Structural Testing with Dynamic Instrumentation	W	T	J. Misurda; J. A. Clause; J. L. Reed; B. R. Childers; M. L. Soffa	University of Pittsburgh (Pittsburgh – USA) University of Virginia (Charlottesville – USA)	Quasi-experiment	Y	Programs	48
	71	Is Mutation an Appropriate Tool for Testing Experiments?	F	E	J. H. Andrews ; L. C. Briand; Y. Labiche	University of Western Ontario (London – Canada) Carleton University (Ottawa – Canada)	Quasi-experiment	Y	Programs	323
	72	An Empirical Study of Fault Localization for End-User Programmers	D	E	J. R. Ruthruff; M. M. Burnett; G. Rothermel	University of Nebraska (Lincoln – USA) Oregon State University (Corvallis – USA)	Quasi-experiment	Y	Spreadsheets	32
	73	Locating causes of program failures	D	A	H. Cleve; A. Zeller	Saarland University (Saarbrüken – Germany)	Case Study	Y	Programs	324
	74	An Empirical Evaluation of Test Case Filtering Techniques Based On Exercising Complex Information Flows	W	E	D. Leon; W. Masri; A. Podgurski	Case Western Reserve University (Cleveland – USA) American University of Beirut (Beirut – Lebanon)	Quasi-experiment	Y	Programs	37
2006	75	Improving Test Suites for Efficient Fault Localization	D	A	B. Baudry; F. Fleurey; Y. Le Traon	IRISA (Rennes – France) France Télécom (Lannion – France)	Exploratory case study	Y	Programs	81
	76	Automated, Contract-based User Testing of Commercial-Off-The-Shelf Components	B	A	L. C. Briand; Y. Labiche; M. M. Sówka	Carleton University (Ottawa – Canada) Simula Research Laboratory (Lysaker – Norway)	Exploratory case study	Y	Programs	28
	77	An Intensional Approach to the Specification of Test Cases for Database Applications	B	A	D. Willmor; S. M. Embury	University of Manchester (Manchester – UK)	Exploratory case study	Y	Databases	40
2007	78	Regression Test Selection for AspectJ Software	R	A	G. Xu; A. Rountev	Ohio State University (Columbus – USA)	Case study	Y	Programs	39
	79	Feedback-directed Random Test Generation	A	A	C. Pacheco; S. K. Lahiri; M. D. Ernst; T. Ball	Massachusetts Institute of Technology (Cambridge – USA) Microsoft Research (Redmond – USA)	Quasi-experiment	Y	Programs	242
	80	Compatibility and regression testing of COTS-component-based software	R, A	A	L. Mariani; S. Papagiannakis; M. Pezzè	University of Milan – Bicocca (Milano – Italy)	Exploratory case study	Y	Programs	37
	81	A Technique for Enabling and Supporting Debugging of Field Failures	D	A	J. A. Clause; A. Orso	Georgia Institute of Technology (Atlanta – USA)	Exploratory case study	Y	Programs	47
	82	GoalDebug: A Spreadsheet Debugger for End Users	D	E	R. Abraham; M. Erwig	Oregon State University (Corvallis – USA)	Exploratory case study	Y	Spreadsheets	26
	83	Using GUI Run-Time State as Feedback to Generate Test Cases	A, M	A	X. Yuan; A. M. Memon	University of Maryland (College Park – USA)	Case study	Y	Programs	67
	84	Automated Generation of Context-Aware Tests	A	A	Z. Wang; S. Elbaum; D. S. Rosenblum	University of Nebraska (Lincoln – USA) University College London (London – UK)	Exploratory case study	Y	Programs	39
	85	Hybrid Concolic Testing	A	A	R. Majumdar; K. Sen	University of California (Berkeley – USA) University of California (Los Angeles – USA)	Exploratory case study	Y	Programs	154
2008	86	Testing Pervasive Software in the Presence of Context Inconsistency Resolution Services	A, W	A	H. Lu; W. K. Chan; T. H. Tse	The University of Hong Kong (Hong Kong – China) City University of Hong Kong (Hong Kong – China)	Quasi-experiment	Y	Programs	28
	87	ARTOO: Adaptive Random Testing for Object-Oriented Software	A	A	I. Ciupa; A. Leitner; M. Oriol; B. Meyer	ETH Zurich (Zurich – Switzerland)	Case study	Y	Programs	79
	88	The Effect of Program and Model Structure on MC/DC Test Adequacy Coverage	M	E	A. Rajan; M. W. Whalen; M. P. E. Heimdahl	University of Minnesota (Minneapolis – USA) Rockwell Collins Inc. (Cedar Rapids – USA)	Quasi-experiment	Y	Programs	25
	89	Sufficient Mutation Operators for Measuring Test Effectiveness	F	E	A. S. Namin; J. H. Andrews ; D. J. Murdoch	University of Western Ontario (London – Canada)	Exploratory case study	Y	Programs	41

Legend for Tech: A: Test case generation. B: Functional (black-box) testing. D: Debugging. F: Fault-based testing. S: Testing strategy. I: Fault injection/tolerance. M: Model-based testing. R: Regression testing. W: Structural (white-box) testing. Legend for Type: A: Approach proposal. E: Evaluation. T: Tool and infrastructure.

Table 16
Testing-related papers published in the ICSE proceedings from 1987 to 2011 (4/4).

Year	#	Title	Tech	Type	Authors	Affiliations	Type of Eval. According to Zannier et al.	Eval	Type of Subjects	Citations (Google Scholar)
2009	90	Taming Coincidental Correctness: Coverage Refinement with Context Patterns to Improve Fault Localization	F, W	A	X. Wang; S.C. Cheung; W. K. Chan; Z. Zhang	Hong Kong University of Science and Technology (Hong Kong – China) City University of Hong Kong (Hong Kong – China) The University of Hong Kong (Hong Kong – China)	Case study	Y	Programs	48
	91	Lightweight Fault-Localization Using Multiple Coverage Types	F, W	E	R. Santelices; J. A. Jones; Y. Yu; M. J. Harrold	Georgia Institute of Technology (Atlanta – USA) University of California (Irvine – USA)	Exploratory case study	Y	Programs	69
	92	HOLMES: Effective Statistical Debugging via Efficient Path Profiling	D	A	T. M. Chilimbi; B. Liblit; K. Mehra; A. V. Nori; K. Vaswani	Microsoft Research (Redmond – USA) University of Wisconsin (Madison – USA) Microsoft Research (Bangalore – India)	Case Study	Y	Programs	88
	93	Maintaining and Evolving GUI-Directed Test Scripts	B	A	M. Grechanik; Q. Xie; C. Fu	Accenture Technology Labs (Chicago – USA)	Quasi-experiment	Y	Programs	34
	94	MINTS: A General Framework and Tool for Supporting Test-suite Minimization	W, B	A	H. -Y. Hsu; A. Orso	Georgia Institute of Technology (Atlanta – USA)	Case study	Y	Programs	20
	95	WISE: Automated Test Generation for Worst-Case Complexity	A	A	J. Burnim; S. Juvekar; K. Sen	University of California (Berkeley – USA)	Exploratory case study	Y	Programs	12
2010	96	An Exploratory Study of Fault-Proneness in Evolving Aspect-Oriented Programs	F	E	F. C. Ferrari; R. Burrows; O. A. L. Lemos; A. F. Garcia; E. Figueiredo; N. Cacho; F. Lopes; N. Temudo; L. Silva; S. Soares; A. Rashid; P. C. Masiero; T. Batista; J. C. Maldonado	University of São Paulo (São Carlos – Brazil) Pontifical Catholic University of Rio de Janeiro (Rio de Janeiro – Brazil) Lancaster University (Lancaster – UK) Federal University of São Paulo (São José dos Campos – Brazil) Federal University of Rio Grande do Norte (Natal – Brazil) University of Pernambuco (Recife – Brazil) Federal University of Pernambuco (Recife – Brazil)	Exploratory case study	Y	Programs	16
	97	Test Generation through Programming in UDITA	A, B	A	M. Gligoric; T. Gvero; V. Jagannath; S. Khurshid; V. Kuncak; D. Marinov	University of Illinois (Urbana – USA) Ecole Polytechnique Fédérale (Lausanne - Switzerland) University of Texas (Austin – USA)	Exploratory case study	Y	Programs	36
	98	Detecting Atomic-Set Serializability Violations in Multithreaded Programs through Active Randomized Testing	A	A	Z. Lai; S.C. Cheung; W. K. Chan	Hong Kong University of Science and Technology (Hong Kong – China) City University of Hong Kong (Hong Kong – China)	Exploratory case study	Y	Programs	20
	99	Falcon: Fault Localization in Concurrent Programs	F	A	S. Park; R. W. Vuduc; M. J. Harrold	Georgia Institute of Technology (Atlanta – USA)	Case study	Y	Programs	20
	100	Practical Fault Localization for Dynamic Web Applications	F, A	A	S. Artzi; J. Dolby; F. Tip; M. Pistoia	IBM T.J. Watson Research Center (Yorktown Heights – USA)	Exploratory case study	Y	Programs	18
	101	From Behaviour Preservation to Behaviour Modification: Constraint-Based Mutant Generation	F	A	F. Steimann; A. Thies	University of Hagen (Hagen – Germany)	Exploratory case study	Y	Programs	4
	102	Is Operator-Based Mutant Selection Superior to Random Mutant Selection?	F	E	L. Zhang; S. -S. Hou; J. -J. Hu; T. Xie; H. Mei	Peking University (Beijing – China) North Carolina State University (Raleigh – USA)	Case study	Y	Programs	11
	103	Using Symbolic Evaluation to Understand Behavior in Configurable Software Systems	W	E	E. Reisner; C. Song; K. -K. Ma; J. S. Foster; A. Porter	University of Maryland (College Park – USA)	Case study	Y	Programs	28
2011	104	Camouflage: Automated Anonymization of Field Data	A, S	A	J. A. Clause; A. Orso	University of Delaware (Newark – USA) Georgia Institute of Technology (Atlanta – USA)	Case study	Y	Programs	16
	105	aComment: Mining Annotations from Comments and Code to Detect Interrupt Related Concurrency Bugs	D	A	L. Tan; Y. Zhou; Y. Padioleau	University of Waterloo (Waterloo – Canada) University Of California (San Diego – USA) Facebook Inc. (Palo Alto – USA)	Exploratory case study	Y	Programs	9
	106	Coverage Guided Systematic Concurrency Testing	S	A	C. Wang; M. Said; A. Gupta	NEC Laboratories America, Inc. (Princeton – USA) Western Michigan University (Kalamazoo – USA)	Exploratory case study	Y	Programs	7
	107	Angelic Debugging	D	A	S. Chandra; E. Torlak; S. Barman; R. Bodik	IBM Research (Hawthorne – USA) University of California (Berkeley – USA)	Case Study	Y	Programs	7
	108	Program Abstractions for Behaviour Validation	B	A	G. de Caso; V. Braberman; D. Garbervetsky; S. Uchitel	University of Buenos Aires (Buenos Aires – Argentina) Imperial College (London – UK)	Exploratory case study	Y	Programs	4
	109	Automated Cross-Browser Compatibility Testing	M	A	A. Mesbah; M. R. Prasad	University of British Columbia (Vancouver – Canada) Fujitsu Laboratories of America (Sunnyvale – USA)	Case study	Y	Programs	10
	110	A Framework for Automated Testing of JavaScript Web Applications	A, W	T	S. Artzi; J. Dolby; S. H. Jensen; A. Møller; F. Tip	IBM T.J. Watson Research Center (Yorktown Heights – USA) Aarhus University (Aarhus – Denmark)	Case study	Y	Programs	17
	111	Precise Identification of Problems for Structural Test Generation	W	A	X. Xiao; T. Xie; N. Tillmann; J. de Halleux	North Carolina State University (Raleigh – USA) Microsoft Research (Redmond – USA)	Case study	Y	Programs	10

Legend for Tech: A: Test case generation. B: Functional (black-box) testing. D: Debugging. F: Fault-based testing. S: Testing strategy. I: Fault injection/tolerance. M: Model-based testing. R: Regression testing. W: Structural (white-box) testing. Legend for Type: A: Approach proposal. E: Evaluation. T: Tool and infrastructure.

References

Andrews, J.H., Briand, L.C., Labiche, Y.,2005. Is mutation an appropriate tool for testing experiments? In: Proceedings of the 27th International Conference on Software Engineering (ICSE). ACM Press, St. Louis, MO, USA, pp. 402–411.

Boehm, B., Rombach, H.D., Zelkowitz, M.V., 2005. Foundations of Empirical Software Engineering: The Legacy of Victor R. Basili. Springer-Verlag New York Inc., Secaucus, NJ, USA.

Campanha, D.N., Souza, S.R.S., Maldonado, J.C.,2010. Mutation testing in procedural and object-oriented paradigms: an evaluation of data structure programs. In: Proceedings of the 2010 Brazilian Symposium on Software Engineering, SBES'10. IEEE Computer Society, Washington, DC, USA, pp. 90–99.

Delamaro, M.E., Chaim, M.L., Vincenzi, A.M.R., Jino, M., Maldonado, J.C.,2011. Twenty-five years of research in structural and mutation testing. In: Proceedings of the 25th Brazilian Symposium on Software Engineering (SBES). IEEE Computer Society Press, São Paulo, SP, Brazil, pp. 40–49.

Deng, Y., Frankl, P., Chays, D.,2005. Testing database transactions with AGENDA. In: Proceedings of the 27th International Conference on Software Engineering, ICSE'05. ACM, New York, NY, USA, pp. 78–87.

Durelli, V.H.S., Araujo, R.F., Silva, M.A.G., Oliveira, R.A.P., Maldonado, J.C., Delamaro, M.E.,2011. What a long, strange trip it's been: past present and future perspectives on software testing research. In: Proceedings of the 25th Brazilian Symposium on Software Engineering (SBES). IEEE Computer Society Press, São Paulo, SP, Brazil, pp. 30–39.

Feitelson, D.G., 2007. Experimental computer science (Guest Editor's Introduction). Communications of the ACM 50, 24–26.

Freeman, P.A., 2008. Back to experimentation. Communications of the ACM 51, 21–22.

Garcia, A., 2011. CBSoft 2011 – SBES – Call for Papers. Available from: http://www.each.usp.br/cbsoft2011/ingles/sbes/chamada_sbes_en.html (accessed 20.01.12).

Graves, T.L., Harrold, M.J., Kim, J.-M., Porter, A., Rothermel, G.,1998. An empirical study of regression test selection techniques. In: Proceedings of the 20th International Conference on Software Engineering, ICSE'98. IEEE Computer Society, Washington, DC, USA, pp. 188–197.

Hsueh, M.-C., Tsai, T.K., Iyer, R.K., 1997. Fault injection techniques and tools. IEEE Computer 30, 75–82.

IEEE, 1990. IEEE Standard Glossary of Software Engineering Terminology, Standard 610.12. Institute of Electric and Electronic Engineers.

Jacso, P., 2009. Google Scholar's Ghost Authors. Available from: http://www.libraryjournal.com/article/CA6703850.html (accessed 17.10.12).

Juristo, N., Moreno, A.M., Vegas, S., Solari, M., 2006. In search of what we experimentally know about unit testing. IEEE Software 23, 72–80.

Juristo, N., Moreno, A., Vegas, S., Shull, F., 2009. A look at 25 years of data. IEEE Software 26, 15–17.

Kim, J.-M., Porter, A., 2002. A history-based test prioritization technique for regression testing in resource constrained environments. In: Proceedings of the 24th International Conference on Software Engineering, ICSE'02. ACM, New York, NY, US, pp. 119–129. ISBN 1-58113-472-X, http://doi.acm.org/10.1145/581339.581357, doi:10.1145/581339.581357.

Lai, Z., Cheung, S.C., Chan, W.K.,2008. Inter-context control-flow and data-flow test adequacy criteria for NesC applications. In: Proceeding of the 16th ACM SIGSOFT International Symposium on Foundations of Software Engineering (FSE). ACM, Atlanta, GA, USA, pp. 94–104.

Lemos, O.A.L., Ferrari, F.C., Eler, M.M., Masiero, P.C., Maldonado, J.C.,2011. Evaluation studies of software testing research in the Brazilian Symposium on Software Engineering. In: Proceedings of the 25th Brazilian Symposium on Software Engineering (SBES). IEEE Computer Society Press, São Paulo, SP, Brazil, pp. 56–65.

Mathur, A.P., 2007. Foundations of Software Testing. Addison-Wesley Professional, Canada.

Myers, G.J., Sandler, C., Badgett, T., Thomas, T.M., 2004. The Art of Software Testing, 2nd edn. John Wiley & Sons, Hoboken, NJ, USA.

Petersen, K., Feldt, R., Mujtaba, S., Mattsson, M.,2008. Systematic mapping studies in software engineering. In: Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering (EASE). The British Computer Society, Bari, Italy, pp. 1–10.

Rothermel, K.J., Cook, C.R., Burnett, M.M., Schonfeld, J., Green, T.R.G., Rothermel, G.,2000. WYSIWYT testing in the spreadsheet paradigm: an empirical evaluation. In: Proceedings of the 22nd International Conference on Software Engineering, ICSE'00. ACM, New York, NY, USA, pp. 230–239.

Rountev, A., Milanova, A., Ryder, B.G.,2003. Fragment class analysis for testing of polymorphism in Java software. In: Proceedings of the 25th International Conference on Software Engineering, ICSE'03. IEEE Computer Society, Washington, DC, USA, pp. 210–220.

Rountev, A., Milanova, A., Ryder, B.G., 2004. Fragment class analysis for testing of polymorphism in Java software. IEEE Transactions on Software Engineering 30 (6), 372–387.

Sjoberg, D.I.K., Hannay, J.E., Hansen, O., By Kampenes, V., Karahasanovic, A., Liborg, N.-K., Rekdal, A.C., 2005. A survey of controlled experiments in software engineering. IEEE Transactions on Software Engineering 31, 733–753.

Wong, W.E., Horgan, J.R., London, S., Mathur, A.P.,1995. Effect of test set minimization on fault detection effectiveness. In: Proceedings of the 17th International Conference on Software Engineering, ICSE'95. ACM, New York, NY, USA, pp. 41–50.

Zannier, C., Melnik, G., Maurer, F.,2006. On the success of empirical studies in the international conference on software engineering. In: Proceedings of the 28th International Conference on Software engineering (ICSE). ACM Press, Shanghai, China, pp. 341–350.

Otávio Lemos received his M.Sc. (2005) and D.Sc. (2009) degrees in computer science from the University of São Paulo (USP). He was a visiting researcher at UC Irvine from January to July 2007, and from June to August 2012. He's currently an Assistant Professor at the Federal University of São Paulo (UNIFESP), doing research on software testing, reuse, and experimentation.

Fabiano Ferrari received his D.Sc. degree in Computer Science from the University of São Paulo (USP), Brazil, in 2010. He is currently an Assistant Professor at the Computing Department of the Federal University of São Carlos (UFSCar). His research interests include: testing of Object-Oriented and Aspect-Oriented software, software metrics, Experimental Software Engineering and Agile Methods.

Marcelo Eler received his PhD in computer science from the University of São Paulo (ICMC-USP) in 2012. He was a visiting researcher at ISTI-CNR, Italy, from septemper 2010 to February 2011. He has been recently hired as an Assistant Professor at University of São Paulo (EACH-USP) and his research interests include: software reuse (components and services), software testing and aspect oriented programming.

José Maldonado is a Professor of the Department of Computer Systems of the Institute of Mathematics and Computer Science of the University of São Paulo (USP), Brazil, where he is one of the leaders of the Software Engineering Research Group. His main research interests are software verification and validation, experimental software engineering, critical embedded systems and distance learning.

Paulo Masiero is a Professor of the Department of Computing Systems of the Institute of Mathematics and Computer Science of the Universidade de São Paulo, Brazil, where he is one of the leaders of the Software Engineering Research Group. His main research interests are testing of aspect-oriented programs, web services and embedded systems, software product lines, software design.