ESEC/FSE: G: Systematizing the Meta-Analytical Process in Software Engineering

A reflection initiated by the Analysis of the Impact of Programming Languages on Energy Consumption for Mobile Devices

Zamira Kholmatova (Advisor: Giancarlo Succi)
Innopolis University
Innopolis, Russia
z.kholmatova@innopolis.university

CCS CONCEPTS

• **General and reference → Computing standards, RFCs and guidelines**.

KEYWORDS

Meta-analysis in software engineering, evidence-based software engineering, experimentation in software engineering

ACM Reference Format:
Zamira Kholmatova (Advisor: Giancarlo Succi). 2021. ESEC/FSE: G: Systematizing the Meta-Analytical Process in Software Engineering *A reflection initiated by the Analysis of the Impact of Programming Languages on Energy Consumption for Mobile Devices.* In *Proceedings of The 28th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE 2020).* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 MOTIVATION

This research has started with an investigation of the impact of programming languages on energy consumption for mobile devices through a meta-analysis. During the investigation, we faced a lack of studies and the obsolescence of data. Among the 6 studies we found, three of them investigated the energy consumption of the devices which is out-of-date. This problem left us with only three studies. But, besides this issue, there is also a lack of meta-analytical works in software engineering which resulted in the lack of commonly agreed processes for doing it. All mentioned obstacles pushed us to further investigate the way of systematizing a meta-analytical process. We started with a small, but very important step towards this direction - identification of issues preventing the widespread usage of meta-analysis in software engineering. To find such factors we conducted several projects applying meta-analytical techniques to the software engineering domain.

ESEC/FSE 2020, 8 - 13 November, 2020, Sacramento, California, United States

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

2 EARLY SOLUTION TO THE ORIGINAL PROBLEM

Since the work on the impact of programming languages on energy consumption for mobile devices was our first one which was done using meta-analytical techniques, we conducted it following the guideline presented in the Cochrane Reviewer's Handbook [1]. However, this handbook is mostly related to medical researchers and many techniques using for meta-analysis in medicine are not applicable to the software engineering domain. Despite this fact, in our case, it was possible to apply the random-effects model for continuous outcomes with Hedges' g estimator for the mean difference [2]. But it was beyond our power to deal with the lack of studies for the considered area. If the study contained the obsolescence data - for example, energy consumed by the devices which are out of date, we ignored it even though the initial amount of papers were small. In our original investigation, we did not go deeper into consideration of the mentioned problems. But, as we saw later, further application of meta-analysis will lead us to the same problems for which we do not know the answer. The best way to go further is to formalize the problem and find comprehensive solutions for it.

3 FORMALIZATION OF PROBLEM BEING ADDRESSED

The fast growth of empirical research in software engineering of the last two decades [3, 4] demands techniques to summarize the results of such empirical investigations. Meta-analysis has been suggested as a mechanism to do it even when empirical software engineering was not so widespread [5, 6]. Still, this is not happening despite the conceptual and operational simplicity of meta-analysis and its wide usage in other empirical disciplines, such as medical and social sciences [7, 8]. Our research started from the meta-analytical investigation of energy consumption targets such problem, which can be reformulated in terms of the following research goals:

RG.1: Identify the issues that prevent the widespread adoption of meta-analysis in software engineering,
RG.2: Propose solutions to such issues,
RG.3: Elaborate a comprehensive approach addressing these issues.

Applying systematically meta-analysis to software engineering is not a new idea, as it emerged in the mid '90s with the works of Brooks [9], Hayes [10], Miller [11], etc. The idea evolved and

Kitchenham et al. connected it to their attempt to apply the evidence-based approached common in medicine to software engineering [12]. The evidence-based practice is focused on the synthesis of best quality scientific studies on a specific topic or research question [13, 14] and such proposal has been particularly successful in one of the practices that advocates - the systematic literature review (SLR), which defines a systematic protocol to collect, analyze, and synthesize mostly qualitatively the existing knowledge on a subject as it appears in the scientific literature [15]. So far more than 100 systematic literature reviews have been published only in ACM Computing Surveys [16] and we can claim that SLRs are part of the current research practices of software engineering researchers. Still, SLRs result in mostly qualitative evidence and the problem of quantitative support of such evidence remains open. Using a meta-analysis, it becomes possible to prove statistically the obtained results. Despite the urgent need for this technique, the proper way of doing it is still unclear.

With a help of several meta-analytical studies which were performed in the framework of a graduate course on Advances Statistics, we identified three main issues that hinder the widespread use of meta-analysis in software engineering. Besides describing these problems, we are aimed to present possible solutions to overcome them in future research.

4 BACKGROUND AND RELATED WORKS

Meta-analysis first appeared in the work of Glass (1976) [17], where it is defined as "the statistical analysis of a large collection of results from individual studies for the purpose of integrating the findings". Meta-analysis can be considered as the tool of extraction knowledge from works of different individual researchers. Similarly, the goal of meta-analysis in software engineering should be to quantitatively summarize the results. Despite the simplicity of the general idea, meta-analysis is still not widely applied in the considered area. To find out the reasons for which it is so and to make the first attempts to systematize the process of aggregation of the results we started with analyzing the existing literature in this field.

The search strings we used to find the relevant literature were generated using the PICO approach [18] (Table 1).

Search query	Number of papers
meta analysis AND software engineering	42
combining evidence AND software engineering	7
evidence based software engineering	12

Table 1: Search queries

To retrieve more papers we applied forward and backward snowballing techniques [19]. The number of papers found before and after applying snowballing is shown in Figure 1

All the papers we found can be divided into four groups according to their objectives (Table 2).

In our research, we are more interested in the second category which is exactly about the application of meta-analysis in software engineering. Analyzing even such a small number of articles, we noticed that the primary application of meta-analysis which consists of aggregating the results from replicated experiments was more preferred than conducting a secondary study that aims to extract

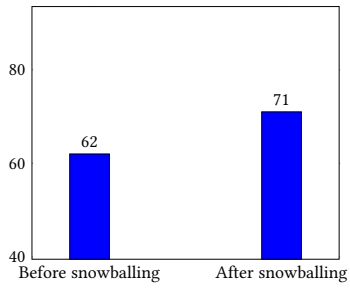

Figure 1: Comparison of number of papers before and after applying snowballing method

the general knowledge from multiple studies coming from different sources. The work of Succi et al. [20] can serve as an example of primary meta-analytical research who used the "weighted estimator of a common correlation" to generalize correlations between object-oriented software metrics coming from public domain Java projects.

№	Category	Number of studies
1	Studies containing general information about meta-analysis	9
2	Studies containing case studies related to the application of meta-analysis in software engineering	24
2	Studies addressing the concept of research synthesis in software engineering	22
3	Studies containing application of systematic literature review to software engineering	16

Table 2: Different kind of studies found during the search

Testing hypotheses based on the data extracted from different primary studies via meta-analysis came to software engineering a bit later - at the beginning of the 21st century. It had already helped to investigate the impact of pair programming on quality, time, and effort [21, 22], of test-driven development on quality and productivity [23], of programming languages on energy consumption [24], the performance of defect prediction models [25]. In all these studies, both fixed and random effects models with an inverse-variance approach for assigning the weights are used, but the way of their application varies from one study to another. For example, even though both Hannay et al. [21] and Umapathy et al. [22] investigated the impact of pair programming on quality, duration, and productivity, their works are different in implementation - in the second work, the meta-analysis was conducted separately for each quality cluster - programming assignment grades, exam scores, and persistence in computer programming courses.

As one can notice, despite that meta-analysis carries out almost for a quarter of a century, it still has no proper guidelines. Standardizing of the meta-analytical process, which supposes finding the issues that have not yet allowed this standardization to be finished, will lead us to the exploration of wider usage of meta-analysis such

as finding factors affecting observed significance and generating new hypotheses [26].

5 UNIQUENESS OF THE APPROACH

5.1 Exploring the problem

To identify some of the factors that prevent a systematization of meta-analysis in software engineering, we conducted several small meta-analytical studies within the context of a graduate course on Advanced Statistics at Innopolis University. The concrete subjects of the small meta-analytical investigations were:

- impact of software reuse on quality and productivity;
- impact of programming languages on energy consumption;
- performance of different machine learning models (Naive Bayes, Random Forest, Support Vector Machine) for software defect prediction.

These meta-analytical studies were based on the following steps:

- formalization of research question,
- search strategy,
- papers selection,
- quality assessment,
- data extraction,
- data analysis,
- selection of a model,
- computing the results.

As mentioned, they were based on [1, 13, 27]. The first six steps are related to the systematic literature review and are performed according to the guideline given by Kitchenham [13]. The last two steps are proposed by us to finalize the meta-analytical process, therefore, in a sense, we introduced a bias in our analysis.

Formalization of research question. The formalization of the research question defines the key research over which metanalysis (and the associated SLR) will be performed.

Search strategy. The search strategy aims to identify as many papers related to the considered topic as possible.

Papers selection. To understand the relevance of papers found, one needs to define inclusion and exclusion criteria for further consideration.

Quality assessment. After the papers were selected, we need to understand the "quality" of the results: to investigate the difference reflecting in studies results, the interpretation of findings, and the strength of inferences.

Data extraction. From the papers passed the quality assessment, one should collect the information needed for the research: papers' metadata, information about methodology, results, etc.

Data analysis. The data analysis process involves the aggregation of the extracted results from primary studies. The aggregation can be both qualitative, which is usually done in SLR, and quantitative, which is can be done through meta-analysis.

Selection of a model. According to the available data, there are different statistical procedures for performing a meta-analysis. For example, two possible approaches for assigning weights to each primary study - sample size and inverse-variance methods [28] and two families of meta-analytical procedures become available - fixed and random effects models [29].

Computing the results. After choosing all mathematical models, one can start to compute the results and derive conclusions. For example, if the main goal was to test the hypothesis via meta-analysis, in this stage, p-values should be calculated and the conclusion about acceptance or rejection of the null hypothesis should be taken.

5.2 Toward a solution

In our research, the difficulties start to appear with the step "Data analysis". While preparing the data to the further meta-analysis, we faced the three major issues:

(1) lack of a standard for metrics,
(2) fast obsolescence of data,
(3) usage of the same datasets.

We faced the first issue during the investigation of the impact of software reuse on quality and productivity. In Table 3, one can see different metrics used in primary studies on this topic: 7 metrics are related to reuse, 8 metrics refer to productivity, and 8 metrics reflect quality. Even though these metrics are used for similar experiments, in most cases, they cannot be transformed into each other. The lack of a standard for metrics makes the analysis of the studies found almost impossible.

The best solution for this issue would be to calculate the metrics that we need from the raw data. However, in most studies, such data is not publicly available. Therefore, we suggest two other possible ways of solving the issue of metrics diversity:

- make clusters of studies according to the metrics they used and conduct a separate meta-analysis on each cluster;
- apply the vote-counting approach, but to the simple, more generic questions, like "Does reuse positively affect quality?", "Does reuse positively affect productivity?"

The second issue was the fast obsolescence of data. We noticed it while investigating the impact of programming language on energy consumption for mobile devices. C, C++, and Java were considered for this investigation, but, nowadays, the most used programming languages for mobile application development are Kotlin and Javascript. Despite their popularity, we found the only paper considered Kotlin [42] and the only paper considered Javascript [43] with a comparison with other languages. Due to the lack of a sufficient number of papers for these two languages, conducting a meta-analytical comparison for them remains not possible.

Also, the other side of this issue was the deprecation of devices, for which energy consumption was measured. In this particular case, we cannot use the paper done, for instance, in 2009. Some operating systems and mobile devices' capacity can be out of date. In this situation, the obsolescence data should be removed from the meta-analysis. This problem is no often met in different fields such as medicine, chemistry, etc. For example, medical treatments can be still effective through decades, and doing a primary study can be expensive. Software engineering is a fast-growing field that allows researchers to conduct a variety of experiments at a low cost. The existence of a strong community as Cochrane in healthcare helps to overcome the problem about the relevance of data. The collaboration between industry, researchers, and people interested

Reuse Metrics	Productivity Metrics	Quality metrics
Reuse rate[30]	Development time [31]	Error density [32]
Reuse level [33]	Effort per module [34]	Defect density [32]
Reuse frequency [33]	Number of non commentary source lines produced per person day [36]	Customer complaint density [37]
New object percentage [35]	Number of new requirements divided by total efforts [38]	Quality index [34]
Reused source instructions [33]	Design and coding time [39]	Number of errors per non commentary source line [36]
External reuse density [37]	Source lines of code per hour [33]	Trouble reports per thousands of lines of code [40]
Internal reuse density [37]	Written lines of code [30]	Size of new or modified code per total size of the component [40]
	Reused lines of code [30]	Defects per thousands of lines of code [41]

Table 3: Different kind of metrics found in a SLR to understand the effects of software reuse on quality and productivity

in software engineering will allow sharing the current trends and creative ideas contributing to the development of this field in all directions.

NASA	PROMISE
[44], [45], [46], [47], [48]	[44], [49], [50]

Table 4: Papers with usage of similar datasets found in SLR to understand the performance of defect prediction models

During the comparison of the performance of different models for defect prediction, we noticed that many studies used the same datasets. We highlighted two datasets used in most studies - NASA and PROMISE. In Table 4, one can see that four papers used NASA, two papers used PROMISE, and one - both of the datasets. From one side, usage of the same data allows us to compare the performance of different models without taking into consideration random sampling errors. It makes the inference more reliable. But, on the other hand, it prevents us from the whole view of the current state of the art: which model performs better under different conditions? Sharing the datasets collected for the experiments or reuse them for different problems will help to solve this problem.

5.3 Threats to validity

Needless to say, the structures of these small meta-analytical projects had intrinsic limitations:

- time: the course was only 7 weeks long,
- breadth: given the time constraint, we selected fairly narrow topics for exploration,
- people: despite being very gifted, the work was done by students and not by professionals,
- approach: we followed the recommendations of Kitchenham et al. [13], for conducting the SLR, but we defined ourselves the steps for the meta-analysis, mutating them from medicine [1, 27].

Despite the uniform protocol, each of the works varies in the search strategy. Some of the students preferred the Google Scholar engine, some of them used digital libraries. This can cause publication bias. Also, the usage of the random effects model in each study assumes that there are many sources of variability, that we did not investigate. It can lower the strengths of these meta-analytical works.

It should be said that we had only three projects where we found the considered limitations preventing the wide usage of meta-analysis. But this is just a beginning. More studies will lead us to more questions that still do not have answers.

6 RESULTS AND CONTRIBUTIONS

With a help of several meta-analytical studies conducted within the framework of the course Advanced Statistics, we identified three major issues that slow down the adoption of meta-analysis in software engineering. They are related to the lack of a standard for metrics, the fast obsolescence of data, and the usage of the same datasets. For each problem, we proposed solutions based on our experience. It should be said that these solutions have some limitations. Firstly, using a clustering approach or vote-counting in a case with a lack of a standard for metrics will give us answers only for more simple research questions. Secondly, the conducted case studies have not been published yet, but this process is already in progress. We expect the publicly available and peer-reviewed results of our experiments carried out in the nearest future.

Needless to say, the systematization of meta-analysis in software engineering is a long-term plan. In our future research we are going to consider different effect sizes and statistical models, start developing the standards for conducting a meta-analysis. But we have already made the first attempt starting with the identification of the current obstacles in this big process and the presentation of our solutions.

The most important thing we noticed is that software engineering needs a strong community that will help to collaborate with people from both academia and industry. Such collaboration will lead us to further standardization of the reporting protocols, investigation of current trends, and wide usage of research results in the industry. We have already started to build such a community within our university giving the students and other people interested in software engineering general ideas about meta-analysis and techniques for aggregation empirical results. Besides all mentioned points, this research gave rise to the topic “ Introduction to meta-analysis ” within the framework of a course of Advances Statistics at Innopolis University.

7 ACKNOWLEDGEMENT

I thank Innopolis University for generously supporting this research, the students of group M20-DS, and instructors of the course Advanced statistics at Innopolis University for conducting several studies on the application of meta-analysis to software engineering data.

REFERENCES

[1] J. P. Higgins, J. Thomas, J. Chandler, M. Cumpston, T. Li, M. J. Page, and V. A. Welch, *Cochrane handbook for systematic reviews of interventions.* John Wiley & Sons, 2019.

[2] M. Borenstein, L. Hedges, and H. Rothstein, "Meta-analysis: Fixed effect vs. random effects," *Meta-analysis. com*, 2007.

[3] B. A. Kitchenham, S. L. Pfleeger, L. M. Pickard, P. W. Jones, D. C. Hoaglin, K. El Emam, and J. Rosenberg, "Preliminary guidelines for empirical research in software engineering," *IEEE Transactions on software engineering*, vol. 28, no. 8, pp. 721–734, 2002.

[4] C. Wohlin, M. Höst, and K. Henningsson, "Empirical research methods in software engineering," in *Empirical methods and studies in software engineering.* Springer, 2003, pp. 7–23.

[5] J. Miller, "Can results from software engineering experiments be safely combined?" in *Proceedings Sixth International Software Metrics Symposium (Cat. No. PR00403).* IEEE, 1999, pp. 152–158.

[6] L. M. Pickard, B. A. Kitchenham, and P. W. Jones, "Combining empirical results in software engineering," *Information and software technology*, vol. 40, no. 14, pp. 811–821, 1998.

[7] K. A. L'ABBÉ, A. S. Detsky, and K. O'ROURKE, "Meta-analysis in clinical research," *Annals of internal medicine*, vol. 107, no. 2, pp. 224–233, 1987.

[8] K. Witte and M. Allen, "A meta-analysis of fear appeals: Implications for effective public health campaigns," *Health education & behavior*, vol. 27, no. 5, pp. 591–615, 2000.

[9] A. Brooks, "Meta analysis—a silver bullet—for meta-analysts," *Empirical Software Engineering*, vol. 2, no. 4, pp. 333–338, 1997.

[10] W. Hayes, "Research synthesis in software engineering: a case for meta-analysis," in *Proceedings Sixth International Software Metrics Symposium (Cat. No. PR00403).* IEEE, 1999, pp. 143–151.

[11] J. Miller, "Applying meta-analytical procedures to software engineering experiments," *Journal of Systems and Software*, vol. 54, no. 1, pp. 29–39, 2000.

[12] B. A. Kitchenham, T. Dyba, and M. Jorgensen, "Evidence-based software engineering," in *Proceedings. 26th International Conference on Software Engineering.* IEEE, 2004, pp. 273–281.

[13] B. Kitchenham, "Procedures for performing systematic reviews," *Keele, UK, Keele University*, vol. 33, no. 2004, pp. 1–26, 2004.

[14] B. Kitchenham, O. P. Brereton, D. Budgen, M. Turner, J. Bailey, and S. Linkman, "Systematic literature reviews in software engineering–a systematic literature review," *Information and software technology*, vol. 51, no. 1, pp. 7–15, 2009.

[15] P. Brereton, B. A. Kitchenham, D. Budgen, M. Turner, and M. Khalil, "Lessons from applying the systematic literature review process within the software engineering domain," *Journal of systems and software*, vol. 80, no. 4, pp. 571–583, 2007.

[16] "Acm computing surveys," http://https://dl.acm.org/journal/csur, accessed: 2021-04-12.

[17] G. V. Glass, "Primary, secondary, and meta-analysis of research," *Educational researcher*, vol. 5, no. 10, pp. 3–8, 1976.

[18] C. M. d. C. Santos, C. A. d. M. Pimenta, and M. R. C. Nobre, "The pico strategy for the research question construction and evidence search," *Revista latino-americana de enfermagem*, vol. 15, no. 3, pp. 508–511, 2007.

[19] C. Wohlin, "Guidelines for snowballing in systematic literature studies and a replication in software engineering," in *Proceedings of the 18th international conference on evaluation and assessment in software engineering*, 2014, pp. 1–10.

[20] G. Succi, R. Spasojevic, J. J. Hayes, M. R. Smith, and W. Pedrycz, "Application of statistical meta-analysis to software engineering metrics data," in *Proceedings of the World Multiconference on Systemics, Cybernetics and Informatics*, vol. 1, 2000, pp. 709–714.

[21] J. E. Hannay, T. Dybå, E. Arisholm, and D. I. Sjøberg, "The effectiveness of pair programming: A meta-analysis," *Information and Software Technology*, vol. 51, no. 7, pp. 1110–1122, 2009.

[22] K. Umapathy and A. D. Ritzhaupt, "A meta-analysis of pair-programming in computer programming courses: Implications for educational practice," *ACM Transactions on Computing Education (TOCE)*, vol. 17, no. 4, pp. 1–13, 2017.

[23] Y. Rafique and V. B. Mišić, "The effects of test-driven development on external quality and productivity: A meta-analysis," *IEEE Transactions on Software Engineering*, vol. 39, no. 6, pp. 835–856, 2012.

[24] Z. Kholmatova, "Impact of programming languages on energy consumption for mobile devices," in *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2020, pp. 1693–1695.

[25] S. Hosseini, B. Turhan, and D. Gunarathna, "A systematic literature review and meta-analysis on cross project defect prediction," *IEEE Transactions on Software Engineering*, vol. 45, no. 2, pp. 111–147, 2017.

[26] J. Yi, V. Ivanov, and G. Succi, "Mining plausible hypotheses from the literature via meta-analysis," in *2019 IEEE/ACM 41st International Conference on Software Engineering: New Ideas and Emerging Results (ICSE-NIER).* IEEE, 2019, pp. 33–36.

[27] M. Egger, G. D. Smith, and A. N. Phillips, "Meta-analysis: principles and procedures," *Bmj*, vol. 315, no. 7121, pp. 1533–1537, 1997.

[28] F. Marín-Martínez and J. Sánchez-Meca, "Weighting by inverse variance or by sample size in random-effects meta-analysis," *Educational and Psychological Measurement*, vol. 70, no. 1, pp. 56–73, 2010.

[29] L. V. Hedges and J. L. Vevea, "Fixed-and random-effects models in meta-analysis." *Psychological methods*, vol. 3, no. 4, p. 486, 1998.

[30] M. T. Baldassarre, A. Bianchi, D. Caivano, and G. Visaggio, "An industrial case study on reuse oriented development," in *21st IEEE International Conference on Software Maintenance (ICSM'05).* IEEE, 2005, pp. 283–292.

[31] W. Frakes and C. Terry, "Software reuse: metrics and models," *ACM Computing Surveys (CSUR)*, vol. 28, no. 2, pp. 415–435, 1996.

[32] W. M. Thomas, A. Delis, and V. R. Basili, "An analysis of errors in a reuse-oriented development environment," *Journal of Systems and Software*, vol. 38, no. 3, pp. 211–224, 1997.

[33] P. Devanbu, S. Karstu, W. Melo, and W. Thomas, "Analytical and empirical evaluation of software reuse metrics," in *Proceedings of IEEE 18th International Conference on Software Engineering.* IEEE, 1996, pp. 189–199.

[34] M. Morisio, D. Romano, and I. Stamelos, "Quality, productivity, and learning in framework-based development: an exploratory case study," *IEEE Transactions on Software Engineering*, vol. 28, no. 9, pp. 876–888, 2002.

[35] R. D. Banker and R. J. Kauffman, "Reuse and productivity in integrated computer-aided software engineering: An empirical study," *MIS quarterly*, pp. 375–401, 1991.

[36] W. B. Frakes and G. Succi, "An industrial study of reuse, quality, and productivity," *Journal of Systems and Software*, vol. 57, no. 2, pp. 99–106, 2001.

[37] G. Succi, L. Benedicenti, and T. Vernazza, "Analysis of the effects of software reuse on customer satisfaction in an rpg environment," *IEEE Transactions on Software Engineering*, vol. 27, no. 5, pp. 473–479, 2001.

[38] B. Deniz and S. Bilgen, "An empirical study of software reuse and quality in an industrial setting," in *International Conference on Computational Science and Its Applications.* Springer, 2014, pp. 508–523.

[39] S. H. Zweben, S. H. Edwards, B. W. Weide, and J. E. Hollingsworth, "The effects of layering and encapsulation on software development cost and quality," *IEEE Transactions on software engineering*, vol. 21, no. 3, pp. 200–208, 1995.

[40] P. Mohagheghi, R. Conradi, O. M. Killi, and H. Schwarz, "An empirical study of software reuse vs. defect-density and stability," in *Proceedings. 26th International Conference on Software Engineering.* IEEE, 2004, pp. 282–291.

[41] A. Gupta, J. Li, R. Conradi, H. Rønneberg, and E. Landre, "A case study comparing defect profiles of a reused framework and of applications reusing it," *Empirical Software Engineering*, vol. 14, no. 2, pp. 227–255, 2009.

[42] P. Schwermer, "Performance evaluation of kotlin and java on android runtime," 2018.

[43] R. Pereira, M. Couto, F. Ribeiro, R. Rua, J. Cunha, J. P. Fernandes, and J. Saraiva, "Energy efficiency across programming languages: how do energy, time, and memory relate?" in *Proceedings of the 10th ACM SIGPLAN International Conference on Software Language Engineering*, 2017, pp. 256–267.

[44] H. Ji, S. Huang, Y. Wu, Z. Hui, and C. Zheng, "A new weighted naive bayes method based on information diffusion for software defect prediction," *Software Quality Journal*, vol. 27, 09 2019.

[45] A. Iqbal, S. Aftab, and F. Matloob, "Performance analysis of resampling techniques on class imbalance issue in software defect prediction," *International Journal of Information Technology and Computer Science*, vol. 11, pp. 44–53, 11 2019.

[46] D. Tomar and S. Agarwal, "Prediction of defective software modules using class imbalance learning," *Applied Computational Intelligence and Soft Computing*, vol. 2016, pp. 1–12, 02 2016.

[47] K. Khatter and A. Kalia, "Investigating software detection methods," in *Proceedings of theSixth International Conference on Industrial Engineering and Operations Management (IEOM)*, 2016, pp. 2248–2258.

[48] A. Iqbal, S. Aftab, I. Ullah, M. S. Bashir, and M. A. Saeed, "A feature selection based ensemble classification framework for software defect prediction," *International Journal of Modern Education and Computer Science(IJMECS)*, vol. 11, No.9, pp. 54–64, 2019.

[49] S. Kumar and S. Rathore, *Types of Software Fault Prediction.* Springer, 06 2018, pp. 23–30.

[50] H. JI, S. HUANG, X. LV, Y. WU, and Y. FENG, "Empirical studies of a kernel density estimation based naive bayes method for software defect prediction," *IEICE Transactions on Information and Systems*, vol. E102.D, pp. 75–84, 01 2019.