Автономная некоммерческая организация высшего образования «Университет Иннополис» (АНО ВО «Университет Иннополис»)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ)
по направлению подготовки
09.04.01 – «Информатика и вычислительная техника»

GRADUATION THESIS (MASTER GRADUATE THESIS)
Field of Study
09.04.01 – «Computer Science»

Направленность (профиль) образовательной программы «Управление большими данными»
Area of Specialization / Academic Program Title: «Data Science»

Тема / Topic	**Применение метааналитических методов к экспериментам программной инженерии - тематическое исследование энергопотребления / Application of meta-analytical techniques to software engineering experimentation — a case study on energy consumption**

Работу выполнил / Thesis is executed by	**Холматова Замира Шухратовна / Kholmatova Zamira Shukhratovna**	подпись / signature
Руководитель выпускной квалификационной работы / Graduation Thesis Supervisor	**Суччи Джианкарло / Succi Giancarlo**	подпись / signature

Иннополис, Innopolis, 2021

Contents

List of Tables

List of Figures

Abstract

In this work, we propose a protocol for a meta-analysis, which is a tool for the generalization of knowledge in different scientific areas. It is already adapted for such fields as medicine, psychology, epidemiology, business. In software engineering, meta-analysis is promoted as a tool for aggregating results from families of experiments rather longer than for the generalization of knowledge coming from different studies. For the second purpose of the application of meta-analysis, a way of performing it is not yet clarified. Therefore, we attempt to make a small step towards this direction - to propose our preliminary protocol to perform a meta-analysis as a secondary study. To see the validity and reliability of a proposed protocol, we conducted a case study based on it.

Chapter 1

Introduction

In the last decades, empirical software engineering became one of the most researched branches of software engineering [1]. The experiments were done on a wide area of subjects - defect prediction [2], [3], code smells detection [4], [5], software reuse [6], [7]. The dramatic growth of empirical research demands the methods for aggregation of the results coming from different sources. For this purpose, in the middle of the '90s, a statistical meta-analysis was proposed with the works of Brooks [8], Hayes [9], Miller [10]. Almost in parallel with the proposal of meta-analysis, software engineers started to develop a concept of "evidence-based software engineering" [11], which was initially developed in medicine. It was used when the individual expert's opinion was not considered more reliable than experimental results. Within a software engineering field, evidence means a synthesis of best quality scientific studies on a specific topic or research question [12], [13]. The main goal of this practice is "to provide the means by which current best evidence from research can be integrated with practical experience and human values in the decision-making process regarding the development and maintenance of software." One of the main methods of evidence-based software engineering - a systematic literature review, is already adapted to the software engineering field [13], [14]. But still, the evidence provided with a help of this concept is mostly qualitative and the information about the methods suitable for aggregating a large collection of quantitative results are not fully provided. However, it is useful to apply a meta-analysis on top of the systematic literature review, where there is a need for aggregation quantitative data. But the representation of the results answering the same questions varies from study to study, making the aggregation process difficult and ambiguous. So, the need of developing the solution while working with quantitative data became vital. In this work, we present our preliminary protocol for the meta-analytical procedures to software engineering and discuss our experience in this area with a presentation of a case study.

Our proposed guideline is based on the following existing sources in different scientific fields:

1. The Cochrane Reviewer's Handbook [15];
2. Guideline proposed by Egger et al. [16].
3. Procedures for performing systematic reviews by Kitchenham [12]

Our goal is to learn from the experience of evidence-based software engineering and application of meta-analysis in other scientific areas to start to develop a common protocol which, through the needed improvements will lead to widespread usage of meta-analysis for different primary studies in software engineering to a level of usage which is similar to one of the systematic literature reviews.

Chapter 2

Literature Review

Primary studies are good sources to see what has been done on a particular topic, but what if the results of different studies contradict each other? Or if there are so many reports about the same experiments, so instead of doing one more study, it is better to summarize existing ones?

Meta-analysis is a technique for aggregating information from different primary studies. It is widely applied in different areas such as medicine, psychology, epidemiology, business, and pedagogic. But in comparison with these disciplines, software engineering suffers from a lack of combining evidence. To identify the previous attempts made in applying the meta-analytical techniques in software engineering, a literature review was done. Only works from scientific journals and conferences were selected to further consideration to make this research more robust.

Based on the analysis of the state of the art, the following research questions were formulated for this literature review:

RQ1: What studies exist in the area of applying meta-analytical techniques for software engineering?

RQ2: What are the existing approaches for the meta-analysis in software engineering?

RQ3: How robust are these methods?

RQ4: Can these methods can be applied to the area of energy-efficient computing, and, if so, how?

This chapter is organized in the following way. Section 2.1 describes the process of paper search: which engine and search strings we used. Section 2.2 contains the inclusion criteria. The quality assessment is considered in Section 2.3. The results of the literature review are presented in Section 2.4.

2.1 Searching process

Since the name of the topic is a bit restricted, we decided to make it broader in order not to miss important results. Instead of searching for the papers related only to the meta-analysis, the Google Scholar was used to search for the papers related to the evidence combining in software engineering.

First of all, we generated search strings using the PICO approach [17]. The search strings are the following:

- meta-analysis AND software engineering;
- combining evidence AND software engineering;
- evidence-based software engineering.

With the help of snowballing [18], we found additional studies.

2.2 Paper selection

The focus on applying meta-analytical techniques in software engineering leads us to the following inclusion criteria:

- considering meta-analysis or other statistical techniques for aggregation the results from the experiments;
- focusing in the area of software engineering;
- written in English;
- published in journals or conference proceedings;
- peer-reviewed.

In this step, the search is done by the title and abstract. Papers not satisfying at least one point of inclusion criteria will not be included in further consideration.

2.3 Evaluation

After the selection process, the quality assessment starts. We need to evaluate the quality of studies passed the inclusion criteria. The categories with question for quality assessment is shown in Tables 2.1, 2.2, 2.3, 2.4, 2.5, and 2.6.

These questions aim to structure the papers found and retrieve the relevant information for future consideration. For each primary study, we look at which statement from each table it corresponds to and give it the appropriate score. After that, all points scored are summed up.

Table 2.1: Evaluation of the objective

Question	Number of points
The objective of a paper is implicitly stated and fully related to our problem	1
The objective of a paper is implicitly stated and partially related to our problem	0.5
The objective of a paper is not implicitly stated or it is difficult to determine the objective	0

Table 2.2: Evaluation of the research process

Question	Number of points
The methodology in a paper is implicitly described, all steps in the research process can be reproduced	1
The methodology in a paper is implicitly described, all steps in the research process can be reproduced, but minor details lack	0.5
The methodology in a paper is not implicitly described, or not all steps in the research process can be reproduced	0

2.4 Outcome of the literature review

For each query mentioned above, about 2-3 million papers were found. Due to the time constraints, only the first 150 papers from each query were considered for inclusion criteria. The papers that did not match the inclusion criteria were excluded. After passing the inclusion criteria, each paper was evaluated according to the questions stated in Tables 2.1, 2.2, 2.3, 2.4, 2.5, and 2.6. We considered only those, that got more than 2.5 points in the evaluation step. The number of papers found by each search string and passed the evaluation step is shown in Table 4.1.

To capture more relevant to our problem papers that were not returned by the search query, we applied snowballing technique. All studies found with

Table 2.3: Evaluation of the presented method

Question	Number of points
The comparison with other methods are presented, the advantages and limitations of the described method are given	1
The comparison with other methods are not presented, but the advantages and limitations of the described method are provided	0.5
There is no evaluation of a described method	0

Table 2.4: Evaluation of the results

Question	Number of points
Results are summarized, conclusions are drawn	1
Results are summarized, and it is possible to draw conclusions	0.5
Results are not are not summarized, or from the given summarization it is not possible to draw conclusions	0

help of this method was also exposed inclusion criteria and evaluation questions. The distribution of papers before and after applying snowballing is shown in Figure 2.1. Overall, 71 papers go for further consideration.

2.4.1 Studies classification

During the evaluation step, 4 major categories were identified:

Table 2.5: Evaluation of the reliability of presented method

Question	Number of points
Results are supported not only with qualitative, but with quantitative assessment	1
Results are supported only with qualitative assessment	0.5
Results are not proven	0

Table 2.6: Evaluation of the answering the research questions

Question	Number of points
The answers for the research questions are fully provided	1
The answers for the research questions are partially provided	0.5
There is no relationship between the stated research questions and provided answers	0

Table 2.7: Search queries

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

- general information about meta-analysis;
- meta-analysis in software engineering;
- research synthesis in software engineering;
- systematic literature reviews in software engineering.

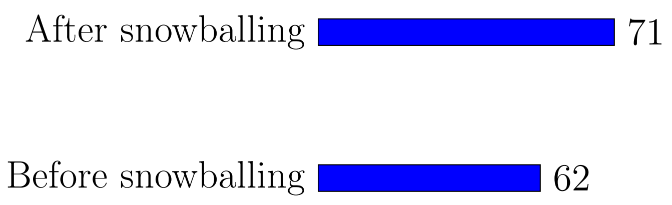
The number of papers related to each of these categories one can see in Table 2.8.

Table 2.8: Categories of papers

Category	Number of papers
General information about meta-analysis	9
Meta-analysis in software engineering	24
Research synthesis in software engineering	22
Systematic literature reviews in software engineering	16

Also, the distributions of the papers found is shown in the Figure 2.2. For the simplicity, the period from 2010 to 2020 was only taken. The number

Figure 2.1: Comparison of number of papers before and after applying snowballing method



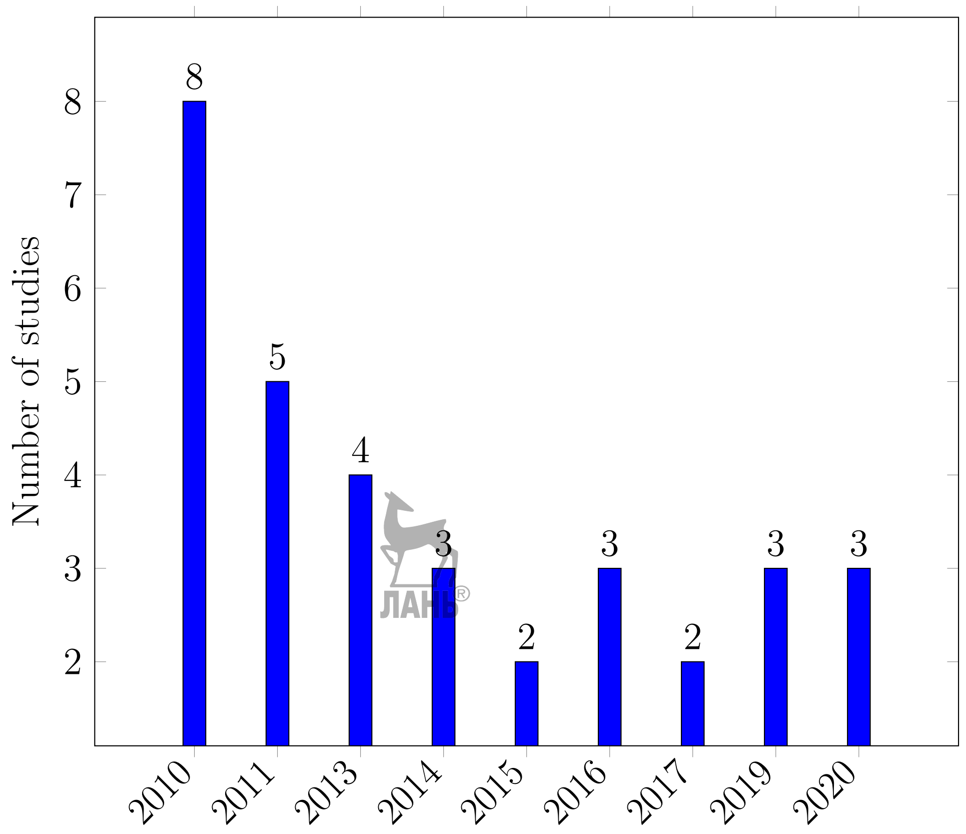
of relevant works in 2010 is significantly bigger than in 2019-2020. This trend is due to the increasing availability of technologies that can allow researchers to implement primary studies.

2.4.2 Analysis of RQ1

To answer RQ1 we need to determine what is the meta-analysis. The evolution of this technique will lead us to the current definition and methods used in this paradigm.

In 1926 Fisher [19] made one of the first attempts to aggregate results from different experiments. He considered the example the potato yield per plant. The land was divided into 36 patches, on which 12 varieties were grown. Each variety had 3 patches scattered over the area. Each patch was divided into three lines, one of which received, in addition to dung, a basal dressing only, the other two received additional dressings of sulphate and chloride of potash respectively. In this example Fisher investigated the effect of the manurial dressings. The significance test that was proposed for this case is now used in the modern meta-analysis.

Figure 2.2: Distribution of papers by years



The first appearance of the term “meta-analysis” was in a work of Glass (1976) [20] to refer to the statistical analysis of a large collection of results from individual studies for the purpose of integrating the findings. He noticed that the number of studies became so large, so instead of doing another primary study, he suggested aggregating the existing ones. The problem was in extracting the knowledge from myriads individual researchers. Meta-analysis connotes a rigorous alternative to the casual, narrative discussions of research studies. It helps to understand what attempts were taken to solve existing problems. In Glass’s opinion, a good review is the intellectual equivalent of original research.

Meta-analysis became a widely applied technique to see common effects across multiple individual studies [21]. It is used in different areas of medicine [22], psychology [23], etc. Only in the middle ’90s, it appeared in software engineering to generalize the results of empirical studies [8]–[10].

From the papers found about meta-analysis in software engineering we can derive two following categories:

- investigation of the possibility to apply different meta-analytical techniques to software engineering data;
- the application of meta-analytical techniques.

The difference between them is the following. The first category contains an investigation of existing approaches for meta-analysis and the way of their application to software engineering [9], [10], [24]. The second one provides the different real cases that step by step shows us how to apply the chosen by the author techniques [25]–[27].

2.4.3 Analysis of RQ2 and RQ3

We decided to combine the answers for these two questions since, during the consideration of different methods, we can analyze their robustness and reliability, advantages, and limitations.

Applying meta-analytical techniques to software engineering is straightforward, but the results can be very unstable, as pointed out by Miller [10]. He investigated various ideas from other disciplines for controlling the variation of the results. They are the following:

- make common criteria for the inclusion or exclusion of studies;
- standardize effect size measures;
- well define variables that transform heterogeneous data set into a group of homogeneous data sets;

- measure similar effect;
- develop study protocol before conducting meta-analysis.

Since the considered field is very young, not all mentioned points are practically proven.

The first works on the application of meta-analytical methods in software engineering consisted in the aggregation of the results of different experiments, not studies.

Such was the work of Succi et al. (2000) [28]. The researchers describe the usage of the "weighted estimator of a common correlation" technique to software measurement data. This meta-analytical technique was applied on 100 public domain Java projects to determine if it is possible to generalize project-specific significant correlations. The researchers used Fisher z-transform for correlation coefficients:

$$z_i = \frac{1}{2} \ln \frac{1 + r_i}{1 - r_i},$$

where r_i is the correlation coefficient of sample i. After this step, they obtained a generalised value representative of the normalised population correlation, z_+:

$$z_+ = w_1 z_1 + w_2 z_2 + \cdots + w_k z_k,$$

where k is the number of samples, w_i is the weight assigned to each element i:

$$w_i = \frac{n_i - 3}{\sum_{j=1}^{k}(n_j - 3)},$$

n_i is the size of sample i.

The weighted estimator for a common correlation is approximately normally distributed with a mean of ζ and a variance of $\frac{1}{N-3k}$. The estimator can be tested for significance by comparing the required significance level from the normal distribution with the result of the test statistic:

$$z_+ - \zeta\sqrt{N-3k}$$

The transformation of the z+ estimate back to standard correlation coefficients representative of the populations ρ-value can be performed using the inverse Fisher transform:

$$z^{-1}(x) = \frac{\exp^{2x} - 1}{\exp^{2x} + 1}$$

The results show that the cyclomatic complexity versus lines of code has a population correlation of 0.696, indicating that a relationship is present. Also, the analysis between the lines of code and the weighted module count shows the presence of a relationship with a population correlation of 0.610. These values confirm that the weighted estimators technique produces high correlations between data where a relationship can be existing.

Different techniques of meta-analysis for families of experiments in software engineering are also shown in the work of Djokic et al. (2001) [29]. The goal of this study is the production of an integrated statement of the findings of the many pieces of research done in the considered area. But the questions of the choice of technology, its adaptation, and implementation remain open.

To assess the quality of such papers that report the attempts of applying meta-analysis to the families of experiments, Kitchenham et al. (2001) [30] did the systematic literature on this topic. To do such studies more reproducible, they presented recommendations for reporting and meta-analyzing families of experiments.

But the problem of such studies that they are still primary ones since the meta-analytical techniques, in this case, is applied to experiments. Reporting the aggregated results from different papers (not experiments) requires conducting the systematic literature review, which makes the study the secondary one.

The different view of applying meta-analytical techniques is presented in the work of Yi et al. (2019) [31]. They suggested not only to generalize the findings across experiments using this technique but also to mine hypotheses. Based on the example with defect prediction techniques, they showed how to do it applying subgroup analysis [32]. However, there is no generalization of the considered approach to other branches of software engineering.

Hannay et al. (2009) [33] conducted separate meta-analyses for the Quality, Duration, and Effort to investigate the effectiveness of pair programming. In each experiment they used Hedges' g estimator [34] to calculate the effect sizes:

$$Hedges'g = \frac{\bar{x_1} - \bar{x_2}}{s_p},$$

where $\bar{x_1}$, $\bar{x_2}$ are the mean of one of the considered aspects (Quality, Duration, Effort) of two groups (pair and individual programming, respectively); s_p is the pooled standard deviation that is based on the standard deviations (s_1 and s_2) and sample sizes (n_1 and n_2) of both groups:

$$s_p = \sqrt{\frac{(n_1-1)s_1^2 + (n_2-1)s_2^2}{(n_1-1)+(n_2-1)}}$$

Since the researchers expected the heterogeneity of considered studies, they used both fixed-effect and random-effect models to calculate the resulting meta-analytic effect sizes [35]. In a fixed-effect analysis it is assumed that all the included studies share a common effect size. Under this model the population effect size and its variance are the following:

$$\bar{T} = \frac{\sum_{i=1}^{k} w_i T_i}{\sum_{i=1}^{k} w_i},\ v = \frac{1}{\sum_{i=1}^{k} w_i},$$

where $w_i = \frac{1}{v_i}$ is the weight assigned to the study i, v_i is the variance for the study i.

Assumption in a random-effect model states that the there is a distribution of true effect sizes [36]. In this case, the population effect size and its variance are the following:

$$\bar{T}^\star = \frac{\sum_{i=1}^{k} w_i^\star T_i}{\sum_{i=1}^{k} w_i^\star},\ v^\star = \frac{1}{\sum_{i=1}^{k} w_i^\star},$$

where $w_i^\star = \frac{1}{v_i^\star}$ is the weight assigned to the study i for the variance $v_i^\star = v_i + \tau^2$. τ^2 is the between-study variance:

$$\tau^2 = \begin{cases} \frac{Q - df}{C}, & \text{if } Q > df \\ 0, & \text{if } Q \le df \end{cases},$$

where $df = k - 1$ represents degrees of freedom, k is the number of samples. Q is the total variance:

$$Q = \sum_{i=1}^{k} w_i (T_i - \bar{T})^2.$$

C is a scaling factor that ensures that τ^2 has the same denomination as within-study variance:

$$C = \sum w_i - \frac{\sum w_i^2}{\sum w_i}$$

Using all these formulas, Hannay et al. calculated the population effect sizes for Quality, Duration, and Effort. The results showed a small significant positive overall effect of pair programming on quality, a medium significant positive overall effect on duration, and a medium significant negative overall effect on effort. The researchers concluded that the results are unstable due to the significant between-study variance and signs of publication bias among published studies on pair programming without pointing out the possible ways to overcome these obstacles.

In 2017 Umapathy et al. [37] also conducted a meta-analysis of pair programming. They used Hedges' g estimator to calculate the effect sizes. 28 independent effect sizes in the domains of programming assignments, exams, passing rates, and affective measures were used during the analysis. The researchers considered only a random-effect model for their meta-analytical study. The results of their research provide compelling evidence that pair programming can have a positive impact on students' programming assignment grades, exam scores, and persistence in computer programming courses. However, these results should be interpreted considering the following limitations:

- publication bias - about 40% of selected studies are from ACM digital library;
- different scales of grades for assignments and exams;
- the random-effect model was not statistically significant.

Both fixed-effect and random-effect models were also used in meta-analytical review on cross-project defect prediction [38]. The Hedges' g estimator was used to calculate the difference in means of f-measure, recall, precision, AUC (area under the curve) between cross-project defect prediction and within project defect prediction models. According to the results of this study, the cross-project defect prediction model can potentially achieve comparative predictive performance within project models when the factors influencing the performance are optimized. Both fixed and random effect models confirm one another in the majority of cases in the considered study. The inclusion of both models demonstrates the validity of the researchers' conclusions.

As one can note, almost the same techniques of the meta-analysis are used in the works found by us. Most of them used meta-analysis as a secondary study, but the review protocol differs from study to study. Such information like search strings, quality assessment criteria will make meta-analytical studies reproducible and help to avoid publication bias.

2.4.4 Analysis of RQ4

The meta-analytical models in software engineering, their advantages, and drawbacks are analyzed, so we can consider the applicability of such methods to the area of energy consumption.

Due to the rapid growth of different electronic devices, energy efficiency became a real concern. Many researchers consider this problem from the software engineering perspective. They investigate the impact of design choices [39], [40], refactoring techniques [41]–[43], caching [44], [45], task allocations methods [46], [47] on energy consumption. However, as pointed out Linares et al. [48], little is known about how much software developers are concerned about energy consumption:

- what aspects of energy consumption they consider important;
- what solutions they have in mind for improving energy efficiency.

Since the experiments related to measuring the energy consumption of different applications require suitable environment and equipment, for many researchers such investigations can be difficult to implement. Here, the aggregation of the results from the existing studies can help to conduct such experiments at a low cost.

From all papers we have analyzed during the search process, only one paper relates to the application of a meta-analysis to the problem of energy-efficient computing - “Impact of programming languages on energy consumption for mobile devices” [49]. This study shows the application of the meta-analytical technique - Hedges’ g estimator - to the area of energy consumption of different languages.

Chapter 3

Methodology

This research aims to investigate meta-analysis as a technique to see the common effects across effect sizes of multiple individual studies [21]. In this chapter, we are aimed to present our protocol for conducting a meta-analysis.

There are several approaches to meta-analysis depending on the desired outcome [15]:

- meta-analysis of dichotomous outcomes;
- meta-analysis of continuous outcomes;
- meta-analysis of ordinal outcomes;
- meta-analysis of counts and rates;
- meta-analysis of time-to-event outcomes.

Our case study is on energy consumption which is measured in Joules. Therefore, a meta-analysis of continuous outcomes will be used in this work.

In this chapter, the proposed protocol is described in Section 3.1. Section 3.2 and Section 3.3 contain information about the random-effects model and the Hedge's g estimator, which will be used for the further meta-analysis.

3.1 A Protocol for Meta-analysis in Software Engineering

The schema of our proposed protocol is presented in Figure 3.1. Since we are considering meta-analysis from the side of aggregation of the results from primary studies, our protocol is based on the one for systematic literature review done by Kitchenham [12].

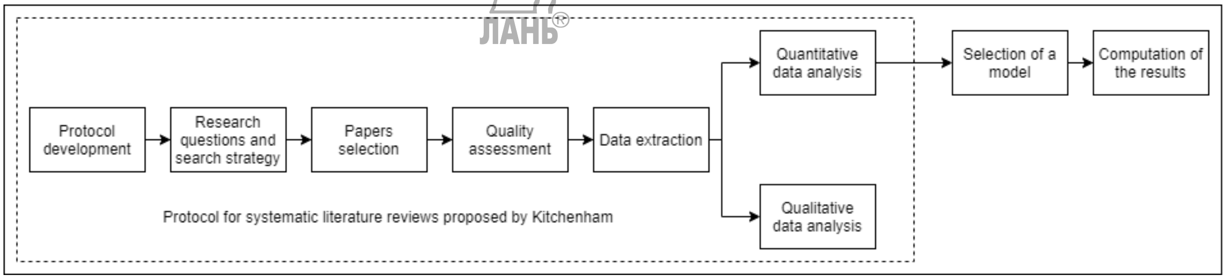

Protocol development
Research questions and search strategy
Papers selection
Quality assessment
Data extraction
Quantitative data analysis
Qualitative data analysis
Selection of a model
Computation of the results
Protocol for systematic literature reviews proposed by Kitchenham

Figure 3.1: Structure of the proposed protocol for performing meta-analysis on software engineering. The dotted line identify the component inherited from the overall structure of the systematic literature review of [13], [50].

The steps which are taken from the guideline for performing systematic literature reviews are the following.

Protocol development. A protocol is a way of defining the procedures needed for the systematic review such as establishing the research questions and search strategy, the study selection, quality assessment, data extraction, and data analysis processes. It also includes tasks for all members involved in the research.

Research questions and search strategy. At the beginning of any systematic review, the clearly stated research questions should be defined. The research questions will reflect the purpose of the investigation being conducted and define the search strategy. The search in systematic reviews aims to identify papers related to the stated research questions. During this process, one should define the engines, libraries, queries, and keywords, that will be used for the search.

Papers selection. After finding all relevant papers, a more detailed selection can be done. One can put inclusion and exclusion criteria. The relevance of the papers found bases on scanning the title and abstract information.

Quality Assessment. In this stage, the search should be done throughout the whole paper and the criteria for being included in the meta-analysis should be defined.

Data Extraction. From the papers passed the quality assessment, the information related to the results of conducted experiments should be derived. This information can be both qualitative and quantitative. The Goal, Question, Metrics approach (known as GQM) can help to derive the objective of primary studies [51].

Data analysis. This stage involves the aggregation of the results from primary studies. This process of aggregation refers to data synthesis. Data synthesis can be qualitative (as it is often presented through systematic review) and quantitative that can be done via meta-analysis. Applying aggregation techniques requires looking carefully at the studies that are going to be summarized: do they address the same research questions/objectives, how they vary in implementation, etc.

The following two steps are proposed by us to finalize the meta-analytical process.

Selection of the model. After the data analysis, we can see what the experiments were done in each selected study. Here, we need to understand the important aspect - the effect size of a study which is "a quantitative reflection of the magnitude of some phenomenon that is used for the purpose of addressing a question of interest" [52]. Effect sizes allow the comparison of effects in a single study and across studies in meta-analysis [53] and their estimators help us to generalize the knowledge going from sample to the population. To do a meta-analysis, we need to define the estimates of effect sizes. The three most used effect sizes are:

- mean difference (can be used if all effect sizes use the same outcome measurement scale),
- standardized mean difference (can be used if some effect sizes use the different outcome measurement scale),
- points biserial correlation.

Energy consumption can be given in different scales: Joules, kJoules, etc. Therefore, the standardized mean difference should be used.

Assigning the weight for each primary study according to its contribution is one of the main parts of meta-analytical process. The most common approach - the inverse-variance method where the weight is calculated as the inverse of the within-study variance in each model [54].

Then we can apply one of the two families of statistical procedures in meta-analysis [55]:

- fixed-effects model;
- random-effects model.

Both models are aimed to make inferences. But with the fixed-effects model, we assume that all studies are measuring the same, common effect size and the variability of the results is explained by the sampling error [36]. In this case, we are thinking that the data comes from the same population, which is more likely to happen with the families of experiments. With the random-effects model, we assume that all studies are measuring the different effect sizes and the variability of the results is explained by the different sources of errors. Studies obtained from a systematic literature review are likely to be more varied than those from a family of experiments.

Computation of the meta-analysis results. After assigning the weights and applying the families of statistical techniques, we can compute the summary effect, which leads to the derivation of the conclusion. If we are considering testing the hypotheses through a meta-analysis, so this stage is about looking at a *p*-value we got. The observed *p*-value helps to decide whether accept or reject the null hypothesis. If we are trying to generate new hypotheses, we can conduct subgroup analysis and based on its results, find the confounding factors.

The most understandable way of representing the result of a meta-analytical work is producing a forest plot [56]. With a help of a forest plot, one can see the information about the study, how many experiments it contains, size, mean, and standard deviation of samples, and the overall effect size.

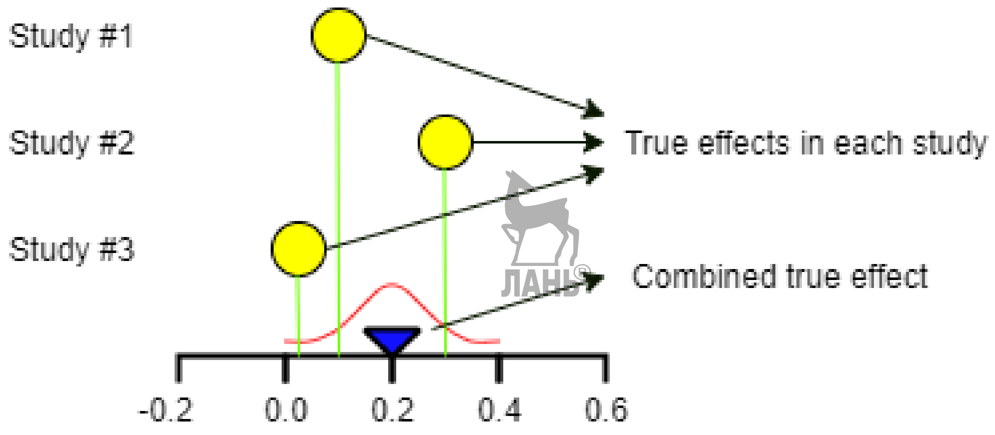
3.2 Random effect model

The random-effects model is the most suitable model to work with data coming from different sources as was already mentioned in Section 3.1. Therefore, it will be used for the meta-analysis upon the systematic literature review on energy consumption.

Under the random effect model there two following assumptions are taken:

- all studies are measuring the different effect sizes;
- the variability of the results is explained not only by the sampling error.

Figure 3.2: True effects in random effect model

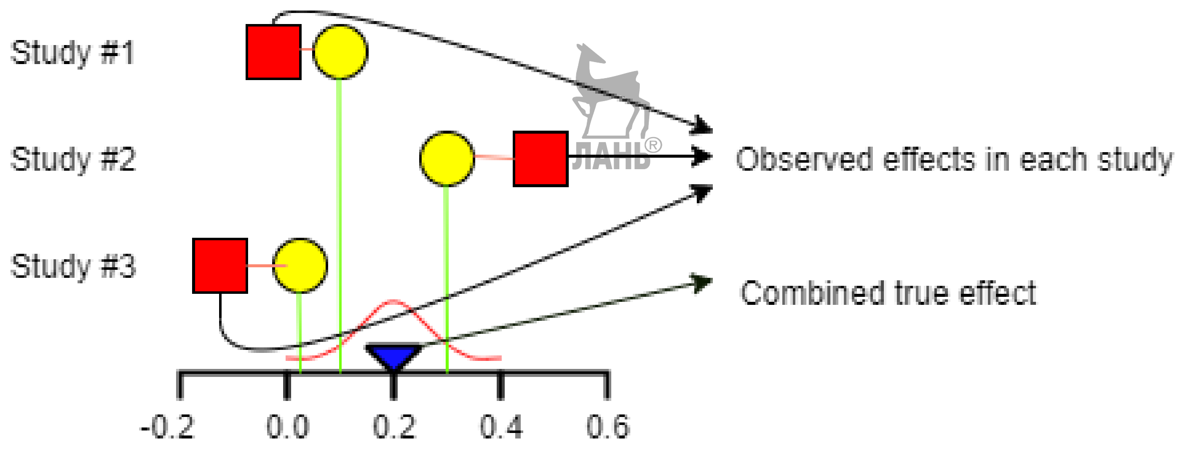


As in Figures 3.2 and 3.3, we assumed that the studies are measuring the different true effect sizes and the observed effect sizes vary due to the different sources of error. And any observed effect size can be described, using the following expression:

$$T_i = \mu + \epsilon_i + \zeta_i,$$

where T_i is the observed effect for the study i, μ is the true effect, ϵ_i is the within-study error, and ζ_i is the between-studies error. Now, the combined effect represents the mean of the population of true effects.

Figure 3.3: Observed effects in random effect model



The weight assigned for each study using inverse-variance approach is:

$$w_i^\star = \frac{1}{v_i^\star}.$$

Now the within-study variance for study i - $v_i^\star = v_i + \tau^2$ - explained not only by within-study, but also by between-study variance.

The between-study variance is computed as

$$\tau^2 = \begin{cases} \frac{Q-df}{C}, & \text{if } Q > df \\ 0, & \text{if } Q \le df \end{cases},$$

where df = (Number of studies) $-$ 1, and $C = \sum w_i - \frac{\sum w_i^2}{\sum w_i}$.

The weighted mean can be computed as:

$$\bar{T}^\star = \frac{\sum_{i=1}^k w_i^\star T_i^\star}{\sum_{i=1}^k w_i^\star}.$$

The variance of the combined effect is defined by the following expression:

$$v^\star = \frac{1}{\sum_{i=1}^{k} w_i^\star}$$

and the standard error of the combined error is:

$$SE^\star = \sqrt{v^\star}.$$

Knowing the combined effect and its standard error, we can define the 95% confidence interval:

$$LL^\star = \bar{T}^\star - 1.96 * SE^\star,$$

$$UL^\star = \bar{T}^\star + 1.96 * SE^\star,$$

where $LL^\star$ is the lower limit, $UL^\star$ is the upper limit for the random effect model.

Finally, the $Z^\star$-value could be computed using the formula below:

$$Z^\star = \frac{\bar{T}^\star}{SE^\star}.$$

The p-value for the one-tailed test is

$$p^\star = 1 - \Phi(Z^\star),$$

and for the two-tailed test:

$$p^\star = 2[1 - (\Phi(|Z^\star|))],$$

$\Phi(Z^{\star})$ is the standard normal cumulative function.

3.3 Hegdes' g estimator

The main challenge in any meta-analytical approach is the estimation of the effect size. The simplest in terms of calculations effect size is the mean difference. But if the data is given in different scales, it is better to standardize it by the standard deviation. For example, we have two groups: programs written in C and programs written in Java. The task is to estimate the mean difference in energy consumption between these groups. One of the possible ways to estimate this parameter is to take the difference of two means in the population where our two groups belong, standardized by the within-sample estimate of the population standard deviation. However, such estimation is likely to be biased due to the unknown population means (the energy consumption of all programs written in C or Java are unknown). To avoid this, we used Hedges' g estimator [57].

$$Hedges'g = \frac{\bar{x_1} - \bar{x_2}}{s_p} * \text{Correction factor},$$

where $\bar{x_1}$, $\bar{x_2}$ are the means of two groups; s_p is the pooled standard deviation that is based on the standard deviations (s_1 and s_2) and sample sizes (n_1 and n_2) of both groups:

$$s_p = \sqrt{\frac{(n_1 - 1)s_1^2 + (n_2 - 1)s_2^2}{(n_1 - 1) + (n_2 - 1)}}$$

Other estimators also exist:

- Cohen's d [58];

- Glass' D [53].

All of them have the same properties as Hedges' g in large samples (i.e., they are equivalent in the limit $(n_1 + n_2) \to \infty$), but Hedges' g has the best properties for small samples: Hedges suggested scaling the biased estimator on the value which depends on the degrees of freedom of two independent groups [34]:

$$\text{Correction factor (for scaling)} = 1 - \frac{3}{4(N-2)-1},$$

where N is the total sample size.

Due to the best properties among different estimators, the Hedges' g is chosen for our meta-analytical case.

Chapter 4

Sample Application

The number of applications for mobile devices is constantly growing. In the first quarter of 2020, Android users had about 2.56 million applications to choose from, and iPhone users - about 1.85 million [59]. Indeed, such sophisticated users also aim at a longer lifespan of batteries. Therefore, the problem of energy spent by mobile devices has turned into the problem of understanding how to develop applications that are "energy-friendly," and this problem has been already partially addressed in the literature on the impact of development approaches [39], [60] and of design choice [61] on energy consumption. Due to the great variety of existing approaches, frameworks, and tools for developing mobile applications, the problem of energy-efficient code can be explored from numerous perspectives. In our study, we focus on the impact of programming language on energy consumption, which we decided to investigate through a meta-analysis.

The investigation was done following the steps described in Section 3.1 from the previous chapter. Each taken step and results obtained in it is explained in a section named with respect to its name from the proposed protocol.

4.1 Protocol development

First of all, the protocol which defined procedures needed for systematic literature review and meta-analysis was developed. Also, before conduction any procedures, the following activities should be done:

1. make a search to understand whether the planned systematic literature review with a meta-analysis has already been done;

2. clarify the objective of a current study;

3. make sure that the conducted investigation will be relevant to the current state of the art in the considered topic (for example, whether the considered methods are still useful).

These activities were done by us (see Chapter 2) making the research reasonable. All the steps which should be reflected with the activities related to them in a protocol are described below in a separate chapters.

4.2 Research questions and search strategy

In our case study, the topic for the investigation was the energy consumption of mobile devices. Since, they are limited in a battery capacity, the problem of energy consumption turned into the investigation the energy consumption of the application written in different programming languages. To capture more information related to the scope of our research, we formulated the following research questions:

- What studies exist in the area of energy consumption in mobile devices?

- The adoption of which programming languages reportedly positively impacts energy consumption?

Using these research question and following PICO approach, we developed the search strings [49]:

- (energy OR power) (consumption OR efficiency) in software development;
- (energy OR power) (consumption OR efficiency) in (mobile devices OR smartphones)

The search was done manually using the Google Scholar engine. The number of papers found for each search query is presented in Table 4.1.

Table 4.1: Search queries

Search query	Number of papers
(energy OR power) (consumption OR efficiency) in software development	74
(energy OR power) (consumption OR efficiency) in (mobile devices OR smartphones)	129

We are aimed to investigate the following null hypotheses:

1. there is no significant difference in energy consumption between Java and C;
2. there is no significant difference in energy consumption between Java and C++.

The alternative ones are the following:

1. there is a significant difference in energy consumption between Java and C;

2. there is a significant difference in energy consumption between Java and C++.

These languages were taken by us for the investigation since all of them are used in mobile application development. C and C++ can be used for applications for iOS, and all of the selected languages are used in mobile application development for Android.

4.3 Paper selection

The focus on the relationship between programming languages and energy consumption leads us to the following three inclusion criteria:

- focusing on the impact of programming languages on energy consumption;
- considering C, C++, and Java;
- containing empirical results from conducted experiments.

In this step, the scanning of papers was done through the title and abstract. From all found articles, we managed to select only 10.

4.4 Quality assessment

Studies that contain full information about their experiments will help us to derive a plausible conclusion. Therefore, the studies which will be used for the further meta-analysis should meet the following criteria:

- the methodology is clearly described, all steps are reproducible;
- empirical results from conducted experiments are presented;

- the results obtained from the experiments are still relevant;
- containing all presented values in joules or watts.

The first criteria will help us to consider the papers with full information about the environment for the experiments which can be considered as a source of the variability. The second point will help us to avoid the future consideration of papers with only qualitative results. It will help us to define studies, for which we can apply meta-analytical techniques for mean differences to determine the best possible energy-efficient solutions. The third point will lead us to the studies which are still relevant and not out-of-date. The fourth criteria ensure that considered empirical values are related to the energy consumption problem and we can work with them as with continuous data.

During the quality assessment, we excluded duplicated and irrelevant studies. If a study misses the exact values, it is also removed from further consideration. Overall, for the further meta-analysis, only 6 papers remained:

[62] R. Pereira, M. Couto, F. Ribeiro, R. Rua, J. Cunha, J. P. Fernandes, 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, which we will refer to as **M1**; ;

[63] S. Abdulsalam, Z. Zong, Q. Gu, M. Qiu, "Using the Greenup, Powerup, and Speedup metrics to evaluate software energy efficiency", in 2015 Sixth International Green and Sustainable Computing Conference, IEEE, 2015, pp. 1-8 (**M2**);

[64] S. Abdulsalam, D. Lakomski, Q. Gu, T. Jin, Z. Zong, Program energy efficiency: The impact of language, compiler and implementation choices, in: International Green Computing Conference, IEEE, 2014, pp. 1–6 (**M3**);

[65] X. Chen, Z. Zong, Android app energy efficiency: The impact of language, runtime, compiler, and implementation, 2016 IEEE International Conferences on Big Data and cloud computing, social computing and networking, sustainable computing and communications, IEEE, 2016, pp. 485–492 (**M4**);

[60] W. Oliveira, R. Oliveira, F. Castor, "A study on the energy consumption of android app development approaches", in: 2017 IEEE/ACM 14th International Conference on Mining Software Repositories, IEEE 2017, pp. 42-52 (**M5**);

[66] M. Couto, R. Pereira, F. Ribeiro, R. Rua, J. Saraiva, Towards a green ranking for programming languages, in: Proceedings of the 21st Brazilian Symposium on Programming Languages, 2017, pp. 1–8 (**M6**).

4.5 Data Extraction

From the studies passed the quality assessment, we derived the necessary information like:

- objective of a primary study;
- methods used in experiments;
- programming languages that were compared;
- environment for the experiments done;

- results from the experiments done.

4.6 Data Analysis

Based on the information we have derived, we noticed that despite the objectives of selected papers were different (for example, the first primary study aimed to identify energy-efficient language, while the second one besides the languages considered structures), all of them aim to identify the energy-efficient solution. Therefore, the comparison is justified.

The methods which were used to measure energy consumption in provided experiments vary from simple sorting algorithms to the whole mobile applications. We considered each of the methods or applications as the same instance without specifications.

Besides C, C++, and Java, most primary studies also considered other programming languages like Python, Ruby, etc.

Tools used in each study were different:

- RAPL and jRAPL (**M1**);
- Intel Power Governor (**M2**);
- Intel Power Governor (**M3**);
- Android Energy Profiler (**M4**);
- the tools made available by Google (Project Volta) (**M5**);
- Intel RAPL and the Qualcomm TrepN frameworks (**M6**).

All the values from the experiments are presented in Joules, kJoules, or nJoules. More detailed information about the provided experiments can be found in each primary study.

4.7 Selection of a model

As was mentioned before, as an effect size for continuous outcomes we used standardized mean difference. The inverse-variance approach was chosen as a method for assigning weights for each primary study. It gives the minimal possible variability taking into consideration both sample size and its variance.

Since we have data that came from different primary studies, it is reasonable to use a random-effects model. Using the random-effects model we assume that all studies are measuring the different effect sizes and the variability of the results is explained by the different sources of errors, which is most likely to happen in the case of literature review.

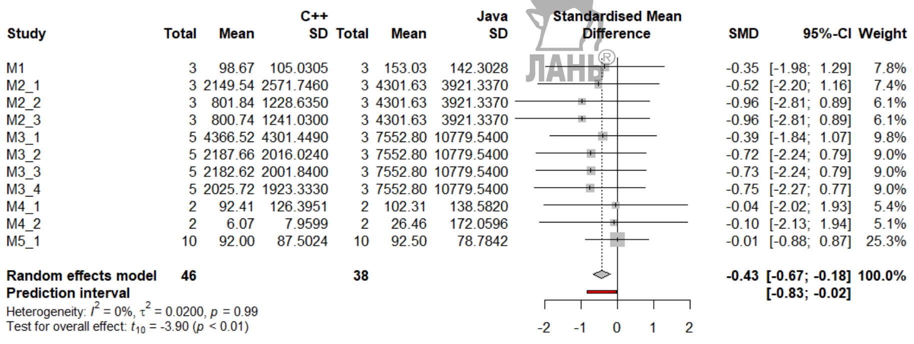
Since we are comparing the means of energy consumed by different programming languages with small sizes, we chose the Hedges' g estimator.

4.8 Computation the Result of Meta-analysis

To conduct our meta-analysis, we used the function "metacont" from library "meta" provided in R. Figures 4.1 and 4.2 show the mean differences of energy consumption between C, C++, and Java while developing mobile applications.

If the study contains more than one experiment, we differentiate them using underscore. For example, M2_1 and M2_1 in Figure 4.2. Column "To-

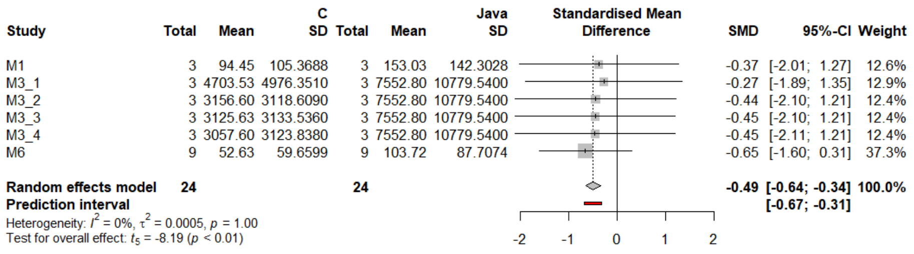
Figure 4.1: Forest plot for mean difference of energy consumption (in joule) between using C++ and Java



Study	C++ Total	C++ Mean	C++ SD	Java Total	Java Mean	Java SD	SMD	95%-CI	Weight
M1	3	98.67	105.0305	3	153.03	142.3028	-0.35	[-1.98; 1.29]	7.8%
M2_1	3	2149.54	2571.7460	3	4301.63	3921.3370	-0.52	[-2.20; 1.16]	7.4%
M2_2	3	801.84	1228.6350	3	4301.63	3921.3370	-0.96	[-2.81; 0.89]	6.1%
M2_3	3	800.74	1241.0300	3	4301.63	3921.3370	-0.96	[-2.81; 0.89]	6.1%
M3_1	5	4366.52	4301.4490	3	7552.80	10779.5400	-0.39	[-1.84; 1.07]	9.8%
M3_2	5	2187.66	2016.0240	3	7552.80	10779.5400	-0.72	[-2.24; 0.79]	9.0%
M3_3	5	2182.62	2001.8400	3	7552.80	10779.5400	-0.73	[-2.24; 0.79]	9.0%
M3_4	5	2025.72	1923.3330	3	7552.80	10779.5400	-0.75	[-2.27; 0.77]	9.0%
M4_1	2	92.41	126.3951	2	102.31	138.5820	-0.04	[-2.02; 1.93]	5.4%
M4_2	2	6.07	7.9599	2	26.46	172.0596	-0.10	[-2.13; 1.94]	5.1%
M5_1	10	92.00	87.5024	10	92.50	78.7842	-0.01	[-0.88; 0.87]	25.3%
Random effects model	**46**			**38**			**-0.43**	**[-0.67; -0.18]**	**100.0%**
Prediction interval								**[-0.83; -0.02]**	

Heterogeneity: $I^2 = 0\%$, $\tau^2 = 0.0200$, $p = 0.99$
Test for overall effect: $t_{10} = -3.90$ ($p < 0.01$)

Figure 4.2: Forest plot for mean difference of energy consumption (in joule) between using C and Java



Study	C Total	C Mean	C SD	Java Total	Java Mean	Java SD	SMD	95%-CI	Weight
M1	3	94.45	105.3688	3	153.03	142.3028	-0.37	[-2.01; 1.27]	12.6%
M3_1	3	4703.53	4976.3510	3	7552.80	10779.5400	-0.27	[-1.89; 1.35]	12.9%
M3_2	3	3156.60	3118.6090	3	7552.80	10779.5400	-0.44	[-2.10; 1.21]	12.4%
M3_3	3	3125.63	3133.5360	3	7552.80	10779.5400	-0.45	[-2.10; 1.21]	12.4%
M3_4	3	3057.60	3123.8380	3	7552.80	10779.5400	-0.45	[-2.11; 1.21]	12.4%
M6	9	52.63	59.6599	9	103.72	87.7074	-0.65	[-1.60; 0.31]	37.3%
Random effects model	**24**			**24**			**-0.49**	**[-0.64; -0.34]**	**100.0%**
Prediction interval								**[-0.67; -0.31]**	

Heterogeneity: $I^2 = 0\%$, $\tau^2 = 0.0005$, $p = 1.00$
Test for overall effect: $t_5 = -8.19$ ($p < 0.01$)

tal" represents the number of available methods or applications written in the selected languages in each primary study.

The mean and standard deviation in each case was approximated using values provided for every method or application in these primary studies. I^2, as a measure for quantifying heterogeneity, describes the percentage of variability in point estimates [67]. The low value of I^2 in our plots means that the selected studies can be considered as homogeneous [68].

The diamond at the bottom of each plot represents the summary effect. In Figures 4.1 and 4.2, it does not cross the zero. This means that we can reject the null hypotheses that:

- there is no significant difference in energy consumption between C and

Java;

- there is no significant difference in energy consumption between C++ and Java.

Summing up, we can reject the original null hypothesis that there is no significant difference in energy consumption between Java and C/C++.

Chapter 5

Evaluation and Discussion

We have described the preliminary potential guideline for meta-analytical review and presented it in Figure 3.1. Also, we applied it in our case study (Chapter 4) to understand its reliability. It goes without mentioning that this work has some limitations.

One of the main limitations is that the case study is limited to the articles that we found by using only the Google Scholar engine. It is not guaranteed that we incorporated all papers related to the considered subject. From the examination of the papers found, it became clear that energy consumption is very diverse in devices for which we measure it, in methods, measurements, and research approaches. Then, the limited number of papers that we have incorporated, indeed, lowers the power of conducted case study.

During the application of a proposed protocol to our case study, we found several critical issues preventing the widespread of meta-analysis in software engineering. The first issue that we faced was the lack of a standard in the metrics being used even for very similar experiments. For example, if a study has considered the amount of energy consumption as a metric, the measurement could be given in Watts (not in Joules) despite that Watts are used to measure the power consumption. The variety of metrics used is the following:

- energy consumption in Joules [63];
- energy consumption in Watts [69];
- energy consumption in mAmper*hour [70];
- battery usage in mWatt*hour [71];
- percentage of energy consumption [44].

The second issue was the fast obsolescence of the data in primary studies. We investigated the impact of programming languages on energy consumption for mobile devices. In this research, we considered C, C++, and Java as programming languages for mobile application development [65], [66], but nowadays, among the most used languages for mobile application development are JavaScript and Kotlin, for which we, unfortunately, did not find a sufficient number of papers: only by one paper for the comparison of JavaScript [62] and Kotlin [72] with other languages. Such a problem is not so often met in medicine or chemistry. For example, substances, vitamins, or drugs used in the previous century can be also used now. It can be not the case in most experiments in software engineering. Most of the methods used in the previous decade can be obsolescent for now. Software engineering is a fast-growing field, and 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.

For each issue we have found during our case study we suggested our solutions.

Problem with obsolescent data can be solved just by removing irrelevant data from a consideration. But this serves as a temporal solution. The long-term plan involves interaction between industry, researchers, and people interested in software engineering which will allow sharing the current trends and creative ideas contributing to the development of this field in all directions. The existence of a strong community as Cochrane in healthcare helps to overcome the problem about the relevance of data.

The best solution for the issue related to the variety of metrics would be to calculate the metrics that we need from the raw data. Furthermore, we emphasize the importance of sharing the raw data. It will make the decision-making process in studies more transparent and help to avoid publication bias and duplication of the experiments. However, in most studies, such data is not publicly available. Therefore, we suggest other possible ways of solving this problem:

- make clusters of studies according to the metrics they used and conduct separate meta-analyses on each cluster;
- apply the vote-counting approach to the simple, more generic questions, like "Does Java consume more energy?"

But both suggested solutions will prevent us from the whole view of the current state.

Also, we did not consider in detail different effect sizes that can be used during a meta-analysis. The choice of the suitable metric depends on the needs of the experiment and the statistical characteristics of considered samples (e.g. their size, distribution). In our case, the mean difference was the most suitable effect size. However, further investigation of this matter could give us ways to improve the quality of research using other possible options.

In addition to the various effect sizes, we also did not consider different approaches for assigning the weights for primary studies (for example, sample size method) and describe only random-effects model for a meta-analysis since in case with data coming from a systematic literature review it is highly likely to meet the different sources of variability in presented results. However, assuming that sharing of our experience in the area of application meta-analytical techniques to software engineering data will be useful, we supported our preliminary guideline with a case study.

With regard to our case study, the number of studies in the considered area is limited, thus not allowing to draw conclusions; we have noticed that Java showed a significant increase in consumption. Our study has therefore indicated that it would be extremely profitable to perform additional studies that might reinforce such tendency, thus making it non-significant or, on converse, could present results going in opposite direction.

Chapter 6

Conclusion

In this work, we have considered the problem of the application of meta-analysis to software engineering experimentation. We started with a literature review, investigating the works already done in this area. With a help of a literature review, we have found that meta-analysis was used mostly as a primary study than as a secondary one. Also, we did not find a unified guideline for conducting a meta-analysis as a secondary study. Therefore, to take a step toward systematizing this process we suggested our preliminary protocol for a meta-analysis upon the systematic literature review.

To see the work of the proposed protocol we decided to conduct a case study on the impact of programming languages on energy consumption. Since the data for a meta-analysis came from different studies, we chose the random-effects model. Due to the small number of studies that we found, we used the Hedges' g as the estimator of the effect size.

During the case study, we faced two problems for which we proposed our solutions. Needless to say, the systematization of meta-analysis in software engineering is a long-term plan. Different effect sizes and statistical models, the standards for conducting a meta-analysis should be done for this purpose. But we have already made the first attempt starting with the presentations of our preliminary guideline, 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.

Bibliography cited

[1] J. Siegmund, N. Siegmund, and S. Apel, "Views on internal and external validity in empirical software engineering", in *Proceedings of the 37th International Conference on Software Engineering - Volume 1*, ser. ICSE '15, Florence, Italy: IEEE Press, 2015, pp. 9–19, ISBN: 9781479919345.

[2] M. D'Ambros, M. Lanza, and R. Robbes, "Evaluating defect prediction approaches: A benchmark and an extensive comparison", *Empirical Software Engineering*, vol. 17, no. 4, pp. 531–577, 2012.

[3] S. Lessmann, B. Baesens, C. Mues, and S. Pietsch, "Benchmarking classification models for software defect prediction: A proposed framework and novel findings", *IEEE Transactions on Software Engineering*, vol. 34, no. 4, pp. 485–496, 2008.

[4] F. A. Fontana, P. Braione, and M. Zanoni, "Automatic detection of bad smells in code: An experimental assessment.", *J. Object Technol.*, vol. 11, no. 2, pp. 5–1, 2012.

[5] N. Moha, Y.-G. Guéhéneuc, L. Duchien, and A.-F. Le Meur, "Decor: A method for the specification and detection of code and design smells", *IEEE Transactions on Software Engineering*, vol. 36, no. 1, pp. 20–36, 2009.

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

[7] I. J. Mojica, B. Adams, M. Nagappan, S. Dienst, T. Berger, and A. E. Hassan, “A large-scale empirical study on software reuse in mobile apps”, *IEEE software*, vol. 31, no. 2, pp. 78–86, 2013.

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

[9] 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.

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

[11] 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.

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

[13] 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.

[14] 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.

[15] 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.

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

[17] M. B. Eriksen and T. F. Frandsen, "The impact of patient, intervention, comparison, outcome (pico) as a search strategy tool on literature search quality: A systematic review", *Journal of the Medical Library Association: JMLA*, vol. 106, no. 4, p. 420, 2018.

[18] 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.

[19] R. A. Fisher, "Statistical methods for research workers", in *Breakthroughs in statistics*, Springer, 1992, pp. 66–70.

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

[21] I. Shrier, J.-F. Boivin, R. W. Platt, R. J. Steele, J. M. Brophy, F. Carnevale, M. J. Eisenberg, A. Furlan, R. Kakuma, M. E. Macdonald, *et al.*, "The interpretation of systematic reviews with meta-analyses: An objective or subjective process?", *BMC medical informatics and decision making*, vol. 8, no. 1, pp. 1–8, 2008.

[22] A.-B. Haidich, "Meta-analysis in medical research", *Hippokratia*, vol. 14, no. Suppl 1, p. 29, 2010.

[23] F. L. Schmidt, "What do data really mean? research findings, meta-analysis, and cumulative knowledge in psychology.", *American psychologist*, vol. 47, no. 10, p. 1173, 1992.

[24] M. Shepperd, "Combining evidence and meta-analysis in software engineering", in *Software engineering*, Springer, 2010, pp. 46–70.

[25] V. Garousi, G. Giray, E. Tüzün, C. Catal, and M. Felderer, "Aligning software engineering education with industrial needs: A meta-analysis", *Journal of Systems and Software*, vol. 156, pp. 65–83, 2019.

[26] M. I. Azeem, F. Palomba, L. Shi, and Q. Wang, "Machine learning techniques for code smell detection: A systematic literature review and meta-analysis", *Information and Software Technology*, vol. 108, pp. 115–138, 2019.

[27] H. Moayedi, M. Mosallanezhad, A. S. A. Rashid, W. A. W. Jusoh, and M. A. Muazu, "A systematic review and meta-analysis of artificial neural network application in geotechnical engineering: Theory and applications", *Neural Computing and Applications*, vol. 32, no. 2, pp. 495–518, 2020.

[28] 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.

[29] S. Djokic, G. Succi, W. Pedrycz, and M. Mintchev, “Meta analysis—a method of combining empirical results and its application in object-oriented software systems”, in *OOIS 2001*, Springer, 2001, pp. 103–112.

[30] B. Kitchenham, L. Madeyski, and P. Brereton, “Meta-analysis for families of experiments in software engineering: A systematic review and reproducibility and validity assessment”, *Empirical Software Engineering*, vol. 25, no. 1, pp. 353–401, 2020.

[31] 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.

[32] X. Sun, J. P. Ioannidis, T. Agoritsas, A. C. Alba, and G. Guyatt, “How to use a subgroup analysis: Users’ guide to the medical literature”, *Jama*, vol. 311, no. 4, pp. 405–411, 2014.

[33] 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.

[34] D. Enzmann, “Notes on effect size measures for the difference of means from two independent groups: The case of cohen’sd and hedges’g”, *January*, vol. 12, p. 2015, 2015.

[35] M. Borenstein, L. V. Hedges, J. P. Higgins, and H. R. Rothstein, “A basic introduction to fixed-effect and random-effects models for meta-analysis”, *Research synthesis methods*, vol. 1, no. 2, pp. 97–111, 2010.

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

[37] 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.

[38] 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.

[39] L. Corbalan, J. Fernandez, A. Cuitiño, L. Delia, G. Cáseres, P. Thomas, and P. Pesado, “Development frameworks for mobile devices: a comparative study about energy consumption”, in *Proceedings of the 5th International Conference on Mobile Software Engineering and Systems*, 2018, pp. 191–201.

[40] A. Noureddine and A. Rajan, “Optimising energy consumption of design patterns”, in *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*, IEEE, vol. 2, 2015, pp. 623–626.

[41] M. Gottschalk, M. Josefiok, J. Jelschen, and A. Winter, “Removing energy code smells with reengineering services”, *INFORMATIK 2012*, 2012.

[42] J. J. Park, J.-E. Hong, and S.-H. Lee, “Investigation for Software Power Consumption of Code Refactoring Techniques”, in *SEKE*, 2014, pp. 717–722.

[43] R. Morales, R. Saborido, F. Khomh, F. Chicano, and G. Antoniol, “Earmo: An energy-aware refactoring approach for mobile apps”, *IEEE Transactions on Software Engineering*, vol. 44, no. 12, pp. 1176–1206, 2017.

[44] K. Dutta and D. Vandermeer, “Caching to reduce mobile app energy consumption”, *ACM Transactions on the Web (TWEB)*, vol. 12, no. 1, pp. 1–30, 2017.

[45] I. Malavolta, K. Chinnappan, L. Jasmontas, S. Gupta, and K. A. K. Soltany, “Evaluating the Impact of Caching on the Energy Consumption and Performance of Progressive Web Apps”, in *7th IEEE/ACM International Conference on Mobile Software Engineering and Systems 2020*, 2020.

[46] Y. Zhang, X. Hu, and D. Z. Chen, “Task scheduling and voltage selection for energy minimization”, in *Proceedings 2002 Design Automation Conference (IEEE Cat. No. 02CH37324)*, IEEE, 2002, pp. 183–188.

[47] K. Li, X. Tang, and K. Li, “Energy-efficient stochastic task scheduling on heterogeneous computing systems”, *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 11, pp. 2867–2876, 2013.

[48] M. Linares-Vásquez, G. Bavota, C. Bernal-Cárdenas, R. Oliveto, M. Di Penta, and D. Poshyvanyk, “Mining energy-greedy api usage patterns in android apps: An empirical study”, in *Proceedings of the 11th Working Conference on Mining Software Repositories*, 2014, pp. 2–11.

[49] 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.

[50] B. A. Kitchenham, D. Budgen, and P. Brereton, *Evidence-based software engineering and systematic reviews*. CRC press, 2015, vol. 4.

[51] E. Hernandes, A. Zamboni, S. Fabbri, and A. D. Thommazo, “Using gqm and tam to evaluate start-a tool that supports systematic review”, *CLEI Electronic Journal*, vol. 15, no. 1, pp. 3–3, 2012.

[52] K. Kelley and K. J. Preacher, “On effect size.”, *Psychological methods*, vol. 17, no. 2, p. 137, 2012.

[53] C. O. Fritz, P. E. Morris, and J. J. Richler, “Effect size estimates: Current use, calculations, and interpretation.”, *Journal of experimental psychology: General*, vol. 141, no. 1, p. 2, 2012.

[54] F. Marın-Martınez 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.

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

[56] S. Lewis and M. Clarke, “Forest plots: Trying to see the wood and the trees”, *Bmj*, vol. 322, no. 7300, pp. 1479–1480, 2001.

[57] L. V. Hedges, “Distribution theory for glass’s estimator of effect size and related estimators”, *journal of Educational Statistics*, vol. 6, no. 2, pp. 107–128, 1981.

[58] M. E. Rice and G. T. Harris, “Comparing effect sizes in follow-up studies: Roc area, cohen’s d, and r.”, *Law and human behavior*, vol. 29, no. 5, p. 615, 2005.

[59] *Number of apps available in leading app stores as of 1st quarter*, Visited on 23rd July 2020, 2020. [Online]. Available: https://www.statista.com/statistics/276623/%20number-of-apps-available-in-leading-app-stores.

[60] W. Oliveira, R. Oliveira, and F. Castor, “A study on the energy consumption of android app development approaches”, in *2017 IEEE/ACM 14th International Conference on Mining Software Repositories (MSR)*, IEEE, 2017, pp. 42–52.

[61] A. McIntosh, S. Hassan, and A. Hindle, “What can android mobile app developers do about the energy consumption of machine learning?”, *Empirical Software Engineering*, vol. 24, no. 2, pp. 562–601, 2019.

[62] 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.

[63] S. Abdulsalam, Z. Zong, Q. Gu, and M. Qiu, “Using the greenup, powerup, and speedup metrics to evaluate software energy efficiency”, in *2015 Sixth International Green and Sustainable Computing Conference (IGSC)*, IEEE, 2015, pp. 1–8.

[64] S. Abdulsalam, D. Lakomski, Q. Gu, T. Jin, and Z. Zong, “Program energy efficiency: The impact of language, compiler and implementation choices”, in *International Green Computing Conference*, IEEE, 2014, pp. 1–6.

[65] X. Chen and Z. Zong, “Android app energy efficiency: The impact of language, runtime, compiler, and implementation”, in *2016 IEEE international conferences on big data and cloud computing (BDCloud), social computing and networking (socialcom), sustainable computing and com-*

munications (sustaincom)(BDCloud-socialcom-sustaincom), IEEE, 2016, pp. 485–492.

[66] M. Couto, R. Pereira, F. Ribeiro, R. Rua, and J. Saraiva, “Towards a green ranking for programming languages”, in *Proceedings of the 21st Brazilian Symposium on Programming Languages*, 2017, pp. 1–8.

[67] J. P. Higgins and S. G. Thompson, “Quantifying heterogeneity in a meta-analysis”, *Statistics in medicine*, vol. 21, no. 11, pp. 1539–1558, 2002.

[68] J. P. Higgins, S. G. Thompson, J. J. Deeks, and D. G. Altman, “Measuring inconsistency in meta-analyses”, *Bmj*, vol. 327, no. 7414, pp. 557–560, 2003.

[69] P. A. Rego, F. A. Trinta, M. Z. Hasan, and J. N. de Souza, “Enhancing offloading systems with smart decisions, adaptive monitoring, and mobility support”, *Wireless Communications and Mobile Computing*, vol. 2019, 2019.

[70] D. Li and W. G. Halfond, “An investigation into energy-saving programming practices for android smartphone app development”, in *Proceedings of the 3rd International Workshop on Green and Sustainable Software*, 2014, pp. 46–53.

[71] B. Westfield and A. Gopalan, “Orka: A new technique to profile the energy usage of android applications”, in *2016 5th International Conference on Smart Cities and Green ICT Systems (SMARTGREENS)*, IEEE, 2016, pp. 1–12.

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