Noname manuscript No.
(will be inserted by the editor)

On the search for industry-relevant regression testing research

Nauman bin Ali[1] · Emelie Engström[2] · Masoumeh Taromirad[3] · Mohammad Reza Mousavi[3,4] · Nasir Mehmood Minhas[1], Daniel Helgesson[2] · Sebastian Kunze[3] · Mahsa Varshosaz[3]

Received: date / Accepted: date

Abstract Regression testing is a means to assure that a change in the software, or its execution environment, does not introduce new defects. It involves the expensive undertaking of rerunning test cases. Several techniques have been proposed to reduce the number of test cases to execute in regression testing, however, there is no research on how to assess industrial relevance and applicability of such techniques. We conducted a systematic literature review with the following two goals: firstly, to enable researchers to design and present regression testing research with a focus on industrial relevance and applicability and secondly, to facilitate the industrial adoption of such research by addressing the attributes of concern from the practitioners' perspective. Using a reference-based search approach, we identified 1068 papers on regression testing. We then reduced the scope to only include papers with explicit discussions about relevance and applicability (i.e. mainly studies involving industrial stakeholders). Uniquely in this literature review, practitioners were consulted at several steps to increase the likelihood of achieving our aim of identifying factors important for relevance and applicability. We have summarised the results of these consultations and an analysis of the literature in three taxonomies, which capture aspects of industrial-relevance regarding the regression testing techniques. Based on these taxonomies, we mapped 38 papers reporting the evaluation of 26 regression testing techniques in industrial settings.

Keywords Regression testing · industrial relevance · systematic literature review · taxonomy · recommendations

N. B. Ali
Blekinge Institute of Technology
E-mail: nauman.ali@bth.se

E. Engström
Lund University
E-mail: emelie.engstrom@cs.lth.se
[1]Blekinge Institute of Technology
[2]Lund University
[3]Halmstad University
[4]University Leicester

1 Introduction

Regression testing remains an unsolved and increasingly significant challenge in industrial software development. As a major step towards quality assurance, regression testing poses an important challenge for the seamless evolution (e.g., continuous integration and delivery) of large-scale software. Similarly, dealing with variability (e.g., in software product lines/product variants) makes regression testing of industrial software a non-trivial matter. Testing is highly repetitive at all levels and stages of the development, and for large and complex systems precision in regression test scoping becomes crucial.

These challenges have led to a large body of academic research. There is even a multitude of systematic literature reviews classifying and analysing the various proposed techniques for regression testing. For example, there are eleven literature reviews on regression testing published since 2010 (Rosero et al. (2016); Felderer and Fourneret (2015); Engström et al. (2010a); Zarrad (2015); Kazmi et al. (2017); Harrold and Orso (2008); Catal (2012); Yoo and Harman (2012); Qiu et al. (2014); Singh et al. (2012); Catal and Mishra (2013)).

Despite this extensive body of research literature, research results have shown to be hard to adopt for the practitioners (Rainer et al. (2005, 2006); Rainer and Beecham (2008); Engström and Runeson (2010); Engström et al. (2012); Ekelund and Engström (2015)). First of all, some results are not accessible for practitioners due to the discrepancies in terminology between industry and academia, which in turn makes it hard to know what to search for in the research literature. Furthermore, many empirical investigations are done in controlled experimental settings that have little in common with the complexity of an industrial setting. Hence, for practitioners, the relevance of such results is hard to assess. Engström and Runeson (2010) surveyed regression testing practices, which highlighted the variation in regression testing contexts and the need for holistic industrial evaluations.

There are today a significant number of industrial evaluations of regression testing. Unfortunately, also these results are hard to assess for the practitioners, since there are no conceptual models verified by practitioners to interpret, compare, and contrast different regression testing techniques. Engström et al. (2012) conducted an in-depth case study on the procedures undertaken at a large software company to search for a relevant regression testing technique and to evaluate the benefits of introducing it into the testing process at the company. This study further emphasises the need for support in matching the communication of empirical evidence in regression testing with guidelines for identifying context constraints and desired effects that are present in practice.

To respond to this need, in this paper, we review the literature from a relevance and applicability perspective. Using the existing literature reviews as a seed set for snowball sampling Wohlin (2014), we identified 1068 papers on regression testing, which are potentially relevant for our study. To gain as many insights as possible about relevance and applicability we have focused the review on large-scale industrial evaluations of regression testing techniques, as these studies in many cases involve stakeholders and are more likely to report these issues.

Both relevance and applicability are relative to a context, and we are not striving to find a general definition of the concepts. In our study, we are extracting factors that may support a practitioner (or researcher) in assessing relevance and applicability in their specific cases. We define relevance as a combination of desired

(or measured) effects and addressed context factors and include every such factor that have been reported in the included studies. Similarly, applicability, or the cost of adopting a technique, may be assessed by considering the information sources and entities utilised for selecting and/or prioritising regression tests. For each of these facets, we provide a taxonomy to support classification and comparison of techniques with respect to industrial relevance and applicability of regression testing techniques.

The original research questions stem from an industry-academia collaboration[1] (involving three companies and two universities) on decision support for software testing. Guided by the SERP-test taxonomy Engström et al. (2017), a taxonomy for matching industrial challenges with research results in software testing, we elicited nine important and challenging decision types for testers, of which three are instances of the regression testing challenge as summarised by Yoo and Harman (2012): regression test minimisation, selection, and prioritisation. These challenge descriptions (i.e., the generic problem formulations enriched with our collaborators' context and target descriptions) guided our design of the study.

To balance the academic view on the regression testing literature, we consulted practitioners in all stages of the systematic review (i.e., defining the research questions, inclusion and exclusion criteria, as well as the taxonomies for mapping selected papers).

The main contributions provided in this report are:

- three taxonomies designed to support the communication of regression testing research with respect to industrial relevance and applicability, and
- a mapping of 26 industrially evaluated regression testing techniques (in total 38 different papers) to the above-mentioned taxonomies.

The remainder of the paper is structured as follows: Section 2 summarises previous research on assessing the industrial relevance of research. It also presents an overview of existing systematic literature reviews on regression testing. Research questions raised in this study are presented in Section 3. Section 4 and Section 5 detail the research approach used in the study and its limitations, respectively. Sections 6 to 8 present the results of this research. Section 9 and Section 10 present advice for practitioners and academics working in the area of regression testing. Section 11 concludes the paper.

2 Related work

In this section, we briefly describe related work that attempts to evaluate the relevance of software engineering research for practice. We also discuss existing reviews on regression testing with a particular focusing on the evaluation of the industrial relevance of proposed techniques.

2.1 Evaluation of the industry relevance of research

Software engineering being an applied research area continues to strive to establish the industrial practice on scientific foundations. Along with the scientific rigour

[1] EASE- the Industrial Excellence Centre for Embedded Applications Software Engineering `http://ease.cs.lth.se/about/`

and academic impact, several researchers have attempted to assess the relevance and likely impact of research on practice.

Ivarsson and Gorschek (2011) proposed a method to assess the industrial relevance of empirical studies included in a systematic literature review. The criteria for judging relevance in thier proposal evaluates the realism in empirical evaluations on four aspects: 1) subjects (e.g. a study involving industrial practitioners), 2) context (e.g. a study done in an industrial settings), 3) scale (e.g. evaluation was done on a realistic size artifacts) and 4) research method (e.g. use of case study research). Several systematic reviews have used this approach to assess the applicability of research proposals in industrial settings (e.g. Ali et al. (2014); Munir et al. (2014)).

Other researchers have taken a different approach and have elicited the practitioners' opinion directly on individual studies (Carver et al. (2016); Franch et al. (2017); Lo et al. (2015)). In these studies, the practitioners were presented a summary of the articles and were asked to rate the relevance of a study for them on a Likert scale.

The *Impact project* was one such initiative aimed to document the impact of software engineering research on practice Osterweil et al. (2008). Publications attributed to this project, with voluntary participation from eminent researchers, covered topics like configuration management, inspections and reviews, programming languages and middle-ware technology. The approach used in the project was to start from a technology that is established in practice and trace its roots, if possible, to research Osterweil et al. (2008). However, the last publications indexed on the project page[2] are from 2008. One of the lessons learned from studies in this project is that the organisations wanting to replicate the success of other companies should *"mimic successful companies' transfer guidelines"* (Osterweil et al. (2008); Rombach et al. (2008)). Along those lines, the study presently read attempts to identify regression testing techniques with indications of value and applicability from industrial evaluations Jr. and Riddle (1985).

To address the lack of relevance, close industry-academia collaboration is encouraged (Jr. and Riddle (1985); Osterweil et al. (2008); Wohlin (2013)). One challenge in this regard is to make research more accessible to practitioners by reducing the communication-gap between industry and academia Engström et al. (2017). SERP-test Engström et al. (2017) is a taxonomy designed to support industry academia communication by guiding interpretation of research results from a problem perspective.

2.2 Reviews of regression testing research

We identified eleven reviews of software regression testing literature (Rosero et al. (2016); Felderer and Fourneret (2015); Engström et al. (2010a); Zarrad (2015); Kazmi et al. (2017); Harrold and Orso (2008); Catal (2012); Yoo and Harman (2012); Qiu et al. (2014); Singh et al. (2012); Catal and Mishra (2013)). Most of these reviews cover regression testing literature regardless of the application domain and techniques used. However, the following four surveys have a narrow scope: Qiu et al. (2014) and Zarrad (2015) target testing web-based applications,

[2] `https://www.sigsoft.org/impact.html`

and Felderer and Fourneret (2015) focus on identifying security-related issues, while Catal (2012) only considers literature where researchers have used Genetic Algorithms for regression testing. The tertiary study by Garousi and Mäntylä (2016) only maps the systematic literature studies in various sub-areas of software testing including regression testing. Instead of focusing only on regression testing research, Narciso et al. (2014) reviewed the literature on test case selection in general. They identified that only six of the selected studies were performed on large-scale systems, and only four of these were industrial applications.

In the most recent literature review, Kazmi et al. (2017) reviewed empirical research on regression testing of industrial and non-industrial systems of any size. They mapped the identified research to the following dimensions: evaluation metrics used in the study, the scope of the study, and what they have termed as the theoretical basis of the study (research questions, regression testing technique, SUT, and the dataset used). Their approach indicates a similar aim as other literature reviews: to identify "the most effective" technique considering the measures of "cost, coverage and fault detection". However, they do not take into consideration the aspect of the relevance and likely applicability of the research for industrial settings.

Among the identified reviews, only five discuss aspects related to the industrial application (Yoo and Harman (2012); Engström et al. (2010a); Catal and Mishra (2013); Singh et al. (2012); Harrold and Orso (2008)). Catal and Mishra (2013) found that 64% of the included 120 papers used datasets from industrial projects in their evaluation. They further recommend that future evaluations should be based on non-proprietary data sets that come from industrial projects (since these are representative of real industrial problems) Catal (2012). Yoo and Harman (2012) identified that a large majority of empirical studies use a small set of subjects largely from the SIR[3] repository. They highlight that it allows comparative/replication studies, and also warn about the bias introduced by working with the same small set of systems. Similarly, Engström et al. (2010a) concluded that most empirical investigations are conducted on small programs, which limits the generalisation to large-systems used in industry. Singh et al. (2012) also found that 50% of the 65 selected papers on regression test prioritisation included in their review use SIR systems. Furthermore, 29% of the studies use the same two systems from the repository.

Harrold and Orso (2008) reviewed the state of research and practice in regression testing. Authors presented the synthesis of main regression testing techniques and found that only a few techniques and tools developed by the researchers and practitioners are in use of industry. They also discussed the challenges for regression testing and divided the challenges into two sets (transitioning challenges and technical/conceptual issues). Along with the review of research on regression testing authors also presented the results of their discussions (an informal survey) with researchers and practitioners. They were intended to understand the impact of existing regression testing research and the major challenges to regression testing.

Unlike existing literature reviews, this study has an exclusive focus on research conducted in industrial settings. This study provides taxonomies to assist researchers in designing and reporting research to make the results more useful

[3] Software Infrastructure Repository http://sir.unl.edu/

for practitioners. Using the proposed taxonomies to report regression testing research, will enable synthesis in systematic literature reviews and help to take the field further. One form of such synthesis will be the technological-rules Storey et al. (2017) (as extracted in this paper) with an indication of the strength-of-evidence. For practitioners, these taxonomies allow reasoning about the applicability of research for their own unique context. The study also presents some technological-rules that are based on the results of this study which practitioners can consider research-informed recommendations from this study.

3 Research questions

In this study, we aggregate information on regression testing techniques that have been evaluated in industrial contexts. Our goal is to structure this information in such a way that it supports practitioners to make an informed decision regarding regression testing with a consideration for their unique context, challenges, and improvement targets. To achieve this goal we posed the following research questions:

RQ1: *How to describe an industrial regression testing problem?* Regression testing challenges are described differently in research (Rothermel and Harrold (1996)) and practice (Engström and Runeson (2010)). To be accessible and relevant for practitioners, research contributions in terms of technological rules (Storey et al. (2017)) need to be interpreted and incorporated into a bigger picture. This, in turn, requires alignment in both the abstraction level and the terminology of the academic and industrial problem descriptions. To provide support for such alignment, we develop taxonomies of desired effects and relevant context factors by extracting and coding knowledge on previous industrial evaluations of regression testing techniques.

RQ2: *How to describe a regression testing solution?* Practitioners need to be able to compare research proposals and assess their applicability and usefulness for their specific contexts. For this purpose, we extract commonalities and variabilities of research proposals that have been evaluated in industry.

RQ3: *How does the current research map to such problem description?* To provide an overview of the current state of the art, we compare groups of techniques through the lens of the taxonomies developed in RQ1 and RQ2.

4 Method

To capture what information is required to judge the industrial-relevance of regression testing techniques, we relied on: 1) industrial applications of regression testing techniques reported in the literature, 2) existing research on improving industry-academia collaboration in the area of software testing, 3) and close cooperation with practitioners.

To develop the three taxonomies presented in Section 6 and 7 and arrive at the results presented in Section 8 we conducted a systematic literature review of regression testing research, interleaving interaction with industry practitioners throughout the review process.

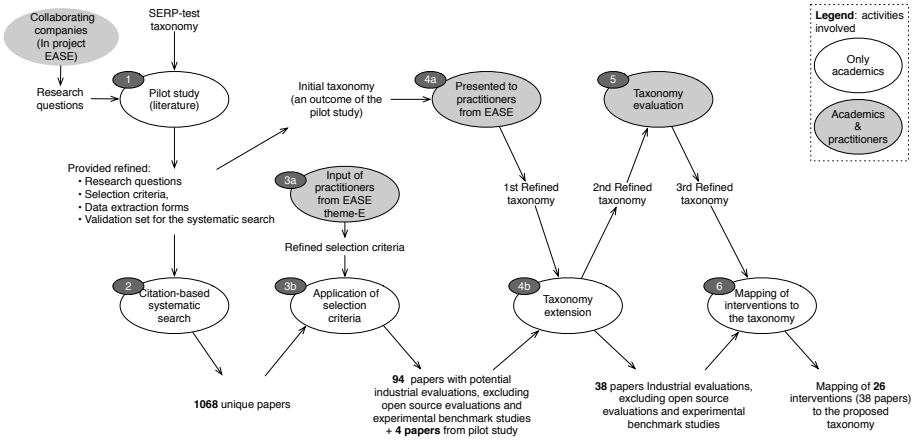

Fig. 1 A description of the flow of activities including alternately reviewing the literature and interacting with practitioners from the research project EASE.

The process followed can be divided into six steps, which are visualised in Figure 1. Research questions were initially formulated within a research collaboration on decision support for software testing (EASE). To validate the research questions and the approach of constructing a SERP taxonomy Engström et al. (2017) a pilot study was conducted (Step 1, Section 4.3). Based on the pilot study, a preliminary version of the taxonomy was presented to the researchers and practitioners in EASE, together with a refined study design for the extensive search. Through the extensive search of the literature (Step 2, Section 4.4) we identified 1068 papers on regression testing. This set was then reduced (Step 3, Section 4.5) by iteratively coding and excluding papers while refining the taxonomy (Step 4, Section 4.6). Finally, the constructed taxonomies were evaluated in a focus group meeting (Step 5, Section 4.7) and the regression testing techniques proposed in the selected papers were mapped to the validated version of the taxonomies (Step 6, Section 4.8).

4.1 Practitioners' involvement

As shown in Figure 1 (ovals with shaded background), practitioners were involved in three steps. For validating the selection criteria (Step 3a) a subset of selected papers was validated with practitioners. In Step 4a, the initial taxonomy was presented to EASE partners in a meeting. This meeting was attended by five key stakeholders in testing at the case companies. In Step 5, for taxonomy evaluation, we relied on a focus group with three key practitioners. The three practitioners came from two companies which develop large-scale software-intensive products and proprietary hardware. The participating companies are quite different from each other; *Sony Mobile Communications* has a strict hierarchical structure, well-established processes and tools, and is globally distributed, while the development at *Axis Communications AB, Sweden* still has the entrepreneurial culture of a small company and has less strictly defined processes. The profiles of the practitioners involved in the study are briefly summarized below:

Practitioner P1 is working at Axis. He has over eight years of experience in software development. At Axis, he is responsible for automated regression testing from unit to system-test levels. His team is responsible for the development and maintenance of the test suite. The complete regression test suite comprises over 1000 test cases that take around 7 hours to execute. He was also involved in previous research-based initiatives to improve regression testing at Axis Ekelund and Engström (2015).

Practitioner P2 also works at Axis communications. He has over 12 years of software development and testing experience. He is responsible for both automated and manual regression testing at the system-test level. He has recently overseen a complete assessment and review of the manually executed test-cases in the regression test suite.

Practitioner P3, works at Sony Mobile Communications. He has over 18 years of experience in software development with responsibilities primarily include software testing and overall automation and verification strategies. His current role as verification architect covers testing at all levels including regression testing. Within the EASE project, he has collaborated with researchers in several research-based investigations at his company.

4.2 Need for a literature review

The broader context of this study is a collaborative research project EASE (involving two academic and three industrial partners) working towards decision support in the context of software testing. As shown in Figure 1, the research questions and the need for a systematic literature review were identified in the context of this project. We considered the literature to answer the following two questions in the pilot study:

1. *Have existing systematic literature reviews taken into consideration the industrial relevance and applicability of regression testing techniques?* We identified 11 systematic literature studies (Rosero et al. (2016); Felderer and Fourneret (2015); Engström et al. (2010a); Zarrad (2015); Kazmi et al. (2017); Harrold and Orso (2008); Catal (2012); Yoo and Harman (2012); Qiu et al. (2014); Singh et al. (2012); Catal and Mishra (2013)), and they have been briefly discussed in Section 2. These studies have not addressed the research questions of interest for the current study.
2. *Are there sufficient papers reporting an industrial evaluation of regression testing techniques?* Once we had established the need to analyse the existing research from the perspective of industrial relevance, we conducted a pilot study to:
   - identify if there are sufficiently many published papers to warrant a systematic literature review,
   - develop an initial taxonomy that serves as a data extraction form in the main literature review, and
   - identify a set of relevant papers that serve as a validation set for our search strategy in the main literature review.

4.3 Pilot study

By manually searching through recent publications of key authors (identified in previous literature reviews discussed in Section 2) and by skimming through the top most-relevant results of keyword-based searches in Google Scholar, we identified 36 papers. Using a data extraction form based on the SERP-test taxonomy Engström et al. (2017), data were extracted from these papers. Data extraction was done independently by at least two reviewers and results were consolidated by discussion. This validation was considered useful for two reasons: firstly, through the cross-validation, we developed a shared understanding of the process. Secondly, since the results were to be used as a guide for data extraction in the main literature review, it was necessary to increase the reliability of this initial step.

The pilot study indicated that sufficient literature exists to warrant a systematic literature review. The results of analysing the extracted information were useful for formulating the data extraction forms for the main literature review.

4.4 Search strategy

Using the following search string, we identified the existing systematic literature studies on regression test optimization as listed in Table 1:
("regression test" OR "regression testing") AND ("systematic review" OR "research review" OR "research synthesis" OR "research integration" OR "systematic review" OR "systematic overview" OR "systematic research synthesis" OR "integrative research review" OR "integrative review" OR "systematic literature review" OR "systematic mapping" OR "systematic map"))

Additionally, we also used Yoo and Harman (2012) survey as it has a thorough coverage (with 189 references) and is the most-cited review in the area of regression testing. Using the references in the papers listed in Table 1, and the citations to these papers were retrieved in August 2016 from Google Scholar. We identified a set of 1068 papers as potentially relevant papers for our study. One of the systematic reviews, by Kazmi et al. (2017) as discussed in Section 2, was not used for snowball-sampling as it was published yet when the search was conducted.

ID	No. of References.	No. of Citations
Felderer and Fourneret (2015)	75	5
Qiu et al. (2014)	69	1
Zarrad (2015)	71	0
Catal (2012)	24	4
Singh et al. (2012)	80	14
Catal and Mishra (2013)	24	25
Engström et al. (2010a)	73	135
Narciso et al. (2014)	46	1
Rosero et al. (2016)	59	0
Yoo and Harman (2012)	189	515

Table 1: Systematic literature studies used as start-set for snowball sampling

Using the 36 papers identified in the pilot-study (see Section 4.3) as the validation-set for this study, we calculated the precision and recall (Zhang et al.

(2011); Kitchenham et al. (2015)) for our search strategy. 36 papers in a validation-set are reasonable for assessing the search strategy of a systematic literature review Kitchenham et al. (2015).

Recall = *100 * (# of papers from the validation-set identified in the search) / (total # of papers in the validation set).*

Precision = *100 * (total # of relevant papers (after applying the selection criteria) in the search results) / (total # of search results).*

$$Recall = \frac{32}{36} * 100 = 89\%$$

Only four of the papers in the pilot-study were not identified by our search strategy (Marijan et al. (2013); Marijan (2015); Saha et al. (2015); Xu et al. (2015)). These papers neither cite any of the literature reviews nor were they included by any of the literature reviews comprising the starting set for search in this study. We also included these four papers to the set of papers considered in this study.

As shown in Figure 1, after applying the selection criteria 94 relevant papers were identified. These papers were used to extend the taxonomy. Using this number, we calculated the precision of our search strategy as follows:

$$Precision = \frac{94}{1068} * 100 = 8\%$$

An optimum search strategy should maximise both precision and recall. However, our search strategy had high recall (with 89% recall it falls in the high recall range, i.e. $\geq$ 85% Zhang et al. (2011)) and low precision. The precision value was calculated considering the 94 papers that were used in extending the taxonomies.

The value of recall being well above the acceptable range Zhang et al. (2011) of 80% adds confidence to our search strategy. Furthermore, such low value of precision is typical of systematic literature reviews in software engineering e.g. approx. 5% Catal (2012) approx. 2% (Edison et al. (2013); Ali et al. (2014)), and below 1% (Engström et al. (2010a); Qiu et al. (2014); Singh et al. (2012)).

4.5 Selection of papers to include in the review

We applied a flexible design of the study and inclusion criteria were iteratively refined. The notion of "industrial" was further elaborated after the first iteration. To make the set of papers more manageable, we decided to exclude open source evaluations and industrial benchmark studies. The assumption was that such reports contain less information about application context and limitations in terms of technology adoption. The following inclusion-exclusion criteria were the ones finally applied:

- **Inclusion criteria:** peer-reviewed publications that report empirical evaluations of regression testing techniques in industrial settings. It was detailed as the following, include papers that:
  - are peer-reviewed (papers in conferences proceedings and journal articles)
  - report empirical research (case studies, experiments, experience reports ...)
  - report research conducted in industrial settings (i.e. uses a large-scale software system, involves practitioners or reports information on the real application context including the process).

  - investigate regression testing optimization techniques (i.e. regression test selection, prioritization, or minimization/ reduction/ maintenance)
- **Exclusion:** exclude papers that:
  - are non-peer reviewed (Ph.D. thesis, technical reports, books etc.)
  - report a non-empirical contribution (analytical/ theoretical/ proposals)
  - report evaluation in non-industrial settings.

We decided to use lines of code (LOC), if reported, as an indicator for the scale of the problem instead of the number of test cases in the test suite or turnaround time of a test suite (and similar metrics) for the following reasons:

- LOC/kLOC is the most commonly reported information regarding the size of a SUT.
- Size and execution time of individual test cases in a test suite varies a lot, therefore, an aggregate value reporting the number of test cases or the execution time of test cases is not very informative.

Techniques that work well on a small program may work on large programs. However, this is yet to be demonstrated. Practitioners seem to trust the results of research conducted in environments similar to their Zelkowitz et al. (1998). Previous research on assessing the industrial relevance of research has also relied on the realism in the evaluation setting regarding the research method, scale, context and users (Ali et al. (2014); Ivarsson and Gorschek (2011)).

We performed pilot selection on three papers to validate the selection criteria and to develop a shared understanding among the authors. Each author independently applied the selection criteria on the these randomly chosen papers. We discussed the decisions and reflected on the reasons for any discrepancies among the reviewers in a group format.

After the pilot-selection, remaining papers were assigned to each author randomly to apply selection criteria. Inclusion-exclusion was performed at three levels of screening: 'Titles only', 'Titles and abstracts only', and 'Full text'. If in doubt, the general instruction was to be more inclusive and defer the decision to the next level. Each excluded paper was evaluated by at least two reviewers.

Additionally, to validate that the studies we were selecting were indeed relevant, during the paper selection phase of this study, a sample of eight papers from the included papers was shared with practitioners. They labelled the paper as relevant or irrelevant for their companies and also explained their reasoning to us. This helped us to improve the coverage of information that practitioners are seeking, which they consider will help them make informed decisions regarding regression testing.

After applying the selection criteria on 1068 paper and excluding open source and industrial benchmarks we had 94 remaining papers. Four papers from the pilot-study were also added to this list. These 98 papers were randomly assigned to the authors of this paper for data-extraction and taxonomy extension. After full-text reading and data extraction, 38 papers were included as relevant papers (see list in Table 2), which represent 26 distinct techniques. All excluded papers were reviewed by an additional reviewer.

Study ID	Reference.	Study ID	Reference.
S1	Ekelund and Engström (2015)	S20	Rogstad et al. (2013)
S2	Saha et al. (2015)	S21	Krishnamoorthi and Mary (2009)
S3	Marijan et al. (2013)	S22	Tahvili et al. (2016)
S4	Marijan (2015)	S23	Janjua (2015)
S5	Buchgeher et al. (2013)	S24	Engström et al. (2010b)
S6	Skoglund and Runeson (2005)	S25	Wikstrand et al. (2009)
S7	White et al. (2008)	S26	Engström et al. (2011)
S8	White and Robinson (2004)	S27	Vöst and Wagner (2016)
S9	Zheng et al. (2006b)	S28	Huang et al. (2009)
S10	Zheng et al. (2007)	S29	Srivastava and Thiagarajan (2002)
S11	Zheng (2005)	S30	Hirzel and Klaeren (2016)
S12	Zheng et al. (2006a)	S31	Pasala and Bhowmick (2005)
S13	Wang et al. (2013b)	S32	Herzig et al. (2015)
S14	Wang et al. (2017)	S33	Li and Boehm (2013)
S15	Wang et al. (2015)	S34	Anderson et al. (2014)
S16	Wang et al. (2016)	S35	Lochau et al. (2014)
S17	Wang et al. (2013a)	S36	Devaki et al. (2013)
S18	Wang et al. (2014)	S37	Carlson et al. (2011)
S19	Rogstad and Briand (2016)	S38	Gligoric et al. (2014)

Table 2: The list of papers included in this study

4.6 Taxonomy extension

Table 3 presents an abstraction of the data extraction form, which was based on the first version of our taxonomy that was developed in the pilot study (see Step-4 onwards in Figure 1 that produced the "1st Refined taxonomy" and Section 4.3 for details of the pilot study). We followed the following steps to validate the extraction form and to develop a shared understanding of it:

1. Select a paper randomly from the set of potentially relevant papers.
2. All reviewers independently extract information from the paper using the data extraction form.
3. Compare the data-extraction results from individual reviewers.
4. Discuss and resolve any disagreements and if needed update the data extraction form.

This process was repeated three times before we were confident in proceeding with data extraction on the remaining set of papers.

The extracted information was used to develop extensions of SERP-test taxonomy Engström et al. (2017) relevant to our focus on regression testing techniques. Separate taxonomies for "addressed context factors", "evaluated effects" and "utilised information sources" were developed (shown as step 4.2 in Figure 1). The initial version of these taxonomies was developed in a workshop where six of the authors participated. Each of the taxonomies were then further refined by two of the authors and reviewed independently by a different pair of authors. This resulted in what is referred to as "2nd refined taxonomy" in Figure 1. This version of the taxonomy was further validated with practitioners, which is discussed in the following section.

Item	Value	Remarks
1) Meta information		
2) Description of testing technique		
3) Scope of technique		
4) High-level Effect/Purpose		
5) Characteristics of the SUT		
6) Characteristics of the regression testing process		
7) Required sources of information		
8) Type of required information		
9) Is this an industrial study?		
10) If yes, could the SUT be categorised as closed source?		
11) Is the paper within the scope of the study? If not, please explain the reason.		

Table 3: Data extraction form

4.7 Taxonomy evaluation

Once data analysis was complete, and we had created the three taxonomies presented in Section 6, Section 7 and Section 8, we conducted a focus group with three key stakeholders from the companies (brief profiles are presented in Section 4.1). In this focus group, moderated by the second author, we systematically collected practitioners' feedback on the context and effect taxonomies because these two taxonomies are supposed to describe the practitioners' need.

Practitioners were asked to assess the relevance of each of the nodes in the taxonomies (as presented in Table 4) and grade these from 1 to 3, where 1) means very relevant (i.e. we are interested in this research), 2) possibly relevant and 3) means irrelevant (i.e. we are not interested in such research). The practitioners were asked to respond based on their experience and not only based on their current need.

The feedback from the practitioners was taken into account, and some refinements to the taxonomies were made based on it. As this is primarily a literature review, we decided not to add factors that were not presented in the included papers although initial feedback pointed us to relevant factors in the studies. Neither did we remove factors completely from the taxonomies (although we removed some levels of detail in a couple of cases). The feedback was mainly used to evaluate and improve understandability of the taxonomies and changes were mainly structural.

4.8 Mapping of techniques to taxonomy

As shown in Figure 1 after incorporating the feedback from the practitioners in the taxonomy, we mapped the 26 techniques to our multi-faceted taxonomy. The reviewer(s) (one of the authors of the study) who were responsible for data extraction from the papers reporting the technique mapped the paper to the taxonomy. Two additional reviewers validated the mapping, and disagreements were resolved through discussion and by consulting the full-text of the papers. The results of the mapping are presented in Table 5.

5 Limitations

In this section, we discuss validity threats, our mitigation actions, and the limitations of the study.

5.1 Coverage of regression testing techniques:

To identify regression testing techniques that have been evaluated in industrial settings, we used snowball sampling search strategy. Snowball sampling has been effectively used to extend systematic literature reviews (Felizardo et al. (2016)). The decision to pursue this strategy was motivated by the large number of systematic literature studies (as discussed previously in Section 2) available on the topic. Some of these reviews (e.g. Yoo and Harman (2012) and Engström et al. (2010a)) are well cited, indicating visibility in the community. This increases the likelihood of finding recent research on the topic.

The search is not bound to a particular venue and is restricted to citations indexed by Scopus and Google Scholar before August 2016. We choose Scopus and Google scholar because of their comprehensive coverage of citations Thelwall and Kousha (2017). We are also confident in the coverage of the study as out of the 36 papers in the validation set, only four were not found (see Section 4).

To reduce the possibility of excluding relevant studies, we performed pilot selection on a randomly selected subset of papers. Furthermore, all excluded papers were reviewed independently by at least two of the authors of this paper. In cases of disagreement, the papers were included in the next phase of the study, i.e. data extraction and analysis.

5.2 Confidence in taxonomy building process and outcome

The taxonomies presented in this paper were based on data extracted from the included studies. To ensure that no relevant information was omitted, we tested the data extraction form on a sample of papers. This helped to develop a shared understanding of the form.

Furthermore, to increase the reliability of the study, the actual data extraction (from all selected papers) and the formulation of facets in the taxonomies were reviewed by two additional reviewers (authors of this paper).

As shown in Figure 1, the intermediate versions of the taxonomy were also presented to practitioners and their feedback was incorporated in the taxonomy. Possible confounding effects of their participation is due to their representativeness. The impact of practitioner feedback was mainly on the understandability and level of detail of the proposed taxonomies and a confounding effect could be that the understandability of the taxonomy is dependant of dealing with a context similar to our practitioners'. The two companies are large-scale and the challenges they face are typical for such contexts Ali et al. (2012); Engström et al. (2017). All participants have many years of experience of testing (as described in Sec 4.1). Therefore, their input is considered valuable for improving the validity of our study, which focuses on regression testing research of large-scale software systems.

The taxonomies presented were sufficient to capture the description of challenges and proposed techniques in the included studies and the practitioners consulted in this study. However, new facets may be added by both researchers and practitioners to accommodate additional concerns or aspects of interest.

5.3 Accuracy of the mapping of techniques and challenges

All mappings of included papers to the various facets of the three taxonomies were reviewed by an additional reviewer. Disagreements were discussed, and the full-text of the papers was consulted to resolve them. Despite these measures, there is still a threat of misinterpretation of the papers, which could be further reduced for example by consulting the authors of the papers included in this study to validate our classification. However, due to practical reasons we did not implement this mitigation strategy.

6 RQ1 – Regression testing problem description

In response to RQ1, we created taxonomies of addressed context factors and desired effects investigated in the included papers.

The taxonomies created in this study follow the SERP-taxonomy architecture Petersen and Engström (2014), i.e. they cover four facets, *intervention*, *context constraints*, *objective/effect* and *scope*. A SERP-taxonomy, should include one taxonomy for each facet. In our case, we create the regression testing taxonomies by extending an existing SERP-taxonomy (i.e. SERP-test Engström et al. (2017)) by adding the details specific to regression testing. More precisely, we develop extensions for three out of four SERP facets: *context factors* (extends context in SERP-test), *desired effects* (extends objective\improvements in SERP-test) and *utilised information entities and attributes* (extends intervention in SERP). We do not extend the scope taxonomy further since regression testing is in itself a scope entity in SERP test, which all reviewed techniques target.

The taxonomy creation was done in three steps (considering both the researcher's and the practitioner's perspective on the regression testing challenge): firstly we, together with our industry partners, defined an initial set of factors and targets which were important to them; secondly we extracted information regarding these factors in the included papers and extended the taxonomies with details provided in the reports, and finally we evaluated the extended taxonomies in a focus group meeting with our industry partners to get feedback on its relevance and understandability to them in their search for applicable regression testing techniques. The items of the final taxonomies are visible in Table 4

At the highest abstraction level, all categories of our proposed taxonomies were considered relevant when describing a regression testing challenge (i.e. characteristics of the system, the testing process and test suite and people related factors in the context taxonomy and similarly improved coverage, efficiency, effectiveness and awareness in the effect taxonomy).

The taxonomies reported in this paper are the revised version that addresses the feedback from this focus group. Due to the dual focus when creating the taxonomies, we believe they could provide guidelines for both researchers and

practitioners in defining the real-world regression testing problems they address, or wish to address consistently to support the mapping between research and practice.

6.1 Investigated context factors

The purpose of the context taxonomy can be summarised as: *Provide support for identifying characteristics of an industrial environment that make regression testing challenging and hence support the search for techniques appropriate for the context.*

Table 4 shows a taxonomy of contextual factors that were investigated in the included papers, as well as considered relevant by our industry partners. To be classified as an investigated context factor the mere mentioning of it in general terms was not considered sufficient, only in cases where the authors of the study include a discussion or explanation of the effect a factor has on regression testing and why it is considered in their study we include it as an investigated context factor.

Since we only include factors that have been discussed in the literature, the context taxonomy is not extensive but can still be used as a guideline for describing regression testing problems and solutions. We identified three main categories of relevant contextual factors (*system related, process related*, and *people related*) that have been addressed in the included papers.

6.1.1 System related context factors

System related context factors include factors regarding the system (or subsystem) under test, such as *size, complexity* and *type*. How size and complexity are measured varies in the studies, but a common measure of size is lines of code. Factors that are reported to add to the complexity are *heterogeneity* and *variability* (e.g. in software product lines). Some techniques are designed to address the specific challenges of applying regression testing to a certain type of systems (e.g. *web-based systems, real-time systems, embedded systems, databases* or *component-based systems*).

In the focus group meeting, *embedded systems* as a type of system were considered to be a relevant factor, characterising the regression testing challenges, but the other suggested system types were not - mainly on account of them not being applicable to the specific situation of the practitioners in the group. We interpret that the abstraction level is relevant and choose to keep the system types in the context taxonomy only where an explanation of what makes the context challenging from a regression testing perspective is given in any of the included studies (i.e. system types that are mentioned but not explained from a regression testing challenge perspective are removed from the taxonomy). A similar approach was used for the other system related factors of which only one, *variability*, was considered very important by the practitioners.

Context taxonomy		Factors along Study ID
Investigated context factors	System-related factors	**Size** e.g. Large-scale S1, S2, S5 – S35 **Complexity** e.g. Heterogeneneous S1, S6, S9 – S12, S19, S20, S26, S29, S30, S31, S35 or Customizeable or using product-line approach S3, S4, S6, S13 – S18, S24 – S26, S35, S38 **Type of the system** e.g. Web-based/SOA S3, S4, S6, S30, S31, S36 Real time S3, S4, S7,S8, S13 – S18, S27 Embedded S1, S24 – S27 Database applications S19, S20, S36 Component-based S6, S9,S10,S11,S12, S31
	Process-related	**Testing process** (e.g. Continuous S1, S3, S4, S26) **Test technique** e.g. Manual testing S5, S33, Combinatorial S19, S20 Scenario-based testing S6 or Model-based testing S35
	People-related factors	**Cognitive factors** e.g. lack of experience S13 – S18, S26 or that new tools need to be easy to implement and use S22 **Organizational factors** (e.g. Distributed development S6)

Effect taxonomy		References
Desired effects	Test Coverage	**Feature-coverage** S3,S4, S13 – S18 **Input (Pairwise)** S19,S20
	Efficiency and effectiveness	**Reduction of test suite** S5 – S18, S23 – S25, S27 S28, S30 – S32, S35 – S37 **Reduced testing time** S1, S3, S4, S5, S7, S8, S13 – S18, S23, S28, S29, S30, S32, S33, S36 **Improved precision** S1, S7 – S12, S24, S25 **Decreased time for fault detection** S2 – S4, S21, S22, S26, S29, S37 **Reduced need for resources** S2, S13 – S18, S29, S30 **Fault detection capability** S7, S8, S13 – S21 – S4, S24 – S26, S28, S29, S34 **Severe fault detection** S3, S4, S21 **Reduced cost of failure** S9 – S12, S19, S20, S33
	Awareness	**Transparency of testing decisions** S26

Information taxonomy		References
Utilised information entities and attributes	Requirements	**No. of changes in a requirement**, **Fault impact**, **Subjective implementation complexity**, **Perceived completeness**, **Perceived traceability** S21 **Customer assigned priority** S21, S33
	Design artefacts	**System models** S13 – S18, S27, S35 **Code dependencies** S19,S20S37
	Source code	**Code changes/ Revision history** S1, S2, S5, S7,S8, S24, S25, S38 **Source file** S2, S7, S8, S30, S37 **No. of Contributors** S32
	Intermediate code	**Class dependencies** S6 **Code changes (method or class)** S2, S6, S28
	Binary code	**Revision history** S6, S29 **Component changes** S9 – S12, S31 **Binary files** S6, S9 – S12, S23, S29
	Test cases	**Target variant** S26 **Type of test** S26 **Model coverage** S13 – S20 **Functionality coverage** S3, S4**Static priority** S26 Age S26 **Fault detection probability (estimated)** S22, S29, S33 **Execution time (estimated)** S22, S29 **Cost (estimated)** S22, S33 **Link to requirements** S21, S22 **Link to faults** S21 – S4 **Link to source code** S6 – S8
	Test Execution	**Execution time** S29, S32 **Database-states** S36 **Invocation counts** S28 **Invocation chains** S28, S31 **Runtime component coverage** S31 **Method coverage** S28 **Code coverage** S5, S23, S29, S37 Gligoric et al. (2014) S38 **Browser states**S36 **Class coverage** S6
	Test reports	**Execution time** S4, S13 – S18, S3, S28 **Verdicts** S1 – S4, S13 – S18, S26, S32, S34 **Severity** S28, S33 **Link to packages and their revisions** S1 **Link to branch** S32 **Build type** S32 **Link to failure** S13 – S18 **Test session** S13 – S18, S26 **Variant under test** S32
	Issues	**Link to fixed file / link to source code** S24, S25 **Fix-time** S32 **Link to test case** S24, S25, S37 **Failure severity** S3, S4

Table 4: A taxonomy of context, effect and information factors addressed in the included papers and considered relevant by our industry partners

6.1.2 Process related context factors

Process related context factors include factors of the development or testing process that may affect the relevance of a regression testing technique, such as currently used *processes* and *testing techniques*. Some regression testing techniques address new regression testing challenges arising with highly iterative development strategies such as continuous integration (which also was the first and main challenge identified by our collaborators and a starting point for this literature review). How testing is designed and carried out (e.g. *manual, black-box, combinatorial* or *model based*) may also be crucial for which regression testing technique is relevant and effective.

Of the testing process characteristics, *use of a specific tool* was considered irrelevant while the *use of testing technique* (all three examples) was considered very important. Thus, we removed the testing tool as a context characteristic and kept the examples of testing techniques, *manual testing*, and *combinatorial testing*. Black box testing was removed as it is covered by the information taxonomy. From the literature, we added two more examples of test techniques that affect the applicability of regression testing solutions, *Scenario based testing* and *Model based testing*. *The frequency of full regression test* (i.e. how often is the complete regression test suite run) was considered important, and we rephrased it to *continuous testing* in the final taxonomy. Also, *size* and *long execution times of test suites* were considered important but since it is covered by the desired effects, we removed it from the context taxonomy.

6.1.3 People related context factors

People related context factors refer to factors that may cause, or are caused by, distances between collaborating parties and stakeholders in the development and testing of the software system. The terminology used stems from Bjarnason et al. (2016). *Cognitive* context factors include the degree of knowledge and awareness, while *organisational* factors include factors that may cause, or are caused by, differences in goals and priorities between units.

People related issues were important to all participants in the focus group, but the message about which factors were most important was mixed. *Ease of use* got the highest score. A new node *Cultural distances* was proposed as well, however, we have not found any such considerations in the selected set of papers, and thus did not include it in the taxonomy. This branch of the taxonomy showed to have overlaps with the effect taxonomy (e.g. *Lack of awareness* and *Need for quick feedback*), and we decided to remove such nodes from the context taxonomy and add them to the effect taxonomy instead.

6.1.4 General reflection

A reflection about the overall context taxonomy is that it is not obvious which characteristics are relevant to report from a generalisation perspective. Even in industrial studies, the problem descriptions are in many cases superficial and many context factors are mentioned without any further explanation as to why they are relevant from a regression testing perspective. Some factors mentioned are crucial only to the technology being evaluated, and not necessarily an obstacle preventing

the use of other technologies. One such example is the type of programming language - it was initially added to the taxonomy, as it is a commonly reported aspect of the cases used for empirical evaluation. However, it was finally removed as it was considered a part of the constraints of a solution, rather than characterising trait of the addressed problem context.

6.2 Desired effects

The desired effect of a technique is basically about the reported types of improvement(s) achieved by applying the technique, such as 'improving efficiency' or 'decreasing execution time'. To be recognised as a desired effect, in our setting, the effect of the technique has to be evaluated in at least one (industrial/large scale) case study, rather than just being mentioned as a target of the technique without any evidence. Accordingly, the effect has to be demonstrated as a *measurement* showing the effect of the proposed technique on regression testing.

Table 4 shows a taxonomy of effect (-target) factors. The proposed effect taxonomy provides a categorisation of the various effects (improvements) identified in the research while simultaneously, it meets the level of information (or detail) required (or considered relevant) by our industrial partners. The improvements (effects) of techniques are categorised into three main categories: *Improved test coverage*, *Improved efficiency and effectiveness* and *increased awareness*.

6.2.1 Improved test coverage

Improved coverage refers to the effects aiming at improving (increasing) the coverage of any type of *entity* by the selected test suite. The type of entity, which is under consideration, depends on the context and the proposed solution. We identified two main desired effects for coverage in the included papers, namely *increased feature coverage* and *improved combinatorial-input coverage (Pairwise)*.

6.2.2 Improved efficiency and effectiveness

Efficiency and effectiveness cover cost reduction factors such as *reduced number of test cases* and *reduced execution time* with a consideration for how comprehensively faults are detected. In principle, efficiency does not look into how well a testing technique reveals or finds errors and faults. Improving only the efficiency of a technique will lead to a testing activity that requires less amount of time or computational resources, but it may not be effective (i.e. comprehensively detect faults). Efficiency and Effectiveness are often distinguished in the research literature, while in practice they are equally important objectives and are most often targeted at the same time. Thus, we treat them as one class in our taxonomy. *Reduction of test suite* often leads to a set of test cases requiring less resource (memory) and less amount of time to be generated, executed, and analysed. Note that test suite reduction in the research literature is often referred to as a technique as such (Yoo and Harman (2012)). It is then used interchangeably with test suite maintenance referring to the permanent removal of test cases in contrast to the temporary removal or ordering of test cases in "test case selection" or "test case prioritisation. However, "reduction of the number of test cases" is at the same

time the most common measure of the effectiveness of a regression test selection technique in industrial evaluations. It is used in evaluation of regression testing techniques when comparing with the current state of practice (both in the maintenance case and the selection case) in a particular context. Thus, we add it as a desired effect in our taxonomy.

Reduction of testing time considers any time/resource-related aspect, also referred to as 'cost' in some studies. *Improved precision* refers to the ability of a selection technique in avoiding non-fault revealing test cases in the selection. High precision results in a reduction of test suite while it also indicates a large proportion of fault detecting test cases. Hence, precision is considered a measure of both efficiency and effectiveness. *Decreased time for fault detection* i.e. the aim is to reduce the time it takes to identify faults, which is relevant when reflecting on the outcomes of a prioritisation technique for regression testing. *Reduced need for resources* i.e. reduces the consumption of a resource e.g. memory consumption. *Improved fault detection capability* also referred to as 'recall' or 'inclusiveness', measures how many faults are detected regardless of their severity. *Improved detection of severe faults* refers to the extent to which a technique can identify severe faults in the system. *Reduced cost of failures*, here the focus is on the consequence (measured in cost factors) of false negatives in the selection.

6.2.3 Increased awareness

Increased awareness refers to improvements related to the testing process (activities) per se and the people involved in the process. *Improved transparency of testing decisions* has been considered in the existing research and identified as a relevant target by our industrial partners. It aims at transparently integrating regression testing techniques into daily/normal development activities such that the stakeholders understand the working of the technique and trust the recommendations regarding the test-cases they produce.

6.2.4 General reflection

A general reflection regarding the effect-taxonomy is that "what is measured in research is not always what matters in practice". The taxonomy was initially based solely on the different variants of measurements used in the studies and rather fine-grained in some aspects. Different levels of code coverage are for example a popular measurement in literature but were not considered relevant by the practitioners in the focus group. All proposed coverage metrics except feature coverage were considered irrelevant by the participants. Our interpretation is *not* that code coverage is considered useless as a test design technique, but that improving code coverage is not a driver for applying regression testing (not for the practitioners and not for any of the stakeholders in the industrial evaluations). Although code coverage is not presented as a desired effect in our taxonomy, it still appears as a characteristic of a technique (information attribute) since there are some examples of techniques in our included papers that utilise measures of code coverage to propose a regression testing scope.

Regarding the variants of measurements under effectiveness and efficiency, the granularity level was considered too high and many of the nuances in the measurements were hard to interpret from a practical point of view. Only three of the low-level categories were considered relevant for at least one of the participants, 'detection of severe faults' was important for all three while 'precision' and 'test suite reduction' were important to one of the participants.

7 RQ2 – Regression testing solution description in terms of utilised information sources

To answer RQ2, i.e., *"how to describe a regression testing solution?"*, we considered the following choices for developing a taxonomy: 1) based on the underlying assumptions (e.g., history-based and change-based), 2) based on the techniques (e.g., firewall, fixed-cache, and model-based), or 3) based on the required information (e.g., test case execution information, code complexity, and version history).

We decided in favour of the third option, in particular, because it allows for reasoning about what information is required to implement a regression testing technique. From a practitioner's point of view the concerns regarding: a) whether a technique would work in his/her context, and b) whether it can help achieve the desired effect, are already covered with the context and the effect taxonomy. Thus, while the effect and context taxonomies enable narrowing down the choice of techniques, the aim of the information taxonomy is to support practitioners in reasoning about the technical feasibility and the estimated cost of implementing a technique in their respective unique context.

Hence, the purpose of the information taxonomy can be summarised as *to provide support in comparing regression testing solutions by pinpointing relevant differences and commonalities among regression testing techniques (i.e., the characteristics affecting the applicability of a technique)*. We consider this classification particularly useful for practitioners as it helps one identify relevant techniques in their context. For example, if a certain test organisation does not have access to source code, they can focus on techniques that do not require it.

Similarly, knowing what information is required to implement a technique, the interested reader can: 1) identify if this information is currently available in their organisation 2) investigate how to derive it from other available information sources, or 3) analyse the feasibility of collecting it. Hence, a practitioner can make an informed decision about the applicability of a technique in their context by considering the possibility and the cost of acquiring the required information.

The information taxonomy (as shown in Table 4) uses the structure <entity, information> to identify what information is required to use a certain technique. Thus, we coded the entities and the utilised information about their various attributes/facets used by each of the techniques. Some examples of entities, in this case, are design artefacts, requirements or source code. The respective information about the various attributes/facets of these three example entities may include dependencies between various components, the importance of a requirement to a customer or code metrics.

From the papers included in this review, the following nine entities (and different information regarding them) were used by the regression testing techniques:

1) Requirements, 2) Design artefacts, 3) Source code 4) Intermediate code, 5) Binary code, 6) Closed defect reports, 7) Test cases, 8) Test executions, and 9) Test reports.

7.1 Requirements

Very few regression testing techniques included in this study have used information related to requirements (such as the importance of required functionality for the customer). Only two papers explicitly make use of information regarding the requirements Krishnamoorthi and Mary (2009); Li and Boehm (2013). Such information can be coupled with requirement coverage (i.e., the number of requirements exercised by a test case) to optimise regression testing with respect to the actual operational use of the SUT Krishnamoorthi and Mary (2009); Tahvili et al. (2016).

The information about several attributes of requirements such as their priority and the complexity of their implementation are stored in requirement management systems. However, the information regarding requirement coverage may as well be stored in the test management system with respect to their corresponding test cases.

One reason for the lack of techniques utilizing requirements as input for regression testing could be that often the traceability information from requirements to source code to test cases is not maintained Uusitalo et al. (2008). Furthermore, it is significantly more difficult to recreate these traceability links than, e.g., linking source code to test cases.

The following five techniques are based on the requirements and feature coverage by test-cases: RTrace Krishnamoorthi and Mary (2009), MPP Marijan et al. (2013); Marijan (2015), TEMSA Wang et al. (2016, 2013a, 2014), and FTOPSIS Tahvili et al. (2016).

FTOPSIS Tahvili et al. (2016) uses multi-criteria decision-making as well as fuzzy logic, where both objective and subjective (expert judgement) data about requirements can be used to prioritise test cases in a regression suite. Krishnamoorthi and Mary's approach RTrace Krishnamoorthi and Mary (2009) expects an explicit link between test cases and requirements for their proposal. However, they do not describe how the case companies were documenting this information. They also do not suggest how such links can be generated. TEMSA Wang et al. (2016, 2013a, 2014) develop and use feature models and component family models, to ensure feature coverage in regression test selection of a software product line system. MPP Marijan et al. (2013); Marijan (2015) uses the coverage of functionality of the system by individual test cases as a criterion to prioritise test cases.

7.2 Design artefacts

Wang et al. Wang et al. (2013b, 2017, 2015, 2016, 2013a, 2014) use feature models and component feature models. These models along with an annotated classification of test cases are used for test case selection. Similar to the approach of Wang et al., Lochau et al. Lochau et al. (2014) also exploit models (in their case, delta-oriented component test models and integration test models). They also used

existing documentation from the industrial partners and interviews with practitioners to develop and validate these models.

For an automotive embedded system, Vöst and Wagner Vöst and Wagner (2016) propose the use of system architecture (system specifications) for test case selection. System specifications and test case traces were used to create a mapping between components and test cases using them.

7.3 Source code

In IEEE standard for software test documentation, regression testing is defined as: "selective retesting of a system or component to verify that modifications have not caused unintended effects and that the system or components still complies with its specified requirements" IEEE (1998). Therefore, several regression testing techniques attempt to leverage available information to localise changes in a software system that can then inform the decision of which test cases to run and in which order. Some such techniques, work with source code and its version history to identify the change. Using source code has two advantages, first, several static and dynamic approaches exist to link test-cases to source code (e.g. once we have localised the change, identifying change traversing test-cases to select or prioritise is possible). The second advantage is that most commercial organisations use a configuration management system. Thus, the techniques that utilise revision history are relevant for industrial settings.

For example, FaultTracer Gligoric et al. (2014), CCB Buchgeher et al. (2013), Fix-cache Engström et al. (2010b); Wikstrand et al. (2009), EFW White et al. (2008); White and Robinson (2004), and Difference-Engine Ekelund and Engström (2015) utilise revision history of source code for regression testing. Similarly, THEO Herzig et al. (2015) uses the number of contributors to the code base as input.

Several techniques require access to actual source code to work. Carlson et al. Carlson et al. (2011) propose the use of a clustering approach that computes and uses code metrics as one of the criteria for test case prioritisation. REPiR Saha et al. (2015) uses information retrieval techniques to prioritise test cases that cover the changes in source code. It relies on the likelihood that similar keywords are used in source code literal and comments as in the test cases.

GWT-SRT Hirzel and Klaeren (2016) instruments source code to be able to generate traces that are used to connect test-cases and source code. This information along with control flow graphs (to isolate code changes) are used for selective regression testing in the context of web applications.

7.4 Intermediate and binary code

In cases when the source code is either not available, or it is not feasible to use it, there are some techniques that work on intermediate and binary code instead of source code to localise change between two versions of a software system. REPiR Saha et al. (2015) and CMAG Huang et al. (2009) use intermediate code to identify changes. While I-BACCI Zheng et al. (2006b, 2007); Zheng (2005); Zheng et al. (2006a), Echelon Srivastava and Thiagarajan (2002), OnSpot Janjua (2015) and

Pasala and Bhowmick's proposed technique Pasala and Bhowmick (2005) work with binaries to localise change.

7.5 Issues

Some techniques utilise information typically residing in issue management systems Engström et al. (2010b); Wikstrand et al. (2009); Herzig et al. (2015). Provided that an issue originates in a fault revealed by a test case, the attributes of that issue may be used to recreate a link between the said test case and the source files that were updated to fix the issue Engström et al. (2010b); Wikstrand et al. (2009). Herzig et al. Herzig et al. (2015) utilise information about the closed defect reports (e.g. the time it took to fix the issue) in a cost model weighing the cost of running a test case against skipping it. The technique described by Marijan et al. Marijan et al. (2013); Marijan (2015) uses the severity information from defect reports, prioritising test cases that reveal faults of high severity.

Among the included papers, no proposal presents an approach to recreate links between defect reports and mapping to related test cases. Therefore, if practitioners want to use one of the techniques that leverage fault coverage by test cases or other fault-related information (like the severity of faults etc.) they must document and maintain the links between these artefacts.

7.6 Test cases

Test cases refer to the specification of the tests and are static information entities (i.e., the information is documented at the design of the test and stored and maintained in a repository typically a test management system). 50% of the evaluated techniques rely on such information. What attributes of the test specifications being used vary between the different techniques, but it could be divided into traceability information and properties of the test case per se.

Traceability information is typically used for coverage optimisation selection Wang et al. (2016); Rogstad and Briand (2016); Marijan (2015); Krishnamoorthi and Mary (2009); Tahvili et al. (2016); Carlson et al. (2011); Buchgeher et al. (2013); Skoglund and Runeson (2005); White and Robinson (2004); Zheng et al. (2007); e.g. links to other information entities such as source code and requirements, or explicit coverage targets such as model coverage Wang et al. (2016); Rogstad and Briand (2016) or functionality coverage Marijan (2015).

Three techniques utilise the property attributes of the test cases (e.g age and estimated cost) solely for test prioritisation Engström et al. (2011); Li and Boehm (2013); Srivastava and Thiagarajan (2002) and hence they are not dependent on static traceability links. Two are risk-based Engström et al. (2011); Li and Boehm (2013) while one recreates traceability links dynamically Srivastava and Thiagarajan (2002), see Section 7.7.

7.7 Test executions

Test executions refer to an implicit information entity, meaning that its information attributes may be dynamically collected but are not stored and maintained

for other purposes. Just as for the 'Test cases' described above the attributes of the 'Test executions' could be divided into coverage information (e.g. 'invocation chains' Huang et al. (2009); Pasala and Bhowmick (2005), 'covered system states' Devaki et al. (2013), 'runtime component coverage' Pasala and Bhowmick (2005) and 'code coverage' Huang et al. (2009); Carlson et al. (2011); Buchgeher et al. (2013); Srivastava and Thiagarajan (2002); Janjua (2015); Gligoric et al. (2014); Skoglund and Runeson (2005)) and intrinsic properties of the executions (e.g. 'execution time' Herzig et al. (2015); Srivastava and Thiagarajan (2002), or 'invocation counts' Huang et al. (2009))

Dynamically collected coverage information is used for similarity-based and coverage-based optimisation of regression tests Buchgeher et al. (2013); Devaki et al. (2013); Carlson et al. (2011) as well as change-based prediction of regression faults Srivastava and Thiagarajan (2002); Janjua (2015); Gligoric et al. (2014) while dynamically collected property attributes of the test executions are typically used for history-based cost optimisation of regression tests faults Huang et al. (2009); Herzig et al. (2015).

7.8 Test reports

Test reports refer to the static records of the test executions, this information could either be automatically captured or entered manually by the testers. Such attributes are used for history-based optimisation of regression tests and most commonly used for regression test optimisation are verdicts Herzig et al. (2015); Ekelund and Engström (2015); Marijan (2015); Engström et al. (2011); Anderson et al. (2014); Wang et al. (2016), time stamps Marijan (2015); Wang et al. (2016); Huang et al. (2009) and links to the tested system configuration Ekelund and Engström (2015); Herzig et al. (2015). Several information attributes of the test reports are similar to the test execution attribute or the test case attribute, but differ in how it is derived and maintained. As an example, test execution time could be an attribute of all three test information entities but as an attribute of a test case it is an estimation; as an attribute of a test execution, it is measured at runtime; and as an attribute of the test report, it is further recorded and maintained.

8 RQ3 – Mapping of current research

26 different techniques (reported in 38 papers) were classified under three taxonomies: context, effect and information (see Table 4). This mapping (see Table 5) helps to select techniques that address relevant context factors and deliver the target benefits for a given scope (regression test selection, prioritization or minimization). The information taxonomy helps to reason about whether the information is available or can be reasonably acquired in the unique context of a particular company. We demonstrate the use of this taxonomy in Section 9 in the form of technological rules Storey et al. (2017) derived from this mapping (see Section 9.

Technique	Study ID	Scope			Addressed context factors			Desired effects			Utilised information (entities)								
		Selection	Prioritization	Minimization	System-related	Process-related	People-related	Test coverage	Efficiency and Effectiveness	Awareness	Requirements	Design artefacts	Source code	Intermediate code	Binary code	Test cases	Test execution	Test reports	Issues
TEMSA	S13 – S18	✓	✓	✓	✓		✓	✓	✓			✓				✓		✓	
History based prioritization (HPro)	S26	✓	✓		✓	✓	✓		✓	✓						✓		✓	
classification tree testing (DART)	S19, S20	✓			✓	✓		✓	✓			✓				✓			
I-BACCI	S9 – S12	✓			✓				✓						✓				
Value_based	S33		✓		✓	✓			✓		✓					✓		✓	
multi-perspective prioritisation (MPP)	S3, S4		✓		✓	✓		✓	✓							✓		✓	✓
RTrace	S21		✓		✓				✓		✓					✓			
Echelon	S29		✓		✓				✓						✓	✓	✓		
Information retrieval (REPiR)	S2		✓		✓				✓				✓	✓					
EFW	S7, S8	✓		✓	✓				✓				✓			✓			
Fix-cache	S24, S25	✓			✓				✓				✓						✓
Most Frequent Failures	S34	✓			✓				✓									✓	
Continuous Multi-scale Additional Greedy prioritisation (CMAG)	S28	✓	✓		✓				✓					✓			✓	✓	
GWT-SRT	S30	✓			✓				✓				✓						
Clustering (based on coverage, fault history and code metrics)	S37	✓	✓						✓			✓	✓				✓		✓
FTOPSIS	S22		✓		✓		✓		✓							✓			
Difference engine	S1	✓			✓	✓			✓				✓					✓	
Change and coverage based (CCB)	S5	✓			✓	✓			✓				✓				✓		
Similarity based minimisation	S36			✓	✓				✓								✓		
THEO	S32	✓			✓				✓				✓				✓	✓	✓
DynamicOverlay / OnSpot	S23	✓			✓				✓						✓		✓		
Class firewall	S6	✓			✓	✓	✓		✓					✓	✓	✓	✓		
model-based architectural regression testing	S35	✓			✓	✓			✓			✓							
component interaction graph test case selection	S31	✓			✓				✓						✓		✓		
keyword-based-traces	S27	✓			✓				✓			✓							
FaultTracer	S38	✓			✓								✓				✓		

Table 5: Mapping of techniques to the taxonomy

8.1 Addressed context factors

In terms of system-related factors, when describing the context where the proposed techniques are applicable, 22 of the included techniques consider the scalability of techniques for large-scale software systems. Another 13 techniques take the complexity and the type of system under test into account. 9 techniques consider process related context factors. While only 4 techniques consider people-related factors.

8.2 Desired effects

Most techniques (25 out of the total 26) target improved effectiveness and efficiency in terms of finding known faults. Three techniques focus on improving coverage (which is considered a proxy for achieving confidence in the testing or confirming the quality of a system). Only one technique targets increased awareness in the decision-making process in the scope of regression testing.

8.3 Information sources

Except regression testing techniques that rely solely on the history of test-cases, most techniques use information beyond the test cases to select, prioritise or minimise the regression testing suite. Such approaches rely on documented/generated links between test cases and other artefacts. The 26 techniques included in this study utilise information contained in nine different types of software engineering artefacts.

Only two and five techniques use information related to requirements and design artefacts, respectively. Attributes of source code are used in nine techniques while seven techniques only rely on intermediate or binary code among other information sources. Ten techniques use information about test cases. Moreover, ten and eight techniques use information from test executions and test reports. Only four techniques make use of the issue reports.

9 Suggestions for practitioners

Practitioners may utilise the results of this study in different ways. The mappings of techniques to each category may guide the practitioner looking for relevant research. The taxonomies could also be used to compare a set of possible solutions found in the research, or those designed by engineers at a company.

In Table 4, three taxonomies for describing regression testing problems and techniques were introduced. In Table 5, we present a mapping of the included papers to the three taxonomies. A practitioner can use the taxonomies and the mapping to identify recommendations that are likely to help them design a solution in their unique context. The mapping of techniques to the three taxonomies may be read as technological rules Storey et al. (2017) i.e., "To achieve <**effect**> in <**context**> apply <**technique**>", which in turn should be interpreted as recommendations (or advise) extracted from research.

Such rules could be formulated in detail for a single technique (i.e. one row in the mapping, as in example TR1) or with fewer details for a set of techniques (by including only the common denominators for that set, TR2) or in more general terms by selecting nodes at a higher level in the taxonomy (TR3). Three illustrative examples (TR1-3) are given:

TR 1: To reduce the regression test suite and testing time when regression testing large scale software systems utilise the following information attributes: #contributors of a piece of code, measured execution time, verdict, build type, variant under test and link to tested branch from test reports and fix time in issue reports. This example was extracted from an evaluation of a tool called THEO (Herzig et al. (2015)).

TR 2: To increase feature coverage, reduce testing time and improve fault detection capability when regression testing customisable, real-time systems, utilise information about verdicts and execution time in test reports. (This rule is based on the intersection of the classification of two different techniques, Multi-perspective prioritisation Marijan (2015) and TEMSA Wang et al. (2015), which have been evaluated in several studies Marijan et al. (2013); Marijan (2015); Wang et al. (2013b, 2017, 2015, 2016, 2013a, 2014).)

TR 3: To improve efficiency and effectiveness when regression testing large scale complex systems, utilise information attributes of the test reports. (This rule is a generalisation of TR2 and is supported by the same studies and another two Ekelund and Engström (2015); Engström et al. (2011).)

From the research communication perspective, we argue that formulating such rules (albeit by compromising some details) will help to communicate our research in particular to industry practitioners.

By selecting the most important aspects of the two problem-defining taxonomies (i.e. desired effects and addressed context factors), for the organisation, one or more research-based recommendations (in terms of technological rules) may be extracted from the mapping in this study together with pointers to the matching literature. These recommendations could then be used as input to the design of the specific solution for the organisation.

10 Recommendations for researchers

As for testing, in general, the value of a regression testing technique could be measured either in terms of its ability to increase confidence in testing or in terms of its ability to improve fault detection with limited time and resources. Those high-level goals are shared by researchers and practitioners but with some variations when it comes to details and priorities Minhas et al. (2017). The recommendations given here are based on our comparison of what we found in the literature and the feedback from our industry partners.

Evaluate coverage of regression testing techniques at the feature level Confidence is achieved by measuring any type of coverage. However, of the facets of our effect taxonomy, coverage was the least important for the practitioners and at the detailed level of our taxonomy only feature coverage was considered relevant. This is also reflected in the literature Marijan (2015); Saha et al. (2015); Wang et al.

(2016). Few techniques were evaluated with respect to coverage and of the few the majority focused on feature coverage.

Focus evaluation on the detection of severe faults and reduction of cost From the practitioners' perspective, the ability of a technique to detect severe faults and to reduce cost in terms of man- and machine-time was considered more important. While confidence, and coverage, always being priorities in the design of new tests, the pursuit of trust in the regression test suite is often the root cause of increasing costs and decreasing the effectiveness of regression testing - as more and more test cases are added just in case Engström and Runeson (2010). Hence, the main driver for most industrial studies on regression testing is cost reduction and precision of regression testing. 70% of the techniques in our study were evaluated with respect to cost reduction. Furthermore, effectiveness should be considered in terms of the number of severe faults, rather than in the number of faults in general as there is also a cost-benefit trade-off in the issue backlog. Only 40% of the techniques in our study were evaluated with respect to fault detection and of those only two considered severity Krishnamoorthi and Mary (2009); Marijan (2015).

Report addressed context factors in industrial evaluations To support generalisation of results between industrial contexts, relevant contextual factors need to be identified and clearly reported. Here relevant context factors denote context factors that are either causing the problems to be solved or affecting the applicability or effect of the solution. Such relevance may be observed or assumed by the stakeholder or the researcher.

Despite being a selection of industrial evaluations, reporting the context factors seems to be of low priority. In 10% of the evaluations, we did not find any context descriptions at all. For the remaining 90% at least system factors such as size, complexity and type of system under test are reported. Only 30% described the current process, which will affect (or be affected by) the adoption of the technique and only 15% reported people-related factors.

Rather than providing a general and extensive description of the case context, as proposed in previous research Petersen and Wohlin (2009) we recommend a careful selection of context factors to report and discuss. Such selection could be guided by the context-taxonomy provided in Table 4.

Study if and how the proposed context factors are relevant In most cases, the relevance of the context factors described in the included papers is assumed rather than observed. Furthermore, they are described on a rather high abstraction level. Thus, there is room for more research on the relevance of various context factors for regression testing.

Study people-related factors As a research area struggling to gain traction in industry Kapfhammer (2011); Juzgado et al. (2004); Bertolino (2007) we believe that there is a need to investigate people-related factors. The need for transparency and understandability that leads to trust in the results of regression testing techniques was highlighted by the industry partners. Only one study in our included set has investigated these factors Engström et al. (2011). Future research in this direction that takes into account the needs and concerns of potential users may increase the likelihood of successful adoption of regression testing techniques in the industry.

11 Conclusion

In this paper, we report an extensive search for and an in-depth analysis of applicability and relevance of regression testing techniques reported in the literature. We focused on techniques that have been applied to large-scale software systems in industrial contexts. Based on the literature review and in collaboration with our industrial partners, we propose three taxonomies that enable practitioners and researchers to assess and compare the relevance and applicability of regression testing techniques for a given industrial context.

The taxonomies extend three out of the four facets of the SERP-test taxonomy Engström et al. (2017): i.e. addressed context factors, desired improvements and characteristics of the techniques from an applicability point of view (required information and underlying assumptions). It allows for characterisation of regression techniques and helps to determine which of these techniques could be suitable in a given context and to indicate what benefits could be expected from its application. The identification of information needs for these techniques further assists a reader to reason about the implications regarding the cost of adoption Ali (2016). In this report, we apply the taxonomies on the 38 papers that are included in the review. However, initially, we identified more than 1000 papers on regression testing and there are many techniques, not included in the review, that may still be relevant and applicable in some industrial contexts. The aim of the taxonomies is to support assessment also of them and of new proposals or adaptations of techniques.

In our interaction with the practitioners, we identified some discrepancies in researchers focus and the perceived needs in the industry. One such example is the commonly used measure of effectiveness in terms of various levels of code coverage. Some types of information (such as coverage information) that are extensively studied in the literature were found to be only partially relevant to the practitioners (e.g., only at the feature level) for evaluating the benefits of a regression testing technique. At the same time, some extremely relevant information in choosing technique (e.g. the context information) is completely neglected in some existing studies.

For academia, this study can help to use evaluation criteria and contexts that are representative of industrial needs. This work also supports reporting the research results to facilitate readers making an informed decision with a consideration for relevance to their context, potential benefits and the likely investment required to use the technique.

Acknowledgements The authors would like to thank the industry partners from Axis Communications AB, Sweden and Sony Mobile Communications, Sweden for their time, input and feedback throughout this study. We also thank Serge Demeyer for reviewing the manuscript and providing improvement proposals. This work has been supported by ELLIIT, a Strategic Area within IT and Mobile Communications, funded by the Swedish Government. Support has also come from EASE, the Industrial Excellence Centre for Embedded Applications Software Engineering. The work of Mohammad Reza Mousavi and Masoumeh Taromirad has been partially supported by (Vetenskapsrådet) award number: 621-2014-5057 (Effective Model-Based Testing of Concurrent Systems) and by the Swedish Knowledge Foundation (Stiftelsen for Kunskaps- och Kompetensutveckling) in the context of the AUTO-CAAS HÖG project (number: 20140312).

References

Ali NB (2016) Is effectiveness sufficient to choose an intervention?: Considering resource use in empirical software engineering. In: Proceedings of the 10th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, ESEM, pp 54:1–54:6

Ali NB, Petersen K, Mäntylä M (2012) Testing highly complex system of systems: an industrial case study. In: Proceedings of ACM-IEEE International Symposium on Empirical Software Engineering and Measurement, ESEM, pp 211–220

Ali NB, Petersen K, Wohlin C (2014) A systematic literature review on the industrial use of software process simulation. Journal of Systems and Software 97:65–85

Anderson J, Salem S, Do H (2014) Improving the effectiveness of test suite through mining historical data. In: Proceedings of the 11th Working Conference on Mining Software Repositories, ACM Press, pp 142–151

Bertolino A (2007) Software testing research: Achievements, challenges, dreams. In: Proceedings of the Workshop on the Future of Software Engineering, FOSE, pp 85–103

Bjarnason E, Smolander K, Engström E, Runeson P (2016) A theory of distances in software engineering. Information & Software Technology 70:204–219

Buchgeher G, Ernstbrunner C, Ramler R, Lusser M (2013) Towards tool-support for test case selection in manual regression testing. In: Workshops proceedings of the 6th IEEE International Conference on Software Testing, Verification and Validation, ICST, pp 74–79

Carlson R, Do H, Denton A (2011) A clustering approach to improving test case prioritization: An industrial case study. In: Proceedings of the 27th IEEE International Conference on Software Maintenance, ICSM, IEEE, pp 382–391

Carver JC, Dieste O, Kraft NA, Lo D, Zimmermann T (2016) How Practitioners Perceive the Relevance of ESEM Research. In: Proceedings of the 10th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement - ESEM '16, ACM Press, Ciudad Real, Spain, pp 1–10

Catal C (2012) On the application of genetic algorithms for test case prioritization: a systematic literature review. In: Proceedings of the 2nd international workshop on Evidential assessment of software technologies, ACM, pp 9–14

Catal C, Mishra D (2013) Test case prioritization: a systematic mapping study. Software Quality Journal 21(3):445–478

Devaki P, Thummalapenta S, Singhania N, Sinha S (2013) Efficient and flexible GUI test execution via test merging. In: Proceedings of the International Symposium on Software Testing and Analysis, ISSTA, pp 34–44

Edison H, Ali NB, Torkar R (2013) Towards innovation measurement in the software industry. Journal of Systems and Software 86(5):1390–1407

Ekelund ED, Engström E (2015) Efficient regression testing based on test history: An industrial evaluation. In: Proceedings of IEEE International Conference on Software Maintenance and Evolution, ICSME, pp 449–457

Engström E, Runeson P (2010) A qualitative survey of regression testing practices. In: Proceedings of the 11th International Conference on Product-Focused Software Process Improvement PROFES, pp 3–16

Engström E, Runeson P, Skoglund M (2010a) A systematic review on regression test selection techniques. Information & Software Technology 52(1):14–30

Engström E, Runeson P, Wikstrand G (2010b) An empirical evaluation of regression testing based on fix-cache recommendations. In: Proceedings of the 3rd International Conference on Software Testing, Verification and Validation, ICST, pp 75–78

Engström E, Runeson P, Ljung A (2011) Improving regression testing transparency and efficiency with history-based prioritization - an industrial case study. In: Proceedings of the 4th IEEE International Conference on Software Testing, Verification and Validation, ICST, pp 367–376

Engström E, Feldt R, Torkar R (2012) Indirect effects in evidential assessment: a case study on regression test technology adoption. In: 2nd international workshop on Evidential assessment of software technologies, pp 15–20

Engström E, Petersen K, Ali NB, Bjarnason E (2017) SERP-test: a taxonomy for supporting industry-academia communication. Software Quality Journal 25(4):1269–1305

Felderer M, Fourneret E (2015) A systematic classification of security regression testing approaches. International Journal on Software Tools for Technology Transfer 17(3):305–319

Felizardo KR, Mendes E, Kalinowski M, de Souza ÉF, Vijaykumar NL (2016) Using forward snowballing to update systematic reviews in software engineering. In: Proceedings of the 10th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, ESEM, pp 53:1–53:6

Franch X, Fernandez DM, Oriol M, Vogelsang A, Heldal R, Knauss E, Travassos GH, Carver JC, Dieste O, Zimmermann T (2017) How do Practitioners Perceive the Relevance of Requirements Engineering Research? An Ongoing Study. In: 2017 IEEE 25th International Requirements Engineering Conference (RE), IEEE, Lisbon, Portugal, pp 382–387

Garousi V, Mäntylä MV (2016) A systematic literature review of literature reviews in software testing. Information & Software Technology 80:195–216

Gligoric M, Negara S, Legunsen O, Marinov D (2014) An empirical evaluation and comparison of manual and automated test selection. In: Proceedings of ACM/IEEE International Conference on Automated Software Engineering, ASE, pp 361–372

Harrold MJ, Orso A (2008) Retesting software during development and maintenance. In: Proceedings of Frontiers of Software Maintenance FoSM, IEEE, pp 99–108

Herzig K, Greiler M, Czerwonka J, Murphy B (2015) The art of testing less without sacrificing quality. In: Proceedings of the 37th International Conference on Software Engineering, IEEE Press, ICSE '15, pp 483–493

Hirzel M, Klaeren H (2016) Graph-walk-based selective regression testing of web applications created with google web toolkit. In: Gemeinsamer Tagungsband der Workshops der Tagung Software Engineering (SE), pp 55–69

Huang S, Chen Y, Zhu J, Li ZJ, Tan HF (2009) An optimized change-driven regression testing selection strategy for binary java applications. In: Proceedings of ACM symposium on Applied Computing, ACM, pp 558–565

IEEE (1998) IEEE standard for software test documentation IEEE Std. 829-1983

Ivarsson M, Gorschek T (2011) A method for evaluating rigor and industrial relevance of technology evaluations. Empirical Software Engineering 16(3):365–395

Janjua MU (2015) Onspot system: test impact visibility during code edits in real software. In: Proceedings of the 10th Joint Meeting on Foundations of Software Engineering, ESEC/FSE, pp 994–997

Jr STR, Riddle WE (1985) Software technology maturation. In: Proceedings of the 8th International Conference on Software Engineering, pp 189–200

Juzgado NJ, Moreno AM, Vegas S (2004) Reviewing 25 years of testing technique experiments. Empirical Software Engineering 9(1-2):7–44

Kapfhammer GM (2011) Empirically evaluating regression testing techniques: Challenges, solutions, and a potential way forward. In: Proceedings of the 4th IEEE International Conference on Software Testing, Verification and Validation, ICST, pp 99–102

Kazmi R, Jawawi DNA, Mohamad R, Ghani I (2017) Effective regression test case selection: A systematic literature review. ACM Comput Surv 50(2):29:1–29:32

Kitchenham BA, Budgen D, Brereton P (2015) Evidence-Based Software Engineering and Systematic Reviews. Chapman & Hall/CRC

Krishnamoorthi R, Mary SASA (2009) Factor oriented requirement coverage based system test case prioritization of new and regression test cases. Information & Software Technology 51(4):799–808

Li Q, Boehm B (2013) Improving scenario testing process by adding value-based prioritization: an industrial case study. In: Proceedings of the International Conference on Software and System Process, ACM, pp 78–87

Lo D, Nagappan N, Zimmermann T (2015) How practitioners perceive the relevance of software engineering research. In: Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering - ESEC/FSE 2015, ACM Press, Bergamo, Italy, pp 415–425

Lochau M, Lity S, Lachmann R, Schaefer I, Goltz U (2014) Delta-oriented model-based integration testing of large-scale systems. Journal of Systems and Software 91:63–84

Marijan D (2015) Multi-perspective regression test prioritization for time-constrained environments. In: Proceedings of IEEE International Conference on Software Quality, Reliability and Security, QRS, pp 157–162

Marijan D, Gotlieb A, Sen S (2013) Test case prioritization for continuous regression testing: An industrial case study. In: Proceedings of IEEE International Conference on Software Maintenance, pp 540–543

Minhas NM, Petersen K, Ali NB, Wnuk K (2017) Regression testing goals-view of practitioners and researchers. In: 24th Asia-Pacific Software Engineering Conference Workshops (APSECW), 2017, IEEE, pp 25–31

Munir H, Moayyed M, Petersen K (2014) Considering rigor and relevance when evaluating test driven development: A systematic review. Information and Software Technology 56(4):375–394

Narciso EN, Delamaro ME, de Lourdes dos Santos Nunes F (2014) Test case selection: A systematic literature review. Int J Software Eng Knowl Eng 24(4):653–676

Osterweil LJ, Ghezzi C, Kramer J, Wolf AL (2008) Determining the impact of software engineering research on practice. IEEE Computer 41(3):39–49

Pasala A, Bhowmick A (2005) An approach for test suite selection to validate applications on deployment of COTS upgrades. In: Proceedings of the 12th Asia-Pacific Software Engineering Conference, APSEC, pp 401–407

Petersen K, Engström E (2014) Finding relevant research solutions for pracitical problems - the SERP taxonomy architecture. In: International Workshop on Long-term Industrial Collaboration on Software Engineering (WISE 2014)

Petersen K, Wohlin C (2009) Context in industrial software engineering research. In: Proceedings of the 3rd International Symposium on Empirical Software Engineering and Measurement, IEEE Computer Society, Washington, DC, USA, ESEM '09, pp 401–404

Qiu D, Li B, Ji S, Leung HKN (2014) Regression testing of web service: A systematic mapping study. ACM Comput Surv 47(2):21:1–21:46

Rainer A, Beecham S (2008) A follow-up empirical evaluation of evidence based software engineering by undergraduate students. In: Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering

Rainer A, Jagielska D, Hall T (2005) Software engineering practice versus evidence-based software engineering research. In: Proceedings of the ACM Workshop on Realising evidence-based software engineering (REBSE '05), pp 1–5

Rainer A, Hall T, Baddoo N (2006) A preliminary empirical investigation of the use of evidence based software engineering by under-graduate students. In: Proceedings of the 10th International Conference on Evaluation and Assessment in Software Engineering

Rogstad E, Briand LC (2016) Cost-effective strategies for the regression testing of database applications: Case study and lessons learned. Journal of Systems and Software 113:257–274

Rogstad E, Briand LC, Torkar R (2013) Test case selection for black-box regression testing of database applications. Information & Software Technology 55(10):1781–1795

Rombach HD, Ciolkowski M, Jeffery DR, Laitenberger O, McGarry FE, Shull F (2008) Impact of research on practice in the field of inspections, reviews and walkthroughs: learning from successful industrial uses. ACM SIGSOFT Software Engineering Notes 33(6):26–35

Rosero RH, Gómez OS, Rafael GDR (2016) 15 years of software regression testing techniques - A survey. Int J Software Eng Knowl Eng 26(5):675–690

Rothermel G, Harrold MJ (1996) Analyzing regression test selection techniques. IEEE Trans Software Eng 22(8):529–551

Saha RK, Zhang L, Khurshid S, Perry DE (2015) An information retrieval approach for regression test prioritization based on program changes. In: Proceedings of the 37th IEEE/ACM International Conference on Software Engineering, ICSE, pp 268–279

Singh Y, Kaur A, Suri B, Singhal S (2012) Systematic literature review on regression test prioritization techniques. Informatica (Slovenia) 36(4):379–408

Skoglund M, Runeson P (2005) A case study of the class firewall regression test selection technique on a large scale distributed software system. In: Proceedings of the International Symposium on Empirical Software Engineering, ISESE, pp 74–83

Srivastava A, Thiagarajan J (2002) Effectively prioritizing tests in development environment. In: Proceedings of ACM SIGSOFT International Symposium on Software Testing and Analysis, ACM, ISSTA '02, pp 97–106

Storey MD, Engström E, Höst M, Runeson P, Bjarnason E (2017) Using a visual abstract as a lens for communicating and promoting design science research in software engineering. In: Proceedings of ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, ESEM, pp 181–186

Tahvili S, Afzal W, Saadatmand M, Bohlin M, Sundmark D, Larsson S (2016) Towards earlier fault detection by value-driven prioritization of test cases using fuzzy TOPSIS. In: Latifi S (ed) Information Technology: New Generations, Springer International Publishing, pp 745–759

Thelwall M, Kousha K (2017) ResearchGate versus Google Scholar: Which finds more early citations? Scientometrics 112(2):1125–1131
Uusitalo EJ, Komssi M, Kauppinen M, Davis AM (2008) Linking requirements and testing in practice. In: Proceedings of the 16th IEEE International Requirements Engineering, RE, IEEE, pp 265–270
Vöst S, Wagner S (2016) Trace-based test selection to support continuous integration in the automotive industry. In: Proceedings of the International Workshop on Continuous Software Evolution and Delivery, CSED, pp 34–40
Wang S, Ali S, Gotlieb A (2013a) Minimizing test suites in software product lines using weight-based genetic algorithms. In: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO, pp 1493–1500
Wang S, Gotlieb A, Ali S, Liaaen M (2013b) Automated test case selection using feature model: An industrial case study. In: Proceedings of the 16th International Conference on Model-Driven Engineering Languages and Systems, MODELS, pp 237–253
Wang S, Buchmann D, Ali S, Gotlieb A, Pradhan D, Liaaen M (2014) Multi-objective test prioritization in software product line testing: an industrial case study. In: Proceedings of the 18th International Software Product Line Conference, SPLC, pp 32–41
Wang S, Ali S, Gotlieb A (2015) Cost-effective test suite minimization in product lines using search techniques. Journal of Systems and Software 103:370–391
Wang S, Ali S, Yue T, Bakkeli Ø, Liaaen M (2016) Enhancing test case prioritization in an industrial setting with resource awareness and multi-objective search. In: Proceedings of the 38th International Conference on Software Engineering, ICSE 2016, Austin, TX, USA, May 14-22, 2016 - Companion Volume, pp 182–191
Wang S, Ali S, Gotlieb A, Liaaen M (2017) Automated product line test case selection: industrial case study and controlled experiment. Software and System Modeling 16(2):417–441
White LJ, Robinson B (2004) Industrial real-time regression testing and analysis using firewalls. In: Proceedings of the 20th International Conference on Software Maintenance, ICSM, pp 18–27
White LJ, Jaber K, Robinson B, Rajlich V (2008) Extended firewall for regression testing: an experience report. Journal of Software Maintenance 20(6):419–433
Wikstrand G, Feldt R, Gorantla JK, Zhe W, White C (2009) Dynamic regression test selection based on a file cache an industrial evaluation. In: Proceedings of the International Conference on Software Testing Verification and Validation, ICST, IEEE, pp 299–302
Wohlin C (2013) Empirical software engineering research with industry: top 10 challenges. In: Proceedings of the 1st International Workshop on Conducting Empirical Studies in Industry, CESI, pp 43–46
Wohlin C (2014) Guidelines for snowballing in systematic literature studies and a replication in software engineering. In: Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering, EASE, pp 38:1–38:10
Xu Z, Kim Y, Kim M, Cohen MB, Rothermel G (2015) Directed test suite augmentation: an empirical investigation. Softw Test, Verif Reliab 25(2):77–114
Yoo S, Harman M (2012) Regression testing minimization, selection and prioritization: a survey. Softw Test, Verif Reliab 22(2):67–120
Zarrad A (2015) A systematic review on regression testing for web-based applications. JSW 10(8):971–990
Zelkowitz MV, Wallace DR, Binkley D (1998) Culture conflicts in software engineering technology transfer. In: NASA Goddard Software Engineering Workshop, Citeseer, p 52
Zhang H, Babar MA, Tell P (2011) Identifying relevant studies in software engineering. Information & Software Technology 53(6):625–637
Zheng J (2005) In regression testing selection when source code is not available. In: Proceedings of the 20th IEEE/ACM International Conference on Automated Software Engineering, ASE, pp 452–455
Zheng J, Robinson B, Williams L, Smiley K (2006a) Applying regression test selection for cots-based applications. In: Proceedings of the 28th International Conference on Software Engineering, ICSE, pp 512–522
Zheng J, Robinson B, Williams L, Smiley K (2006b) A lightweight process for change identification and regression test selection in using COTS components. In: Proceedings of the 5th International Conference on Commercial-off-the-Shelf (COTS)-Based Software Systems, ICCBSS, pp 137–143

Zheng J, Williams L, Robinson B (2007) Pallino: automation to support regression test selection for cots-based applications. In: Proceedings of the 22nd IEEE/ACM International Conference on Automated Software Engineering, ASE, pp 224–233

Author Biographies

Nauman bin Ali is a senior lecturer at Blekinge Institute of Technology. He is involved in empirical research in the field of software engineering. His research interests include Lean software development, software testing, software process simulation, software metrics and evidence-based software engineering. He received his Ph.D., (2015) in software engineering from Blekinge Institute of Technology, Sweden

Emelie Engström is an associate senior lecturer of Software Engineering at Lund University and a member of the Software Engineering Research Group. Her research interests include empirical software engineering, especially focused on software quality, operational decision support, and industry-academia collaboration. She received her PhD in 2013 from Lund University, Sweden.

Masoumeh Taromirad is a postdoctoral researcher at the Department of Computer Engineering, Sharif University of Technology, Iran. She received her PhD in computer science from the University of York, UK, in 2015, and was a postdoctoral researcher at Halmstad University. She is generally interested in software engineering and particularly, in systems verification and testing, model-driven engineering, and software development methodologies.

Mohammad Mousavi is a professor of Data-Oriented Software Engineering at the Department of Informatics, University of Leicester, UK. He got his Ph.D. in Computer Science from Eindhoven University of Technology, The Netherlands in 2005. Since then he held positions at Reykjavik University (postdoctoral researcher), Eindhoven University of Technology (assistant and associate professor), Delft University of Technology (guest faculty member), Halmstad University (professor of Computer Systems Engineering), and Chalmers / University of Gothenburg (guest professor of Software Engineering). Mohammad's main research area is in model-based testing, particularly applied to software product lines and cyber-physical systems.

Nasir Mehmood Minhas is a PhD candidate working at Software Engineering Research Lab (SERL) Sweden, Blekinge Institute of Technology, Karlskrona Sweden. Previously he was serving as Assistant Professor of Software Engineering at University Institute of Information Technology (UIIT), PMAS Arid Agriculture University Rawalpindi, Pakistan, and he has a teaching experience of more than 18 years. His research interests are software testing, software requirement engineering, formal methods in software engineering and software process.

Daniel Helgesson holds a M. Sc. from Chalmers University. He spent about fifteen years in the trenches of the telecom industry, after which he returned to academia. He is currently working as Ph.D.-student at the department of computer science at Lund University, with a research focus on cognitive dimensions of software engineering.

Sebastian Kunze was a Ph.D. student at the Centre for Research on Embedded Systems at the Halmstad University in Sweden. Nowadays, he is working for Netlight Consulting AB in test automation, machine learning, and cloud computing to deliver independent solutions and tangible results based on the clients business objectives.

Mahsa Varshosaz is a PhD candidate at the Centre for Research on Embedded Systems (CERES) at the School of Information Technology in Halmstad University. She received her Master'fs degree in software engineering from University of Tehran in 2012. Her research interests include model-based testing and verification of software systems, in particular systems with variability.