How Practitioners Perceive the Relevance of ESEM Research

Jeffrey C. Carver
University of Alabama
carver@cs.ua.edu

Oscar Dieste
Universidad Politecnica de Madrid
odieste@fi.upm.es

Nicholas A. Kraft
ABB Corporate Research
nicholas.a.kraft@us.abb.com

David Lo
Singapore Management University
davidlo@smu.edu.sg

Thomas Zimmermann
Microsoft Research
tzimmer@microsoft.com

ABSTRACT

Background: The relevance of ESEM research to industry practitioners is key to the long-term health of the conference. Aims: The goal of this work is to understand how ESEM research is perceived within the practitioner community and provide feedback to the ESEM community ensure our research remains relevant. Method: To understand how practitioners perceive ESEM research, we replicated previous work by sending a survey to several hundred industry practitioners at a number of companies around the world. We asked the survey participants to rate the relevance of the research described in 156 ESEM papers published between 2011 and 2015. Results: We received 9,941 ratings by 437 practitioners who labeled ideas as Essential, Worthwhile, Unimportant, or Unwise. The results showed that overall, industrial practitioners find the work published in ESEM to be valuable: 67% of all ratings were essential or worthwhile. We found no correlation between citation count and perceived relevance of the papers. Through a qualitative analysis, we also identified a number of research themes on which practitioners would like to see an increased research focus. Conclusions: The work published in ESEM is generally relevant to industrial practitioners. There are a number of topics for which those practitioners would like to see additional research undertaken.

CCS Concepts

•**General and reference** → **Surveys and overviews; Reference works;**

Keywords

Survey, Industrial Relevance, ESEM Conference

ESEM '16 Ciudad Real, Spain

DOI:

1. INTRODUCTION

The gap between research and practice is an oft-discussed topic both in general and in the ESEM community. Many software engineering researchers, including those within the ESEM community, desire for research results to have an impact on practice. Unfortunately, industrial participation in many conferences (including ESEM), is often quite low, which has the potential to limit the impact. This dichotomy raises two important questions: (1) Is research published at ESEM relevant to practitioners' needs? and (2) How can ESEM authors make our research even more relevant to practitioners' needs? These questions are often asked by funding agencies and even researchers themselves. Answers to these questions will provide important insight into the differences between the perceptions of practitioners and researchers. This insight will also help chart a course of action to bridge the gap between these communities. In this work, we hope to make researchers aware of problems that matters to practitioners and reach out to practitioners to give them a voice into the research community.

Along these same lines, Lo et al. [11] performed a successful empirical study to assess how practitioners at Microsoft perceive the relevance of software engineering papers published in ICSE and FSE between 2009-2014. To understand ESEM's relevance to practitioners, we replicate that study by considering papers published in ESEM between 2011-2015. We also expand the population to include practitioners at various companies around the world. Our goal is to measure the degree of disconnect (if any) between ESEM researchers and practitioners.

To make this process feasible, we ask practitioners to review short (1-2 sentence) summaries of each paper rather than asking them to read the whole paper (or even the whole abstract). First, we asked the authors of each paper to write short descriptions of their work. Many authors responded with a summary. In cases where the authors did not provide a summary, we created one. We also created a second summary for each paper to study whether the author of the summary (paper author or researcher) or the level of detail in the summary would lead the summary to be viewed differently by practitioners. Next, we sent these summaries to a wide variety of practitioners from around the world. The largest sets of respondents were from Microsoft and ABB, but we also had respondents from a number of smaller orga-

nizations across the world. The survey presented each practitioner with a randomly chosen set of 32 summaries to read and rate as: *Essential*, *Worthwhile*, *Unimportant*, *Unwise*, or *I don't understand*. We also gathered qualitative feedback about the summaries and about future research directions. The survey received a total of 437 usable responses.

In this paper, we investigate: (1) whether ESEM research is relevant to industrial practitioners, (2) which research ideas are most highly-rated, (3) whether papers with one or more industrial authors are more relevant to practitioners, (4) whether the type of summary impacts the ratings, and (5) what types of research practitioners would like to see emphasized in ESEM research. Our findings highlight that work published in ESEM are relevant to industrial practitioners. Additionally, our study also identifies a few most highly-rated research ideas and the impact of (or lack of) various factors on the ratings.

The primary contributes of this paper include:

1. A formal investigation of the perception software engineering practitioners have about research work published in the last five years of ESEM.
2. An analysis of the types of research practitioners found to be most relevant
3. Recommendations to help the ESEM community better bridge the gap between research and practice.

The remainder of the paper is structured as follows. Section 2 describes background materials including descriptions of related work. Section 3 explains the study design. Section 4 reports the findings of the study. Section 5 discusses additional analyses. Section 6 concludes the paper and explains future work.

2. BACKGROUND

In this section, we highlight related studies that can be categorized into two families: studies assessing impact of software engineering research, and those assessing practitioner perceptions on various issues.

2.1 Studies Assessing Impact of Software Engineering Research

Numerous studies focus on the assessing impact of software engineering research [4,7,8,21–23]. Ryder and Soffa [22] investigated how research on exception handling helps shape current practice. Ryder et al. [23] reported research studies that have impacted features of modern programming languages. Estublier et al. [8] described research work done in the university and industry on software configuration management and their impact on practice. Clarke and Rosenblum [4] investigated how runtime assertion checking developed over time and how it has been used in the industry. Emmerich et al. [7] explored middleware technologies and elaborated how research has impacted practice. Rombach et al. [21] described relationship between research and practice focusing on software inspection, review, and walk-through.

These studies are part of the ACM SIGSOFT IMPACT project [18] which aims to identify how software engineering research has influenced practice in a significant way. In this work, we do not focus on assessing the impact of a research work. Impact is often known only many years later after a work has been completed. Rather, we aim to assess the relevance of research as perceived by practitioners. Our purpose is to assess the degree of disconnect between current research and developer perceptions. We need to stress that research work marked as unwise or unimportant by developers may in turn be viewed as relevant or important many years later. However, effort may be needed to better communicate these research works to practitioners to reduce their initial skepticism, potentially fostering adoption.

2.2 Studies Assessing Practitioner Perceptions

Many studies have investigated how practitioners perceive certain matters/issues [1, 13, 14, 19]. Misirli et al. [14] interviewed 12 practitioners after deploying effort estimation and defect prediction solutions to their company to gather their feedback. Bavota et al. [1] investigated how practitioners perceive code coupling which is often viewed negatively by the research community. Palomba et al. [19] performed a study that investigated developer perception on code smell, while Meyer et al. [13] investigated developer perception on productivity. Similar to these papers, we also seek to understand practitioner perceptions. However, we focus on answering a different question: How do practitioners perceive the relevance of recent ESEM papers?

The most related (and recent) work is the study by Lo et al. [11] mentioned in Section 1. In that paper, researchers sought to understand how Microsoft developers perceived the relevance of research from five years of ICSE and FSE papers. The researchers developed short summaries each of the published papers and asked practitioners to rate those summaries as *Essential*, *Worthwhile*, *Unimportant*, or *Unwise*. Overall, the results showed that practitioners had a positive view of the research published in these two venues. Lo et al. also described a number of reasons given by practitioners to explain why they viewed some research topics as "unwise", including: *no need for a tool*, *empirical study nonactionable*, *generalizability issues*, *cost vs. benefit*, *questionable assumptions*, *better solutions exist*, and *side effects*.

We replicate the Lo et al. study and address some of its limitations. Specifically, we: (1) broaden participation beyond Microsoft, (2) ask authors to provide summaries (rather than only researchers), and (3) ask practitioners to provide guidance to the research community about the most important types of problems in need of research.

3. STUDY DESIGN

We replicated the design of Lo et al.'s original study [11], with some modifications to explore additional research questions. The following subsections describe the research questions, the process for selecting and summarizing the papers, the process for selecting participants, the method for feedback elicitation, and our data analysis process.

3.1 Research Questions

In this study, we seek to answer some of the same questions in Lo et al.'s original study plus pose a few new ones.

The Industrial Relevance of ESEM Research.

The primary goal is to obtain an overall understanding of the relevance of ESEM research to industrial practitioners. Therefore, the primary question: **RQ1: What is the relevance of ESEM research to industrial practitioners?**

To provide additional insight into this top-level result, we pose a number of additional questions. First, we do not expect that every ESEM paper has the same level of relevance. Therefore, the next research question is: **RQ2: What are the most highly rated research ideas?**

Second, a number of ESEM authors have direct ties to industry (often through the research arm of an organization). It is quite possible that papers with an industrial co-author may have more direct relevance to industry. To explore this potential impact, the next research question is: **RQ3: Do papers with one or more industrial authors have higher industrial relevance than other papers?**

The Impact of Paper Summarization.

One of the limitations of the original study was that the researchers, rather than the authors, summarized the papers. It is possible that the perspective from which the summary is written may influence the perceived relevance of work. To explore whether the content of the summary has any effect on the perceived relevance, we pose two additional questions. First, we wanted to understand the effect if the author summarized the paper rather than the researchers. That lead to this research question: **RQ4: Do the results change depending whether the summary was written by the author or by researchers?**

Second, it is possible that the level of detail contained in the summary could affect the perceived relevance. That led to the next research question: **RQ5: Do the results change depending on whether the researcher's summary is more or less detailed?**

Guidance to Researchers.

Finally, there is often a gap between the focus of research studies and the needs of practitioners The last research question seeks to bridge this gap: **RQ6: What research problems do practitioners think are most important to be focused on by the research community?**

3.2 Paper Selection and Summarization

We selected all 161 Full Papers and Industrial Experience papers from the 2011-2015 editions of the *International Symposium on Empirical Software Engineering and Measurement (ESEM)*. As in the original study, we provided the participants with summaries of the papers so they would not have to read the entire paper. The paper summarization process is one area where we made changes to the design of the original study. To provide data to answer questions RQ4 and RQ5, we created two summaries for each paper for a total of 322 summaries.

We asked authors of the included papers to summarize their own paper based on these guidelines:

- Summarize the paper in a way that will be understandable to general practitioners. Focus on the research question(s) or topic(s) rather than specific results or research methodology. Avoid use of technical jargon.
- The summary should still uniquely refer to your paper rather than being applicable to a whole subarea.
- The summary should be no longer than 240 characters.
- Do not include the paper title or authors names.

For the authors that responded to this request, we labeled the summary as an *Author Summary*. We then created our own summary for these papers, following the same guidelines, but without looking at the *Author Summary*. We labeled these summaries as *Research Summary*. The authors provided summaries for approximately 1/3 of the papers. To answer RQ4, we compare the ratings of the *Author Summaries* to the ratings of the *Researcher Summaries*. Note that some summaries exceeded the 240 character limit.

For the remaining papers, we, the researchers, developed two summaries using different approaches. First, we created a summary following the same guidelines we sent to the authors. We labeled these summaries as *Researcher (Simple)*. Then we created a more detailed version of the summary that included information about the type of study conducted and the type/number of participant involved. We labeled these summaries as *Researcher (Detailed)*. To answer RQ5, we compare the ratings of the *Researcher (Simple)* summaries to the ratings of the *Researcher (Detailed)* summaries. Note that the survey did not indicate to the respondent which type of summary s/he was reading.

To illustrate the difference between simple and detailed summaries, here are two versions for the same paper (note that the *Author Summaries* and corresponding *Researcher Summaries* are more similar to the *Simple* summary):

[Simple] A study of the impact of System Sequence Diagrams and System Operation Contracts on the quality of the domain model.

[Detailed] A family of four controlled experiments comprising 55 groups of undergraduate students, aimed at evaluating the impact of System Sequence Diagrams and System Operation Contracts on the quality of the domain model.

3.3 Participant Selection

To broaden the industrial perspective from Lo et al.'s survey, we solicited industrial participants from a number of organizations across the world. We used our contacts to solicit full-time development staff. We wanted to ensure that the respondents had sufficient background to provide useful feedback, so we restricted our participants to those in technical roles within their organizations. To protect the anonymity of the respondents, we did not ask them to indicate their employer, so we cannot report the exact breakdown of respondents. Based on the data, we can estimate that of the 437 responses approximately 230 were from Microsoft, 100 were from ABB, and the remainder were from other companies including: Apple, Boeing, Google, Home Depot, NetApp, SAIC, CAPS, Command Alkon, Netflix, and McLeod Software.

3.4 Feedback Elicitation

Following the success of Lo et al.'s survey, we designed a web-based survey to gather feedback from the participants. The survey was completely self-contained, in that they respondents did not have to access materials outside of the survey to answer the questions. We limited the survey questions to numerical, rating-scale, or short answers as done in the original survey and suggested by Kitchenham and Pfleeger [10]. We gathered the following information.

Demographics

- *Primary Work Area*: Development, Test, Program Manager, Other
- *Role*: Individual contributor, Lead, Architect, Manager, Executive, Other
- *Experience in years* [Decimal]
- *Major in Computer Science* [Boolean]
- *Has advanced degree (MSc, PhD, etc.)* [Boolean]

Quantitative Ratings of Research For each participant, we randomly selected 32 summaries from the collection of 322

summaries described above. For each summary the respondent answered "*In your opinion, how important are the following pieces of research?*" Using the rating categories in Lo et al.'s paper, which were drawn from earlier work of Begel and Zimmermann [2], the participants could label each research idea as "*Essential*", "*Worthwhile*", "*Unimportant*", "*Unwise*", or "*I Don't Understand*".

Relevance and adoption are two different concepts. In this study, our goal was to judge how relevant the industrial practitioners perceived the research to be rather than their ability to adopt the ideas. Adoption of new methods can be influenced by many factors outside the control of the survey respondent. But, regardless of those factors, if the respondent does not find the idea relevant, they will be less likely to push for adoption.

Qualitative Feedback To provide a deeper understanding of practitioners perception of ESEM research, we asked for two types of qualitative feedback. First, to understand the rationale behind the ratings, we randomly chose two of the summaries the participant rated and asked them to "*provide a brief explanation for why you found it either relevant or not to your work.*" Second, we gave the participants an opportunity to provide guidance to the research community about topics of interest. We asked them "*Suppose that you could provide guidance to a team of software engineering researchers, what problems should they focus on first?*"

3.5 Data Analysis

We used the same measures as Lo et al. [11] to characterize the perspectives that practitioners have on software engineering research. We measure the proportion of ratings that are Essential (best response), Worthwhile (positive feedback), Unimportant (negative feedback), or Unwise (worst response), respectively. More formally, let E, W, Ui, and Uw denote the number of essential, worthwhile, unimportant, and unwise ratings received.

- **E-score:** The percentage of "Essential" ratings.

$$E\text{-}Score = \frac{E}{E + W + Ui + Uw}$$

- **EW-score:** The percentage of "Essential" or "Worthwhile" ratings.

$$EW\text{-}Score = \frac{E + W}{E + W + Ui + Uw}$$

- **U-Score:** The percentage of "Unwise" ratings.

$$U\text{-}Score = \frac{U}{E + W + Ui + Uw}$$

The statistics can be computed for different groups, e.g., all ratings, ratings by certain demographics, ratings for specific conferences, or ratings for individual papers.

To analyze the qualitative feedback regarding the types of problems respondents would like to see software engineering researchers studying, we used card sorting techniques. Specifically to identify themes of what problems software engineering researchers should focus on, we used an open card sort [24]. Card sorting is widely used to create mental models and derive taxonomies from data. Two authors jointly sorted the cards.

4. RESULTS

This section is organized around the six research questions posed in Section 3.1. We had to remove five papers from the analysis due to problems with the summaries. Therefore, the results are based upon the ratings of 156 papers.

4.1 Relevance of ESEM Research

This section answers *RQ1: What is the relevance of ESEM research to industrial practitioners.* The survey respondents provided a total of 9,941 ratings. Figure 1 shows the percentage of papers included in each of the rating (e.g.: *Essential*) categories, across different population subgroups (Defined in Section 3.4). Based on this data, we can make a few observations from this data:

- 67% of the ratings were *Essential* or *Worthwile*, while only 5% were considered to be *Unwise*.
- Considering the *Primary Work Area*, the Developers and Testers were fairly consistent, with Program Managers having a significantly lower EW-score (60%).
- Considering *Experience*, we have considered low experienced participants (ExpLow) those below the 25th percentile of experience (4.28 years). High experienced (ExpHigh) are those above the 75th percentile (15.5 years). The results are similar for the Low and Medium experienced participants. The high experience participants had a significantly lower EW-score (57%).
- Whether someone *has an advanced degree* does not affect the overall ratings.
- Those practitioners who *Major in Computer Science* have a slightly higher overall rating (68% to 63%) that those who did not.
- Considering *Role*, Managers had a significantly lower EW-score (57%).

The most significant result has to do with high experienced participants' and managers' ratings. They give consistently lower, statistically significant scores than any other group. Absolute differences are around 9%. Their scores are similar to each other.

The scores for the 2011-2015 editions of the ESEM conference are stable over time, as shown in figure 2. The EW-score exceeds 60% in all editions, exhibiting only small departures from the 5-year average of 66%. The papers rated as *Essential* experience a modest increment (3%-8%, depending on the edition).

4.2 Highly Rated Research Ideas

This section answers *RQ2: What are the most highly rated research ideas?* We examined the ratings for each summary to identify which summaries described work that was most relevant to practitioners. Note that we considered the two summaries for each paper separately. Table 1 highlights the ten highest rated summaries. The papers are sorted in terms of their E-score (descending) followed by EW-score (descending). Again, we can make some interesting observations regarding these highly-rated ideas:

- Only one paper (the paper represented by S1 and S4) had both summaries in the top ten. (Note that because of this duplication, the total number of papers discussed in the remainder of this section is nine rather than 10). This result suggests that how the summary is written may have an effect on the perceived relevance of the paper (see Section 4.4).

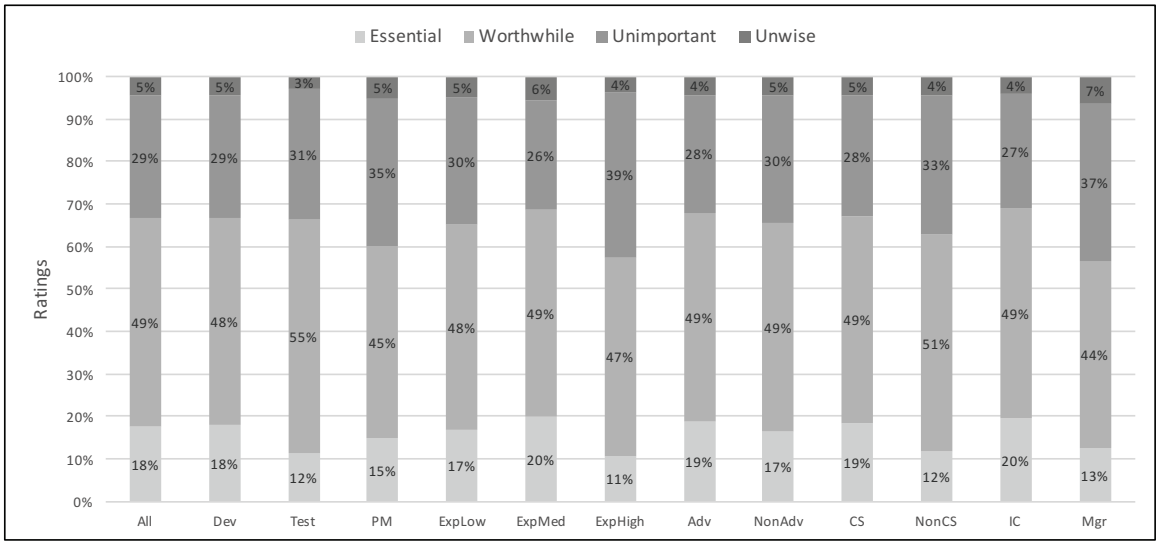

Figure 1: Ratings for some representative population subgroups (from left to right: All [participants], Dev[elopers], Test[ers], P[rogram] M[anagers], participants with: low, medium and high experience, holding an advanced degree on CS or not, majored in CS or not, and individual contributors and managers)

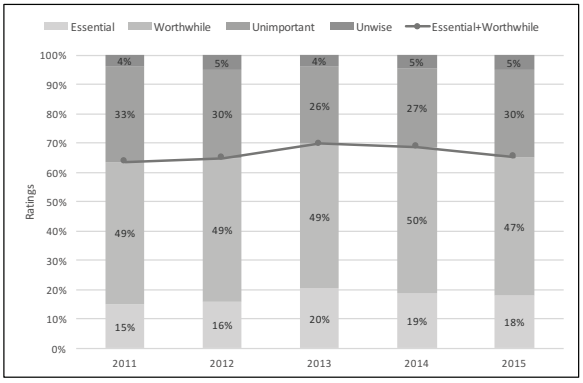

Figure 2: Rating for the 2011-2015 editions

- The papers represented by these summaries cover diverse topics in the SWEBOK, including (number of papers in parenthesis): Software Construction (3), Testing (2), Professional Practice (2), Computing Foundations (1), and Software Engineering Economics (1).
- Half of the top ten summaries were written by the paper authors. For four of those five papers, the summary written by the researchers was ranked much lower (over 100). The only exception is S5 where the researcher summary was only ranked as number 15.
- Two of the papers, including the highest rated paper, have no citations.
- The number of highly rated summaries varied across years: 2013 - 5, 2014 - 2, 2015 - 1, and 2011 - 1.
- Only one of the papers represented by these summaries won the Best Paper Award (S7).

4.3 Industry Co-Authorship

This section answers *RQ3: Do papers with one or more industrial authors have higher industrial relevance?* When a research has industry partners, it is likely that some themes for industry influence how the research is performed and reported. It is also highly likely that the presence of industry partners implies that the research has been conducted in or in close connection to industry. Paper with industry co-authors could be seen by practitioners as more relevant than purely academic papers. In our sample, 51 out of 156 papers (33%) had an industrial co-author.

Table 2 shows the ratings for papers with and without industry co-authors. Papers with industry co-authors have higher ratings, but differences are marginal and in all cases non-significant. The EW-scores (69% and 65%, respectively) are in line with the EW-score of the overall population (67%). Furthermore, of the papers with highly rated summaries (Table 1), three, including the top paper, had industry co-authors (the same ratio papers with industrial co-authors in the overall sample). So, having an industrial co-author does not seem to affect whether a summary is highly-rated.

The ESEM conference has an industry track, where industry research and experience reports are published. In our sample, 44 out of 161 papers (27%) were from the industry track. Another possibility is that the papers published in the industry track were rated higher by practitioners than the papers published in the main (research) track. However, the EW-score for the industry track is 67%, whereas for the main research track is 66%. The difference is not statistically significant. Furthermore, of the papers with highly rated summaries (Table 1), two of nine, including the top paper, were from the industry track (22%). Again, the type of paper does not seem to affect the overall rating.

4.4 The Impact of Paper Summaries

This section answers *RQ4: Do the results change depending on whether the summary was written by the author or by the researchers?* and *RQ5: Do the results change depending on whether the researcher's summary is more or less detailed?*

The authors of 95 papers provided their own paper sum-

Table 1: Highly rated research ideas

	Paper Summary	Type	Total	E-Score	EW-Score	U-Score	Rank Other Summary
S1	A study on the cost effectiveness of unit testing. [6]	R(S)	39	0.59	0.79	0.05	4
S2	Empirical study on which features of an API are most critical to achieve good usability: clear names, simple type hierarchies, accurate documentation, and the right amount of flexibility. [20]	A	35	0.57	0.94	0.00	127
S3	A historical data mining investigation of 1600 open-source software bugs to identify how users report bugs, including what information is provided, how frequently, and their consequences. [5]	R	30	0.47	0.83	0.03	48
S4	A case study on the cost effectiveness of unit testing, in terms of early defect detection and cost savings, conducted in a financial institution. [6]	R(D)	31	0.45	0.74	0.00	1
S5	An empirical simulation to understand how far state-of-the-art speech translation technology (i.e., speech recognition + machine translation) is from being useable in synchronous distributed meetings where each party is free to speak in their own native language, in order to grant everyone equal 'communication powers' to steer discussion. [3]	A	30	0.43	0.80	0.03	15
S6	A case study to investigate how automated test case generation tools process can contribute to the effectiveness and efficiency of testing, when they are used in industrial environments. Effort required to deploy such generation tools is also evaluated. [16]	A	27	0.41	0.89	0.00	186
S7	An empirical study that explores the strengths and weaknesses of four approaches to parallel programming (Chapel, Cilk, Go, and Threading Building Blocks) with respect to source code size, coding time, execution time, and speedup. [15]	R(D)	28	0.39	0.89	0.00	28
S8	A study of commits that introduced vulnerabilities in the Apache HTTP code, with the purpose of analyzing whether developers could have identified, prevented or caught them. [12]	R(S)	18	0.39	0.78	0.00	23
S9	Software project pricing based on evidence-based trends of functional size measurement and cost, without expert judgments. A real-life pilot in a globally distributed setting resulted in improved project transparency and satisfied stakeholders. [9]	A	23	0.39	0.65	0.04	302
S10	Open Source development methods can benefit specialized domains where core developers include subject matter experts (doctors/clinicians) who also use the product. Despite small size of the community we found a very high degree of responsiveness to issues raised by users (new and old). The implication is that a few experts and a small core of dedicated programmers can achieve success using an Open Source approach in a specialized domain. [17]	A	29	0.38	0.72	0.00	192

*Note: A = Author, R = Researcher, R(S) = Researcher (Simple), R(D) = Researcher (Detailed)

Table 2: Ratings of papers with and without industry co-authors

	E	W	Ui	Uw
Has industry co-authors	19%	49%	27%	4%

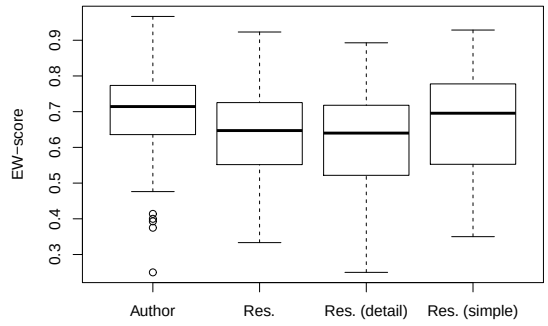

Figure 3: Impact of summaries

Table 3: Impact of summaries

	E	W	Ui	Uw
Authors	21%	49%	26%	4%
Researchers	16%	48%	31%	5%
Researchers (Simple)	17%	51%	28%	4%
Researchers (Detailed)	15%	48%	32%	5%

maries, leaving $156 - 95 = 61$ papers for which the researchers created the summaries. We would expect that authors know their work better than anyone else. The summaries created by the researchers may influence (positive or negative) the participants' ratings. To assess this potential bias and answer RQ4, we created summaries for some papers for which we already had an author's summary (as described in Section 3.2). Table 3 shows the overall scores. Authors' summaries are regarded as more *essential* and less *unimportant* than researchers' summaries. While those differences are statistically significant (using paired t-tests, $\alpha = .05$), the box-plot in Figure 3, where the EW-scores are displayed, shows that the differences are not very large. One interest-

ing caveat is that while only 30% of the summaries were written by authors (95/322), five of the top ten summaries are author summaries.

To answer RQ5, we also created simple and detailed summaries for papers without an author's summary (as described in Section 3.2). The results, also displayed in Table 3 and Figure 3, mimic the former ones. Simple summaries get better rankings, resembling those created by authors. However, differences are again statistically significant.

To provide more insight to the results in the previous section, and better understand the impact of the content of the summary, we analyzed the 10 papers for which the two summaries had the largest difference in EW-score. Initially we hoped to analyze the qualitative data explaining the ratings, but we did not receive enough to be meaningful. Interestingly, none of those papers were represented among the most highly rated summaries (Table 1). We can make a few observations about these papers related to the discussions in the prior subsections: (1) For 5 of these papers, one summary was from an author and in four of those cases, the author summary had the higher rating; and (2) For the other five papers that had only researcher summaries, in four of the five cases the Simple summary had a higher rating.

In addition, we looked at the papers related to S2, S6, S9, and S10 from Table 1 to see why the other summary might have been rated so much lower. We cannot draw definitive conclusions from only four papers, but we observed the following trends: (1) summaries that indicate that the study was done done in a realistic setting or describes the type of participants as experts were rated highly; (2) summaries that give a company name or the number of subjects (when that number is low) are rated lower.

4.5 Guidance to Software Engineering Researchers

This section answers *RQ6: What research problems do practitioners think are most important to be focused on by the research community?*

As the last question in the survey, we asked practitioners "*Suppose that you could provide guidance to a team of software engineering researchers, what problems should they focus on?*" Of the participants, 223 responded to the question. A large number of participants misunderstood the question, were uninspired ("I have no idea."), or provided a response that was too generic ("Focus on solving real problems the industry has.") or too broad ("Understanding the problem. / Designing solutions to the problem. / Identify potential problems in each solution."). We excluded 60 of such responses from the qualitative analysis; 163 responses remained. To analyze the responses, two authors jointly coded the responses.

In the following, we report the frequency of each code in the qualitative analysis. However, please note that a higher frequency does not necessarily imply a higher importance because of the qualitative nature of the question. We now briefly discuss the themes that emerged from the responses. The themes can be grouped into higher-level categories of: software, process, developers, and users.

Software. Several open problems were related to properties of the software itself. Survey participants wanted to know about Code Quality (15×), Sustainability (13×), Architecture (10×), Software Design (6×), and Usability (1×).

For *Code Quality* the participants asked for different ways to improve code quality, for example "An analysis of how and why and what can be prevented for code submits that broke the build in the last 2-3 years.", "How can we improve code quality to prevent bugs which can be caught earlier in the development process"; and "Define good qualities of software and help developers to implement software with great quality."

The responses related to *Sustainability* were about how to ensure the long-term success of software, e.g., to "minimize future application software obsolescence" or "produce software that is maintenance friendly.". Specific problems that were mentioned were dead code ("Identify and eliminate dead code in a big legacy project"), code maintenance and technical debt ("The biggest problem in most software engineering systems is code maintenance and technical debt").

The responses related to *Architecture* were about how to architect software that has high scalability, performance, portability, security, and/or energy efficiency. Responses related to the *Design* were about how to "design software with low coupling and high cohesion in a clear and simplistic nature" and "ways to effectively incorporate new knowledge into an existing design."

Process. The main problems related to the software development process fell into the categories of Testing (13×), Estimation (12×), Process Improvement (8×) as well as Success/Failure of Projects (9×) and Cost/Benefit Analysis of Technologies (7×)

Many responses related to *Testing* concerned improving test automation and making testing a fluid part of the development process. Others comments were about "Analyzing the impact of unit testing on architecture quality", "What testing techniques should be used according to the programming paradigm used.", or "How do different testing approaches translate to increased or decreased user satisfaction?"

The responses related to *Estimation* were about fault prediction and effort estimation, for example "Correctness, fault prediction", "Practical methods for time estimation", and "Predicting security vulnerabilities before they can occur."

Several responses were related to *Process Improvements*, for example "Focus on why we aren't adhering to the software dev lifecycle and how we could improve on our agile processes." and "I'd focus more on a way to make the whole software process more efficient."

Participants were interested in learning about what properties defines the *Success/Failure of Projects*, for example "More post-mortems on failed and successful systems. Open source can provide some examples, but commercial systems are well worth understanding. How often is technical success (the product performed as it was supposed to) paired with actual failure (market, adoption, etc.)?"

A related topic was about providing a *Cost-Benefit Analysis of Technologies*, for example "Cost-benefit analysis of buzzwordy trending techniques... For example, proving with irrefutable evidence that TDD or Pair Programming present cost-savings to the company. The same applies to microservices architectures, hybrid apps (i.e. using Reactive Native opposed to native code), etc."

Participants mentioned several other problems related to process topics such as Agile Development (3×), Measurement, Documentation, Tool Usage (each 2×), Continues Delivery, Bug Triage, Code Review, and Debugging (each 1×).

Developers of a software. The main problems re-

lated to the developers of a software were about Productivity (16×), Communication and Coordination (15×), People (12×), and Knowledge Management (7×).

For *Productivity* participants wanted to know how developers spend their time and what could be done to make them more productive and to spend more time coding, for example "Analyzing the amount of time developers use on configuring/fixing/using tools rather than actually developing.", "Focus first on how to make engineers to be more productive in the engineering process.", and "Everything I need to do to get to the coding stage is annoying. This includes setting up your environment, building code, pulling changes from other developers and dependencies. If researchers could find ways to maximize my coding time, that would be brilliant."

For *Communication and Coordination* participants mentioned the need for better communication and coordination tools, for example "Enabling strong, clear communication tools and practices between developers" and "How can we be effective in working in globally distributed development teams."

Participants also mentioned challenges related to *People* with respect to the performance and attitude of individuals, especially in teams. Example responses are "how to improve the success rate of software project managers" and "how to maintain an attitude of team first, instead of IC first. How to overcome social gaps in order to enable a team to be a team, and not just Individual Contributors".

As a specific opportunity to improve communication and coordination, the survey participants mentioned *Knowledge Management*, for example "Ways of transferring knowledge between developers. If they all are proactive, and If only one of them are" and "Improve the ability of developers to express intent of code. A large percentage of bugs are the result of a disconnect between developer intent and code implementation."

Other problems related to the developers were Onboarding (3×), Education, Team Dynamics (each 2×), and Hiring (1×)

Users of a software. The survey participants wanted to know about Requirements (21×) and about Customers (10×). For *requirements* an important problem to be addressed by research was "Getting, understanding, and anticipating the requirements for a software system with a very large user-base with competing needs."

Learning more about *customers* was also mentioned several times, for example "A study that would correlate customer satisfaction to code changes/checkins would be interesting." or "Understand how our customers use our products."

5. DISCUSSION

This section discusses several important aspects of our work: the relationship between citation count and perceived relevance (Section 5.1), whether the highly-rated summaries address problems important to practitioners (Section 5.2), and the limitations of this work (Section 5.3).

5.1 Citation Count vs. Perceived Relevance

Citations are often used to assess the impact and relevance of research publications. To determine whether citation counts and the practitioners' ratings are aligned, we collected the citation counts for all papers using Google Scholar as of March 3rd 2016, and calculated yearly averages.

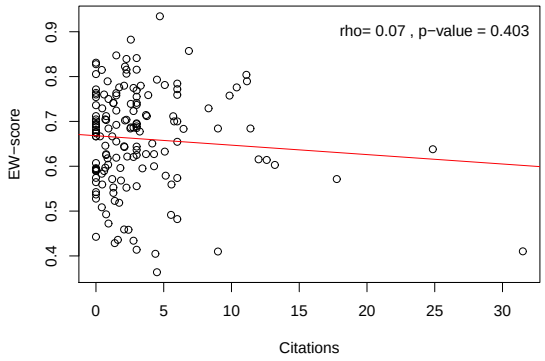

Figure 4: Scatter plot comparing citation counts (yr. averages) and EW-scores

This approach may be unfair for the 2015 publications, as only 5 months have passed since publication of those papers. Since many of those papers were available as pre-prints since June/July 2015, we decided to include them in this analysis.

Figure 4 shows a scatter plot comparing citation counts and EW-scores. The regression line is almost horizontal, showing no relationship between citation counts and EW-scores. The Spearman correlation $\rho = 0.07$ confirms the visual assessment. The plots corresponding to the other scores (E, W, Ui, Uw) are similar, both visually and statistically.

Given the previous result, we can find instances of papers with high E-scores, but having few or no citations (including the highest rated paper), and vice versa. Using the same approach as Lo et al., we divided the E-score for the paper (by combining the results from both summaries) by the citation count. Table 4 shows the top-five studies favored by practitioners (i.e. high E-score but low citation count), and also the top five favored by academia (i.e. high citation count but low E-score). The summaries are either the author summary (if provided) or the detailed researcher summary. For this analysis we excluded papers from 2015 because they have not yet had time to obtain enough citations to make the comparison fair. In addition, we excluded papers with high E-scores but no citations because the division yields an ∞ value.

A few interesting observations about these papers include: (1) None of the papers in either set appear in the list of the highly rated research ideas (Table 1), which is based solely on the summary's E-score; (2) Two of the five papers favored by practitioners were in the industry track, while none of the papers favored by academia were; (3) Two of the papers favored by academia were Methodology papers while none of the papers favored by industry were; and (4) The papers favored by practitioners seem to address concrete needs while those favored by academia seem to address why something happens or to acquire knowledge.

5.2 Relationship between Highly Rated Summaries and Practitioner Needs

To further understand whether the research published at ESEM is addressing the needs of practitioners, we map the summaries in Table 1 to the guidance provided by practition-

Table 4: Top studies favored by practitioners and academia

Favored by practitioners
A study on Eclipse and Spring Framework to characterize software files by their change histories at commit level and correlate such history to their defect proneness. History is built on architectural changes and changes in lines of code.
A study of 16 teams of three professional SW developers, each team implementing the same web application within two days. How do process and result depend on the languages and frameworks used or on a "platform culture"?
An approach to recommend bug fixers for new coming bugs based on developers' past activeness related to particular components of software products.
A mixed research conducted in 10 companies, using several interviews and questionnaires, aimed at surveying the software measurement practices and experiences related to software engineering in Finland.
Empirical study predicting software defects using history of collaboration among developers, testers, and customers. This approach may be used to better plan for the upcoming releases, helping managers to make evidence based decisions.
Favored by academia
A qualitative study to gain an understanding of the main challenges developers face in practice when they build mobile apps by interviewing 12 senior mobile developers as well as 188 survey respondents from the mobile development community.
We did an empirical study to understand the strengths and weaknesses of applying model driven engineering in large companies.
We present Debsources: a platform to publish and search the source code of the Debian distribution. We discuss our experiences with the instance running at http://sources.debian.net, which spans all current and historical Debian releases
A brief tutorial on how to apply 'Thematic Synthesis' for the synthesis of Software Engineering empirical studies.
A comparison of two different systematic literature reviews, one conducted using database search vs. another using backward snowballing for primary study identification.

ers in Section 4.5. The mapping shows that seven of the nine papers represented by the highly ranked summaries clearly map to the types of problems that practitioners mentioned. In addition, those papers map to topics that were most frequently mentioned by practitioners. The specific mapping is as follows:

- S1, S4: Map to "Process", specifically "Testing" (13x) and "Cost/Benefit Analysis of Technologies" (7x)
- S2: Maps to "Developers of a software" (Productivity, 16x) and "Software" (Software Design, 6x)
- S3: Most closely maps to "Users of a software" (Customers, 10x) and "Process" (Bug Triage, 1x)
- S5: Maps to "Developers of a software", specifically "Communication and Coordination" (15x)
- S6: Maps to "Process", specifically "Testing" (13x) and "Cost/Benefit Analysis of Technologies" (7x)
- S7: Maps to "Process", specifically "Cost/Benefit Analysis of Technologies" (7x)
- S8: Maps to "Software", specifically "Code Quality" (15x)
- S9: Maps to "Process" (Measurement, 2x) and "Users of a software" (Customers, 10x)
- S10: Maps to "Users of a software" (Requirements, 21x and Customers, 10x) and "Process" (but not really to a specific topic in that group)

This is an encouraging result as it shows that ESEM researcher work on topics that are needed by practitioners. However, we believe that there are opportunities to improve the discoverability of ESEM papers to further bridge the gap between research and practice.

5.3 Limitations

This work has similar limitations as the original work by Lo et al. [11]. The statistics reported in this paper depend on the summaries provided to the survey participants. In the original study, Lo and colleagues created the summaries. In this paper, we follow a hybrid approach: some summaries were created by the authors of the orginal ESEM papers, while other summaries were created by us. This allowed us to empirically assess the impact of the source of summaries and improve the process to solicit feedback from practitioners. Summaries created by the authors of the original ESEM papers have comparable, slightly higher relevance scores than summaries created by us.

The findings in this paper come from a limited number of companies and two companies (Microsoft, ABB) account for a significant number of the responses. We acknowledge that perspectives of practitioners in other companies and/or industries may be different. Ideally the survey would be send to a representative panel of practitioners. Even though the statistics and insights in this paper come only from a limited number of companies, we believe that they are still useful and representative of the needs of software engineers. Compared to the original study, we significantly increased the number of companies.

Lastly, we focused on assessing research work's *perceived* relevancy in the eyes of engineers. Perceived relevancy does not mean that a research work will be adopted by practitioners or will have high impact. Only time can tell the success of research in these dimensions. However, collecting data about perceived relevance is a rapid way to solicit feedback from practitioners without waiting several years.

6. CONCLUSION

In this paper we gathered feedback from 437 practitioners regarding the relevance of research published at ESEM from 2011-2015. Similar to the results of Lo et al.'s original paper on ICSE/FSE research, the practitioners generally viewed the ESEM research positively, which is encouragingg. We found that, in general, the rating of a summary does depend on whether the summary is written by the paper author or by the researchers, but that difference is relatively small. We also found that, in general, papers with direct industry ties, either via a co-author or by being in the industry track, do not have better ratings than other papers. Although, when we examine the studies favored by practitioners, the influence of industry ties becomes a bit more pronounced.

Our analysis of the qualitative guidance provided by practitioners provided a number of research ideas that the ESEM community should be pursuing. It was refreshing to see that the majority of the papers that were highly ranked were actually addressing the most important of these topics.

The next steps are to further repeat this work for other SE communities with the goal to improve the process of how

feedback can be collected from practitioners. For example, to improve discoverability of research, one could show links to the papers that a participant rated highly, or even search for related papers based on what they put into the question about needs for research. Other possible improvements are to streamline the collection of paper summaries (e.g., as part of the camera ready process) and sharing practitioner feedback with the authors of individual papers.

Acknowledgments

Thanks to the researchers who contributed paper summaries and to everyone who responded to our survey!

7. REFERENCES

[1] G. Bavota, B. Dit, R. Oliveto, M. D. Penta, D. Poshyvanyk, and A. D. Lucia. An empirical study on the developers' perception of software coupling. In *35th Intl. Conf. Soft. Engg.*, pages 692–701, 2013.

[2] A. Begel and T. Zimmermann. Analyze this! 145 questions for data scientists in software engineering. In *36th Intl. Conf. Soft. Engg.*, pages 12–13, 2014.

[3] F. Calefato, F. Lanubile, R. Prikladnicki, and J. a. H. S. Pinto. An empirical simulation-based study of real-time speech translation for multilingual global project teams. In *8th Intl. Symp. on Empirical Soft. Engg. and Measurement*, pages 56:1–56:9, 2014.

[4] L. A. Clarke and D. S. Rosenblum. A historical perspective on runtime assertion checking in software development. *ACM SIGSOFT Software Engineering Notes*, 31(3):25–37, 2006.

[5] S. Davies and M. Roper. What's in a bug report? In *8th Int. Symp. on Empirical Soft. Engg. and Measurement*, pages 26:1–26:10, 2014.

[6] D. Delgado and A. Martinez. Cost effectiveness of unit testing: A case study in a financial institution. In *7th Intl. Symp. on Empirical Soft. Engg. and Measurement*, pages 340–347, 2013.

[7] W. Emmerich, M. Aoyama, and J. Sventek. The impact of research on the development of middleware technology. *ACM Trans. Softw. Eng. Methodol.*, 17(4), 2007.

[8] J. Estublier, D. B. Leblang, A. van der Hoek, R. Conradi, G. Clemm, W. F. Tichy, and D. W. Weber. Impact of software engineering research on the practice of software configuration management. *ACM Trans. Softw. Eng. Methodol.*, 14(4):383–430, 2005.

[9] H. Huijgens, G. Gousios, and A. van Deursen. Pricing via functional size - a case study of a company's portfolio of 77 outsourced projects. In *Intl. Symp. on Empirical Soft. Engg. and Measurement*, pages 1–10, 2015.

[10] B. A. Kitchenham and S. L. Pfleeger. *Guide to Advanced Empirical Software Engineering*, chapter Personal Opinion Surveys, pages 63–92. Springer London, London, 2008.

[11] D. Lo, N. Nagappan, and T. Zimmermann. How practitioners perceive the relevance of software engineering research. In *10th Foundations of Soft. Engg. and European Soft. Engg. Conf.*, pages 415–425, 2015.

[12] A. Meneely, H. Srinivasan, A. Musa, A. R. Tejeda, M. Mokary, and B. Spates. When a patch goes bad: Exploring the properties of vulnerability-contributing commits. In *Intl. Symp. on Empirical Soft. Engg. and Measurement*, pages 65–74, 2013.

[13] A. N. Meyer, T. Fritz, G. C. Murphy, and T. Zimmermann. Software developers' perceptions of productivity. In *22nd Intl Symp. on Foundations of Soft. Engg.*, pages 19–29, 2014.

[14] A. T. Misirli, B. Caglayan, A. Bener, and B. Turhan. A retrospective study of software analytics projects: In-depth interviews with practitioners. *IEEE Software*, 30(5):54–61, 2013.

[15] S. Nanz, S. West, K. S. D. Silveira, and B. Meyer. Benchmarking usability and performance of multicore languages. In *Int. Symp. on Empirical Soft. Engg. and Measurement*, pages 183–192, 2013.

[16] C. D. Nguyen, B. Mendelson, D. Citron, O. Shehory, T. E. J. Vos, and N. Condori-Fernandez. Evaluating the fittest automated testing tools: An industrial case study. In *Intl Symp. on Empirical Soft. Engg. and Measurement*, pages 332–339, 2013.

[17] J. Noll, S. Beecham, and D. Seichter. A qualitative study of open source software development: The open emr project. In *Intl. Symp. on Empirical Soft. Engg. and Measurement*, pages 30–39, 2011.

[18] L. J. Osterweil, C. Ghezzi, J. Kramer, and A. L. Wolf. Determining the impact of software engineering research on practice. *IEEE Computer*, 41(3):39–49, 2008.

[19] F. Palomba, G. Bavota, M. D. Penta, R. Oliveto, and A. D. Lucia. Do they really smell bad? A study on developers' perception of bad code smells. In *30th IEEE Intl. Conf. on Soft. Maint. and Evolution*, pages 101–110, 2014.

[20] M. Piccioni, C. A. Furia, and B. Meyer. An empirical study of api usability. In *Intl. Symp. on Empirical Soft. Engg. and Measurement*, pages 5–14, 2013.

[21] H. D. Rombach, M. Ciolkowski, D. R. Jeffery, O. Laitenberger, F. E. McGarry, and F. Shull. 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, 2008.

[22] B. G. Ryder and M. L. Soffa. Influences on the design of exception handling ACM SIGSOFT project on the impact of software engineering research on programming language design. *ACM SIGSOFT Software Engineering Notes*, 28(4):29–35, 2003.

[23] B. G. Ryder, M. L. Soffa, and M. M. Burnett. The impact of software engineering research on modern programming languages. *ACM Trans. Softw. Eng. Methodol.*, 14(4):431–477, 2005.

[24] T. Zimmermann. Card-sorting: From text to themes. In *Perspectives on Data Science for Software Engineering*. Morgan Kaufmann, 2016.