Zhang L, Tian JH, Jiang J *et al.* Empirical research in software engineering — A literature survey. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 33(5): 876–899 Sept. 2018. DOI 10.1007/s11390-018-1864-x

Empirical Research in Software Engineering — A Literature Survey

Li Zhang[1,2], *Senior Member, CCF*, Jia-Hao Tian[1], *Member, CCF*, Jing Jiang[1,*], *Member, CCF*, Yi-Jun Liu[1,2]
Meng-Yuan Pu[1,2], and Tao Yue[3], *Senior Member, IEEE*

[1] *State Key Laboratory of Software Development Environment, Beihang University, Beijing 100191, China*

[2] *College of Software, Beihang University, Beijing 100191, China*

[3] *Simula Research Laboratory, Martin Lingesvei 25, 1364 Fornebu, Norway*

E-mail: {lily, ttstr, jiangjing, lyj_mika, pumengyuan}@buaa.edu.cn; tao@simula.no

Received March 5, 2018; revised May 14, 2018.

Abstract Empirical research is playing a significant role in software engineering (SE), and it has been applied to evaluate software artifacts and technologies. There have been a great number of empirical research articles published recently. There is also a large research community in empirical software engineering (ESE). In this paper, we identify both the overall landscape and detailed implementations of ESE, and investigate frequently applied empirical methods, targeted research purposes, used data sources, and applied data processing approaches and tools in ESE. The aim is to identify new trends and obtain interesting observations of empirical software engineering across different sub-fields of software engineering. We conduct a mapping study on 538 selected articles from January 2013 to November 2017, with four research questions. We observe that the trend of applying empirical methods in software engineering is continuously increasing and the most commonly applied methods are experiment, case study and survey. Moreover, open source projects are the most frequently used data sources. We also observe that most of researchers have paid attention to the validity and the possibility to replicate their studies. These observations are carefully analyzed and presented as carefully designed diagrams. We also reveal shortcomings and demanded knowledge/strategies in ESE and propose recommendations for researchers.

Keywords empirical software engineering, empirical method, systematic mapping study

1 Introduction

Empirical software engineering (ESE) studies software-related artifacts in order to characterize, understand, evaluate, predict, control, manage and improve them via qualitative and quantitative analyses[1]. ESE methods have been widely applied and well-recognized in software engineering.

With the increasing popularity of empirical methods, ESE has gained wider identification and higher recognition. Universities have started ESE courses, and relevant academic institutions have established special ESE groups such as the Empirical Software Engineering Group of Microsoft Research[①]. In order to promote ESE and improve the quality of ESE research, it is necessary to obtain a comprehensive understanding of ESE, e.g., knowing in which context and for what purposes empirical studies with case studies should be conducted.

Over the past five years, articles published in International Conference of Software Engineering (ICSE), Empirical Software Engineering Journal (EMSE), IEEE Transaction on Software Engineering (TSE), and Symposium on Empirical Software Engineering and Measurement (ESEM) have reported a large number of empirical evaluations[2]. Especially, in ICSE 2016[②], the program committee announced that among all the accepted papers, ESE was the most interesting research

Survey

Special Section on Software Systems 2018

This work was supported by the National Natural Science Foundation of China under Grant Nos. 61672078 and 61732019, and the National Key Research and Development Program of China under Grant No. 2018YFB1004202.

[*] Corresponding Author

[①] https://www.microsoft.com/en-us/research/group/empirical-software-engineering-group-ese/, Aug. 2018.

[②] http://2016.icse.cs.txstate.edu/program/main-conference, July 2018.

area with a total of 32 related papers accepted. This clearly shows that there is an increasing trend of conducting empirical studies in various software engineering fields.

To identify the application of empirical methods in the SE context from those venues, we select a representative journal (EMSE) and a representative conference (ESEM). In fact, it is difficult to define whether a paper is an empirical research paper as there is no widely accepted common understanding on it. EMSE and ESEM are well-established venues particularly focusing on ESE. EMSE was founded by Basili *et al.* in 1996, while ESEM was organized first in 2008. Both of them have considerable numbers of publications from senior ESE researchers every year.

In this paper, we conduct a systematic mapping study on 538 articles to obtain an overview of the application of empirical research methods in the recent five years. Among the 538 selected articles, 260 of them are from ESEM and the other 278 articles are from EMSE, which are all published between January 2013 and November 2017.

The main contributions of our mapping study are as follows.

- We obtain an understanding of where (in which sub-fields) and how empirical research methods have been applied. Results show that experiment, case study and survey are the three most frequently used empirical methods.
- We illustrate the data sources of empirical studies. An interesting finding is that the open source projects have provided data for over half of the research papers we have investigated. Thus, open source projects have proved to be one of the most popular data sources and there is an increasing trend of using open source projects in ESE.
- We observe that the most popular open source platforms are Apache, GitHub, and SourceForge. The most adopted open source projects are Eclipse, Fire Fox, and Linux kernel.
- We identify supporting approaches and tools for data collecting/processing/analysis in ESE. We evaluate researchers' awareness of threats to validity and their considerations on replications of their experiments. Results show that more than 90% of articles have paid attention to threats to validity and experiment replications.

Our research contributions could help researchers in finding supporting approaches/tools and improving the quality of their studies.

This paper is organized as follows. In Section 2 we introduce related work. Section 3 describes the design and method of our study. In Section 4, we present the results and analysis. Section 5 discusses our findings and threats to validity. At last, Section 6 concludes the paper and discusses the future work.

2 Related Work

Empirical research has been studied in various contents. Borgs *et al.*[3] performed a systematic mapping study in 2015 to identify ESE mechanisms including methodologies, tools and guidelines to help and encourage researchers to choose proper mechanisms from the lists they provided. In their study, in total 375 mechanisms were identified.

In 2017, Cosentino *et al.*[4] explored the software engineering researches related to GitHub through a mapping study, attested the high activity of research in the field of the open source collaboration. The authors revealed a set of shortcomings and proposed actions to mitigate them. And in [4], the importance of open source platforms and projects as data sources of conducting ESE research was emphasized.

There are other studies focusing on specific empirical methods such as replications[5], specific topics (e.g., bug reports[6]), and concerning sub-fields of software engineering such as software testing[7]. Bezerra *et al.*[5] conducted a systematic review to extract and synthesize data from reported replications. They found an increasing trend of replications and identified several limitations of the studied replications. Zhang *et al.*[6] presented an exhaustive survey on bug-report analysis. In [7], Zhang *et al.* presented a literature survey on tasks, challenges and future directions of bug resolution in software maintenance process. In 2017, Ahmad *et al.*[8] conducted a systematic mapping study on empirical research in cloud-based software testing.

In our previous work[9], we investigated the publications of 250 EMSE articles from January 2013 to June 2017. Via qualitative and quantitative analyses, we found that the most applied empirical methods were experiment, case study and survey. We studied the involved sub-areas, data sources selection, data processing tools and technologies in ESE. In this paper, we overcome the limitation of extracting data from only EMSE and extend the article inclusion ending date to November 2017. As a result, additional 260 ESEM papers and 18 EMSE papers are taken into account in this paper. To compare with the findings reported in

[9], there are several differences. For instance, the percentage of articles adopting the survey method has increased from 8% to 17.4%). Moreover, the open source platform, GitHub, is more popular than SourceForge, as a data provider, which is however not the case as reported in [9].

3 Method

3.1 Research Questions

To get a better understanding of ESE methods, we identify and address the following four research questions.

*RQ*1. Which research fields/sub-fields of software engineering are most concerned in the selected papers?

ESE is a sub-field of software engineering and it focuses on finding solutions for software engineering problems by using empirical research methods and qualitative/quantitative analysis. It is necessary to find out the most popular topics in ESE, sub-fields and subtopics most relevant to ESE.

*RQ*2. What are the most common ESE methods and types of research purposes? What is the application status of ESE methods in different sub-fields?

This paper aims at identifying the most applied ESE methods and the sub-fields of software engineering in which these empirical methods are adopted. Types of empirical research purposes in software engineering, domain significance and typical scenarios are also addressed.

*RQ*3. What are the characteristics of data sources? What are the commonly applied data collection, data processing and data analysis tools?

The book "Experimentation in Software Engineering"[10] presents different data collection methods and data processing strategies according to different ESE methods. We intend to investigate if data collection and data processing in ESE have been influenced by various factors such as the wide use of the Internet, the rapid development of data mining technologies and the popularity of open source projects hosting platforms and open source communities.

Furthermore, the following questions are expected to be answered. What percentage do open source and industrial projects take respectively among all data sources? What is the trend of using open source/industrial projects as data sources? Which open source projects are the most common choices of researchers? In which software engineering sub-fields are empirical methods more likely to be adopted? How many projects/cases are used in one research? What are the most commonly used mathematical statistical methods and data analysis tools in ESE?

*RQ*4. How concerned are researchers about the validity and the possibility to replicate empirical studies?

This research question concerns researchers' understanding and awareness of the validity and the possibility to replicate ESE studies. First, we present researchers' understanding of the validity of their studies by checking if they explicitly discussed the validity/limitations of their studies. Then, we investigate which types of threats to validity concern researchers the most. Meanwhile, we pay close attention to if they take the "possibility to replicate the study" into account, by checking whether they have provided resources or other relevant information for replications.

3.2 Article Selection

Considering the large number of articles in ESE, it is a challenge to collect and analyze all of them in details. Moreover, our research content is quite broad. Therefore, systematic mapping and scoping[11] are more suitable than systematic literature review. Mapping study is often chosen when aiming at obtaining the general overview of the current technical development or practice level of a research area. As a result, we decide to adopt systematic mapping and scoping to analyze selected articles from EMSE and ESEM. Some of the important characteristics of EMSE and ESEM are discussed below.

- As two important venues in ESE, EMSE and ESEM focus on the application of empirical methods in software engineering, and research contributions of papers published in these two venues cover various subfields of software engineering. These two venues are representative and therefore, they could, to a great extent, reflect the current status of ESE.

They both have enough papers with great quality. EMSE has published papers for 20 years, with six volumes per year, and most of the articles are 30∼40 pages. ESEM has been established since 2007, and more than 600 articles have been published until December 2017. Their impact factors in software engineering domain have significantly increased in last few years. EMSE has been in the Q1/Q2 area of the Journal Citation Report for the last four years.

Articles from other venues are not included. Empirical methods are widely used in research articles in SE field. But it is difficult to clearly tell whether they are empirical research papers. To avoid negative effects on the validity of our research (discussed in Section 5), we exclude research papers from other venues.

To identify the research features and provide an overview of ESE over the last five years, totally 538 papers are selected, i.e., 250 EMSE papers from January 2013 to November 2017 and 260 ESEM papers from 2013 to 2017.

3.3 Data Acquisition

The data acquisition procedure is shown in Fig.1. In the first step, we collect a total of 538 research articles from EMSE and ESEM, excluding the periodical editors' articles, guidelines, with the acknowledgement of EMSE and the introduction papers, post papers of ESEM. A preliminary data summary table is designed according to the research questions, and it is improved iteratively during the data collection process. In the second step, we randomly select sample articles from the 538 primary articles. In the third step, the data summary table is filled out with the information collected from the sample articles. The obtained empirical research data is analyzed in step 4. Step 5 is to modify the data summary table by iteratively refining the process from step 2 to step 5. We obtain the final data summary table once this iterative process is complete. The sixth step is to fill out the data summary table of all of the 538 articles. In the last step, we analyze the collected data using statistical methods, and present our research results.

In the second step, more than 90 sample articles are selected. To extract data from the articles, every article is read and data tables are filled by the authors of this article and volunteers (postgraduate students or Ph.D. candidates studying ESE in Beihang University, Beijing) respectively. The results from participants are carefully compared by discussing differences. Notice that this method is also widely used in social science[12]. Analyses of the collected data are performed carefully by consistently considering the four research questions and ensuring the possibility to replicate our research in the future.

Table 1 describes the structure of our data extraction form. Notice that the values remarked as "□" mean they are for multiple-choice while the values remarked as "○" mean they are for single-choice. As shown in Table 1, software engineering domains are classified into the following 12 sub-fields described in [13]: software requirement (SR), software design (SD), software construction (SC), software testing (ST), software maintenance (SM), software configuration management (SCM), software engineering management (SEM), software engineering process (SEP), software engineering models and methods (SEM&M), software quality (SQ), software engineering professional practice (SEPP), and software engineering economics (SEE). Articles that do not belong to these 12 sub-fields are classified into the "OTHER" type after agreed by all the participants. Therefore, one paper probably covers more than one sub-field. For example, in 2013, Delgado and Martinez[14] identified the prevented defects by unit tests and then performed an analysis on cost and savings. According to SWEBOK 3.0, this article touches upon both software testing and software engineering economics. Keeping in mind the definitions of the sub-fields of SE from SWEBOK 3.0, we try to arrange the selected papers in groups by reading the titles and going through their abstracts.

The empirical methods applied in empirical studies are decided mainly according to authors' declaration

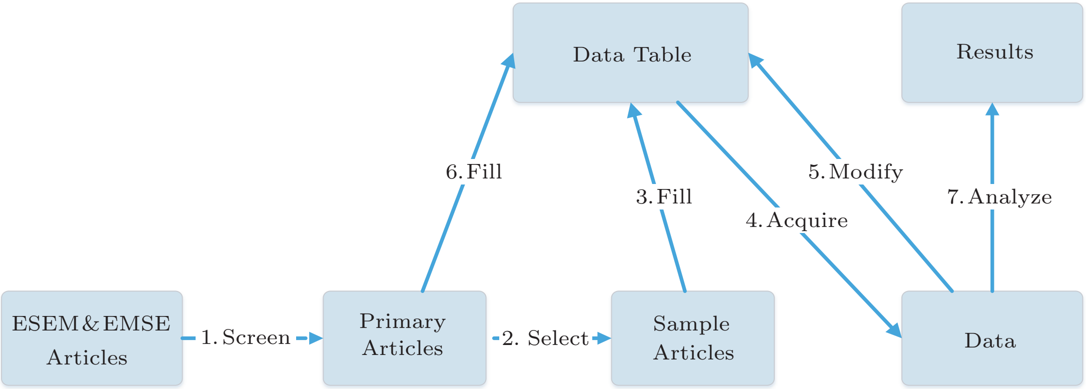


Fig.1. Data extraction table formation process.

Table 1. Data Extraction Form

Item	Value	Remark
Resource	Journal/conference name, publish date	
Title	Title	
Author	Author name	
Sub-field	□ Software requirement □ Software design □ Software construction □ Software testing □ Software maintenance □ Software configuration management □ Software quality □ Software engineering management □ Software engineering process □ Software engineering models and methods □ Software engineering professional practice □ Software engineering economics □ OTHER	Multiple-choice
Topic	Extract topics from abstracts referring to SWEBOK[13]	
Empirical method	□ Controlled experiment □ Quasi-experiment □ Case study □ Replication experiment □ Simulation □ Survey □ Literature review □ Systematic mapping study □ Pilot study □ Systematic literature review □ OTHER	Multiple-choice
Research purpose	□ Exploratory research □ Explanatory research □ Technical validation □ OTHER	Multiple-choice
Data resource	□ Industrial project □ Open source project □ Open test set/data set □ Industrial standard □ Laboratory project □ OTHER	Multiple-choice
Data collection method	□ Questionnaire □ Interview □ Observation □ Archive data □ Preprocess □ OTHER	
Data processing method	Data processing and data analysis methods mentioned in the article	
Project and tool used	Projects, software and mathematics tools mentioned in the article	
Chart and figure used	Charts and figures used for analyses in the article	
Validity	○ No, the article does not contain discussion on threats of validity ○ Yes, the article contains threats of validity (weakness, disadvantage, limitation) ○ Yes, the article conducts analysis about threats of validity, including: □ Construct validity □ Internal validity □ External validity □ Conclusion validity	○ For single-choice □ For multiple-choice
Consideration on possibility to replicate the studies	○ Yes ○ No	Single-choice

and/or the definitions of empirical methods. Typically, almost all the selected papers adopt the following empirical methods: experiment (including quasi-experiment and controlled experiment), case study, survey, simulation, replication, systematic literature review, system mapping study, and pilot study. Methods different from the above ones are classified into the "OTHER" type.

In our previous work[9], three common types of research purposes were defined: explanatory research, exploratory research, and technical validation. In explanatory studies, researchers use data to interpret a phenomenon or result. While the discovery of a phenomenon, problem, or law, is called exploratory research. Technical validation is often used for evaluating technologies including strategies, algorithms, models, methods, tools, etc. If there are other types of research goals, we classify them as "OTHER" and provide explanations.

Data sources are classified into industrial projects, open source projects, open test sets/datasets, industrial benchmarks, laboratory projects, and "OTHER" sources.

The researchers' data collection methods include questionnaire, interview, observation, data archiving, pre-processing, crawler, and "OTHER". In order to understand the empirical research in data processing and analysis, we extract data processing and analysis methods/tools from each article, as well as the analysis of the results such as histograms and pie charts.

A fundamental question concerning results from an experiment is how valid the results are, and this refers to another important aspect to consider in empirical research, namely validity, which helps researchers to measure the study results' trustworthiness. It can be sorted into structural validity, internal validity, external validity, and conclusion validity (reliability)[15].

Possibility to replicate the investigation is an intrinsic and important property of empirical research, and the purpose of a replication is to show that the result from the original experiment is valid for a larger population. A replication becomes a "true" replication if it is possible to replicate both the design and the results[10]. In recent years, researchers in ESE have gradually gained the awareness of its importance. An empirical study that considers the possibility of replications enhances the credibility of the study. The replication package of our primary literature, statistics, figures and charts is available online[3].

4 Analysis and Results

In this section, the collected data is presented, and we answer each research question with extracted data. The acquired data from these articles is normalized and organized into charts and diagrams, so as to get clear results through analyses.

4.1 RQ1. Which Research Fields/Topics of Software Engineering Are Most Concerned in Selected Papers?

4.1.1 *Publications in EMSE and ESEM*

First of all, to describe the general state of EMSE and ESEM publications in the last five years, we collect all the articles by systematically searching the journal website, conference website. Excluding the periodical editors' articles, guidelines, acknowledgement of EMSE, and introduction papers and post papers of ESEM, totally 538 articles are obtained. The number of papers published per year is shown in Fig.2. From Fig.2 we can see that the number of papers published by EMSE has increased significantly since 2013. In 2017, it reaches 69 (the issue published in December 2017 was excluded because our data collection is finished by November 2017), while the number of ESEM papers keeps at 50∼65, except in 2015 when there were only 36 papers. From 2015, the total number of both venues per year presents a smoothly growing trend.

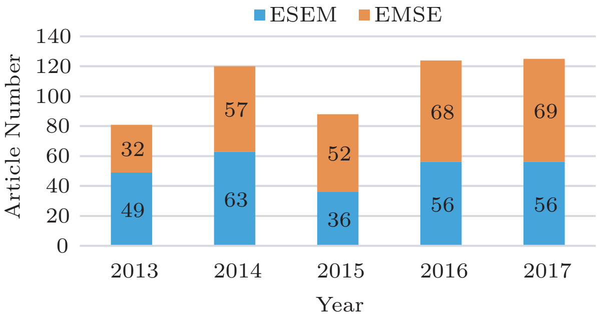


Fig.2. Number of articles published in ESEM and EMSE in Jan. 2013∼Nov. 2017.

4.1.2 *Covered Sub-Fields of SE*

As presented in Section 3, software engineering domains are classified into 12 sub-fields according to SWEBOK 3.0. Any article that does not match the description of the mentioned 12 sub-fields is placed in the "OTHER" category, while an article can cover multiple sub-fields. Therefore we have already taken this factor into account and designed our methodology accordingly. Fig.3 shows the results of our classification.

As can be seen from Fig.3, empirical research papers cover basically all the sub-fields of software engineering. The most popular sub-fields are software maintenance, software quality, and software testing. In the first place,

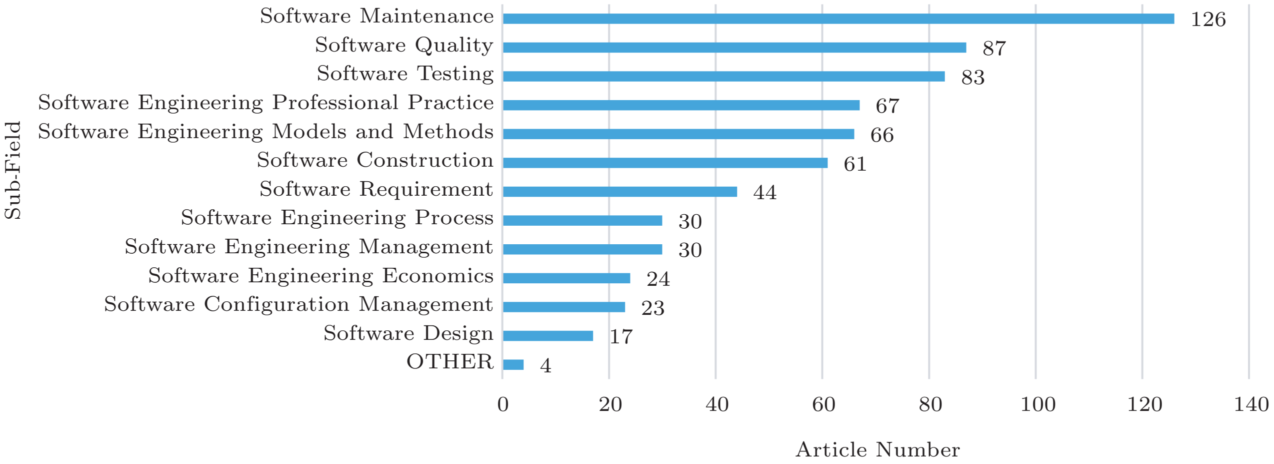


Fig.3. Article numbers in different sub-fields of SE.

[3] http://t.beihangsoft.cn/lily/ch/index.html, July 2018.

empirical research methods are mostly applied in software maintenance, with 126 articles (23.4%). This can be the result of the two EMSE special issues of program comprehension and reverse engineering. Software quality and software testing have 87 (16.2%) and 83 (15.4%) articles, respectively. Articles in software engineering professional practice, software engineering models and methods, and software construction are also considerable. The number of articles in software engineering professional practice is 67 (12.5%), while software models and methods have 66 (12.3%) articles and the number of articles in software construction is 61 (11.3%). Article numbers in these three sub-fields are also considerable. There are 30 (5.6%) articles in software engineering process, 30 (5.6%) articles in software engineering management, 24 (4.5%) articles in software engineering economics, 23 (4.3%) articles in software configuration management, and 17 (3.2%) articles in software design. In the five sub-fields above, the article numbers are relatively smaller than those of the sub-fields discussed before. There are only four articles in the "OTHER" category since they do not match the description of any of the 12 sub-fields.

Summary. In this subsection, we review the overall publications of EMSE and ESEM in the recent five years. Empirical studies in these research papers cover all software engineering sub-fields, and empirical research methods are most applied in software maintenance, software quality, and software testing.

4.2 RQ2. What Are the Most Frequently Used ESE Methods and Types of Research Purposes?

4.2.1 Empirical Research Methods in Software Engineering

Table 2 presents the number of articles adopting each empirical method. Among the 538 articles, nine empirical methods are used, as shown in Table 2. Experiment (quasi-experiment and controlled experiment) has been used in 264 (49.1%) articles. There are 168 articles adopting controlled experiment among the 264 articles, and the other articles use quasi-experiment, which is often used when it is impossible to randomly assign treatments to subjects[16].

The adoption of case study ranks the second, and it is used in 185 (34.4%) articles. Case study is an empirical method for investigating phenomena in real-world environment[17]. However, the distinction between case study and experiment is not precisely defined. We mainly rely on authors' claims to distinguish these two methods in our mapping study. However, we shall do the classification not only according to the authors' declarations, but also by evaluating the methodology in each article by ourselves. When there is no explicit indication of which of the two research methods is used, one principle we follow is that if there are artificial control variables in the research, it is classified as an experiment. For example, Haller *et al.*[18] presented a tool for detecting and classifying advanced data structures used in binary files, and evaluated the accuracy of the tool in 10 real-world applications. The authors claimed that they used case studies to perform research in 10 cases. But according to our judgment, the adopted empirical method is experiment.

Survey is adopted in 94 (17.5%) articles. Survey is a method to collect and summarize evidence from a large representative sample of the overall population of interest[19]. In recent years, there has been increasing emphasis on human aspects in software engineering research and practices[20]. The common forms of survey method are questionnaire, interview, etc.[10] As an independent and complete method, survey should include research goal, design, implementation, data analysis, conclusion formation, and validity analysis.

Questionnaire is also used together with other research methods. However, if it is only used for data collection without reaching a conclusion, it will not be

Table 2. Empirical Methods Used in the Selected Articles

No.	Method	Article Number	Primary/Secondary	Remark
1	Experiment	264	Primary	168 adopt controlled experiment, the others adopt quasi-experiment
2	Case study	185	Primary	
3	Survey	94	Primary	
4	Literature review methodology	47	Secondary	29 adopt systematic literature review, 18 for systematic mapping study
5	Replication experiment	18	Secondary	
6	Pilot study	20	Primary	
7	Simulation	5	Primary	
8	OTHER	1		Action research

considered as a research method. Instead it is just regarded as a data collecting approach. This will be discussed in Subsection 4.3.2.

The literature review methodology is a kind of secondary study[21] based on the experience of publications. Literature review has been adopted in 47 (8.7%) articles. Among them, 29 use systematic literature review (SLR for short) while 18 use systematic mapping.

Replications of experiments[22] appear in 18 articles, and five of them are included in the replication experiment special issue of EMSE in April 2014. Pilot study is a small-scale study in real-world environment[23], and it is used in 20 selected articles. For example, in 2017, Wang[24] conducted a pilot study to test the study design and environment in order to ensure the effectiveness and feasibility before conducting a large-scale investigation. In 2015, Octaviano *et al.*[25] proposed strategy of "score citation automatic selection (SCAS)", and a small-scale pilot study was then used to evaluate the accuracy and error of the strategy. There are five articles adopting the simulation experiment method, which is often used when it is too difficult to conduct an experiment.

We find that in addition to the empirical methods listed in Table 1, there is another empirical method, namely action research, with which researchers carry out studies as participants in real-world projects[26].

According to Table 2, in the original studies, the most commonly adopted research methods are experiment, case study, and survey, and these three methods appear in more than 95% of the articles. Despite the number of articles using the survey method is relatively low, questionnaire and interview are used in 187 articles as data collection methods. Data collection methods will be discussed in Subsection 4.3.2. We consider experiment, case study and survey very important, and recommend that researchers should master these three research methods. Fig.4 illustrates the percentage of articles adopting experiment, case study and survey from 2013 to 2017. According to Fig.4, articles using experiment and survey have been increasing, while the adoption trend of case study falls slightly.

4.2.2 *Features of Empirical Methods Adoption in Different Sub-Fields*

The adoption of various empirical research methods in different sub-fields has been analyzed, and the result is shown in Table 3. Experiment and case study are applied in all the sub-fields of SE.

Experiment is used mostly in software maintenance, software quality, software construction, and software requirement. One of the reasons for abundant use of experimentation in these sub-fields is that it is easy for researchers to control variables and get access to historical data in order to evaluate the validity of new approaches or technologies. For example, Shin and Williams[27] performed experiments to assess how the error prediction method worked on predicting vulnerability with open source browser Firefox data. In 2013, Raja[28] applied a text mining technique to the defect report of open source software to study the quantity of the defect, and the result was compared with that of manual statistics. Experiment is the most frequently used method in the software requirement sub-field. In software requirement, evaluation is often subjective. Therefore, for some experiments, questionnaires or interviews are often needed as supplement. For example, Özlem and Carver[29] examined the impact of individual factors on requirements inspections by providing the participants with experimental materials and allowing them to find errors in the required documents and record them in the error list.

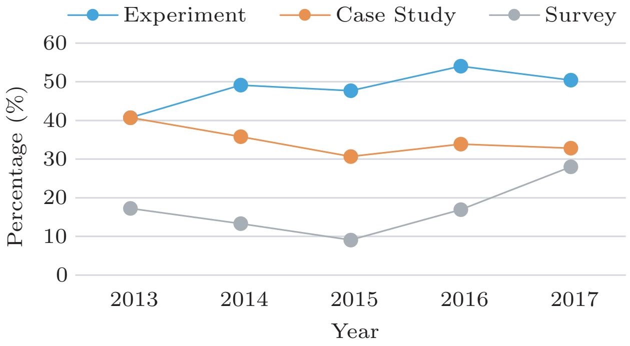


Fig.4. Percentage and trend of articles adopting experiment, case study and survey respectively.

Case study is adopted more frequently than the other empirical methods in software engineering process, software engineering management, software engineering professional practice, and software engineering economics. Researches in these fields are normally impacted by multiple factors that are difficult to control.

Difficulties in simulation, long time cycles and high cost may make experiment unsuitable in software engineering process, software engineering management, software professional practice, and software economics. In these sub-fields, case study is more appropriate to be used because it extends the controlling on multiple factors. For example, Estler *et al.*[30] conducted a case study in 2014. In their case study they collected data

Table 3. Applied Times of Empirical Methods in Different Sub-Fields of Software Engineering

Sub-Field	Experiment	Case	Survey	Literature	Replication	Pilot	Simulation	OTHER
		Study		Review		Study	Experiment	
Software requirement	30	10	8	2	2	3	2	0
Software design	12	4	1	4	1	0	1	0
Software construction	32	25	9	3	3	1	0	0
Software testing	36	25	10	3	2	4	0	0
Software maintenance	77	41	9	5	2	2	.3	0
Software configuration management	9	14	3	0	1	0	0	0
Software engineering management	15	15	8	4	4	0	0	0
Software engineering process	10	14	7	2	0	0	0	0
Software engineering models and methods	36	14	10	11	3	5	1	0
Software quality	48	33	6	3	3	0	1	1
Software engineering professional practice	15	26	32	3	2	5	0	1
Software engineering economics	12	9	2	1	1	1	0	0
OTHER	2	1	0	1	0	1	0	0

of 66 real-world global software projects via questionnaires and interviews, in order to discover the difference between projects developed by agile flows (Scrum, XP, etc.) and projects developed by structured flows. In this case, the experiment method was not selected because it was difficult to control the other factors except those in software engineering process. As mentioned above, experiment is seldom used in software engineering process with few exceptions. For instance, in the research of Chen *et al.*[31], a new semi-automatic software process evaluation approach using machine-learning technologies was proposed. In order to verify the validity of this approach, a contrast experiment was conducted in nine real-world industrial projects to evaluate the implementation of the defect management process, and the authors compared the results with existing approaches.

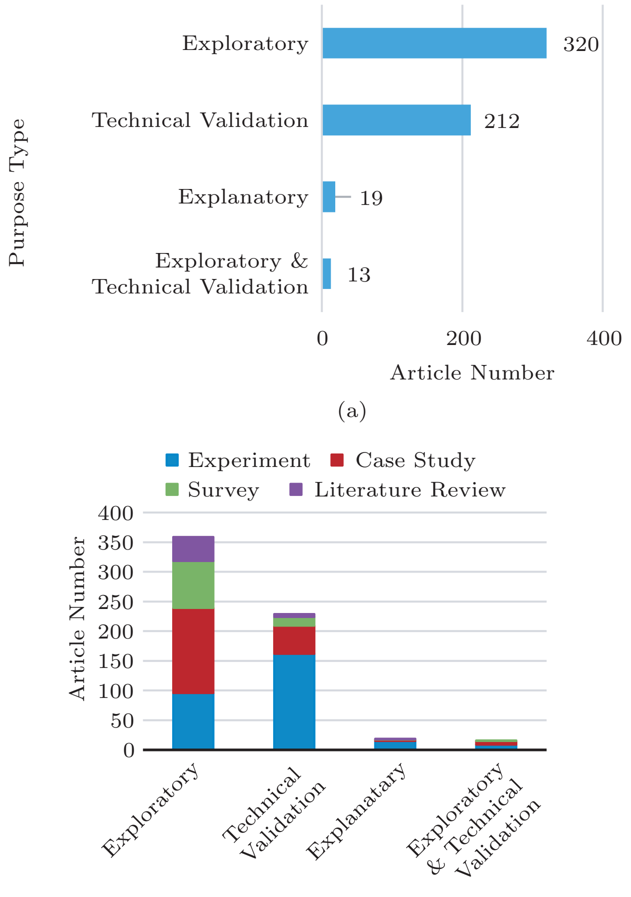
The number of articles using the survey method is smaller than those using experiment and case study. Survey is more widely used in software construction and software maintenance. It is often used to explore the qualitative study of a topic, and to quickly understand participants' perspectives on specific topics. Results are also influenced by many factors such as the choice of participants. For example, in 2016 Chen *et al.*[32] conducted a survey to investigate viewpoints of the participants on refactoring. The result showed that the participants had various viewpoints on refactoring and issues expected to be solved in the future were presented.

There is no obvious field significance in the application of simulation experiments and pilot studies. Simulation experiment is used in software requirements, software maintenance, and software quality. Pilot study is often used in software construction, software testing, and software maintenance. For instance, Unterkalmsteiner *et al.*[33] studied the selection of test cases and evaluated the performance of different information through experiments retrieval methods. Before their experiments, a pilot study was conducted to confirm whether the information retrieval method could be used in the choice of test cases.

4.2.3 *Types of Purpose/Paradigms of ESE Studies*

There are three types of research paradigms: exploratory, explanatory, and technical validation. Fig.5(a) presents the distribution of the articles for different types of research purposes. According to Fig.4, 320 (59.3%) articles are exploratory. This type of purpose is to identify problems or discover possible patterns by observing phenomena based on evidence or data; 212 (39.4%) of the articles belong to technical validation, which can be used to evaluate new strategies, algorithms, models, methods, tools and so on; 19 articles are explanatory, such as studying causal relationships existing in software engineering. In addition, in 13 articles, exploratory studies were conducted before the technical validations.

Fig.5(b) presents the numbers of empirical methods (experiment, case study, survey, and literature review) adopted in researches with different types of purposes. As shown in Fig.5(b), case study is the most frequently used empirical method in exploratory researches. For example, Capiluppi and Izquierdo-Cortázar[34] explored the relationship between the developers' workload and the concentration of development time from the Linux open source case. In technical validation, the experiment method is most frequently adopted. It is suitable for technical validation of new methods. Explanatory study is usually used to explain the causal relationship between phenomenon and laws, such as [29] and [35]. There are 10 articles in which the authors conducted exploratory researches to identify problems and then proposed approaches as solutions before performing technical validations, such as [36] and [37].



(a)

(b)

Fig.5. Research purpose types of articles. (a) Numbers of articles for different research purpose types. (b) Empirical methods in articles of different purpose types.

In empirical research, researchers sometimes combine multiple empirical methods to achieve different research purposes. The examples are as follows.

- There are five articles adopting both literature review and survey. For example, in 2014, Smite *et al.*[38] firstly used the systematic literature review to explore the problems encountered in the new terminology of global software engineering, such as the ambiguity of new terminology classification. Then the authors solved the problems through a survey, namely conducting interviews to global software experts.

- In 19 articles, survey and case study are used together. For example, in 2013, Greiler and Deursen[39] firstly figured out the problem of the test suite plugin in the architecture design through a survey, and then proposed a method to solve the problem. At last the authors evaluated the validity of the proposed method through a case study.

- There are eight articles using case studies and experiments together. For instance, in the research of Callaú *et al.*[40], a case study of a large Smalltalk code base was carried out to find out the problems in dynamic feature classification. Then an automatic dynamic classification approach was proposed and was then validated by an experiment.

- The only article combining literature review, experiment and case study is the article of Cheung *et al.*[41] In this article, firstly, the authors used literature review to explore the limitations of the cloning web page detection technology and relevant tools. Then they used case study to collect data from Google, Yahoo, and Twitter and solve the problems they encountered in cloning web pages. At last, a cloned web page detecting tool was presented, and the authors used an experiment to evaluate the effectiveness of the tool. Results showed that the proposed tool provided higher accuracy than tools proposed by other researchers.

4.2.4 Research Purpose Types in Different Sub-Fields

For further acknowledgement about research purposes of empirical research in different sub-fields, we collect related statistics of researches in each sub-field, and the results are shown in Table 4. In most of the sub-fields of SE, articles with the exploratory research purpose take the largest percentage. However, in software maintenance, software testing and software design, articles aiming at technical validation are more than exploratory articles. In software maintenance and software quality, the number of articles aiming at explaining the cause or effect of some statues is larger than that of any other sub-fields.

4.2.5 Summary

The top three frequently used empirical methods in software engineering researches are experiment, case study, and survey.

In all software engineering sub-fields, experiment and case study are used. Experiment is the most popular in software maintenance, software quality, software construction, and software requirements. Case study is used more than other empirical methods in software engineering process, software engineering management, and software professional practice.

Table 4. Research Purpose Types in the Sub-Fields of Software Engineering

Sub-Fields/Research Purpose Type	Exploratory	Technical Validation	Explanatory
Software requirement	23	18	2
Software design	8	9	0
Software construction	44	18	1
Software testing	37	46	4
Software maintenance	58	67	4
Software configuration management	19	5	0
Software engineering management	28	8	2
Software engineering process	20	9	1
Software engineering models and methods	36	28	0
Software quality	48	35	8
Software engineering professional practice	54	11	2
Software engineering economics	17	7	7
OTHER	2	1	1

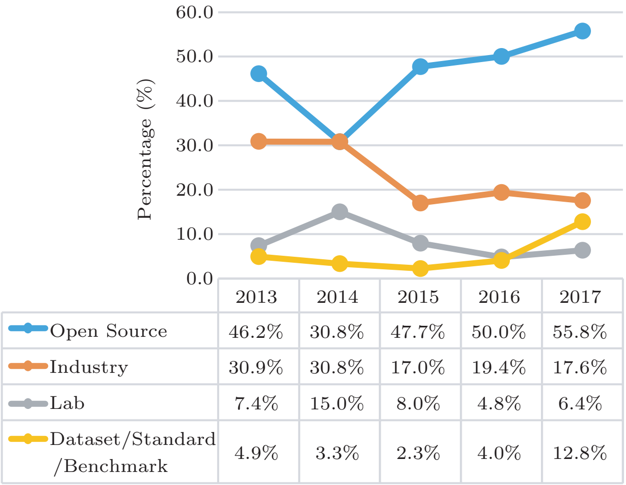
The most common empirical research purpose is exploratory, which is closely followed by technical validation researches in the second place. Most of the experiments are used for technical validation, while most of the case studies are used for exploratory purposes.

4.3 RQ3. What Are the Characteristics of Data Sources? What Are the Commonly Applied Data Collection, Data Processing and Data Analysis Methods/Tools?

4.3.1 Data Sources of Empirical Research

Data sources of literature reviews are published papers, and data of survey is often gathered by interview or questionnaire, which will be discussed in details in Subsection 4.3.2. In this subsection, we focus on all the primary data sources, while excluding the secondary data of 47 literature reviews. We find that out of the 538 articles, 489 articles' data sources consist of open source projects, industrial projects, lab projects, and standards/datasets/test-suits/benchmarks. The result is depicted in Fig.6.

An article may have multiple data sources, such as [42], in which the researchers used data from both open source and industrial projects. According to our statistics, 53.4% of the articles use open source projects data for empirical research, including open source software code, process data, and data from open source communities, etc.



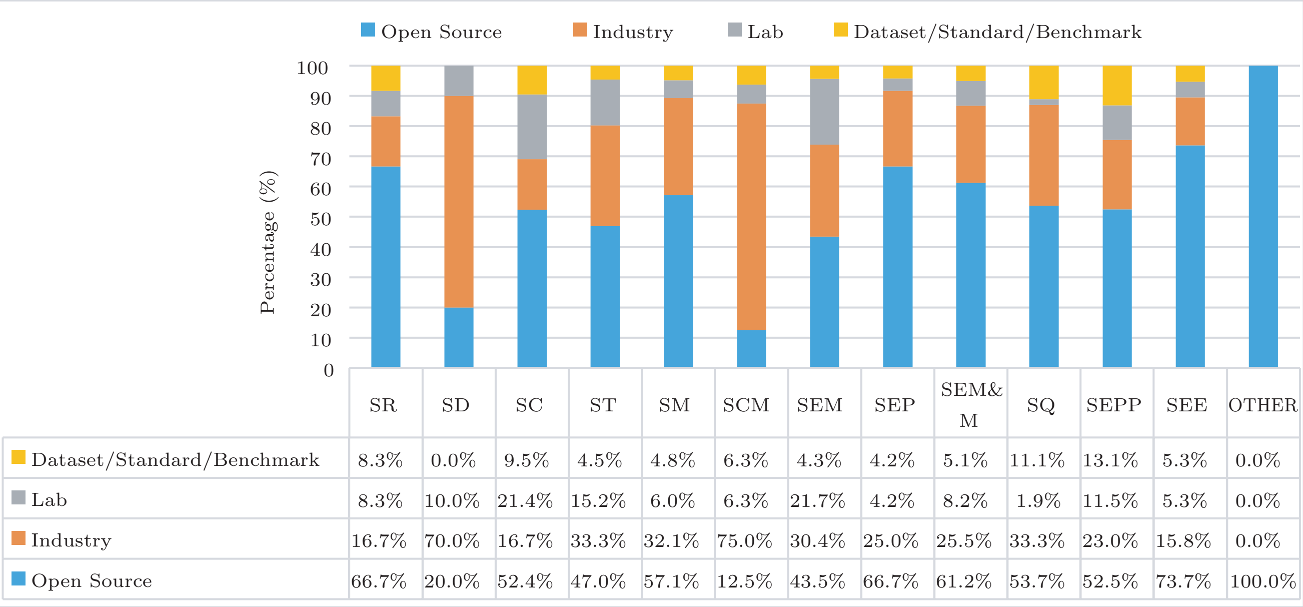
	2013	2014	2015	2016	2017
Open Source	46.2%	30.8%	47.7%	50.0%	55.8%
Industry	30.9%	30.8%	17.0%	19.4%	17.6%
Lab	7.4%	15.0%	8.0%	4.8%	6.4%
Dataset/Standard /Benchmark	4.9%	3.3%	2.3%	4.0%	12.8%

Fig.6. Percentage and trend of data sources used in empirical research.

Fig.6 presents the trend of different data sources in empirical research in the selected articles from 2013 to 2017. In 2015 and 2016, the percentage of papers using open source projects kept around 50%, while in 2017 it rose up to 56%. Generally, there is an obvious rising trend of using open source projects as data sources from 2014.

Among all the surveyed articles, 28.7% of them use industrial projects and the percentage is gradually decreasing. There are 10.6% of articles using laboratory projects. There are also 7.2% of studies using standard datasets or open-test sets/benchmarks, which reveals the lack of present standard datasets and open test sets/benchmarks.

We also conduct analyses across sub-fields and project sources. Fig.7 shows the percentage of the four kinds of data sources from the articles in each sub-field. It can be perceived that open source projects are adopted more than any other kind of data source in articles of the following sub-fields: software requirement, software construction, software testing, software maintenance, software engineering process, software quality, software engineering professional practice, and software engineering economics. In the sub-fields of software design and software configuration management, industrial projects are mostly used. While in the sub-fields of software testing and software engineering management, the percentages of articles using industrial projects and open source projects are close. Thus, the researches focusing on costs, processes, and new development models or methodologies are more likely to use industrial projects as data source. Meanwhile in software construction, software maintenance and software configuration management, it is easier for researchers to obtain



	SR	SD	SC	ST	SM	SCM	SEM	SEP	SEM&M	SQ	SEPP	SEE	OTHER
Dataset/Standard/Benchmark	8.3%	0.0%	9.5%	4.5%	4.8%	6.3%	4.3%	4.2%	5.1%	11.1%	13.1%	5.3%	0.0%
Lab	8.3%	10.0%	21.4%	15.2%	6.0%	6.3%	21.7%	4.2%	8.2%	1.9%	11.5%	5.3%	0.0%
Industry	16.7%	70.0%	16.7%	33.3%	32.1%	75.0%	30.4%	25.0%	25.5%	33.3%	23.0%	15.8%	0.0%
Open Source	66.7%	20.0%	52.4%	47.0%	57.1%	12.5%	43.5%	66.7%	61.2%	53.7%	52.5%	73.7%	100.0%

Fig.7. Data sources of empirical studies in different sub-fields.

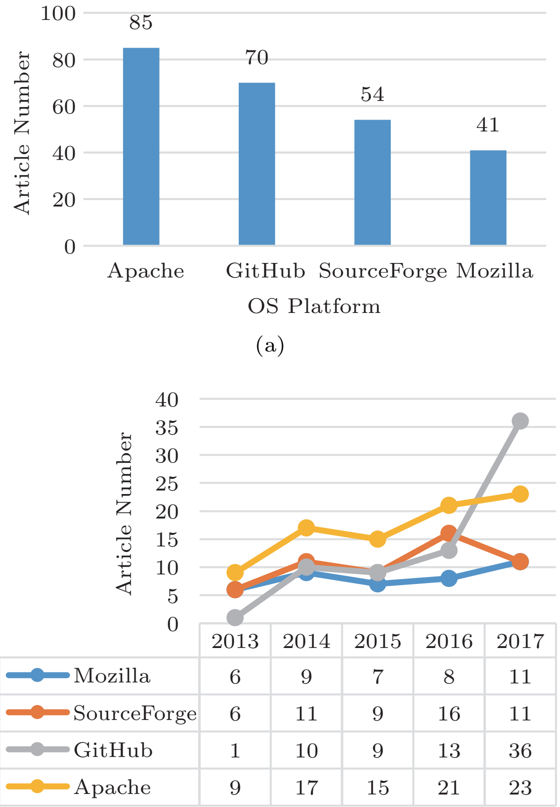
data from open source projects. Open source software is geographically and timely dispersed. Open source projects are often developed by contributors with different backgrounds. Software construction and software maintenance are currently the sub-fields mostly related to open source projects, in which there are many open source projects as proper research materials. Test-related data is available in both open source projects and industrial projects. Choosing a lab project as a data source is less frequent in ESE, because it is difficult to simulate the complex software engineering problems, and it is hard for other researchers to conduct replication studies. We also see that standard datasets/test sets/benchmarks are even fewer, indicating that available standard datasets for research in software engineering are insufficient. In general, researchers from most software sub-fields tend to use open source projects. Specially, there tends to be more lab projects and standard datasets/test-sets in the software requirement sub-field than the other sub-fields.

1) *Most Frequently Used Open Source Projects in Empirical Research.* From the analysis in previous subsections, we find that open source projects are used in almost all software engineering sub-fields. With the rapid development of open source platforms and open source communities, some famous open source software products such as Linux, Apache Web are even more widely used than the industrial ones of the same kind. Meanwhile, due to the easier access to open source software and the convenience to conduct replicated researches, more and more researchers begin to use them. In order to facilitate a better use of open source software in ESE, further analysis on the use of open source software in all 489 primary research articles is conducted.

Fig.8(a) lists the top four open source platforms providing project data for articles, including Apache, GitHub, SourceForge, and Mozilla. As shown in Fig.8(a), there are 85 articles using data from Apache projects. Apache projects belong to Apache Software Foundation (ASF)[4]. ASF is a non-profit organization that provides an open source for Apache projects. HTTP Server, IBatis, Tomcat, Wicket, Maven2, and Ant, are often used for experimental research in Apache projects, such as Derby, Xerces, httpd, Hadoop, $literal, Struts, Lucene, Axis2_c, Pluto, SOLR, Joda-time, Hibernate, Cocoon, JMeter, and cpptasks. Projects from the open source hosted platform GitHub Systems, such as MDG, JBidwatcher, dnsjava, Closure Library, and SproutCore, have provided research materials for 70 articles. Fifty-four articles use open source projects from SourceForge, which is an open source software platform and warehouse. For example, in 2013, Raja[28] used text clustering in defect reports of FileZilla, jEdit, PHP, MyAdmin, pidgin and Slash from SourceForge to predict the defect resolution time. In 2015 Arcuri and Fraser[43] randomly extracted 100 Java projects from

[4] https://www.apache.org/, July 2018.

SourceForge to study the parameter value setting problem of a searching algorithm. In addition, projects from the Mozilla Nonprofit Foundation are used in 41 articles, including Firefox, Firefox Extensions Firebug and Rhino. It is visible from Fig.8(b) that there has been an obviously increasing trend of using GitHub in recent years, while articles using Apache and Mozila are also increasing.



(a)

	2013	2014	2015	2016	2017
Mozilla	6	9	7	8	11
SourceForge	6	11	9	16	11
GitHub	1	10	9	13	36
Apache	9	17	15	21	23

(b)

Fig.8. Use of open source platforms. (a) Most used open source (OS) platforms in the 489 articles. (b) Trend of using Apache, GitHub, SourceForge and Mozilla.

The most frequently used open source projects are identified through further statistics. Fig.9 lists the top 10 frequently used open source projects. They are Eclipse, jEdit, Linux kernel, JHotDraw, Lucene, ArgoUML, Firefox, Tomcat, PostgreSQL, httpd and Rhino. There are 57 articles using Eclipse project data, such as Eclipse source code, version information, bug reports[44], and open source plug-ins on Eclipse. jEdit is a text editor developed in Java language. For example, in article [45], a data fusion model for feature positioning was proposed using jEdit evaluation. Linux kernel is a computer operating system kernel written in C language. The Linux kernel method, kernel source code, and version information can be used as the data source of empirical research. JHotDraw is a two-dimensional GUI framework. In 2014, Bavota *et al.*[46] used JHotDraw data to evaluate an automated class refactoring approach. Lucene is a full-text search engine toolkit. ArgoUML is a UML model tool. Firefox is a browser for Windows, Linux, and OS X platforms. Tomcat is a web application server, PostgreSQL is a relational database management system, and httpd is Apache Hypertext Transfer Protocol (HTTP). Rhino is a 3D modeling software. These commonly used open source projects have some advantages in common: complete documents, continuously updated version, wide coverage of users, and high code quality.

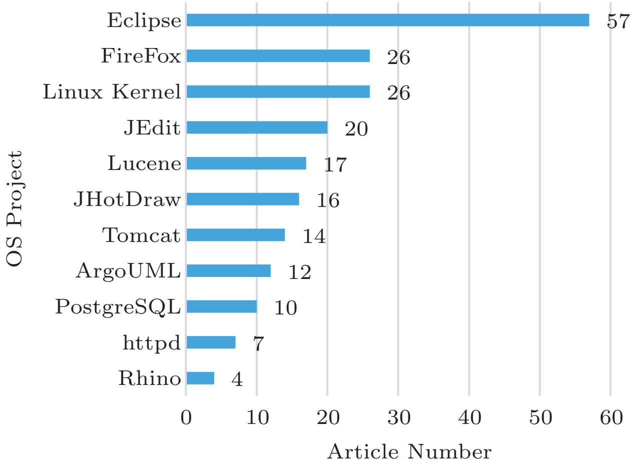


Fig.9. Most frequently used open source projects in the 489 articles.

2) *Number of Studied Projects/Cases in a Single Empirical Research.* Both researchers and practitioners are concerned with the proper number of projects for an empirical research, which is also one of the most controversial topics in this field. In this paper, we count the number of projects used in researches for different research purposes, and results are shown in Table 5. The project number refers to the number of projects used in each article. We have also organized the distribution of projects according to their use in various sub-fields of SE. We tabulate our results in Table 6.

Table 5. Number of Projects Used in an Article of Different Purpose Types

	Maximum	Minimum	Median
Exploratory	7 365	1	3.0
Technical validation	3 469	1	3.5
Explanatory	21	1	3.5
Exploratory & technical validation	900	1	6.0
Total	7 365	1	3.0

Table 6. Descriptive Statistics of the Numbers of Projects Used Across Different Sub-Fields

Sub-Field	Maximum	Minimum	Median
Software requirement	2 117	1	14.0
Software design	9	1	2.0
Software construction	1 160	1	7.5
Software testing	3 286	1	3.0
Software maintenance	7 365	1	3.0
Software configuration management	5	1	1.0
Software engineering management	20	1	3.0
Software engineering process	1 547	1	4.0
Software engineering models and methods	2 217	1	3.0
Software quality	7 365	1	4.5
Software engineering professional practice	1 385	1	3.0
Software engineering economics	51	1	3.0
OTHER	29	3	5.0

In some of the articles, a large number of projects were studied. For example, Zhu *et al.*[47] collected 832 408 repositories from GitHub in 2012, which include the meta-data describing attributes. In [48], 17 877 apps mined from the BlackBerry and Google app stores in 2014 were used to evaluate their proposed similarity measure technique. Mcilroy *et al.*[49] used 10 713 projects of GooglePlay to find the update frequency of those mobile applications, so as to provide useful advices for developers in 2016. In the technical validation of Allix *et al.*[50], the authors selected 52 000 Android apps from Google Play to verify their detection methods they proposed to solve the problem of malware detection. There are another three articles which adopted over 10 000 projects. In these articles, too many projects are selected, which could greatly affect the average number. In order to make the statistical results reflect the real situation, the maximum (the largest number of projects used in one article), the minimum (the smallest number of projects used in one article), and the median of project numbers used in each article are calculated without average.

The numbers of projects in various researches are presented in two perspectives, namely type of research paradigm and research sub-field, as shown in Table 5 and Table 6.

- *Type of Research Paradigms and Case/Project Quantity.* We can see from Table 5 that the median of the number of projects used in a technical validation or exploratory study is literally higher. To verify a new method, or to find a result from a phenomenon, researchers usually need multiple projects to perform their researches. The researches that perform interpretative studies are designed to verify the causal relationship between phenomena and laws, and the number of chosen projects will be less. The medians vary in [3, 6]. The variances of exploratory studies and technical validations are greater.

From further analysis, we find that, of all the 538 articles, there are only 14 articles (2.7%) using more than 1 000 projects, six of which are for technical validation. For example, Fraser and Arcuri[51] studied 1 385 open source projects from SourceForge and Googlecode to validate the general defect prediction model they proposed. Eight articles are exploratory studies. For instance, Vasilescu *et al.*[52] explored the evolution of the ecosystem contributors' workload and involvement in projects and activities by studying 1 316 GNOME projects. There are 80 articles using one project, accounting for 15.7% of articles. Among these 80 articles, 42.5% of them are for technical validation, and 50% of them are for exploratory studies.

- *Sub-Field and Case/Project Number.* From Table 6, we can see that there is a big difference between the maximum value and the minimum value in some sub-fields. The result shows that in most sub-fields, there is a maximum of more than 1 000 projects, while in software design, software configuration management, software engineering management, and software engineering economics, the maximums are relatively smaller. This may result from the difficulty to get the data that researchers need in these three sub-fields.

Median is representative since it is not affected so much by the maximums and the minimums. After analyzing the medians of project number in each sub-field, we find that first, the median of project number in software requirement is 14, and then software construction has a median of 7.5. Up next are software quality, software engineering process, software engineering economics, and the medians in these sub-fields are between 3 and 5. As shown in Fig.6, articles in these sub-fields use open source projects very frequently, which makes the medians larger than those of other sub-fields. Software configuration management and software design have the lowest medians, while over half of the articles in each of the two areas choose to use one single project. This probably indicates that in these two sub-fields, using only one project in research can also be basically approved by peer researchers. It may be difficult to obtain the required project data in these two sub-fields.

4.3.2 Methods for Data Collection

We review all the data collection tools or methods demonstrated in the articles. The result is shown in Table 7.

The result shows that in primary studies, data archiving is the most frequently used data collection method, which is adopted in 164 articles. Then questionnaire is used in 148 articles and up next is interview, which is used in 108 papers. Questionnaire and interview are the most commonly used data collection methods in surveys, and they are also frequently used in experiments and case studies. Questionnaire and interview are combined in 23 papers. In 46.4% of the articles, researchers collect data or information from people, such as article [53].

Questionnaire is used in articles from all the sub-fields as a flexible method. Most common forms of online questionnaire include mail-based questionnaire on website. Interview is often used to collect data in surveys and case studies. It can be classified into structured, semi-structured and unstructured interview[1]. Structured interview and survey both have clear questions. Interview in general is more flexible and accessible than questionnaire, and meanwhile more time consuming.

Data archiving is used in experiments, case studies and surveys, and it is a very frequently used data collecting method in experiments. Data archiving refers to, for example, minutes of meeting, documents from different development phases, failure data, organizational charts, financial records, and other previously collected measurements in an organization.

There are 17 articles adopting crawlers to collect data and four articles using observations. For instance, Kosti *et al.*[54] conducted a behavioral research on participants after obtaining their personalities and project preference by observation.

Table 8 shows the number of articles which use different collection methods in various sub-fields. It is evident in Table 8 that interview is the most frequently used in software quality, software maintenance, and software construction.

Questionnaire is mostly used in articles of software engineering models and methods, software testing, software quality, and software maintenance. In sub-fields where human is treated as research objects, questionnaire and interview are more likely to be adopted. Data archiving is most frequently used in software quality and software maintenance.

4.3.3 Methods and Tools for Data Processing and Analysis

In this subsection, we illustrate the methods and tools used in data processing and analysis. Firstly, we present the mostly used figures and charts. Trends or characteristics can be easily revealed through them. Then, popular statistical tests are listed. At last, the advantages and applying conditions of adopted supporting tools for data analysis are compared.

1) *Usage of Figures and Charts.* We find almost all articles using visual aids to describe their results. This makes the results clearer. Specific uses of different kinds of figures and charts are shown in Fig.10. Tables are mostly used in almost all articles. Bar charts, boxplots, line charts, scatter diagrams and histograms are also frequently used. For this reason, we think these charts are analytical tools that empirical researcher should master.

2) *Usage of Mathematical Statistics.* Empirical studies always produce a large amount of data, which makes data analysis an indispensable link. Various statistical methods are available during data analysis. Table 9 shows the mathematical statistics methods used in these articles. Wilcoxon test, t-test, Mann-Whitney test, variance analysis (ANOVA), and Kruskal-Wallis test are the most commonly used statistical tests. As we can see, they all belong to hypothesis test. Researchers should master these methods in order to select an ap-

Table 7. Number of Articles Using Different Data Collection Methods While Using Different Empirical Methods

Empirical Method	Questionnaire	Interview	Data Archiving	Crawler	Observation
Experiment	62	15	87	6	1
Case study	31	46	56	9	2
Survey	46	42	12	2	1
Pilot study	8	5	4	0	0
Simulation	1	0	4	0	0
OTHER	0	0	1	0	0
Total	148	108	164	17	4

Table 8. Number of Articles Using Different Data Collection Methods in Each Research Sub-Field

Sub-Field	Interview	Questionnaire	Data Archiving	Crawler	Observation
Software requirement	5	10	10	2	1
Software design	5	5	4	0	0
Software construction	17	14	19	5	0
Software testing	14	24	15	0	0
Software maintenance	17	20	51	2	1
Software configuration management	9	7	10	0	0
Software engineering management	7	15	9	0	0
Software engineering process	6	7	9	1	0
Software engineering models and methods	16	27	23	3	0
Software quality	20	24	29	0	0
Software engineering professional practice	5	12	13	1	1
Software engineering economics	6	6	7	1	0
OTHER	0	0	1	0	0

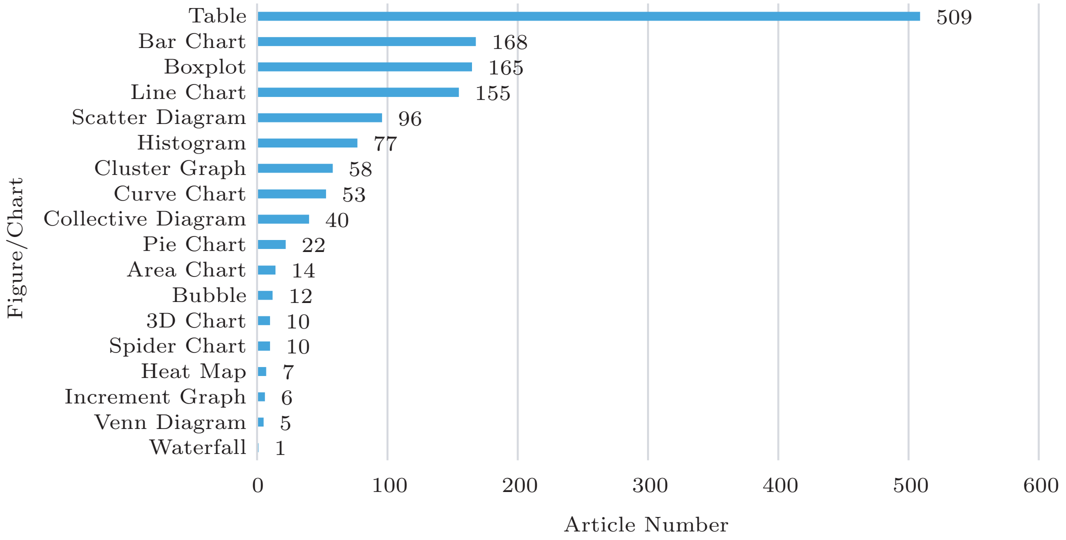


Fig.10. Figures and charts used in empirical studies.

Table 9. Statistical Tests Used in Empirical Research

Sample Number	Sample Capacity	Statistical Test	Article Number
1	Large (normal distribution)	Chi-square test	17
1 or 2	Large (normal distribution)	t-test	117
2	Large (normal distribution)	F test	27
		paired t-test	8
	Small (not normal distribution)	Wilcoxon test	143
		Mann-Whitney test	78
	No restriction	Sign test	12
2 or more	Large (normal distribution)	ANOVA	72
	Small (not normal distribution)	Kruskal-Wallis test	35

propriate one for statistical analysis during the study. Statistical hypothesis[5] is a testable hypothesis on the basis of observing a process that is modeled via a set of random variables. It is a method of statistical inference. Commonly, two statistical datasets are compared, or a dataset obtained by sampling is compared with a synthetic dataset from an idealized model.

The comparison is deemed statistically significant if the relationship between the datasets would be an unlikely realization of the null hypothesis according to a

[5] https://en.wikipedia.org/wiki/Statistical_hypothesis_testing, July 2018.

threshold probability — the significance level. Hypothesis tests are used in determining what outcomes of a study would lead to a rejection of the null hypothesis for a pre-specified level of significance.

Here are the test methods' spheres of applications.

- *Cases When There Is One Sample or Two Samples.* t-test is the most commonly used parameter test when testing a single sample or two samples with a large sample size. Generally, it is used to test the overall mean value when a single normal population variance is not known or when the two independent normal population variances are unknown but equal.
- *Cases When There Are Two Samples.* Both the Wilcoxon test and the Mann-Whitney test are capable of testing two datasets of small samples with non-normal distribution. However, the Wilcoxon test is a non-parametric test for paired samples, while the Mann-Whitney test is a non-parametric test for two independent samples and it is a test between different groups.
- *Cases When There Are Two or More Samples.* The variance analysis and the Kruskal-Wallis test are all significant tests for analyzing the mean value differences between two or more samples. The difference is that variance analysis is used to analyze the mean difference between two or more samples of a larger scale significance, which is often used to analyze and infer which factors have a significant impact on the study. When the sample size is small, it is more appropriate to choose the Kruskal-Wallis test. The Kruskal-Wallis test is a sequence-based variance analytical method which is often used to test the populations of samples.

3) *Usage of Data Analysis Tools.* Data analysis plays an important role in ESE. It makes statistics well-ordered and provides visual functions which help researchers to conclude results more quickly and effectively. The most commonly used tools are Microsoft Excel, SPSS, R Statistical Analysis Tools, and MatLab. As the most popular office-software, Microsoft Excel is the choice of most researchers for statistical analyzing, which is mentioned in 64 articles. There are 31 articles using SPSS for statistical analysis. Due to its simplicity and various functions, SPSS is frequently used for statistical analysis calculations, data mining, predictive analytics, and decision support tasks. There are 22 articles using R statistical analysis tool. R is favored by researchers as an excellent tool for statistical calculations and statistical charting. Seventeen articles use MatLab, which combines algorithm development, data visualization, data analysis, and other functions to perform data analysis.

The four tools above are mainly for quantitative data analyses. The most commonly used qualitative analysis tool is NVivo[6], and 11 articles use it to analyze stereotyped data. NVivo, as a software that supports qualitative research methods and hybrid research methods, can be used to collect, organize and analyze interviews, which makes it very convenient for researchers to process and analyze qualitative data.

4.3.4 *Summary*

- The mostly used data sources of empirical research are open source projects, followed by industrial projects and laboratory projects. Open source projects and industrial projects provide data for over 80% of the selected empirical research articles. The number of articles using open source projects has been continuously rising in recent years.
- Open source projects act as the primary data source for empirical research in software requirement, software maintenance, software testing, etc.
- The median of project numbers among all articles is 3.
- Questionnaire, interview, and data archiving are the most commonly used data collection methods.

4.4 RQ4. How Concerned Are Researches about Validity and Possibility to Replicate Empirical Studies?

4.4.1 *Researchers' Attention to Validity*

Validity is important to measure the effectiveness of an empirical study. None of the empirical studies can avoid the threats of validity, and researchers should give sufficient considerations on the validity of empirical research in order to mitigate some effectiveness threats. Kosti *et al.*[54] proposed a guideline for validity discussion which classified validity into structural validity, internal validity, external validity, and conclusion validity.

Through the validity analysis in the statistical data extraction table, we find that the authors of EMSE and ESEM articles have a satisfactory understanding and awareness of the validity and the results are showed in Fig.11. The validity threat, weakness, limitation, or other terms with similar meanings, are mentioned in 94.2% of selected articles, and this indicates that most researchers are aware of the validity of empirical

[6] www.qsrintenational.com/nvivo/nvivo-products, Aug. 2018.

research. Only 31 articles (5.8%) do not mention the validity threat. Among the 507 (94.2%) articles that have mentioned threats to validity, 318 (59.1%) articles discuss the validity according to the classification of construct validity, internal validity, external validity and conclusion validity (reliability).

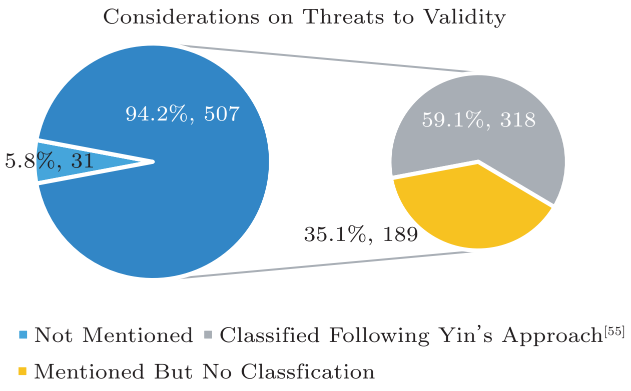


Fig.11. Discussions on threats to validity in the 538 articles.

External validity is discussed most frequently, and external validity evaluation is considered as the metric of whether the research conclusions can be applied to practice[56]. Among the articles that have classified threats to validity, 94% of them discuss the external validity and 86% discuss the internal validity, 73% of the articles discuss the construct validity, and 38% of the articles discuss the conclusion validity (reliability).

4.4.2 Researchers' Awareness of Replications

Possibility of replication is a significant issue in empirical research. An increasing number of researchers are appealing for the attention on the possibility of replication[2]. Therefore, authors' attention paid to the possibility of replication is evaluated.

According to the related items shown in Table 10, the considerations on the possibility of replication are classified into seven categories. We think in the first six categories, the possibility of replication has been taken into account while in the 7th category, the possibility of replication is not considered. Our result shows that in most articles (over 92%) it is taken into account.

Situation 1. Possibility of replication is emphasized and the replication package is available online. Among all the selected articles, 23.4% of them belong to this category.

Situation 2. The list of questionnaire/interview or primary articles is provided, which can be used for replication. There are 12.8% articles falling into this category.

Situation 3. The websites of data sources or projects are given. Most of the articles (41.1%) belong to this category.

Situation 4. The replication resource is collected from previous studies; therefore, it is accessible from previous studies. For example, in [28] the authors declared that the data used in their experiments was collected from two previous studies. This category covers 10.8% of all articles.

Situation 5. Authors may tell readers to ask them for research data. For instance, article [57] makes it clear that the data was accessible if you contacted the authors. This category only includes 0.7% articles.

Situation 6. Authors declared that research data was not available. The confidentiality of the used projects could be the reason. This category covers 3.3% of all the articles.

Situation 7. The possibility of replication was not addressed by authors, which covers 7.8% of all the articles.

4.4.3 Summary

Among all the selected research papers, 94.2% of researchers paid attention on threats to validity and described the specific threats to validity, while in the other 5.8% articles validity threats are not mentioned by authors. Over 92% of the researchers take the possibility of replication into account by providing access to research data. However, there are still less than 8% articles that do not consider the possibility of replication.

Table 10. Considerations About the Possibility of Replication

Situation No.	Have Consideration or Not	Situation	Article No./Percentage
1	Y	Replication package is provided available online	126/23.4%
2	Y	Lists of questionnaire/interviews/articles available	69/12.8%
3	Y	Websites of data sources provided	221/41.1%
4	Y	Replication resources available in previous studies	58/10.8%
5	Y	Asking authors for replication resources	4/0.7%
6	Y	Not available for explicit reasons	18/3.3%
7	N	Possibility of replication is not addressed by authors	42/7.8%

5 Discussion

5.1 Threats to Validity

According to Yin's approach[55], we discuss threats to internal validity, external validity, construct validity and conclusion validity in this subsection.

5.1.1 Internal Validity

This aspect of validity is of concern when causal relations are examined[2]. Both systematic literature reviews and system mapping researches face the common internal threats to validity that all literature research methods share: selection bias and subjectivity. Selecting different articles for literature research may result in different results. In recent years, the quantity of ESE papers has been very large, and different software-engineering journals have different degrees of acceptance of empirical research. How to choose proper ESE articles and how to decide time range of articles are both important factors, which may affect the internal validity. In addition, the internal connection of software engineering and computer science makes subfield classification job more complex.

It is challenging to distinguish empirical articles from the others because of the lack of generally accepted criterion and clear definitions. Meanwhile, the quality of these articles varies. We consider ESEM and EMSE as they are the main venues for publishing empirical research results. The official website of EMSE[7] says: "Empirical Software Engineering provides a forum for applied software engineering research with a strong empirical component, and a venue for publishing empirical results relevant to both researchers and practitioners." ESEM's call for papers writes: "The ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM) is the premier conference for presenting research results related to empirical software engineering." These two venues are widely accepted by the software engineering community in terms of their scopes, and they establish reputations in terms of their peer-reviewing processes. Therefore, we select papers from EMSE and ESEM as our research materials.

Subjectivity refers to the internal validity of the research results when we design and fill in the data summary table. In order to improve the validity of this paper, our extracted data has been modified for many times on the basis of the method shown in Fig.1. To finish the data summary table, it demands multiple rounds of reviews and discussions. For example, different options and decisions on the criteria for judgment are discussed to reduce the ambiguity at most. In order to reduce the intention of human misjudgment, multiple authors participate in the data acquisition process. If the decision results are inconsistent, we put it into the collective discussions, and this approach can reduce the threat to validity from subjectivity.

5.1.2 External Validity

The external validity focuses on the suitability of research results. Our findings are based on the empirical software engineering publications from EMSE and ESEM. And it is unknown about whether our results can be generalized to empirical research papers from other journals or conferences. In the future, we would like to study more empirical research papers and compare their results with our current findings. We will explore whether including more papers will draw different conclusions. Our current results are based on the investigation of 538 articles. We therefore believe the conclusion we draw is representative.

5.1.3 Construct Validity

This aspect of validity reflects to what extent the operational measures studied really represent what the researcher has in mind and what is investigated according to the research questions. While designing the data digest tables, there is a threat of structural validity. We take the iterative design method, randomly select the article to fill in the test, and then fill out the data summary table according to the result. We make it as an iterative process to achieve the final data summary table. Iterative design methods can effectively decrease the threat to the construct validity.

5.1.4 Conclusion Validity

This aspect is concerned with to what extent the data and the analysis are dependent on the specific researchers. Hypothetically, if another researcher conducted the same study later on, the result should be the same[16]. In this paper, the research methods and processes are elaborated in details to ensure that the research process can be reproduced, and we hope researchers can reproduce our study. The threats to conclusion validity in this paper are in the data statistics and analysis process. During these two processes, we use Excel for assistance.

[7] https://link.springer.com/journal/10664, July 2018.

5.2 Illustrations About SLR

In this paper, we also conduct a research on SLR (systematic literature review) articles in recent eight years (2010∼2017). As we mentioned in Subsection 4.2.1, the literature review methodology is a kind of secondary study based on the experience of previous publications. Reading related literature reviews can be an efficient way to obtain an overview of the research statues in certain sub-fields or topics.

The strategy for collecting the relevant literature is two-fold. 1) A keyword search uses "literature + review" as keywords in DBLP (Database Systems and Logic Programming) from 2010∼2017. 2) We filter the search results by selecting articles from software engineering related journals and conferences. After the above two steps, we obtain 256 selected SLR papers. We select DBLP as our literature database because it is an English literature database which contains all kinds of research findings in computer science domain. Most of the important journals on computer science are tracked, as well as the proceedings papers of many well-recognized conferences, and thus it can well present the overall research achievements in various directions.

Firstly, as shown in Fig.12, the article numbers in every year from 2010 to 2017 are listed. There has been an obvious increasing trend of article numbers per year in recent eight years. Specially, the article number of each year during 2015∼2017 is much larger than that of any year before 2015, with an amplification of around 200%. From 2010 to 2014, all these years' article numbers are less than 26. From 2015∼2017, the average article number reaches 49, indicating that in recent three years, researches have been using SLR as their research method more frequently.

Fig.13 presents the distribution of these 256 SLR articles in different journals or conferences. There is a significant article quantity difference between different journals/conferences. Sixty-one articles are from the journal of Information & Software Technology, which is much larger than the article numbers from any other selected venues. As for all the other journals/conferences, their article numbers are less than 25. Among them, 12 journals/conferences have 1∼3 SLR articles, 13 of them have 5∼15, and there are three of them whose SLR article numbers are between 20 and 25. Their SLR article numbers are strongly influenced by the specific demands of articles and the total number of volumes or issues. For example, Journal of Information & Software Technology has a much larger total number of articles, while ACM Computing Survey and Computer Science Review are more likely to accept SLR papers according to their call for papers.

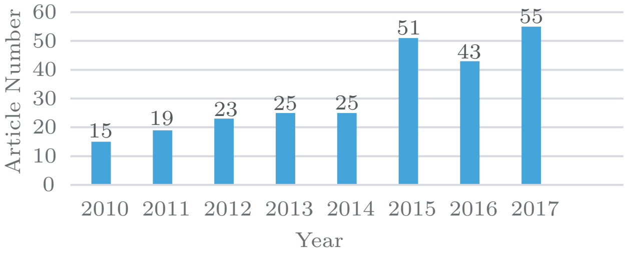


Fig.12. Number of SLR articles of each year on software engineering in 2010∼2017.

5.3 Gains, Problems, and Trends

Through our mapping study, we find that empirical researches cover all the sub-fields of software engineering research area. Experiment, case study, survey and SLR are widely used in all sub-fields of software engineering during every stage of software life cycle. There are some distinctions of their use along with the specific sub-fields they apply to.

The number of articles on software maintenance is much larger than that on any other sub-field. This indicates that software maintenance is the most interesting sub-area to ESE researchers, with significant value to be focused on. Also, this reflects that a lot of potential work and improvements are to be done in this area.

There are three types of research paradigms in ESE, namely exploratory, explanatory and technical validation. In exploratory empirical research, case study is more frequently used while in technical validation researches, experiment is adopted more. Exploratory researches and technical validations appear in all sub-fields of software engineering.

As the main data sources, open source projects and industrial projects provide over 80% data for empirical research. Laboratory projects are much fewer than them. Open source projects as a support for ESE and a sufficient supplement for the lack of standard datasets/test-sets, have become increasingly popular data sources among empirical researchers in recent years.

Currently, the most frequently used open source hosting platforms are Apache, SourceForge, GitHub and Mozilla, while the adoption of GitHub is rising in persistence. We also list the most frequently used open source projects (Fig.9). By analyzing the data collection methods, we give researchers several suggestions

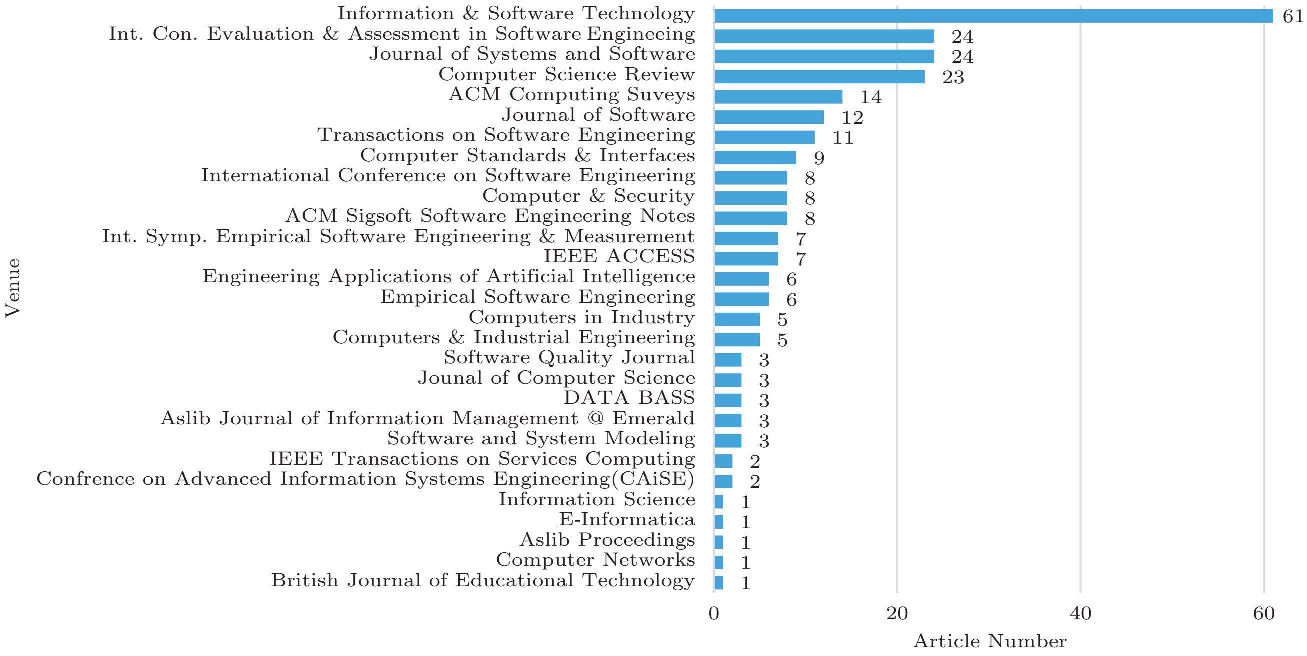


Fig.13. Number of SLR articles in the selected journals and conferences of software engineering field in 2010∼2017.

for getting acknowledge of the data processing methods and tools they should master in Subsection 4.3.3.

We encounter several problems during our research as we present above. Firstly, as we mention in Subsection 5.1, there is a difficulty to distinguish empirical articles from the others because of the lack of generally accepted criterion. The same problem occurs during our literature survey. We find empirical articles in EMSE and ESEM can be classified into two types. In the first scenario, empirical methods are applied as the main/primary measures to find answers to the research questions directly. While in the other scenario, solutions for research questions are proposed first, and then empirical methods are adopted just for assistance to prove the correctness or efficiency of it. But in practice, we find it difficult to precisely decide whether empirical methods are primary or secondary. Second, there is a lack of specification in terminologies. The most typical example is that when an author uses "...case study" as the title of the article, actually it is a controlled experiment conducted in the research.

Fortunately empirical research articles from EMSE and ESEM have more complete structures than those from the other journals or conferences. Meanwhile, researchers have an excellent awareness of validity and the possibility of replication, and thus more than 95% of them consider these two issues. But in other empirical research articles, there are still some obvious shortcomings and inadequacy, such as the lack of basis for questionnaires' design, the lack of analysis on the rationality of choosing participators and insufficient validity or limitation analysis. Therefore, we propose a more systematic education for ESE researchers.

Future Trend. According to our analysis on the EMSE and ESEM articles, there is an increasing trend on adoptions of open source data in empirical research. Among the selected articles, over 80 of them involve mining, analyzing or learning big data. As defined in Wikipedia, "empirical research" is a research using empirical evidence. It is a way of gaining knowledge by means of direct and indirect observation or experience. In the digital era with various big data technologies, it is easier to obtain empirical data (data of real-world projects). Therefore, we think that introducing new technologies such as machine learning and data mining to empirical study is necessary.

6 Conclusions

In this paper, we presented the overall landscape of ESE in the latest five years by reviewing and analyzing 538 research papers: 278 published in EMSE and 260 published in ESEM from January 2013 to November 2017. We found that empirical researches cover every sub-field of software engineering. Our important findings are listed below.

- There are various available empirical methods and they suit different situations. They are chosen by researchers according to the research materials, goals and environment. Among them, experiment, case study, and survey are most frequently used.

- Types of research paradigms are classified into exploratory, explanatory, and technical validation. Exploratory articles are in the first place, technical validation articles are in the second, and explanatory articles are in the third.

- We also identified the data sources for empirical research, such as frequently used open source platforms and projects. Open source resources are found to be the primary data source for empirical research. There is an obvious trend of using open source projects.

- At last we found most researchers have awareness of validity and the possibility of replication while still a few of them (less than 10%) pay little attention.

By identifying detailed implementations at critical points of empirical research, we aim to help researchers in performing researches of better quality. Our findings provide researchers supporting strategies, technologies and skills at each phase of their empirical research. At the same time, we reminded researchers of paying attention on threats to validity analysis and providing readers enough research data or resources to ensure the possibility to replicate their researches.

In the future, we plan to expand the scope of the survey by including more venues, more papers, so as to make our research more complete and convincing. In this paper, we mainly studied adoption of empirical methods in software engineering. We plan to provide detailed guidelines on how to apply empirical methods in various research areas of software engineering, and study more technologies or methods used in empirical studies. Meanwhile, we plan to develop automatic or semi-automatic tools with natural language processing (NLP) functions which assist us in data collecting, processing and analysis.

References

[1] Shull F, Singer J, Sjøberg D I K. Guide to Advanced Empirical Software Engineering. Springer, 2008.

[2] Siegmund J, Siegmund N, Apel S. Views on internal and external validity in empirical software engineering. In *Proc. the 37th International Conference on Software Engineering*, May 2015, pp.9-19.

[3] Borgs A, Ferreira W, Barreiros E, Almeida A, Fonseca L, Teixeira E, Silva D, Alencar A, Soares S. Support mechanisms to conduct empirical studies in software engineering. In *Proc. the 19th International Conference on Evaluation and Assessment in Software Engineering*, April 2015, Article No. 22.

[4] Cosentino V, Izquierdo J L C, Cabot J. A systematic mapping study of software development with GitHub. *IEEE Access*, 2017, 5: 7173-7192.

[5] Bezerra R, Silva F, Santana A, Magalhaes C, Santos R. Replication of empirical studies in software engineering: An update of a systematic mapping study. In *Proc. the 2015 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, October 2015, pp.132-135.

[6] Zhang J, Wang X Y, Hao D, Xie B, Zhang L, Mei H. A survey on bug-report analysis. *Science China Information Sciences*, 2015, 58(2): 1-24.

[7] Zhang T, He J, Luo X, Chan A T S. A literature review of research in bug resolution: Tasks, challenges and future directions. *The Computer Journal*, 2016, 59(5): 741-773.

[8] Ahmad A, Brereton P, Andras P. A systematic mapping study of empirical studies on software cloud testing methods. In *Proc. IEEE International Conference on Software Quality, Reliability and Security Companion*, July 2017, pp.555-562.

[9] Zhang L, Pu M Y, Liu Y J *et al.* Empirical investigation of empirical research methods in software engineering. *Journal of Software*, 2018, 29(5): 1422-1450. (in Chinese)

[10] Wohlin C, Runeson P, Höst M, Ohlsson M C, Regnell B, Runeson P, Wesslén A. Experimentation in Software Engineering. Springer, 2012.

[11] Petersen K, Feldt R, Mujtaba S, Mattsson M. Systematic mapping studies in software engineering. In *Proc. the 12th International Conference on Evaluation and Assessment in Software Engineering*, June 2008, pp.68-77.

[12] Petticrew M, Roberts H. Systematic Reviews in the Social Sciences: A Practical Guide. John Wiley & Sons, 2008

[13] Bourque P, Fairley R E. Guide to the Software Engineering Body of Knowledge (3rd edition). IEEE Computer Society Press, 2014

[14] Delgado D, Martinez A. Cost effectiveness of unit testing a case study in a financial institution. In *Proc. the 2013 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, October 2013, pp.340-347.

[15] Cook T D, Cambell D T. Quasi-Experiment: Design and Analysis Issues for Field Setting. Houghton Mifflin, 1979.

[16] Robert J M. Experimental and quasi-experimental designs for generalized causal inference. *Journal of Policy Analysis and Management*, 2003, 22(2): 330-332.

[17] Runeson P, Ḧost M. Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering*, 2009, 14(2): 131-164.

[18] Haller I, Slowinska A, Bos H. Scalable data structure detection and classification for C/C++ binaries. *Empirical Software Engineering*, 2016, 21(3): 778-810.

[19] Molléri J S, Petersen K, Mendes E. Survey guidelines in software engineering: An annotated review. In *Proc. the 10th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, September 2016, Article No. 58.

[20] Bao L F, Li J, Xing Z C, Wang X Y, Xia X, Zhou B. Extracting and analyzing time-series HCI data from screen-captured task videos. *Empirical Software Engineering*, 2017, 22(1): 134-174.

[21] Petersen K, Vakkalanka S, Kuzniarz L. Guidelines for conducting systematic mapping studies in software engineering: An update. *Information and Software Technology*, 2015, 64: 1-18.

[22] Juristo N, Vegas S. Using differences among replications of software engineering experiments to gain knowledge. In *Proc. the 3rd International Symposium on Empirical Software Engineering and Measurement*, October 2009, pp.356-366.

[23] Monteiro C V, Silva F Q, Capretz L F. The innovative behaviour of software engineers: Findings from a pilot case study. In *Proc. the 10th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, September 2016, Article No. 7.

[24] Wang Y. Characterizing developer behavior in cloud based IDEs. In *Proc. the 2017 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, November 2017, pp.48-57.

[25] Octaviano F R, Felizardo K R, Maldonado J C, Fabbri S C P F. Semi-automatic selection of primary studies in systematic literature reviews: Is it reasonable? *Empirical Software Engineering*, 2015, 20(6): 1898-1917.

[26] Heeager L T, Rose J. Optimising agile development practices for the maintenance operation: Nine heuristics. *Empirical Software Engineering*, 2015, 20(6): 1762-1784.

[27] Shin Y, Williams L. Can traditional fault prediction models be used for vulnerability prediction? *Empirical Software Engineering*, 2013, 18(1): 25-59.

[28] Raja U. All complaints are not created equal: Text analysis of open source software defect reports. *Empirical Software Engineering*, 2013, 18(1): 117-138.

[29] Albayrak Ö, Carver J C. Investigation of individual factors impacting the effectiveness of requirements inspections: A replicated experiment. *Empirical Software Engineering*, 2014, 19(1): 241-266.

[30] Estler H C, Nordio M, Furia C A, Meyer B, Schneider J. Agile vs. structured distributed software development: A case study. *Empirical Software Engineering*, 2014, 19(5): 1197-1224.

[31] Chen N, Hoi S C, Xiao X. Software process evaluation: A machine learning framework with application to defect management process. *Empirical Software Engineering*, 2014, 19(6): 1531-1564.

[32] Chen J, Xiao J, Wang Q, Osterweil L J, Li M. Perspectives on refactoring planning and practice: An empirical study. *Empirical Software Engineering*, 2016, 21(3): 1397-1436.

[33] Unterkalmsteiner M, Gorschek T, Feldt R, Lavesson N. Large-scale information retrieval in software engineering: An experience report from industrial application. *Empirical Software Engineering*, 2016, 21(6): 2324-2365.

[34] Capiluppi A, Izquierdo-Cortázar D. Effort estimation of FLOSS projects: A study of the Linux kernel. *Empirical Software Engineering*, 2013, 18(1): 60-88.

[35] Fucci D, Turhan B. On the role of tests in test-driven development: A differentiated and partial replication. *Empirical Software Engineering*, 2014, 19(2): 277-302.

[36] Mcburney P W, Mcmillan C. An empirical study of the textual similarity between source code and source code summaries. *Empirical Software Engineering*, 2016, 21(1): 17-42.

[37] Mcilroy S, Ali N, Khalid H, Hassan A E. Analyzing and automatically labelling the types of user issues that are raised in mobile app reviews. *Empirical Software Engineering*, 2016, 21(3): 1067-1106.

[38] Šmite D, Wohlin C, Galvina Z, Prikladnicki R. An empirically based terminology and taxonomy for global software engineering. *Empirical Software Engineering*, 2014, 19(1): 105-153.

[39] Greiler M, Deursen A V. What your plug-in test suites really test: An integration perspective on test suite understanding. *Empirical Software Engineering*, 2013, 18(5): 859-900.

[40] Callaú O, Robbes R, Tanter É, Röthlisberger D. How (and why) developers use the dynamic features of programming languages: The case of small-talk. *Empirical Software Engineering*, 2013, 18(6): 1156-1194.

[41] Cheung W T, Ryu S, Kim S. Development nature matters: An empirical study of code clones in JavaScript applications. *Empirical Software Engineering*, 2016, 21(2): 517-564.

[42] Ceccato M, Capiluppi A, Falcarin P, Boldyreff C. A large study on the effect of code obfuscation on the quality of java code. *Empirical Software Engineering*, 2015, 20(6): 1486-1524.

[43] Arcuri A, Fraser G. Parameter tuning or default values? An empirical investigation in search-based software engineering. *Empirical Software Engineering*, 2013, 18(3): 594-623.

[44] Tian Y, Lo D, Xia X, Sun C N. Automated prediction of bug report priority using multi-factor analysis. *Empirical Software Engineering,* 2015, 20(5): 1354-1383.

[45] Dit B, Revelle M, Poshyvanyk D. Integrating information retrieval, execution and link analysis algorithms to improve feature location in software. *Empirical Software Engineering*, 2013, 18(2): 277-309.

[46] Bavota G, Lucia A D, Marcus A, Oliveto R. Automating extract class refactoring: An improved method and its evaluation. *Empirical Software Engineering*, 2014, 19(6): 1617-1664.

[47] Zhu J, Zhou M, Mockus A. Patterns of folder use and project popularity: A case study of GitHub repositories. In *Proc. the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, September 2014, Article No. 30.

[48] Al-Subaihin A A, Sarro F, Black S, Capra M, Harman M, Jia Y, Zhang Y. Clustering mobile apps based on mined textual features. In *Proc. the 10th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, September 2016, Article No. 38.

[49] Mcilroy S, Ali N, Hassan A E. Fresh apps: An empirical study of frequently-updated mobile apps in the Google play store. *Empirical Software Engineering*, 2016, 21(3): 1346-1370.

[50] Allix K, Bissyandé T F, Jérome Q, Klein J, State R, Traon Y L. Empirical assessment of machine learning-based malware detectors for Android — Measuring the gap between in-the-lab and in-the-wild validation. *Empirical Software Engineering*, 2016, 21(1): 183-211.

[51] Fraser G, Arcuri A. 1600 faults in 100 projects: Automatically finding faults while achieving high coverage with EvoSuite. *Empirical Software Engineering*, 2015, 20(3): 611-639.

[52] Vasilescu B, Serebrenik A, Goeminne M, Mens T. On the variation and specialisation of workload: A case study of the GNOME ecosystem community. *Empirical Software Engineering*, 2014, 19(4): 955-1008.

[53] Xia X, Bao L F, Lo D, Kochhar P S, Hassan A E, Z Xing Z C. What do developers search for on the Web? *Empirical Software Engineering*, 2017, 22(6): 3149-3185.

[54] Kosti M V, Feldt R, Angelis L. Archetypal personalities of software engineers and their work preferences: A new perspective for empirical studies. *Empirical Software Engineering*, 2016, 21(4): 1509-1532.

[55] Yin R K. Case Study Research: Design and Methods (4th edition). Sage Publications, 2009.

[56] William B J, Carver J C. Examination of the software architecture change characterization scheme using three empirical studies. *Empirical Software Engineering*, 2014, 19(3): 419-464.

[57] Schulz T, Radliński Ł, Gorges T, Rosenstiel W. Predicting the flow of defect correction effort using a Bayesian network model. *Empirical Software Engineering*, 2013, 18(3): 435-477.

Li Zhang received her Bachelor's, Master's and Ph.D. degrees in computer science and technology from Beihang University, Beijing, in 1989, 1992 and 1996, respectively. She is now a professor in the State Key Laboratory of Software Development Environment, Beihang University, Beijing, where she is leading the expertise area of system and software modeling. Her main research area is software engineering, with specific interest in requirements engineering, software/system architecture, model-based engineering, model-based product line engineering, and empirical software engineering.

Jia-Hao Tian received his Bachelor's and Master's degrees in software engineering from Jilin University, Changchun, in 2012 and 2015 respectively. He is now a Ph.D. candidate in the State Key Laboratory of Software Development Environment, Beihang University, Beijing. His research interests include empirical software engineering, software architecture, and machine learning.

Jing Jiang is an assistant professor in the State Key Laboratory of Software Development Environment, Beihang University, Beijing. Her research interests include software engineering, empirical software engineering, data mining, and recommendation. She received her B.S. and Ph.D. degrees in computer science from Peking University, Beijing, in 2007 and 2012, respectively.

Yi-Jun Liu received her Bachelor's degree in computer science and technology from Huazhong Normal University, Wuhan, in 2016. She is now a postgraduate student in the State Key Laboratory of Software Development Environment, Beihang University, Beijing. Her research interests include empirical software engineering, natural language processing, and knowledge engineering.

Meng-Yuan Pu received her Bachelor's degree in software engineering from East China Normal University, Shanghai, in 2015. She received her Master's degree in software engineering from Beihang University, Beijing. Her research interests include empirical software engineering, software testing and artificial intelligence.

Tao Yue is a chief research scientist and head of Department of Engineering Complex Software at Simula Research Laboratory (SRL), Norway, and she is also affiliated with UiO (University of Oslo). She has received her Ph.D. degree in the Department of Systems and Computer Engineering at Carleton University, Ottawa, Canada, in 2010. Before that, she was an aviation engineer and system engineer for seven years. She has nearly 20 years of experience of conducting industry-oriented research with a focus on Model-Based Engineering (MBE) in various application domains such as Avionics, Maritime and Energy, Communications, Automated Industry, and Healthcare in several countries including Canada, Norway, and China. Tao is on the editorial board of Empirical Software Engineering and Science of Computer Programming. She is on the steering committee of MODELS 2018 and will serve as PC co-chair of MODELS 2019. Tao has been on the program and organization committees of several international conferences (e.g., MODELS, RE and SPLC) and is also actively participating in defining international standards in Object Management Group (OMG), including Precise Semantics for Uncertainty Modeling (PSUM), System Modeling Language (SysML) V2, and UML Testing Profile (UTP) V2. Tao also served as an expert of proposal evaluation committee of an EU H2020 call.