e-Informatica

software engineering journal

2022 | volume 16 | issue 1

e-Informatica

e-Informatica

software engineering journal

2022 volume 16 issue 1

e-Informatica

Wrocław University
of Science and Technology

Faculty of Information and Communication Technology, Department of Applied Informatics
Wrocław University of Science and Technology,
50-370 Wrocław, Wybrzeże Wyspiańskiego 27, Poland

e-Informatica Software Engineering Journal
www.e-informatyka.pl, DOI: 10.37190/e-inf

Editorial Office Manager: Wojciech Thomas

Typeset by Wojciech Myszka with the LaTeX 2$_\varepsilon$ Documentation Preparation System

OFICYNA WYDAWNICZA POLITECHNIKI WROCŁAWSKIEJ
Wybrzeże Wyspiańskiego 27, 50-370 Wrocław
www.oficyna.pwr.edu.pl;
e-mail: oficwyd@pwr.edu.pl; zamawianie.ksiazek@pwr.edu.pl

ISSN 1897-7979

Editorial Board

Martin Shepperd (Brunel University London, UK)
Rini van Solingen (Drenthe University, The Netherlands)
Miroslaw Staron (IT University of Göteborg, Sweden)
Tomasz Szmuc (AGH University of Science and Technology Kraków, Poland)
Guilherme Horta Travassos (Federal University of Rio de Janeiro, Brazil)
Adam Trendowicz (Fraunhofer IESE, Germany)
Burak Turhan (University of Oulu, Finland)
Rainer Unland (University of Duisburg-Essen, Germany)
Sira Vegas (Polytechnic University of Madrit, Spain)
Corrado Aaron Visaggio (University of Sannio, Italy)
Bartosz Walter (Poznan University of Technology, Poland)
Dietmar Winkler (Technische Universität Wien, Austria)
Bogdan Wiszniewski (Gdańsk University of Technology, Poland)
Krzysztof Wnuk (Blekinge Institute of Technology, Sweden)
Marco Zanoni (University of Milano-Bicocca, Italy)
Jaroslav Zendulka (Brno University of Technology, The Czech Republic)
Krzysztof Zieliński (AGH University of Science and Technology Kraków, Poland)

Contents

Self-Adaptation Driven by SysML and Goal Models – A Literature Review
Amal Ahmed Anda, Daniel Amyot 220101
Analysis of Factors Influencing Developers' Sentiments in Commit Logs: Insights from Applying Sentiment Analysis
Rajdeep Kaur, Kuljit Kaur Chahal, Munish Saini 220102
How good are my search strings? Reflections on using an existing review as a quasi-gold standard
Huynh Khanh Vi Tran, Jürgen Börstler, Nauman bin Ali, Michael Unterkalmsteiner 220103
Examining the Predictive Capability of Advanced Software Fault Prediction Models – An Experimental Investigation Using Combination Metrics
Pooja Sharma, Amrit Lal Sangal 220104
A Systematic Review of Ensemble Techniques for Software Defect and Change Prediction
Megha Khanna 220105
A Comparison of Citation Sources for Reference and Citation-Based Search in Systematic Literature Reviews
Nauman bin Ali, Binish Tanveer 220106
Microservice-Oriented Workload Prediction Using Deep Learning
Sebastian Ştefan, Virginia Niculescu 220107
Empirical AI Transformation Research: A Systematic Mapping Study and Future Agenda
Einav Peretz-Andersson, Richard Torkar 220108
Reporting Consent, Anonymity and Confidentiality Procedures Adopted in Empirical Studies Using Human Participants
Deepika Badampudi, Farnaz Fotrousi, Bruno Cartaxo, Muhammad Usman 220109
Reuse in Contemporary Software Engineering Practices – An Exploratory Case Study in A Medium-sized Company
Xingru Chen, Deepika Badampudi, Muhammad Usman 220110

e-Informatica Software Engineering Journal, Volume 16, Issue 1, 2022, pages: 220101, DOI: 10.37190/e-Inf220101

Self-Adaptation Driven by SysML and Goal Models – A Literature Review

Amal Ahmed Andaa*, Daniel Amyot*
*School of Electrical Engineering and Computer Science, University of Ottawa
aanda027@uottawa.ca, damyot@uottawa.ca

Abstract

Background: *Socio-cyber-physical systems* (SCPSs) are a type of cyber-physical systems with social concerns. Many SCPSs, such as smart homes, must be able to adapt to reach an optimal symbiosis with users and their contexts. The Systems Modeling Language (SysML) is frequently used to specify ordinary CPSs, whereas goal modeling is a requirements engineering approach used to describe and reason about social concerns.
Objective: This paper aims to assess existing modeling techniques that support adaptation in SCPSs, and in particular those that integrate SysML with goal modeling.
Method: A systematic literature review presents the main contributions of 52 English articles selected from five databases that use both SysML and goal models (17 techniques), SysML models only (11 techniques), or goal models only (8 techniques) for analysis and self-adaptation.
Result: Existing techniques have provided increasingly better modeling support for adaptation in a SCPS context, but overall analysis support remains weak. The techniques that combine SysML and goal modeling offer interesting benefits by tracing goals to SysML (requirements) diagrams and influencing the generation of predefined adaptation strategies for expected contexts, but few target adaptation explicitly and most still suffer from a partial coverage of important goal modeling concepts and of traceability management issues.

Keywords: adaptation, cyber-physical systems, goal modeling, socio-technical systems, SysML, traceability, uncertainty

1. Introduction

Cyber-physical systems (CPSs) are systems that tightly "integrate physical, software, and network aspects in a sometimes adverse physical environment" [1]. They are composed of hybrid components such as hardware (e.g., sensors, devices, and networks) and software, which can even be integrated at runtime. Horváth [2] observes that the complexity, emergent properties, and adaptability of CPSs have increased substantially in the past decade in order for CPSs to be compatible with different components and changes in their surrounding environment. Moreover, CPSs are characterized by a high level of uncertainty, which is difficult to address with current design methods [2, 3].

Socio-cyber-physical systems (SCPSs) are a type of CPSs that are also socio-technical systems, where human concerns are considered during the development process (i.e., at design time) and during execution (i.e., at runtime). Many SCPSs should ideally be able

Submitted: 26 Apr. 2021; Revised: 11 Aug. 2021; Accepted: 26 Sep. 2021; Available online: 15 Dec. 2021

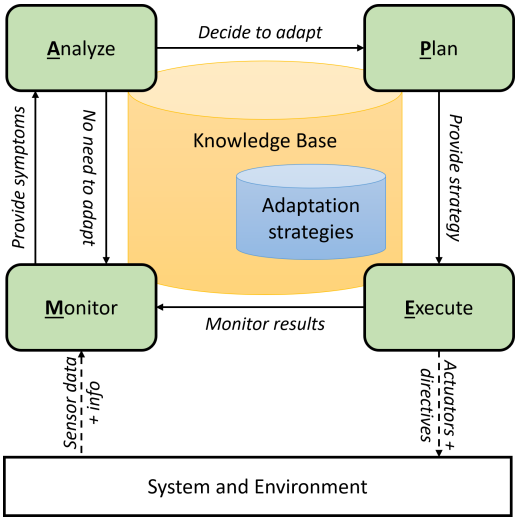


Figure 1. Self-adaptation activities: MAPE-cycle (adapted from [8])

to adapt to changing conditions in order to reach an optimal symbiosis with users (and other stakeholders) and their contexts [2]. Examples include existing systems such as air traffic control systems, and emerging ones such as smart homes/cities [4], human-oriented services exploiting the Internet of Things (IoT) [5], adaptive Systems of Systems (SoS) [6], and Industry 4.0 [7].

Many CPSs and SCPSs monitor their environments, which enables them to detect contexts where the system may no longer accomplish what it was intended to do or meet its goals. *Self-adapting* systems are capable of detecting such situations and change their own behavior accordingly. Kephart and Chess [8] divided the adaptation process into four different activities (Monitor, Analyze, Plan, and Execute), collectively called *MAPE-cycle* and illustrated in Figure 1. These activities share some knowledge and interact with the rest of the system and its environment. The general functionality of each activity is as follows:

- *Monitor*: Gathers information about monitored system features and the environmental context.
- *Analyze*: Analyzes the information and determines whether to activate the planning process and what information should be passed on to it.
- *Plan*: Selects the most suitable adaptation strategy (some might be predefined) depending on the information provided by the analysis activity.
- *Execute*: Executes the selected adaptation strategy, with impact on the system and the environment that again must be monitored.

Some challenges coming with adaptive systems were identified and addressed by Bocanegra et al. [9], Muñoz-Fernández et al. [10], and Horváth [2] with Model-Driven Engineering (MDE) approaches. In particular, *goal modeling*, which enables the description of stakeholder and system goals together with their relationships, is used as part of many MDE approaches to facilitate traceability, deal with uncertainty, manage stakeholder objectives, and support requirements engineering at design time and at runtime. Bocanegra et al. [9] further stated that integrating MDE and goal-oriented requirements engineering is a promising way to solve many self-adaptation challenges. In addition, Muñoz-Fernández et al. suggest that traceability supports reasoning about system behavior and the changes or events that triggered a specific adaptation at runtime [10]. Although traceability in SCPSs was

identified by Bordeleau et al. as a challenge [11], ideally, for self-adaptive systems to be realizable, traceability should be managed synchronously from the beginning (goals) to the end of the system (code). Even if Bocanegra et al. cited the lack or loss of information while transforming (goal) models to code as one weakness of many MDE approaches [9], the benefits of goal modeling in this context tend to outweigh its drawbacks.

MDE is the basis of many Systems Engineering (SE) methods meant to deal with complex, technological obstacles and the heterogeneous nature of multidisciplinary systems [1, 12, 13]. One opportunity here is to include stakeholder goals into targeted systems via modeling. In the same context, the *Systems Modeling Language* (SysML) is a language standardized by the Object Management Group (OMG) [14, 15] and the International Organization for Standardization [16] to support SE methods. SysML reuses part of the Unified Modeling Language (UML), including use case, sequence, activity, and state machine diagrams, and modifies other types of diagrams to produce block and internal block diagrams. Moreover, SysML adds parametric and requirements diagrams to facilitate the connection between system components and their requirements [15]. SysML enables the modeling of software and hardware components as well as their relationships, in a way that simplifies their design [17] and reduces their complexity [18, 19].

SysML modelers can connect requirements to other model elements such as use cases, test cases, and blocks using a few diagram types. Yet, SysML lacks important "social" modeling concepts for SCPSs such as goals and stakeholders' objectives [20]. Cross-disciplinary model fusion and flexible model integration are known to be challenging [21], but a multi-level modeling approach is still a promising avenue in contexts such as Systems of Systems and adaptive SCPS [1, 22, 23]. There exist many goal modeling languages that can help here, including KAOS [24, 25], *i** [26], and the Goal-oriented Requirement Language (GRL) [27], part of the User Requirements Notation (URN) [28]. GRL is discussed further here because this is the only internationally standardized goal modeling language so far and one of the few languages that supports indicators, which enable monitoring in an adaptive context. GRL has also been used in the modeling and design of large CPSs, some with a social aspect but without adaptation (e.g., for collaborative CPSs [29]), some that adapt but without a social aspect (e.g., for unmanned aircraft systems [30]), and some that model adaptive SCPSs (e.g., for smart homes [31]).

GRL helps capturing stakeholders (roles, organizations, systems, etc., collectively named actors), their intentions (goals, softgoals, or tasks), their relationships (AND/OR decomposition, positive/negative contributions, dependencies), and indicators to measure intention satisfaction based on external evidence. Figure 2 illustrates a GRL model of a simplified hybrid car's engine system and its related user's goals. The system needs to select which engine(s) to activate so that speed and distance from other cars are properly controlled while ensuring that the user's concerns (comfortable driving, measured via a vibration indicator, and costs minimized) are satisfied. GRL *actors* (illustrated as ellipses ⬭) are used to capture the system itself as well as its users and other stakeholders. Their *goals* (⬭) should be fulfilled, while their *softgoals* (⬭) point out the non-functional or quality aspects desired. *Tasks* (⬭) capture the alternatives that the system has to chose from in the plan activity. *Indicators* (⬭) are used to monitor internal/external conditions and convert this information into satisfaction levels. Intentions can also be AND/OR-decomposed (⟶), whereas an arrow (⟶) with a negative/positive weight (normalized to a value between −100 and +100) represents the contribution of some element to another one. The color coding (the greener, the better) and the numbers above intentions (between 0 and +100) indicate the current level

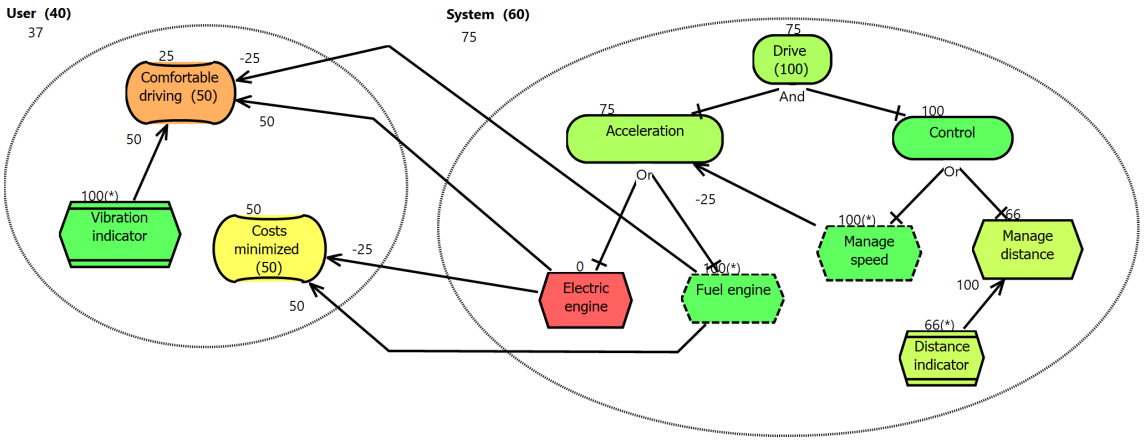


Figure 2. Simplified goal model (in GRL) of a hybrid car's engine

of satisfaction in a given context called a *strategy*, whose initial values (*) are propagated to the other elements and actors based on an automated propagation algorithm [32].

The other goal modeling languages also support distinctive goal and quality concepts, together with AND/OR decomposition, and assignment of goals to actors (or "agents" in other languages). For example, Figure 3 briefly highlights the syntax of the popular KAOS language [24, 25], also used in several adaptive SCPS approaches. Unlike GRL, KAOS neither supports indicators nor contributions, but it includes explicit concepts for *obstacles* and *threats*, akin to goals or tasks that would be the source of negative contributions in GRL.

GRL was initially created to support requirements engineering activities during development; however, it can also be used in a runtime adaptation context [33]. GRL supports a system's dynamic adaptation by connecting goals with requirements, feeding indicators from external sources of information, and providing comprehensive alternative strategies/tasks supporting trade-off analysis [20].

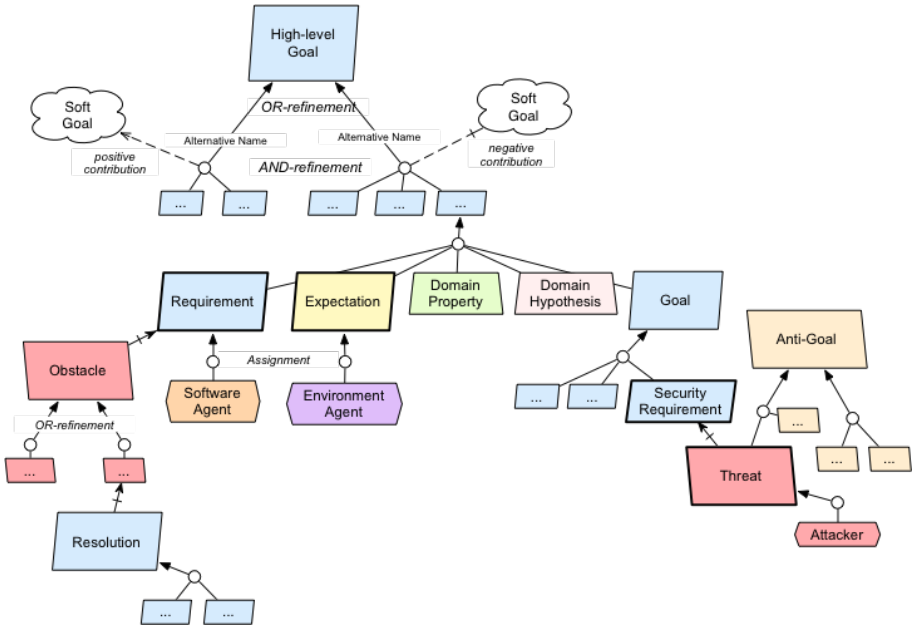


Figure 3. KAOS goal modeling concepts and syntax (from `https://kaos.info.ucl.ac.be/`)

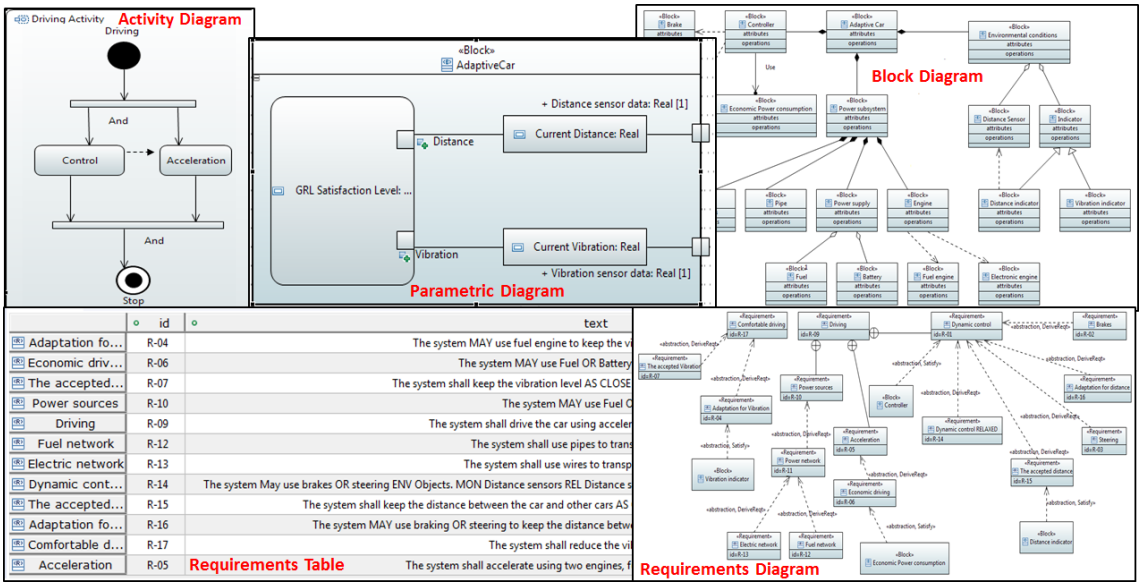


Figure 4. Some SysML diagrams of a simplified hybrid car's engine

To briefly illustrate SysML (more commonly known than goal modeling languages), Figure 4 highlights important types of diagrams and tables for adaptive systems, with content relevant to the simplified hybrid car's engine example. A requirements table contains natural language requirements (functional and non-functional) with their attributes. This information, together with traceability and other relationships (e.g., derivation or satisfaction), can be visualized in a requirements diagram. A block definition diagram shows the system components (software and hardware) as a static structure of blocks with their attributes, operations, constraints, interfaces, and relationships (e.g., containment or generalization). Parametric diagrams describe mathematical functions for constraint blocks. Activity diagrams present the dynamic flow of activities that describe behavioral aspects. SysML also supports other diagrammatic views for use cases, sequences, state machines, internal blocks, and packages.

In order to adjust requirements engineering and modeling techniques for modeling SCPSs, and especially adaptive ones, it is necessary to consider the integration between SysML (which supports system specification) and a goal modeling language that supports social concepts and comprehensive decision making [3, 20, 34, 35]. The objective of this paper is to survey and analyze articles that relate to goal and SysML modeling and their support for self-adaptive systems, especially SCPSs. The focus is on the techniques that combine *both* SysML and goal modeling in that context (and this literature review is quite exhaustive in that regard), but some key techniques that only use SysML or goal modeling are also discussed in order to enable useful comparisons.

The main contributions of this literature review are its consolidation of existing work and an outline of achievements and challenges related to existing techniques. This review is also original in that although there exist general literature surveys and mappings about adaptive systems [36–38], CPS modeling [39], goal modeling [40], and SysML [41, 42], none really goes deep into the combination of goal and SysML modeling for emerging types of adaptive socio-cyber-physical systems. This review will help academics understand what contributions and gaps exist in that research area. It will also raise the awareness of practitioners in the existing techniques for self-adaptation based on SysML and/or goal modeling, while providing guidance in the selection of appropriate techniques in their development context.

This paper is structured as follows. Section 2 describes the methodology used to plan, conduct, and report on the literature review. Section 3 provides the selected resources on Goal/SysML integration, as well as their evaluation methods, concepts, and objectives. In addition, the methods presented for self-adaptation are classified into different categories, and the self-adaptation concepts and dimensions are evaluated and extracted. Results are compared in order to provide insight into their aims, achievements, and challenges. Related literature reviews are discussed in Section 4. The limitations and threats to the validity of this review are explained in Section 5. Finally, Section 6 concludes the paper.

2. Methodology

Based on the systematic literature review approach of Kitchenham and Charters [43], we followed three common steps: planning, conducting, and reporting on the review.

2.1. Planning the review

This step includes setting research questions, identifying the search scope and strategy, as well as formulating quality assessment criteria and data extraction items.

2.1.1. Setting the study goal and research questions

SCPSs combine stakeholder goals, software, and hardware components. Some SCPSs may also be self-adaptive. In this context, the objective of this review is to investigate the possible modeling methods that 1) integrate goal and SysML models, or 2) support self-adaption via SysML only or via the integration of SysML and goal models. The research questions for this objective include two main questions, each of which containing secondary questions.

RQ1. What are the existing methods that integrate goal-oriented models with SysML models?
- **SQ1.1.** Why have these integrations been proposed?
- **SQ1.2.** How do the methods integrate the two types of models?

RQ2. What are the collected methods that support self-adaptation?
- **SQ2.1.** How do the methods support self-adaptive systems?
- **SQ2.2.** What are the roles that each model plays in this adaptation support?

2.1.2. Identifying the search scope and strategy

The search scope combines three areas: 1) the studies that are relevant to goal models and SysML models together, independently of support for self-adaptive systems; 2) the principal studies that use SysML models to support adaptive systems; and 3) important studies (selected manually) that use goal models to support adaptive systems, as a comparison point outside the SysML world.

The searches were more exhaustive for the first two areas (involving SysML) than for the last one. The main strategy for the first two areas is based on automatic searches performed on popular databases. As the topic of the last step (goal models for adaptive systems) is quite wide and already well covered in the literature, a selection based on a domain expert's opinions and on forward citation searches (i.e., recent papers citing previously selected papers,

including from the same authors) was used to highlight the main trends and contributions without being exhaustive.

Data Sources. Five important electronic databases were used to discover scientific papers related explicitly to the research questions: Elsevier's *Scopus* and Clarivate Analytics' *Web of Science* are two wide-scope search engines, *IEEE Xplore* and the *ACM Digital Library* are covering the two main societies publishing on systems modeling, and finally *Google Scholar* is a catch-all academic search engine. Note that Google Scholar discriminates less than the other databases in terms of paper quality, and its query language is less powerful than the ones of the other engines. Together, these databases provide a very high coverage of the literature related to SysML and goal-oriented modeling.

Search Queries. Many synonyms of goal models were used in order to cover the most common goal modeling languages (*i**, GRL, URN, and KAOS). Adaptive, adaptation, socio-technical, and socio cyber were also considered as quasi-synonyms in our context. The automatic search was conducted in two phases, first with a focus on SysML/goal integration and second on SysML models and self-adaptation. Table 1 specifies each search conducted with the related query. Because Google Scholar retrieved thousands of papers (with many false positives), we eliminated adaptation/social terms from the second query to ensure papers only integrating goal and SysML models would be included in our dataset, as we excluded goal models from the fourth query to focus on non-goal-oriented SysML adaptation. These abstract queries were transformed to concrete queries for the different languages used by the databases. With Google Scholar (which retrieves thousands of papers with many false positives), as we were mainly interested in using its results as a complement to the other (and more reliable) databases while minimizing the effort needed to prune out irrelevant papers, only the first 60 papers returned by each query were further inspected. The number 60 was selected based on observing an increasingly high density of false positives as we went down the lists of results, especially after 40 results. The return on the time investment after 60 results was deemed ineffective.

Table 1. Queries used for Goal/SysML integration (1 and 2) and self-adaptation with SysML (3 and 4)

No.	Search	Query
1	SysML and goal models	TITLE−ABS−KEY(SysML AND ("goal oriented" OR "goal model" OR "i star" OR istar OR KAOS OR "user requirements notation" OR URN OR adaptation OR adaptive OR "Socio cyber" OR "Socio technical"))
2	SysML and goal models, using Google Scholar	(SysML AND ("goal oriented" OR "goal model" OR "i star" OR istar OR KAOS OR "user requirements notation" OR URN))
3	SysML models and self-adaption	TITLE−ABS−KEY(SysML AND (adaptation OR adaptive))
4	SysML models and self-adaption, using Google Scholar	(SysML adaptation OR adaptive) −"goal model"

Inclusion and Exclusion Criteria. We used inclusion and exclusion criteria to select which papers to keep. The inclusion criteria were:

1. The article is peer reviewed (no book, patent, tutorial, magazine, or gray literature).
2. The article is written in English.
3. For queries 1 and 2 in Table 1, the article provides or clarifies methods about Goal/SysML integration.
4. For queries 3 and 4 in Table 1, the article includes methods using SysML for self-adaptation support.

The exclusion criteria were:

1. The article duplicates (or is a subset of) another paper in terms of the Goal/SysML integration or self-adaptation methods.
2. The article does not provide any information related to our research questions.

A paper satisfying one of the exclusion criteria or not satisfying all of the inclusion criteria was excluded. Some papers did discuss a combination of goal modeling with SysML modeling, but not their integration or self-adaptation. For example, Tueno Fotso et al. [44] integrate KAOS-like AND/OR goal models with a subset of SysML for the generation of Event-B specifications, but not for adaptive systems.

2.1.3. Quality assessment criteria

We used the checklist in Table 2 to provide a qualitative assessment of each study.

Table 2. Quality assessment criteria and possible values

Code	Quality	Qualitative Score
C1	Is the problem specified clearly?	Yes, No, Partially
C2	Is a method provided?	Yes, No, Partially
C3	Is the presented method original?	Yes, No, Partially
C4	Is the method detailed?	Yes, No, Partially
C5	Is the method complete?	Yes, No, Partially
C6	Is a case study provided?	Yes, No
C7	Does the case study clearly illustrate the method?	Yes, No, Partially
C8	Is self-adaptation handled?	Yes, No, Partially
C9	Is self-adaptation specified in detail?	Yes, No, Partially

2.1.4. Identifying data extraction items

Table 3 details the data items extracted from each selected paper, together with their related research questions from the planning stage.

Table 3. Data extracted from each paper

Questions	Data item
Documentation	Title, Year, Publisher, Authors, Database engine
RQ1	Goal model, Automation, Integrated diagrams, Method realization
RQ1, RQ2	Goal model, Goal concepts, Goal analysis, Objective, Development phase, Environment of the method, Realization type
RQ2	Quality attribute, Realization dimension, Adapted object, Temporal features, Modeling dimension (Goal), Why SysML model

For data documentation, from each article, we collected the title, the publication year, the publisher, the authors' names, and the database engine used to retrieve the article. To answer research questions RQ1 and RQ2, information was collected by posing the following questions:

1. Are the goals integrated with SysML as a model, or as text/requirement?
2. What are the diagram types that were used in the integration?
3. Are common goal modeling concepts considered in the integration? These include goal types, qualitative/quantitative contributions between different types of goals, and goal dependencies.
4. Is goal analysis considered in the integration?
5. What is the purpose of the integration?
6. Is the integration fully automatic, semi-automatic, or manual?
7. What method was utilized when the integration was done?
8. What are the non-functional requirements (NFRs) that were the focus of attention?
9. For which development phase was the integration done?
10. How are the methods realized? This includes the adaptation type and approach, if any.
    a) How is the adaptation type explained from these different perspectives?
        i. When are the alternatives handled? (*closed*: at development phase; or *open*: at runtime).
        ii. Is the method model-based or not?
    b) How is the adaptation approach realized? This is grouped into the following:
        i. The decision-making process decides the adaptation and chooses between alternatives (analysis and selection processes) [37]. Is it *static* and created at development time as rules, or *dynamic* using an equation or algorithm?
        ii. The adaptation approach is based on the phase of the system in which the adaptation approach was included [3]. Is *Making adaptation* included at development time or *Achieving adaptation* included at runtime using learning approaches?
11. What is the object affected by the adaptation process? Three different sets of information related to this object are:
    a) The layer in which the object is located (application, middleware, network, hardware, etc.);
    b) The impacted object (architecture, subsystem, service, component, parameter, etc.); and
    c) The adaptation action, which could be *weak* or *strong* depending on the effect and cost of adaptation. For example, *strong adaptation* includes adaptations that add or change the system architecture or components behaviors at runtime. This result exists because much system time and effort is consumed to achieve the adaptation goals. A *weak adaptation* is related to any inexpensive change. (Cost-impact)
12. When does the adaptation happen before specific events (Proactive) or after specific events (Reactive)? (Temporal adaptation)
13. Does the system monitor specific features or does it monitor its environment continually using sensors? (Temporal monitoring)
14. Is human intervention involved in the adaptation process?
15. How is the adaptation done? For example, using a specific language or algorithm.
16. Goal:
    a) Does the number of goals change at runtime? (Evolution)

   b) Do system goals remain unchanged, change within constraints, or change without constraints? (Flexibility)
   c) How many goals are considered in the adaptation process? (Multiplicity)
   d) Are the goals dependent or independent of each other? (Dependency)
17. What is the reason for the adaptation? (Change)
   a) Is the source of the change *external* (environmental) or *internal* (system)?
   b) Is the change due to functional requirements, to non-functional requirements, or to a technical reason?
18. Was the time spent for adaptation process guaranteed or not? (Timeliness)
19. What was the reason for using a SysML model to specify self-adaptive systems?

2.2. Conducting the review

After having identified the queries and databases engines, the study was conducted along four steps: search, screening, data extraction, and quality assessment.

2.2.1. Search methods

The retrieval of papers to satisfy the conditions we identified included two search methods: 1) goals/ SysML integration and 2) SysML and self-adaptation support. The search method for **goals/SysML integration** consisted of the following steps:

1. The first query was used to capture the papers from the Scopus, IEEE, Web of Science and ACM database engines.
2. Because Google scholar retrieves many irrelevant articles, we used the following strategy on this engine:
   a) The first query was applied to retrieve papers first using the "anywhere" option and second using the "in the title" option. The latter retrieved only 10 papers while the former retrieved 4,200 papers. We considered only the first 60 papers (which are ranked by relevance).
   b) To include further relevant papers, we conducted the second query using the "anywhere" option. This returned 660 papers, and again only the first 60 papers were considered.

The search method for **SysML and self-adaptive systems** consisted of the following steps:

1. The third query was used to capture the papers from the Scopus, IEEE, Web of Science and ACM database engines.
2. Using Google Scholar, the fourth query was used with the two options, "anywhere" and "in the title" separately. The first option led to 3,860 papers, and only the first 60 papers were included in our dataset. The second option retrieved only three papers.

To get a non-exhaustive overview of the role that goals have played in supporting self-adaptation features, our dataset was supplied with 12 primary articles using goals in self-adaptive systems by an expert and two more papers using forward search (snowballing).

2.2.2. Screening

The papers retrieved through the previous step were screened for relevance. The exclusion and inclusion criteria were applied on each paper based on abstracts and conclusions. If the information was insufficient to decide whether a paper was relevant or not, its introduction

and method sections were read. If the information was still insufficient to decide, the full-text of the article also was read. Discussions between both authors were required for nine papers because it was difficult to decide whether they were relevant or not.

2.2.3. Result

Figure 5 illustrates the results of the screening process, with the numbers of results returned for each query available in Table 4. From the goals/SysML search, out of 444 papers returned by the search engines, 33 papers were deemed relevant whereas 411 were rejected, including 58 duplicates[1]. For the SysML and adaptation search, 334 papers (including 18 duplicates) were considered but only eleven articles met our criteria. Most of the papers were found by Scopus, with much duplication by the Web of Science, and the other engines added little value. In addition to these 44 papers, eight papers on goal modeling for adaptive systems

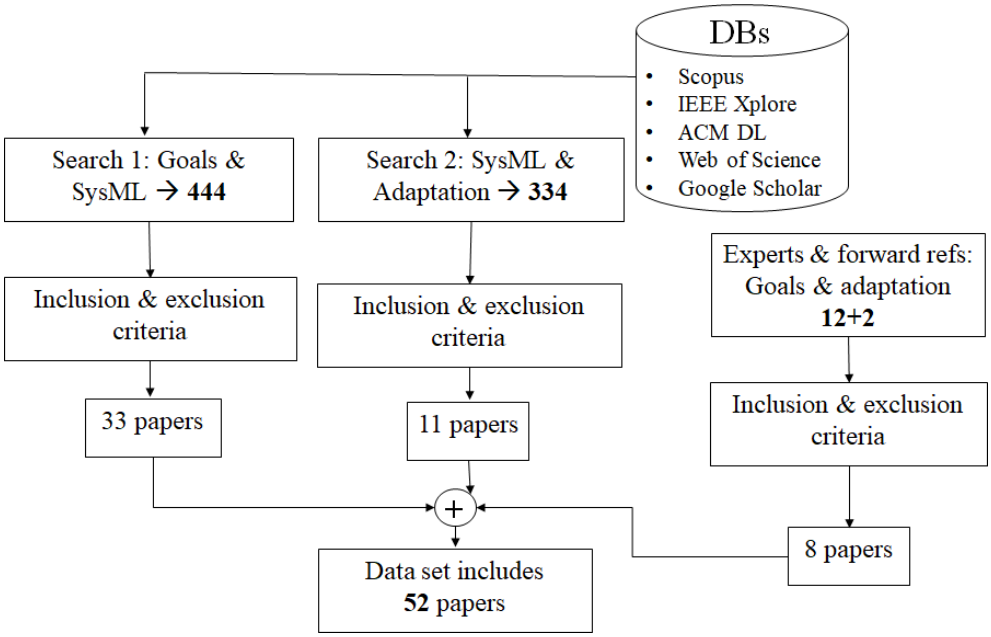


Figure 5. Result of the screening process

Table 4. Results of the searches per databases

Database	Goal and SysML	SysML and adaptation
Scopus	70	62
IEEE Xplore	60	67
Web of Science	28	17
ACM Digital library	23	62
Google Scholar (title only)	23	6
Google Scholar (any field)	60+60 (from 4,200)	60+60 (from 3,860)
Google Scholar (with goals-SysML)	60+60 (660)	-
Total (with duplicates)	**444**	**334**
Total (unique papers)	**386**	**316**

[1]A table with the accepted and rejected papers is available online at http://bit.ly/SysML-Goal-SLR

were included at the suggestion of an expert and with a simple forward search (we did not aim to be exhaustive here). This resulted in 52 papers that were eligible for the analysis.

2.2.4. Data extraction

For each study, we extracted the data items mentioned in Table 3. Extracting this data was done iteratively from the selected studies to accumulate information concerning our research questions.

2.2.5. Quality Assessment Process

The collected studies were compared against the criteria listed in Table 2. We did not evaluate how good the articles were (beyond ensuring they were not coming from a predatory source), but we did evaluate how useful each would be to the study. The result of the quality assessment against the criteria explained in Table 2 is provided in Tables A1 and A2 in Appendix A.

3. Discussion

We classify and present the selected studies in a way that will enable answering our research questions accurately. We split the discussion into five subsections: integration methods, adaptation support methods, adaptation assessment, and challenges.

3.1. Integration methods

To answer the first research question (RQ1: What are the existing methods that integrate goal-oriented models with SysML models?), we used the 33 articles retrieved by the Goal/SysML search (Table A1). We classify the studies according to the applied methods and current objectives. A total of 17 methods, named M1 to M17 in Table 5, are proposed by these articles. The types of goal modeling languages and SysML diagrams used in each method are also listed in Table 5, and the main papers in each collection are highlighted in **bold**.

3.1.1. Languages and diagrams involved

For each method, we extracted the goal modeling language and SysML diagrams used (Table 5). Any additional model was considered out of the scope of the study.

The most commonly used SysML diagrams in the 17 proposed methods are requirements diagrams and block diagrams, in that order. All presented methods but three (M1, M11, M14) connected goals or goal models with requirements diagrams, while nearly half the methods (M1, M2, M9, M11, M12, M13, M17) used block diagrams in their integration.

From a goal model perspective, several different languages were used. The most popular languages in these methods are KAOS [24] (M5, M7, M8) and GRL [28] (M2, M4, M17). OMG's Business Motivation Model [76] was also mentioned once in M6 and RELAX [77] once in M8. Several methods only used textual goals or non-functional requirements (NFRs), with some integrating them more formally as SysML stereotypes (M15 and M16). Instead of integrating goal models themselves, Ingram et al. [45] (M1) used goal model analysis

Table 5. Selected studies and their methods used
(FG = Functional goals; NFG = Non-functional goals)

Research studies	Code	Goal language	SysML diagrams
Ingram et al. [45]	M1	Fault tolerance strategies	Block (and dependency relationships)
Amyot et al. [20]	M2	GRL	Requirements, block
Vanderperren and Dehaene [46]	M3	NFG (no specific notation)	Requirements, use case
Ozkaya [47]	M4	GRL	Requirements
Matoussi et al. [48], Laleau et al. [49]	M5	KAOS (FG)	Requirements
Cui and Paige [50]	M6	Business Motivation Model	Requirements
Gnaho et al. [51, 52], Mammar and Laleau [53], Bousse [54]	M7	KAOS (FG, NFG)	Requirements
Ahmad et al. [55], Ahmad et al. [56–58], Ahmad [59], Ahmad and Bruel [60, 61], Belloir et al. [62]	M8	KAOS (FG, NFG), RELAX	Requirements
Apvrille and Roudier [63, 64], Roudier and Apvrille [65]	M9	Textual goals/NFR	Requirements, block, state machine, parametric
Tsadimas et al. [66]	M10	NFR diagram	Requirements
Spyropoulos and Baras [67]	M11	Textual NFRs	Block, parametric
Badreddin et al. [68]	M12	Textual goals (based on GRL)	Requirements, block, use case
Fan et al. [69]	M13	Textual goals	Requirements, block, activity
Wang [70]	M14	Textual goals (with AND/OR decomposition)	Use case
Lee et al. [71]	M15	Requirements diagrams with goal stereotypes	Requirements
Maskani et al. [72]	M16	Requirements diagrams with goal stereotypes	Requirements
Anda and Amyot [73], **Anda and Amyot [74]**, Anda [75], **Anda and Amyot [31]**	M17	GRL	Requirements, block, internal block, parametric, state machine

results in their integration to increase the confidence of system designers when defining the system architecture. From another perspective, Anda and Amyot [74] (M17) used GRL models and enabled analysis through arithmetic expressions.

3.1.2. Objectives of the integration

In our investigation, extracting information about the adaptation objective is different from extracting information for the integration itself. To answer the secondary question SQ1.1 (Why have these integrations been proposed?), we clustered the studies according to their objectives to figure out which ones were most frequently used in the literature to justify a goal/SysML integration.

Table 6 reports on seven main objectives, together with their related methods and articles. The management of *uncertainty and adaptation* (which is concerned with whether the information being monitored is reliable enough to justify adaptation decision, and with what adaptation will help satisfy goals the best), has attracted the highest number of studies (14), with four different methods. However, the *architecture selection and modeling* objective (which is important at design time to find suitable trade-offs between various non-functional goals such as performance, cost, and reliability of systems and adaptations) is targeted by a more varied set of methods (6). These two important objectives are followed by *formal validation and verification* (to ensure safety, liveness, security, and other such properties), and *traceability* (to manage change effectively and to ensure coverage during quality assurance). Other objectives were mentioned only by one or two papers, namely *process improvement* (e.g., so goals are more explicitly considered), *requirements visualization* (e.g., to see how system requirements trace to or contribute to goals), and *impact assessment of non-functional requirements on functional requirements* (e.g., to consider trade-offs involving both types of requirements).

Table 6. Objectives of Goal/SysML integration and related methods

Objectives	Methods	Articles
Uncertainty & adaptation	M1, M2, M8, M17	[20, 45, 51, 55–62, 73–75]
Architecture selection & modeling	M1, M9, M10, M11, M13, M17	[45, 63–67, 69, 73, 74]
Formal V&V	M5, M7, M8, M15	[48, 49, 53, 54, 58, 71]
Traceability	M6, M12, M14, M16, M17	[31, 50, 68, 70, 72]
Development process improvement	M3, M17	[46, 75]
Requirements visualization	M4	[47]
Impact of NFRs on FRs	M7	[52]

3.1.3. Method characteristics

Integrating goal models with SysML models has different dimensions depending on the objective of the study and the researchers' vision for a specific problem and its solutions. To answer the secondary question SQ1.2 (How do the methods integrate the two types of models?), Table 7 includes information about each main study and the related data that explains the following:

1. Whether the method was automated;
2. Whether the method integrated goals as a model;

3. Whether the main goal concepts are used in the integration;
4. Whether goal analysis was supported; and
5. The method realization (usually through a profile).

Table 7. Extracted data on the integration dimensions
(F = fully automatic, S = Semi-automatic, M = Manual, ? = Unknown)

Research method	Code	Auto	Goal model	Goal concepts	Goal analysis	Method realization
Ingram et al. [45]	M1	?	M	M	S	Profile
Amyot et al. [20]	M2	S	F	S	S	Investigating
Vanderperren and Dehaene [46]	M3	?	M	S	M	Profile
Ozkaya [47]	M4	?	?	?	M	Investigating
Matoussi et al. [48]	M5	?	S	S	M	Profile
Cui and Paige [50]	M6	?	S	S	M	Profile
Gnaho et al. [51, 52]	M7	?	S	S	M	Profile
Ahmad et al. [55]	M8	S	S	S	M	Profile
Apvrille and Roudier [63, 64]	M9	S	M	S	S	Profile
Tsadimas et al. [66]	M10	S	M	S	M	Profile
Spyropoulos and Baras [67]	M11	S	M	S	S	Profile
Badreddin et al. [68]	M12	?	S	S	S	Textual syntax
Fan et al. [69]	M13	S	M	M	M	Profile
Wang [70]	M14	M	S	S	M	Mapping
Lee et al. [71]	M15	M	S	S	M	Profile
Maskani et al. [72]	M16	M	S	S	M	Profile
Anda and Amyot [31, 74]	M17	F	F	F	F	Math functions and RMS

To assess how far the methods go in their integration, Table 7 includes columns that are further explained below. Note however that some methods were still under development or investigating alternatives. As they did not provide sufficient details about their process, the level of automation and the method realization were difficult to assess at times. Most of the studies did not mention how goals or requirements are transferred to extended SysML profiles. Some of them developed specific editors for their methods but did not explain whether the goals or requirements were translated automatically or re-entered manually.

Automation. Does the method support an MDE (automated) approach? Several studies [9] have addressed the advantages of an MDE approach, including information traceability, holistic validation and verification, as well as code generation. These features are not only important to support self-adaptability, but also to improve productivity and system quality [9, 46]. As seen in Table 7, most selected methods used goals (partially) as a model with SysML requirements diagrams. In contrast, Badreddin et al. [68] proposed the only method (M12) that does not support a graphical MDE approach and presented a new language that combines the models using a textual syntax. In many studies, goals have actually been translated to a textual, hierarchical structure using a profiled SysML requirements diagram, (with various degrees of formalization). One method (M17) automatically translates GRL goal models to mathematical functions that can be embedded in SysML models. Some studies used SysML block and parametric diagrams with some goal model analysis such as trade-off analysis.

Goal Modeling Concepts. Were important goals modeling concepts (goals, softgoals, decompositions, actor importance, contribution weights, indicators, etc.) part of the integration with the SysML model? Anda and Amyot [31, 74] proposed the only method (M17),

named CGS4Adaptation, that includes all the elements of goal models in their integration. Goals were integrated with SysML requirements diagrams in most methods but not all goal modeling concepts were mapped. These methods extended requirements diagrams with goal types (functional and/or non-functional) and some goal relationships (mainly AND/OR decomposition). However, quantitative/qualitative contributions between goals, importance of goals to their containing actors, and indicators with parameters are seldom covered. For example, Cui and Paige [50] integrated goals model without considering the quantitative values of the contribution relationships between goals or indicator parameters, whereas Ahmad et al. [55] integrated all types of goals and their relationships except for contribution weights, importance levels, and indicators. This prevents modelers from quantitatively 1) performing goal analysis to guide the selection of alternatives (at design time) and 2) supporting dynamic adaptation at runtime according to user preferences [36].

When dealing with NFRs, the methods presented by Apvrille and Roudier [63, 64], Tsadimas et al. [66], and Spyropoulos and Baras [67] focused on the important role of goal-oriented techniques in system architecture and design selection. However, none actually transformed or linked goals to the design phase. Instead, they broke down system goals into non-functional requirements and linked them to design elements of SysML requirements diagrams. In contrast to these methods, Maskani et al. [72] expanded the requirements profile with security goals and requirements while the related stakeholders, goals, assets, and risks were added as attributes.

Goal Analysis. Trade-off analysis can be conducted through positive and negative contributions between goals during the decision-making process, e.g., to determine which actors will be satisfied or dissatisfied by a particular solution or adaptation strategy. Some methods were used to analyze fault tolerance and security mechanisms using quantitative values in their goal/ SysML integration, but mainly to select the best architecture/design [55, 63, 64] or to include possible choices in the system implementation phase [45, 67]. However, these analyses are limited to static decisions and adaptions, often outside of the SysML model as well. To support goal-based design selection and runtime adaption in a way that is integrated with SysML, Anda and Amyot [73–75, 78] generate arithmetic functions from GRL models that can be inserted in SysML models for simulation and optimization, and in the system code for runtime adaptations.

Method Realization. As seen in Table 7, all but four studies used some level of SysML profiling to map goal concepts to SysML concepts (often using requirements diagrams as a basis). Badreddin et al. [68] however proposed (in M12) integrating both views through a new textual language (fSysML), whereas two other studies were still investigating this aspect. In M17, in addition embedding functions generated form goal models in the SysML models, the authors also support importing both the goal and SysML models into a third-party traceability tool (commonly called a Requirements Management System – RMS) to enable managing traceability links between the elements of the goal and SysML models (blocks and requirements diagrams and their relationships), hence also enabling impact analysis and consistency checks as models evolve [31].

3.2. Adaptation support methods

To answer research question RQ2 (What are the collected methods that support self-adaptation?), we selected additional articles coming from digital libraries and provided by experts. Sub-questions SQ2.1 and SQ2.2 are answered using adaptation concepts and dimensions.

In order to find the methods that support self-adaptation characteristics in a context where adaptation objectives are not explicitly mentioned in some of the studies, we classified the methods using two criteria: self-adaptation properties and adaptation type. These two criteria are respectively based on two classifications: 1) the non-functional requirements that guide a particular system architecture design, and 2) the phase used to realize the adaptation.

3.2.1. Self-adaptation properties

We classified the studies based on the four common *self-** properties of self-adaption [8, 35], namely self-healing (from failures and incorrect states), self-configuration (to changing contexts and resources), self-optimization (to best meet specific goals), and self-protection (to avoid system harm). This classification was done with the help of related quality attributes, as suggested by Mistrik et al. [79] and Salehie and Tahvildari [80]. We extracted the non-functional requirements (NFRs) cited in the 52 eligible studies before we related them to four self-* properties.

For the studies where an adaption rationale was available, we established a mapping to self-* types via NFRs. Table 8 details the results. Sixteen methods support systems in adapting themselves while running by responding to changes that could be external (environmental) or internal (the system itself) [81]. Only four of them [45, 55, 67, 74] integrate goal and SysML models for both system design and self-adaptation. SysML also was hired by another 7 methods to support self-adaptation.

In terms of adaptation approaches that use goal models but not SysML, we find several methods such as those from Morandini et al. [82, 83], Qian et al. [94], Ramnath et al. [95], Baresi et al. [86], and Baresi and Pasquale [87, 88]. Additional diversity is brought by pattern-based and case-based approaches [90, 94].

Table 8. Distribution of self-* properties and non-functional requirements among the studies

Self-*	NFRs	Goal/SysML	Goal	SysML
Self-Healing	Fault diagnosing tolerance	Ingramet al.[45]	Morandini et al. [82, 83]	Bareiß et al. [84], Parri et al. [85]
Self-Configuration	Adaptability, Integrity and Availability	Ahmad [59], Ahmad et al. [55–58], Ahmad and Bruel [60, 61]		
	Adaptability	Anda and Amyot [73, 74], Anda [75]	Baresi et al. [86], Baresi and Pasquale [87, 88]	Hussein et al. [89], Meacham [90]
	Reliability			Ribeiro et al. [91]
Self-Optimization	Resource utilization	Spyropoulos and Baras [67]		Lopes et al. [92], Souza et al. [93]
	Time behavior		Qian et al. [94]	
Self-Protection	Security	Belloir et al. [62]	Ramnath et al. [95]	

3.2.2. Adaptation phase and development

Support for the development of adaptive systems is provided at different levels. Some studies provide analysis and design methods for such systems, but *without* explicit adaptation support. Others that come *with* adaptation support do so either for design-time adaptation or for runtime adaptation. In design-time adaptation, the situations triggering adaptation, the adaptation mechanisms, and the strategies for decision making are already known and applied in the system at design time. Systems that apply runtime adaptation are distinguished by the ability to deal with unpredictable environmental changes while running [80, 90].

Table 9 shows that most of the studies that integrate goal and SysML models target the development adaptive systems for different reasons (i.e., uncertainty reduction, complexity simplification, system validation and verification) other than for adaptation, while most of the methods that target self-adaptation trough SysML models or goal models separately implement their adaption strategies, mechanisms, and decisions at design time (design-time adaptation). Interestingly, runtime adaption in SysML is currently lacking contributions.

Table 9. Distribution of the studies related to development of adaptive systems

Study Category	Without Adaptation Support	With Adaptation Support	
	Analysis and Design Only	Design-Time Adaptation	Runtime Adaptation
Goal/SysML	[31, 46–54, 63–66, 68–72]	[20, 45, 55–62, 67]	[73–75]
SysML and adaptation	[18, 19, 96]	[17, 84, 85, 89–93]	
Goals and Adaptation	[97]	[82, 83, 86–88, 95]	[94]

3.2.3. Adaptation approaches

Tables 10 and 11 summarize the approaches each method applies to meet its objectives. Three methods (in four articles) used the *i** goal modeling language, and one (in four articles) used GRL, a language that originates from *i**. Four methods used the KAOS language (in 11 articles), and RELAX [77] was used in a few instances. Please note that the methods in Tables 10 and 11 are different and independent from the integration methods described in Table 5.

Table 10. Methods using combined Goal/SysML models to represent self-adaptive systems

Method	Overview
Ingram et al. [45]	Employed conditions and roles of a fault tolerance study to choose the best strategy for managing traffic problems.
Ahmad et al. [55]	Used SysML, KAOS and RELAX to manage uncertainty at runtime.
Spyropoulos and Baras [67]	Used trade-off analysis to optimize resource distribution of an Electrical Microgrid system using mathematical algorithms applied in a SysML model. The last model was integrated with the Consol-Optcad optimization tool for early cost and performance estimation.
Anda and Amyot [31, 73, 74], Anda [75]	Transformed GRL and feature models into mathematical functions that can be executed outside of goal modeling tools including SysML, simulation, optimization, and implementation tools. Also, goal and SysML models are imported into an RMS to manage traceability and consistency as models evolve.

Table 11. Methods using SysML models or goal models separately to represent self-adaptive systems

Method	Overview	Using
Morandini et al. [82, 83]	Unified goal model (i*), failure model, and environmental model to support self-adaptation.	Goals
Qian et al. [94]	Combined strategies selection and case-based reasoning self-adaptation approaches. In order to determine the embedded strategies, the lowest level of parameterized goal models was linked with the highest level of softgoals via weighted contribution relationships.	Goals
Ramnath et al. [95]	Linked strategies for attack and protection at the design layers of the proposed architecture.	Goals
Meacham [90]	Combined pattern-based with case-based reasoning approaches where repeated falls were collected and analyzed to identify their patterns, leading to solutions as plans.	SysML
Ribeiro et al. [91]	Modeled real time requirements and managed traceability through extending SysML requirements diagram with relationships and properties. Synchronized relationships were used to represent parallel real-time requirements.	SysML
Bareiß et al. [84]	Modified the SysML meta-model to create a SysML4Pack profile that combines SysML model, OCL [98] and the state machines of OMAC to represent predictable faults of automatic production systems.	SysML
Lopes et al. [92]	Integrated SysML models with trade-off analysis and techno-economical cost-benefit analysis to optimize electricity management, generation, and distribution among customers.	SysML
Soyler and Sala-Diakanda [17]	Included disaster management strategies in a SysML system architecture with continuous feedback from the last disaster data.	SysML
Akbas and Karwowski [18]	Combined dynamic models with agent-based models that were extracted from system design using SysML models.	SysML
Souza et al. [93]	Created a SmartCitySysML profile that extends the profiles of requirement and block in SysML to represent smart city elements.	SysML
Horkoff et al. [97]	Integrated goal models i* with the MAVO framework of [99] to iterate over the analysis process for early uncertainty reduction.	Goals
Baresi et al. [86]	Modified the KAOS language with fuzzy goals (i.e., non-functional goals with uncertainty) leading to a new language called FLAGS, which supports functional models (crisp goals) and adaptive models (fuzzy goals). The crisp goals were formalized through Linear Temporal Logic language (LTL) [100] plus fuzzy temporal operations such as <, >, <=, and *approximately* to express the fuzzy goals.	Goals
Parri et al. [85]	Combined system configurations derived from SysML block definition diagrams (BDD) metadata with a failure model derived from fault tree via digital twins and data analysis agents.	SysML
Baresi and Pasquale [88]	Used service composition based on the Business Process Execution Language (BPEL) [101] to transform the FLAGS/KAOS model in Baresi et al. [86] to membership functions and abstract processes, semi-automatically. These functions trigger the adaptation strategies using Boolean conditions.	Goals
Baresi and Pasquale [87]	Added operators from RELAX Language to the FLAGS language in Baresi and Pasquale [88] to represent the fuzzy goals. Member functions are used in the monitoring process but the adaptation strategies are triggered by conditions associated with the operational model.	Goals

Several integrations (and, at times, extensions) were also done with goal models or SysML models separately, hence answering the sub-question SQ2.2.

Goal Model Integrations with Languages Other than SysML

1. In order to support dynamic adaptive systems, Morandini et al. [82, 83] integrated models of goals, failures, and the environment.
2. To deal with unpredicted changes at runtime, Qian et al. [94] integrated goal models with case-based reasoning.
3. Horkoff et al. [97] integrated goal models with the MAVO framework to reduce uncertainty early.
4. Baresi et al. [86] and Baresi and Pasquale [87, 88] described goals using a formal linear temporal logic (LTL) language and the RELAX language for usage at runtime.
5. Anda and Amyot [31] adapted a model import method [102] to import and trace GRL models into an RMS (IBM Rational DOORS [103]). They also generate functions from goal and feature models that can be embedded in SysML models.

SysML Model Integrations with Non-Goal-Oriented Languages

1. Meacham [90] did an integration of SysML with UML to specify cases of presented patterns, while Soyler and Sala-Diakanda [17] also supported an integration with UML, but this time to represent the structure and behavior of systems in one single environment.
2. Additional relationships and properties were added to SysML requirements diagrams by Ribeiro et al. [91] for representing runtime requirements in a hierarchical way and for managing requirements traceability for system validation and verification.
3. Bareiß et al. [84] used an integration with OMAC state machines, ISA-88 physical models, and OCL constraints for transforming models to code.
4. Lopes et al. [92] provided an integration supporting trade-off analysis and techno-economical cost-benefit analysis when modeling detailed system architectures.
5. System dynamic models and agent-based simulation were integrated by Akbas et al. [19] and Akbas and Karwowski [18] for minimizing system complexity and specifying system agents in a hierarchical structure.
6. Smart city elements including different types of requirements, solutions, processes, stakeholders, problems, and dimensions have been added to a SysML profile by Souza et al. [93] to support a domain-specific modeling process.
7. Real configuration items from SysML BDD properties and diagnostic, predictive, and prescriptive analytics derived from fault tree are integrated by Parri et al. [85] to discover alternative configurations when a runtime violation is detected.
8. Ginigeme and Fabregas [96] derived configuration parameters from the stakeholder's requirements and system design in SysML to be used by a discrete-event simulation (Arena) tool to evaluate the design configurations.
9. SysML BDDs and requirements diagram are imported in an RMS (DOORS) by Anda and Amyot [31] to support consistency and completeness checks (against imported GRL models) as well as more common impact analysis and change management processes.

3.3. Adaptation assessment

In order to answer questions SQ2.1 and SQ2.2 on adaptation methods, we extracted information that identifies terms inspired from existing adaptation taxonomies [37, 80] and modeling dimensions of self-adaptation [81, 104]. Using these terms was helpful in inferring

correct indicators that specify how each method supports self-adaptation and what roles each model plays in this adaptation.

Among the articles collected, some were eliminated from this assessment because their adaptation methods were redundant or not described in sufficient detail. In particular, Akbas and Karwowski [18] and Horkoff et al. [97] designed self-adaptive systems for reducing uncertainty and system complexity. They supported the use of self-adaptive systems but not the use of adaptation where the system re-configures itself to become more usable. In addition, Baresi et al. [86] and Baresi and Pasquale [87, 88] expressed the same methods with different emphases, so we considered them as one method represented by the most detailed paper [87]. Finally, Spyropoulos and Baras [67] provided information about their dynamic decision-making process but not about the adaptation strategies and properties; this paper was excluded from the adaptation properties and dimensions assessment, but kept for the decision-making criterion. As a result, 14 methods are discussed here.

3.3.1. Adaptation terms

The selected terms were defined in Section 2.2.4 on data extraction. Table 12 illustrates the assessment of each adaptation term against related methods and studies. The color coding reflects how positive a result is (green = positive, yellow = neutral, red = negative, and white = unknown or inappropriate).

3.3.2. Adaptation modeling dimensions

Three types of modeling dimensions (goal, change, and mechanisms), proposed by Andersson et al. [81], are used to specify self-adaptive properties. Some of these properties are overlapping with the adaptation taxonomy previously mentioned. Some of the methods, such as those presented by Ahmad et al. [55], Anda and Amyot [74], and Baresi and Pasquale [87], are generic and can be applied to different applications; we estimated their values based on the provided information. We extracted the methods' information related to the chosen modeling dimensions, which is summarized in Table 13.

3.3.3. Assessment results

The surveyed methods handled self-adaptation from several perspectives: adaptation terms and modeling dimensions, early management of uncertainty, the use of different languages to deal with adaptation, frameworks for developing self-adaptive systems, adaptation strategies, and finally decision-making and strategy selection processes. This section provides further assessment of the methods along these six perspectives. A comparison of the methods according to the used models is also provided to highlight the contribution of these models to the ability of systems to self-adapt.

Adaptation Terms and Modeling Dimensions. From Tables 12 and 13, several observations can be made:

- Most of the collected methods realized a *closed* approach of adaptation by including their strategies with system design. Only three were clearly *open*, i.e., more amenable to adaptation to unforeseen situations and contexts.
- All of the collected methods supported their adaptation approach at design time, and they do not enhance or change them at runtime using a learning technique (e.g., based on machine learning).

Table 12. Adaptation terms related to each selected method
(C = Closed, O = Open, ? = Not provided, Y = Yes, N = No, P = Partially, Dy = Dynamic, Sta = Static, M = Making, A = Achieving, Md = Middleware, Ap = Application, Sr = Service, St = Structure, W = Weak, Rt = Reactive, Pt = Proactive, Co = Continuous, Ad = Adaptive)

	Goals and SysML			Goals only				SysML only						
Adaptation terms	Ingram et al. [45]	Anda and Amyot [74]	Ahmad et al. [55]	Baresi and Pasquale [87]	Morandini et al. [83]	Qian et al. [94]	Ramnath et al. [95]	Meacham [90]	Ribeiro et al. [91]	Bareiß et al. [84]	Lopes et al. [92]	Soyler and Sala-Diakanda [17]	Souza et al. [93]	Parri et al. [85]
Adaptation Type	C	O	?	C	C	O	C	C	C	C	?	O	C	C
Model-based	Y	Y	Y	Y	Y	P	Y	Y	Y	Y	Y	Y	Y	Y
Decision (Analyze/Selection process)	?	Dy	Sta	Dy/ Sta	Sta	Dy	Sta	Sta	?	Sta	?	?	?	Dy/ Sta
Adaptation approach	M	M	M	M	M	M	M	M	M	M	M	M	M	M
Layer	Md	Ap	Ap	Ap	Ap	Ap	Ap	Ap	?	?	Md	Ap	Ap	Md
Artifact	Sr/S	Sr	Sr	Sr	Sr	Sr	Sr	Sr	?	St	Sr /St	Sr	Sr	St
Cost-impact	W	W	W	W	W	W	W	W	?	W	W	W	W	W
Temporal adaptation	Rt	Pt /Rt	Rt	Rt	Pt/ Rt	Rt	Rt	Rt	?	Rt	Rt	Rt	Pt/ Rt	Pt/ Rt
Temporal monitoring	Ad	Co	Co	Co	Co	Co	?	Co	?	Co	Co	Co	Co	Co
Human intervention	N	N	N	P	P	N	N	N	?	P	?	?	N	P

– According to their effected layers and artifacts, they supported only a weak adaptation (i.e., no change to the architecture at runtime).
– The closed approaches affect the goals flexibility feature negatively and consequently lead to different ways of managing adaptivity via fixed goals or flexible goals with constraints, as shown in Table 13.
– Most methods that used goals as a model managed flexible goals with constraints because of the conditions that were used to trigger system plans and strategies during the strategy selection process (closed approach and design-time adaptation).
– The collected methods do not support unconstrained goals except for two methods. Qian et al. [94] used methods to generate solutions: 1) goal-reasoning to generate a new solution when the current cases did not match the conditions of the stored cases, and 2) using the average of the similar cases to generate new solutions. However, the

Table 13. Modeling dimension of the selected methods
(Sta=Static, Dy=Dynamic, Rgd=Rigid, Cns=Constrained, Ncn = Unconstrained, Mlti = Multiple, S = Single, E = External, I = Internal, Nfr = Non-functional requirement, Gt = guaranteed, NGt = Not guaranteed, ? = Unknown)

	Goals and SysML			Goals only				SysML only						
Adaptation terms	Ingram et al. [45]	Anda and Amyot [74]	Ahmad et al. [55]	Baresi and Pasquale [87]	Morandini et al. [83]	Qian et al. [94]	Ramnath et al. [95]	Meacham [90]	Ribeiro et al. [91]	Bareiß et al. [84]	Lopes et al. [92]	Soyler and Sala-Diakanda [17]	Souza et al. [93]	Parri et al. [85]
Goal														
Evolution	Sta	Sta	Sta	Dy	Sta	Sta	Sta	Sta	Sta	Sta	Sta	Sta	Sta	Sta
Flexibility	Rgd	Ncn	Cns	Cns	Cns	Ncn	Rgd	Rgd	Rgd	Rgd	Rgd	Rgd	Cns	Rgd
Multiplicity	Mlti	Mlti	Mlti	Mlti	Mlti	Mlti	Mlti	Mlti	Mlti	S	Mlti	Mlti	Mlti	Mlti
Dependency	Mlti	Mlti	Mlti	Mlti	Mlti	Mlti	Mlti	Mlti	Mlti	S	Mlti	Mlti	Mlti	Mlti
Change														
Source	E	E& I	E	E&I	E& I	I	E	E	?	I	E& I	E	E	I
Type	NFR	NFR	NFR	NFR	NFR	NFR	NFR	NFR	NFR	NFR	NFR	?	NFR	NFR
Mechanisms														
Timeliness	Gt	Gt	?	NGt	Gt	NGt	Gt	Gt	?	Gt	Gt	?	Gt	Gt

new solutions could be unsuitable for the current problem and consequently lead to non-guaranteed adaptation timeliness, as shown in Table 13. On the contrary, Anda and Amyot [73, 74] have used a goal reasoning method (without constraints or conditions on its choices) to generate on the fly the best solutions when unforeseen circumstances are encountered at runtime. Since the used mathematical functions include the impact of the current environmental condition on all elements of the goal models and are restricted by the mathematical function of feature models, the created solutions are feasible and the best (the functionality and quality of the system satisfy its stakeholders' objectives) for the current environmental condition.

– Baresi and Pasquale [87] presented the only method that changes the number of system goals during adaptation by adding and deleting goals. As a consequence, the time needed for adaptation is not guaranteed even if the conditions and related plans are already known and embedded in the system at design time.
– Most methods (except three) included work or comments on mechanisms.

To conclude, using goal models in adaptation methods strengthens their flexibility and ability to deal with unknown conditions at runtime. However, this can also lead to the generation of infeasible solutions or unguaranteed adaptation timeliness due to insufficient validity checking of the generated solutions and the new alternatives.

Early Management of Uncertainty. Reducing or eliminating uncertainty before having to manage it is one way to analyze and design self-adaptive systems. To support the decision-making process in analysis and design phases, early in the requirement engineering process, Horkoff et al. [97] presented a formal iterative goal analysis process with a tool that integrated *i** goal models with the MAVO framework [99] to remove unnecessary requirements alternatives. The treatment of uncertainty in general goal modeling is further explored by Alwidian et al. [105].

Language Usage. Self-adaptive systems offer an opportunity for more relaxed language to be used to better specify their requirements, because common patterns such as "the system shall do this" are often too strict that context. This need was addressed in by Ahmad et al. [55], who used the RELAX language [77] as a more formal representation of this idea for monitoring environmental conditions and detecting violations. In addition, the formal language called FLAGS [87, 88] formalizes the KAOS goal modeling language through LTL. In order to represent fuzzy goals with uncertainty, LTL is accompanied by fuzzy temporal operators based on RELAX [87]. This language was used to keep tracking and using the goal model from requirements elicitation to the implementation phase. Anda and Amyot [31, 73, 74] generate arithmetic functions in common languages (including C, Java, and Python) from GRL and feature models, which enable analysis and implementation to be done with a wide range of development tools.

Frameworks for Designing Self-Adaptive Systems. Several approaches and frameworks were presented to design and select an appropriate architecture for self-adaptive systems (SAS). Morandini et al. [83] extended the Tropos framework [106] for Adaptive Systems (Tropos4AS). This framework helps analyzing requirements of SAS from early requirements to the implementation by mapping the goal model of particular actors to architecture agents and by mapping the plan (tasks) to activity diagrams. This framework uses goal, failure, and environmental models. The Tropos goal modeling language, itself based on *i**, was extended to add goal types (achieve, maintain, perform), relationships (sequence, inhibition) and conditions. Code is generated automatically from the models by mapping Tropos4AS terms to Belief-Desire-Intention (BDI) agents, which enable SAS validation and verification via simulation.

To support system reliability, flexibility, and runtime recoverability, Parri et al. [85] proposed a software/hardware framework, called JARVIS, for developing CPSs and Systems of Systems. JARVIS adopts SysML BDDs and fault trees to discover configuration alternatives using digital twins and data analytic agents.

Security strategies can also affect user privacy and cost. For this reason, Ramnath et al. [95] proposed a non-functional framework to deal with adaptive security analysis. The goal model is linked from and to dynamic behavior of the organization via a transaction-based mechanism. Such goal model is used to support trade-off analysis between cost and privacy in order to help with the definition of a secure architecture.

To reduce the complexity of SAS and manage traceability between their components, Soyler and Sala-Diakanda [17] presented a model-based framework exploiting SysML. This framework was selected to capture a Disaster Management System in one single environment using feedback to adapt the embedded strategies, plans, and policies.

Finally, Akbas and Karwowski [18] proposed an agent-based framework that uses a hybrid simulation model to support system design, validation, and verification, as well as to provide quick feedback about the chosen design.

Adaptation Strategies. The collected methods dealt with possible adaptation strategies or configurations through open and closed adaptation.

- *Closed adaptation approaches:* In a closed approach, possible alternatives, strategies, and configurations are embedded in the system during the development phase. Assuming environmental conditions and changes are well-known at design time, the closed methods (Table 12) manage uncertainty through rigid or constrained goals. From the Goal/SysML integration methods, Ingram et al. [45] used fault tolerance analysis and rules to deal with errors. Without considering goal models, Ribeiro et al. [91], Parri et al. [85], Bareiß et al. [84], Souza et al.[93], and Soyler and Sala-Diakanda [17] triggered their embedded strategies, configuration, or plans to respond to internal or environmental changes. Similarly, Morandini et al. [82, 83] represented the goal model in an agent structure while embedding the environmental and failure conditions, alternatives, and plans in agent beliefs and system design. Designing self-adaptive systems with predicted or predictable change management is a characteristic common to these types of methods. One issue here is that they cannot deal with unpredictable changes that could emerge at runtime. On the other hand, they guarantee that the selected adaptation strategy is suitable and timely for a given contextual change (see Tables 12 and 13).
- *Open adaptation approaches:* Open approaches do not solely rely on predetermined adaption strategies and conditions. Feedback can be used to update the embedded strategies, as suggested by Soyler and Sala-Diakanda [17] (although they give little explanation on how to do so). Case-based reasoning is an approach that uses previously stored solutions in solving current similar problems. To deal with unexpected environmental changes, case-based reasoning can be employed to update embedded configurations and strategies. Based on such feedback loop, Qian et al. [94] create new solutions or configurations from the average of the parameters' values of two or more stored cases or from goal reasoning (such as label propagation algorithms [107]). In contrast, Meacham [90] used case-based reasoning to manage fall cases of elderly people and infer their patterns in order to determine the related system reactions. She used stored cases and patterns only while the feedback technique was not applied, in order to continue enhancing the stored cases, as Qian et al. [94] did. To enhance overall system performance, new strategies or configurations can be issued by the optimization method of Anda and Amyot [31, 73], which deals with unexpected conditions at runtime.

Decision-Making and Strategy Selection Processes. The collected methods have not provided much diversity in the decision-making process that triggers the adaptation and the selection of the most suitable strategy (see Figure 1). All the decision processes were encoded inside the system (i.e., static decision-making) and no adaptation was performed on these processes using learning techniques. However, the decision-making process can use different policies: action (static decision), goal and utility (dynamic decision), as well as hybrid policies [37, 108]. These methods realized their decision-making processes as follows.

- *Action policies:* Apply to the process that decides when the adaptation should be done and what the system should do based on the current state, conditions, and actions (if-then logic) [108]. Meacham [90] used a pattern analysis algorithm to trigger the adaptation while Morandini et al. [83] used a goal modeling approach and several types of conditions that trigger the adaptation process. In order to select a suitable recovery strategy, Bareiß et al. [84] used a diagnosis model that compared the current system state with the pre- and post-conditions of each operation state. The if-condition-then-plan technique is used here because it is a simple way for humans to express a rational

logic in the systems. However, action policies become complex in real-world conditions, and additional techniques (e.g., prioritization) are needed to solve policy conflicts in practice [10, 108].

– *Utility and goal policies:* In order to select an optimal adaptation strategy, experts are needed to identify the control variables required by the utility policy approach. Such approach has been used in the decision-making process, providing a flexible way to trigger the optimal adaptation's strategy by exploiting designer knowledge at design time and real monitored data at runtime [108].
Ramnath et al. [95] used utility functions for cost and benefit of the involved stakeholders and trade-off analysis to select a suitable design. From there, related strategies were connected to the security layers of this architecture to be executed at runtime. Similarly, Spyropoulos and Baras [67] used trade-off analysis to get the optimal solution for power allocation in their Microgrid system. In their approach, Lopes et al. [92] added an enterprise service management plan using utility functions to select the best strategy based on the techno-economical costs/benefits and trade-off analysis at design time. However, none of the previous studies used these policies at *runtime* with the real data. Baresi and Pasquale [87] used satisfaction equations and goal reasoning in analyzing system state. However, they were not used in their strategy selection process triggering the possible solutions depending on several conditions attached to the system operations as rules. Similarly, Parri et al. [85] used Fault Tree analysis to detect and predict failures while the suitable configurations were associated to the tree via digital twins. Also, Qian et al. [94] used case-based reasoning in all MAPE activities (see Figure 1) except in the planning process, the latter being supported by goal-based reasoning when it failed. However, they applied goal-based reasoning in generating new configurations only by increasing the weights of the violated goals to get solutions related to these specific goals, but the new solutions could still be unsuitable for the current problem. Hence, although using such a utility function leads to an optimal solution without strategy conflicts, its usability is affected significantly and experts are still required [10, 108]. On the other hand, goal and feature models transformed to mathematical functions by Anda and Amyot [73, 74, 78] are used in MAPE activities to monitor, analyze, and select the suitable strategies at design time and during runtime adaptation.

3.3.4. Self-adaptation, goals, and SysML

The collected articles display distinguished features when classified according to the three categories (Figure 5) initially used to search the literature: goal models (without SysML), SysML models (without goal models), and goal models combined with SysML models.

From Section 3.2.3, goal models are used to reduce uncertainty early and provide adaptation rationale and alternatives. Also, based on Table 9, goals are involved into all the methods that enhance their adaptation solutions at runtime while runtime adaption is not well supported in SysML. In Table 12, two methods used goal models to generate new solutions/strategies when facing unknown conditions at runtime while only one SysML-based method used feedback to create new strategies, but without much detail. Similarly, three methods out of four that provided dynamic decisions exploited goal models while only one method (Parri et. al. [85]) supports dynamic analysis with a strategy selection process using SysML. From a modeling dimension perspective, SysML is involved in all the inflexible methods that use fixed goals in their adaptation approaches. On the other hand, all the goal-based methods manage multiple goals and their dependencies defined in goal models.

Also from Section 3.2.3, SysML provides a suitable environment that reduces the complexity of self-adaptive systems and represents them in one single environment through profiles. Such profiles are used to strengthen domain-specific modeling by adding new terms for new types of systems such as smart cities [93]. However, these profiles do not represent goal model elements and analysis together, and valuable information is lost during the mapping of goal-related concepts to SysML concepts, which reduces the flexibility of theses methods (and leads to rigid or constrained goals instead of goal reasoning enabling selections and trade-offs among goals). Only one method [74] integrates goal model analysis (including goals, softgoals, actors, tasks, indicators, contributions, relationships, and their importance) with SysML models without using profiles. This integration involves mathematical expressions generated from goal models, which enables a flexible method with open adaptation together with dynamic decision in analysis and mapping mechanisms.

3.4. Challenges

The studies have faced several challenges while employing their methods, and some are further explored below.

Usability of Integration. The integration processes are often characterized by remodeling goals with design tools (duplication of work), which not only causes risks of information loss and inconsistencies, but also consumes much development effort and time. One reason is that requirements, goal models, and SysML design artifacts have different and specific environments and tools that deal with their creation, management, and analysis needs. Representing a goal model using another tool with a different purpose (such as s SysML design tool) was a major obstacle faced by most methods. Furthermore, trade-off analysis as well as runtime adaptation selection are other affected features within all these methods because the goal model is not mapped completely and used effectively in the MAPE activities.

Goal Models and MAPE Activities. The MAPE activities (monitor, analyze, plan, and execute, see Figure 1) are not all supported at the same level by the collected methods. Managing and changing system goals at runtime is one suggested solution for conducting trade-off analysis and selecting the best adaptation strategy using real-time variables. However, in these studies, the scalability of the proposed methods is rarely formally assessed.

Goal-based Reasoning at Runtime. The use of goal-based reasoning at runtime differed from one study to another, and it was affected negatively by several factors: 1) transferring only part of a goal model to the design and/or runtime phases (e.g., not transferring contribution linksweights) and 2) handling the reasoning process in several ways (i.e., considering softgoals and tasks only, or violated softgoals only). As a result, the methods' ability to use goal reasoning at runtime for selecting the best (or even just one) suitable solution during the analysis and strategy selection processes was limited, and unsuitable solutions could be generated along the way. Furthermore, goal analysis and trade-off analysis cannot be done at runtime accurately using those methods, which consequently limits the ability to self-adapt in the developed systems by using conditions and implementing inflexible methods that cannot deal with unpredictable contexts.

Unmanageable Traceability. High-level goals are usually more stable than low-level ones, and they help guide the evolution of requirements from elicitation to runtime adaptation [9, 61]. However, to truly unlock the benefits of goal-orientation (including consistency/completeness, conflict, trade-off, and impact analyses), SysML system design components should be linked to goals at all levels [9, 54, 55]. Yet, it is difficult to manage

traceability and consistency between goal and SysML models. Embedding goal models in SysML tools can help, but amount to redeveloping goal-based analysis in such tools, and none of the existing methods really does this. Overall, although establishing links between goals/requirements, system design, and the implementation was an objective of most selected methods, most fail from providing sufficient and practical traceability support except one [31], which uses an external RMS to do so but with low usability due the high number of tools involved (goal modeling environment, SysML tool, and RMS). SysML tool vendors should consider better integrating goal modeling and analysis capabilities in their solutions.

4. Related work

Although a literature review is already about collecting and assessing related work, it is also important to situate it among other literature reviews on related topics.

Zahid et al. [39] have recently published a systematic mapping of semi-formal and formal methods in requirements engineering of (industrial) Cyber-Physical Systems. Although SysML is mentioned on a few occasions, adaptive CPSs are not covered. Surprisingly, goal modeling is not discussed in their review.

There are also generic language-oriented systematic mappings on goal-oriented modeling (e.g., from Horkoff et al. [40]) and SysML (e.g., from Wolny [41, 42]) but they are superficial in their treatment of (self-)adaptive systems.

In contrast, there are several literature surveys and mappings on adaptive systems, whether they are cyber-physical or not. In particular:

- de Lemos et al. [109] provided an important roadmap for software engineering research on self-adaptive systems, which emphasized the identification and representation of goals, the management of the design space, and the validation of models (without mentioning SysML or goal-oriented modeling however) as important challenges. Many of the methods addressed in our review provide contributions in those areas.
- Macías-Escrivá et al. [36] also provided a survey with research challenges, but without details on modeling aspects.
- Krupitzer et al. [37] reviewed software engineering approaches for self-adaptive systems and discuss some goal-oriented methods, but no SysML-based ones, and not to the depth of our own review.
- Yang et al.[110] provided a review of requirements modeling and analysis for self-adaptive systems, where they identified 16 methods, some of which involving goal modeling. No SysML-based method was identified at the time this review was published (2014). Their main assessment was related to the coverage of important modeling and analysis concepts by the methods and the languages they used. CPSs are not mentioned.
- More recently, Porter et al. [38] explored the types of questions that are researched in the literature in relation to self-adaptive systems, instead of methods.

Our literature review is unique in that it is fairly exhaustive in its coverage of goal/SysML integrations and of SysML methods targeting adaptation, with partial coverage of some of the main goal-oriented methods for adaptive systems. It also provides a deeper analysis of multiple research questions and facets of these methods. It finally positions these methods in the CPS domain, with a specific emphasis on emerging types of adaptive socio-cyber-physical systems.

5. Limitations and threats to validity

As highlighted by Feldt and Magazinius [111], the validity of any study depends on the degree of correctness of its conclusions, including threats related to bias and over-generalization. We applied some strategies to mitigate common threats to validity, but several remain, as discussed below.

5.1. Internal validity

The first author selected and reviewed the papers, and extracted the raw data, with supervision and informal consultations and checks from the second author. In addition, one of the methods studied here (M17) also comes from the authors of this literature review. There is hence a risk of bias here. To mitigate this threat, we consulted several experts, including the authors of some of the selected papers, to increase the level of confidence in our assessment of their contributions. We have also used existing assessment criteria from the literature whenever they were available (e.g., from [37, 80, 81, 104]). However, the authors of the many papers reviewed here have not been rigorously surveyed, and hence there is a remaining risk that some of their contributions were classified or assessed incorrectly.

There is also a risk that important and relevant papers have been missed or incorrectly excluded in this literature review. To mitigate this risk, we used different and recognized scientific databases in the areas of systems modeling, with fairly permissive queries (refined over many iterations based on previous results). We also used Google Scholar with different queries and choices to increase our confidence that relevant studies from different sources were included. Precise inclusion and exclusion criteria were defined and used, and both authors were involved in the selection in cases where we were unsure about relevance. Yet, one remaining threat here is that the selected literature was limited to the English language.

We tried to be exhaustive for papers combining goal and SysML modeling, as well as for papers about SysML for self-adaptation. However, we manually selected primary articles (proposed by experts based on citations and reputation, as there were too many such papers) that support adaptation using goal models (Figure 5). One threat here is that many papers related to goals and self-adaptation have not been considered. Yet, the sample we have selected was useful to understand what is being done outside the SysML world, as a comparison point and as an indication of future opportunities.

5.2. External validity

This type of validity is related to the generalization of the results outside of the study's scope [111, 112]. The number of studies that focus on the integration of goal models with SysML models is rather small. If we consider the method granularity, only 17 methods were presented and four of them were specifically targeting adaptation. This is also why we focused on a descriptive presentation of our results, without trying to discuss statistical significance in the answers to our research questions.

What is published in peer-reviewed venues also may not be representative of what practitioners actually use in industry. Generalizing the results of these methods is a threat due to the relative immaturity of the field. We tried to mitigate this threat by systematically including papers on SysML for self-adaptive systems, and manually including primary papers on goal models for self-adaptation, again as comparison points. Still, general

conclusions about the use of goal modeling for adaption (without an integration with SysML) or about the integration of SysML and goal models outside of a CPS context should not be inferred from this literature review.

6. Conclusion

The number, complexity, and importance of socio-cyber-physical systems (SCPSs), which consider the goals of their stakeholders at design time and at runtime, is increasing in our societies [23]. In some SCPSs, the need for adaptability driven by stakeholder goals was partially addressed in the peer-reviewed scientific literature. This paper reviewed 52 publications and assessed methods that integrate goal models with SysML models (or use them separately) to support runtime self-adaption, with a consideration for the SCPS context. The review answers many questions of broad interest both to researchers and to practitioners who are considering the use of goal models, SysML models, or both in SCPSs or self-adaptive systems contexts. The research questions were answered through this review as follows:

RQ1. *What are the existing methods that integrate goal-oriented models with SysML models?* This was answered by Table 5, which presents a total of 17 methods, labeled M1 to M17, extracted from 33 studies. KAOS and GRL are the most frequently mentioned goal modeling languages in that context.

- **SQ1.1.** *Why have these integrations been proposed?* The objective of each study was presented in Table 6, where the common objectives are system architecture selection and modeling, uncertainty and adaptation, as well as traceability and formal validation and verification, in that order.
- **SQ1.2.** *How do the methods integrate the two types of models?* The answer was provided in Table 7 and its explanation in Section 3.1.3, which concluded that mapping parts of goal models to SysML requirements diagrams via profiles (formal or not) is by far the most often used approach through all 17 methods.

RQ2. *What are the collected methods that support self-adaptation?* By classifying the collected methods using NFRs, self-* properties (Table 8), and adaptation phases (Table 9), methods that support self-adaptation are listed and described in Table 10 (for the four approaches that integrate goals and SysML models) and Table 11 (for a sample of 15 approaches that use either SysML or goal models).

- **SQ2.1.** *How do the methods support self-adaptive systems?* This question was answered by Tables 12 and 13, which respectively identify terms inspired from the adaptation taxonomies and modeling dimensions of self-adaptation. The discussion around these tables (Section 3.3) provides insight into how the assessed methods support the activities of self-adaptive systems.
- **SQ2.2.** *What are the roles that each model plays in this adaptation support?* This was answered by exploring the reasons for using each model in each integration in Section 3.2.3, and by discussing the adaptation assessment criteria in Section 3.3.

Although there was much improvement in the last decade, the main results show that mapping goals at design time is common among the collected methods to support traceability, architecture selection, system validation and verification, as well as self-adaptation. However, existing mappings usually suffer from a loss of important information (e.g., contribution links and weights) or an absence of information (e.g., indicators sensing external contexts)

that play key roles in runtime goal analysis and flexible self-adaptation. Goal modeling is actually used sparsely and differently in MAPE activities of adaptive systems. Thus, in addition to consuming time and effort, most of the proposed methods were unable to implement goal-based reasoning in all activities. This consequently leads to situations where incorrect adaptation solutions are produced and used, and in time constraints that cannot be guaranteed. In fact, although modeling goal and SysML models in a single tool could help solve traceability problems and support adaptation, achieving this integration with existing design and analysis tools remains a challenge, as highlighted in Section 3.4.

To address many of the challenges and limitations observed throughout this review, we identify the following research directions.

- Developing and evolving adaption methods for SCPS where the goal models exploit important quantitative information such as contribution weights, importance levels to stakeholders, and indicators that measure different facets of the context. Such information is often necessary in models for data-centric systems and is very important for non-trivial adaptive SCPSs [113]. Such methods exist, but they are seldom integrated with SysML design activities.
- Ensuring that methods exploit the goal models through the MAPE cycle to their fullest extent, especially during runtime adaptation for unforeseen contexts (open approaches). Again, several opportunities have been explored in the goal modeling community but they are yet to be exploited in a SysML modeling and analysis context.
- As most integrated methods only support weak adaptation, there are opportunities to investigate goal-oriented, strong adaptations of component structures and architectures at runtime in an SysML context.
- Improving the usability and scalability of goal/SysML integrations for adaptive systems, with proper tool support, especially as the models grow in size and are frequently modified.
- Enabling (machine) learning during adaptation in integrated goal/SysML methods. None of the current work currently exploits this opportunity.

Despite many observed gaps and challenges, we believe the benefits of goal modeling (potential or actual) combined with SysML for adaptive SCPSs outweigh the identified drawbacks, and that further research will bring innovative and practical solutions in the near future.

Acknowledgment

Amal Ahmed Anda is supported by a scholarship from the Libyan Ministry of Education. We are thankful to the Natural Science and Engineering Research Council of Canada (Discovery program) for their support.

References

[1] B. Tekinerdogan, D. Blouin, H. Vangheluwe, M. Goulão, P. Carreira et al., *Multi-Paradigm Modelling Approaches for Cyber-Physical Systems*. Elsevier Science, 2020.

[2] I. Horváth, "What the Design Theory of Social-Cyber-Physical Systems Must Describe, Explain and Predict?" in *An Anthology of Theories and Models of Design*. Springer, 2014, pp. 99–120.

[3] I.J. Jureta, A. Borgida, N.A. Ernst, and J. Mylopoulos, "The requirements problem for adaptive systems," *ACM Transactions on Management Information Systems (TMIS)*, Vol. 5, No. 3, 2015, p. 17.

[4] A. Smirnov, A. Kashevnik, and A. Ponomarev, "Multi-level self-organization in cyber-physical--social systems: Smart home cleaning scenario," *Procedia CIRP*, Vol. 30, 2015, pp. 329–334, 7th Industrial Product-Service Systems Conference – PSS, industry transformation for sustainability and business.

[5] F. Zambonelli, "Towards a general software engineering methodology for the internet of things," *CoRR*, Vol. abs/1601.05569, 2016. [Online]. http://arxiv.org/abs/1601.05569

[6] E. Cavalcante, T. Batista, N. Bencomo, and P. Sawyer, "Revisiting goal-oriented models for self-aware systems-of-systems," in *2015 IEEE International Conference on Autonomic Computing (ICAC)*, July 2015, pp. 231–234.

[7] M. Sanchez, E. Exposito, and J. Aguilar, "Autonomic computing in manufacturing process coordination in industry 4.0 context," *Journal of Industrial Information Integration*, Vol. 19, 2020, p. 100159. [Online]. https://www.sciencedirect.com/science/article/pii/S2452414X20300340

[8] J.O. Kephart and D.M. Chess, "The vision of autonomic computing," *Computer*, Vol. 36, No. 1, 2003, pp. 41–50.

[9] J. Bocanegra, J. Pavlich-Mariscal, and A. Carrillo-Ramos, "On the role of model-driven engineering in adaptive systems," in *Computing Conference (CCC), 2016 IEEE 11th Colombian*. IEEE, 2016, pp. 1–8.

[10] J.C. Muńoz-Fernández, R. Mazo, C. Salinesi, and G. Tamura, "10 challenges for the specification of self-adaptive software," in *12th International Conference on Research Challenges in Information Science (RCIS)*, May 2018, pp. 1–12.

[11] F. Bordeleau, B. Combemale, R. Eramo, M. van den Brand, and M. Wimmer, "Tool-support of socio-technical coordination in the context of heterogeneous modeling," in *6th Int. Workshop on the Globalization of Modeling Languages (GEMOC), MODELS 2018 Workshops*, 2018, pp. 1–3.

[12] BKCASE Governing Board, "Guide to the Systems Engineering Body of Knowledge (SEBoK) v. 1.9.1," 2014, p. 945. [Online]. https://bit.ly/2PWwxFJ

[13] T. Huldt and I. Stenius, "State-of-practice survey of model-based systems engineering," *Systems Engineering*, 2018, pp. 1–12 (online first).

[14] OMG, "OMG Systems Modeling Language (SysML), Version 1.6," Object Management Group, 2019. [Online]. https://www.omg.org/spec/SysML/

[15] S. Friedenthal, A. Moore, and R. Steiner, *A practical guide to SysML: the systems modeling language*. Morgan Kaufmann, 2014.

[16] ISO, "ISO/IEC 19514:2017 – Information technology – Object management group systems modeling language (OMG SysML)," International Organization for Standardization, 2017. [Online]. https://www.omg.org/spec/SysML/

[17] A. Soyler and S. Sala-Diakanda, "A model-based systems engineering approach to capturing disaster management systems," in *2010 IEEE International Systems Conference*, apr 2010, pp. 283–287.

[18] A.S. Akbas and W. Karwowski, "A systems engineering approach to modeling and simulating software training management efforts," in *25th European Modeling and Simulation Symposium, EMSS 2013*, 2013, pp. 264–269.

[19] A.S. Akbas, K. Mykoniatis, A. Angelopoulou, and W. Karwowski, "A model-based approach to modeling a hybrid simulation platform (work in progress)," in *Proceedings of the Symposium on Theory of Modeling & Simulation – DEVS Integrative*, DEVS '14. San Diego, CA, USA: Society for Computer Simulation International, 2014, pp. 31:1–31:6. [Online]. http://dl.acm.org/citation.cfm?id=2665008.2665039

[20] D. Amyot, A.A. Anda, M. Baslyman, L. Lessard, and J.M. Bruel, "Towards Improved Requirements Engineering with SysML and the User Requirements Notation," in *2016 IEEE 24th International Requirements Engineering Conference (RE)*, sep 2016, pp. 329–334.

[21] G. Mussbacher, D. Amyot, R. Breu, J.M. Bruel, B.H.C. Cheng et al., "The relevance of model-driven engineering thirty years from now," in *Model-Driven Engineering Languages and Systems*, J. Dingel, W. Schulte, I. Ramos, S. Abrahão, and E. Insfran, Eds. Cham: Springer International Publishing, 2014, pp. 183–200.

[22] J.A. Lane and T. Bohn, "Using SysML modeling to understand and evolve systems of systems," *Systems Engineering*, Vol. 16, No. 1, 2013, pp. 87–98.

[23] C. Ncube and S.L. Lim, "On systems of systems engineering: A requirements engineering perspective and research agenda," in *26th International Requirements Engineering Conference (RE)*. IEEE CS, Aug 2018, pp. 112–123.

[24] A. Van Lamsweerde, *Requirements engineering: From system goals to UML models to software*. Chichester, UK: John Wiley & Sons, 2009, Vol. 10.

[25] S. Woldeamlak, A. Diabat, and D. Svetinovic, "Goal-oriented requirements engineering for research-intensive complex systems: A case study," *Systems Engineering*, Vol. 19, No. 4, 2016, pp. 322–333.

[26] E.S.K. Yu, "Towards modelling and reasoning support for early-phase requirements engineering," in *Requirements Engineering, 1997, Proceedings of the Third IEEE International Symposium on*, 1997, pp. 226–235.

[27] D. Amyot and G. Mussbacher, "User Requirements Notation: the first ten years, the next ten years," *JSW*, Vol. 6, No. 5, 2011, pp. 747–768.

[28] ITU-T, "Recommendation Z.151 (10/18): User Requirements Notation (URN) – Language Definition," 2018. [Online]. http://www.itu.int/rec/T-REC-Z.151/en

[29] M. Daun, J. Brings, L. Krajinski, V. Stenkova, and T. Bandyszak, "A GRL-compliant iStar extension for collaborative cyber-physical systems," *Requirements Engineering*, Vol. 26, No. 4, 2021, pp. 325–370.

[30] K. Neace, R. Roncace, and P. Fomin, "Goal model analysis of autonomy requirements for unmanned aircraft systems," *Requirements Engineering*, Vol. 23, No. 4, 2018, pp. 509–555.

[31] A.A. Anda and D. Amyot, "Traceability management of GRL and SysML models," in *SAM'20: 12th System Analysis and Modelling Conference*. ACM, 2020, pp. 117–126.

[32] D. Amyot, S. Ghanavati, J. Horkoff, G. Mussbacher, L. Peyton et al., "Evaluating goal models within the Goal-oriented Requirement Language," *International Journal of Intelligent Systems*, Vol. 25, No. 8, 2010, pp. 841–877.

[33] D. Amyot, H. Becha, R. Bræk, and J.E. Rossebø, "Next generation service engineering," in *First ITU-T Kaleidoscope Academic Conference – Innovations in NGN: Future Network and Services*, 2008, pp. 195–202.

[34] M. Alenazi, N. Niu, W. Wang, and J. Savolainen, "Using obstacle analysis to support SysML-based model testing for cyber physical systems," in *8th Int. Model-Driven Requirements Engineering Workshop (MODRE)*. IEEE CS, 2018, pp. 46–55.

[35] G. Blair, N. Bencomo, and R.B. France, "Models@ run time," *Computer*, Vol. 42, No. 10, 2009.

[36] F.D. Macías-Escrivá, R. Haber, R. del Toro, and V. Hernandez, "Self-adaptive systems: A survey of current approaches, research challenges and applications," *Expert Systems with Applications*, Vol. 40, No. 18, 2013, pp. 7267–7279.

[37] C. Krupitzer, F.M. Roth, S. VanSyckel, G. Schiele, and C. Becker, "A survey on engineering approaches for self-adaptive systems," *Pervasive and Mobile Computing*, Vol. 17, 2015, pp. 184–206.

[38] B. Porter, R.R. Filho, and P. Dean, "A survey of methodology in self-adaptive systems research," in *International Conference on Autonomic Computing and Self-Organizing Systems (ACSOS 2020)*. IEEE, 2020, pp. 168–177.

[39] F. Zahid, A. Tanveer, M.M. Kuo, and R. Sinha, "A systematic mapping of semi-formal and formal methods in requirements engineering of industrial cyber-physical systems," *Journal of Intelligent Manufacturing*, 2021, pp. 1–36.

[40] J. Horkoff, F.B. Aydemir, E. Cardoso, T. Li, A. Maté et al., "Goal-oriented requirements engineering: an extended systematic mapping study," *Requirements Engineering*, Vol. 24, No. 2, 2019, pp. 133–160.

[41] W. Wang, N. Niu, M. Alenazi, and L. Da Xu, "In-place traceability for automated production systems: A survey of PLC and SysML tools," *IEEE Transactions on Industrial Informatics*, Vol. 15, No. 6, 2018, pp. 3155–3162.

[42] S. Wolny, A. Mazak, C. Carpella, V. Geist, and M. Wimmer, "Thirteen years of SysML: a systematic mapping study," *Software & Systems Modeling*, Vol. 19, No. 1, 2020, pp. 111–169.
[43] B. Kitchenham and S. Charters, "Guidelines for performing systematic literature reviews in software engineering," Keele University and Durham University Joint Report, Tech. Rep. EBSE 2007-001, 2007.
[44] S.J. Tueno Fotso, M. Frappier, R. Laleau, A. Mammar, and M. Leuschel, "Formalisation of SysML/KAOS goal assignments with B system component decompositions," in *Integrated Formal Methods*, C.A. Furia and K. Winter, Eds. Cham: Springer International Publishing, 2018, pp. 377–397.
[45] C. Ingram, Z. Andrews, R. Payne, and N. Plat, "SysML fault modelling in a traffic management system of systems," in *System of Systems Engineering (SOSE), 2014 9th International Conference on*. IEEE, 2014, pp. 124–129.
[46] Y. Vanderperren and W. Dehaene, "SysML and systems engineering applied to UML-based SoC design," in *Proc. of the 2nd UML-SoC Workshop at 42nd DAC, USA*, 2005.
[47] I. Ozkaya, "Representing requirement relationships," in *First International Workshop on Visualization in Requirements Engineering, REV 2006*, 2007.
[48] A. Matoussi, F. Gervais, and R. Laleau, "A goal-based approach to guide the design of an abstract Event-B specification," in *16th International Conference on Engineering of Complex Computer Systems (ICECCS)*. IEEE, 2011, pp. 139–148.
[49] R. Laleau, F. Semmak, A. Matoussi, D. Petit, A. Hammad et al., "A first attempt to combine SysML requirements diagrams and B," *Innovations in Systems and Software Engineering*, Vol. 6, No. 1, 2010, pp. 47–54.
[50] X. Cui and R. Paige, "An integrated framework for system/software requirements development aligning with business motivations," in *Proceedings – 2012 IEEE/ACIS 11th International Conference on Computer and Information Science, ICIS 2012*, 2012, pp. 547–552.
[51] C. Gnaho, R. Laleau, F. Semmak, and J.M. Bruel, "bCMS requirements modelling using SysML/KAOS," 2013. [Online]. https://goo.gl/QU9Tgn
[52] C. Gnaho, F. Semmak, and R. Laleau, "An overview of a SysML extension for goal-oriented NFR modelling: Poster paper," in *IEEE 7th International Conference on Research Challenges in Information Science (RCIS)*, may 2013, pp. 1–2.
[53] A. Mammar and R. Laleau, "On the use of domain and system knowledge modeling in goal-based Event-B specifications," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Vol. 9952 LNCS, 2016, pp. 325–339.
[54] E. Bousse, "Requirements management led by formal verification," Master's thesis, Master's thesis, Computer Science, University of Rennes, France, 2012.
[55] M. Ahmad, N. Belloir, and J.M. Bruel, "Modeling and verification of functional and non-functional requirements of ambient self-adaptive systems," *Journal of Systems and Software*, Vol. 107, 2015, pp. 50–70.
[56] M. Ahmad, J.M. Bruel, R. Laleau, and C. Gnaho, "Using RELAX, SysML and KAOS for ambient systems requirements modeling," in *Procedia Computer Science*, Vol. 10, 2012, pp. 474–481.
[57] M. Ahmad, J. Araújo, N. Belloir, J.M. Bruel, C. Gnaho et al., "Self-adaptive systems requirements modelling: Four related approaches comparison," in *Comparing Requirements Modeling Approaches Workshop (CMA@ RE), 2013 International*. IEEE, 2013, pp. 37–42.
[58] M. Ahmad, I. Dragomir, J.M. Bruel, I. Ober, and N. Belloir, "Early analysis of ambient systems SysML properties using Omega2-IFX," in *SIMULTECH 2013*, 2013.
[59] M. Ahmad, "First step towards a domain specific language for self-adaptive systems," in *10th Annual International Conference on New Technologies of Distributed Systems (NOTERE)*. IEEE, 2010, pp. 285–290.
[60] M. Ahmad and J.M. Bruel, "bCMS requirements modelling using RELAX/SysML/ KAOS," in *3rd CMA Workshop at RE'2013*, 2013.

[61] M. Ahmad and J.M. Bruel, "A comparative study of RELAX and SysML/KAOS," Institut de Recherche en Informatique de Toulouse, University Toulouse II Le Mirail, France, Tech. Rep., 2014.

[62] N. Belloir, V. Chiprianov, M. Ahmad, M. Munier, L. Gallon et al., "Using relax operators into an mde security requirement elicitation process for systems of systems," in *Proceedings of the 2014 European Conference on Software Architecture Workshops*. ACM, 2014, p. 32.

[63] L. Apvrille and Y. Roudier, "SysML-Sec: A SysML environment for the design and development of secure embedded systems," *APCOSEC, Asia-Pacific Council on Systems Engineering*, 2013, pp. 8–11.

[64] L. Apvrille and Y. Roudier, "Designing safe and secure embedded and cyber-physical systems with SysML-Sec," in *International Conference on Model-Driven Engineering and Software Development*. Springer, 2015, pp. 293–308.

[65] Y. Roudier and L. Apvrille, "SysML-Sec: A model driven approach for designing safe and secure systems," in *Model-Driven Engineering and Software Development (MODELSWARD), 2015 3rd International Conference on*. IEEE, 2015, pp. 655–664.

[66] A. Tsadimas, M. Nikolaidou, and D. Anagnostopoulos, "Extending SysML to explore non-functional requirements: the case of information system design," in *Proceedings of the 27th Annual ACM Symposium on Applied Computing*. ACM, 2012, pp. 1057–1062.

[67] D. Spyropoulos and J.S. Baras, "Extending design capabilities of SysML with trade-off analysis: Electrical microgrid case study," *Procedia Computer Science*, Vol. 16, 2013, pp. 108–117.

[68] O. Badreddin, V. Abdelzad, T.C. Lethbridge, and M. Elaasar, "FSysML: Foundational executable SysML for cyber-physical system modeling," in *CEUR Workshop Proceedings*, Vol. 1731, 2016, pp. 38–51.

[69] Z. Fan, T. Yue, and L. Zhang, "SAMM: an architecture modeling methodology for ship command and control systems," *Software and Systems Modeling*, Vol. 15, No. 1, 2016, pp. 71–118.

[70] H. Wang, "Multi-Level Requirement Model and Its Implementation For Medical Device," Master's thesis, Master's thesis, Mechanical and Energy Engineering, Purdue University, United States, 2018.

[71] S. Lee, S. Park, and Y.B. Park, "Self-adaptive system verification based on SysML," in *2019 International Conference on Electronics, Information, and Communication (ICEIC)*. IEEE CS, 2019, pp. 1–3.

[72] I. Maskani, J. Boutahar, and S. El Ghazi El Houssaïni, "Modeling telemedicine security requirements using a SysML security extension," in *2018 6th International Conference on Multimedia Computing and Systems*, 2018, pp. 1–6.

[73] A. Anda and D. Amyot, "An optimization modeling method for adaptive systems based on goal and feature models," in *Tenth International Model-Driven Requirements Engineering (MoDRE)*. IEEE, 2020, pp. 11–20.

[74] A.A. Anda and D. Amyot, "Arithmetic semantics of feature and goal models for adaptive cyber-physical systems," in *27th International Requirements Engineering Conference (RE)*. IEEE, 2019, pp. 245–256.

[75] A.A. Anda, "Modeling adaptive socio-cyber-physical systems with goals and SysML," in *26th International Requirements Engineering Conference (RE)*. IEEE CS, 2018, pp. 442–447.

[76] OMG, "Business Motivation Model (BMM), Version 1.3," Object Management Group, 2015. [Online]. https://www.omg.org/spec/BMM/

[77] J. Whittle, P. Sawyer, N. Bencomo, B.H.C. Cheng, and J.M. Bruel, "RELAX: A language to address uncertainty in self-adaptive systems requirement," *Requirements Engineering*, Vol. 15, No. 2, 2010, pp. 177–196.

[78] Y. Fan, A.A. Anda, and D. Amyot, "An arithmetic semantics for GRL goal models with function generation," in *System Analysis and Modeling. Languages, Methods, and Tools for Systems Engineering*, F. Khendek and R. Gotzhein, Eds. Cham: Springer International Publishing, 2018, pp. 144–162.

[79] I. Mistrik, N. Ali, R. Kazman, J. Grundy, and B. Schmerl, *Managing Trade-offs in Adaptable Software Architectures*. Morgan Kaufmann, 2016.

[80] M. Salehie and L. Tahvildari, "Self-adaptive software: Landscape and research challenges," *ACM transactions on autonomous and adaptive systems (TAAS)*, Vol. 4, No. 2, 2009, p. 14.

[81] J. Andersson, R. De Lemos, S. Malek, and D. Weyns, "Modeling dimensions of self-adaptive software systems," *Software engineering for self-adaptive systems*, 2009, pp. 27–47.

[82] M. Morandini, L. Penserini, and A. Perini, "Automated mapping from goal models to self-adaptive systems," in *Proceedings of the 23rd IEEE/ACM International Conference on Automated Software Engineering*. IEEE Computer Society, 2008, pp. 485–486.

[83] M. Morandini, L. Penserini, A. Perini, and A. Marchetto, "Engineering requirements for adaptive systems," *Requirements Engineering*, Vol. 22, No. 1, 2017, pp. 77–103.

[84] P. Bareiß, D. Schütz, R. Priego, M. Marcos, and B. Vogel-Heuser, "A model-based failure recovery approach for automated production systems combining SysML and industrial standards," in *2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA)*, sep 2016, pp. 1–7.

[85] J. Parri, F. Patara, S. Sampietro, and E. Vicario, "A framework for model-driven engineering of resilient software-controlled systems," *Computing*, 2020, pp. 1–24.

[86] L. Baresi, L. Pasquale, and P. Spoletini, "Fuzzy goals for requirements-driven adaptation," in *Requirements Engineering Conference (RE), 2010 18th IEEE International*. IEEE, 2010, pp. 125–134.

[87] L. Baresi and L. Pasquale, "Adaptive goals for self-adaptive service compositions," in *Web Services (ICWS), 2010 IEEE international conference on*. IEEE, 2010, pp. 353–360.

[88] L. Baresi and L. Pasquale, "Live goals for adaptive service compositions," *Proceedings of the 2010 ICSE Workshop on Software*, 2010.

[89] M. Hussein, S. Li, and A. Radermacher, "Model-driven development of adaptive iot systems." in *MODELS (Satellite Events)*, 2017, pp. 17–23.

[90] S. Meacham, "Towards self-adaptive IoT applications: Requirements and adaptivity patterns for a fall-detection ambient assisting living application," in *Components and Services for IoT Platforms*. Springer, 2017, pp. 89–102.

[91] F.G.C. Ribeiro, S. Misra, and M.S. Soares, "Application of an extended SysML requirements diagram to model real-time control systems," in *International Conference on Computational Science and Its Applications*. Springer, 2013, pp. 70–81.

[92] A.J. Lopes, R. Lezama, and R. Pineda, "Model Based Systems Engineering for Smart Grids as systems of systems," in *Procedia Computer Science*, Vol. 6, 2011, pp. 441–450.

[93] L.S. Souza, S. Misra, and M.S. Soares, "SmartCitySysML: A SysML Profile for Smart Cities Applications," in *Computational Science and Its Applications – ICCSA 2020*. LNCS 12254, Springer, 2020, pp. 383–397.

[94] W. Qian, X. Peng, B. Chen, J. Mylopoulos, H. Wang et al., "Rationalism with a dose of empiricism: combining goal reasoning and case-based reasoning for self-adaptive software systems," *Requirements Engineering*, Vol. 20, No. 3, 2015, pp. 233–252.

[95] R. Ramnath, V. Gupta, and J. Ramanathan, "RED-Transaction and Goal-Model Based Analysis of Layered Security of Physical Spaces," in *Computer Software and Applications, 2008. COMPSAC'08. 32nd Annual IEEE International*. IEEE, 2008, pp. 679–685.

[96] O. Ginigeme and A. Fabregas, "Model based systems engineering high level design of a sustainable electric vehicle charging and swapping station using discrete event simulation," in *2018 Annual IEEE International Systems Conference (SysCon)*. IEEE, 2018, pp. 1–6.

[97] J. Horkoff, R. Salay, M. Chechik, and A. Di Sandro, "Supporting early decision-making in the presence of uncertainty," in *Requirements Engineering Conference (RE), 2014 IEEE 22nd International*. IEEE, 2014, pp. 33–42.

[98] J.B. Warmer and A.G. Kleppe, *The object constraint language: Precise modeling with UML (Addison-Wesley Object Technology Series)*. Addison-Wesley Professional, 1998.

[99] R. Salay, M. Famelis, and M. Chechik, "Language independent refinement using partial modeling," in *Fundamental Approaches to Software Engineering*, J. de Lara and A. Zisman, Eds. Springer Berlin Heidelberg, 2012, pp. 224–239.

[100] A. Pnueli, "The temporal logic of programs," in *Foundations of Computer Science, 1977., 18th Annual Symposium on*. IEEE, 1977, pp. 46–57.

[101] W. Emmerich, B. Butchart, L. Chen, B. Wassermann, and S. Price, "Grid service orchestration using the business process execution language (BPEL)," *Journal of Grid Computing*, Vol. 3, No. 3-4, 2005, pp. 283–304, cited By 104.

[102] A. Rahman and D. Amyot, "A DSL for importing models in a requirements management system," in *4th International Model-Driven Requirements Engineering Workshop (MoDRE)*. IEEE CS, 2014, pp. 37–46.

[103] IBM, "Rational DOORS v9.6.1," 2018. [Online]. http://goo.gl/yGWpze

[104] B.H.C. Cheng, R. De Lemos, H. Giese, P. Inverardi, and J. Magee et al., "Software Engineering for Self-Adaptive Systems: A Research Roadmap," in *Software engineering for self-adaptive systems*, Vol. LNCS 5525. Springer, 2009, pp. 1–26.

[105] S.A. Alwidian, M. Dhaouadi, and M. Famelis, "A vision towards a conceptual basis for the systematic treatment of uncertainty in goal modelling," in *SAM'20: 12th System Analysis and Modelling Conference*, A. Gherbi, W. Hamou-Lhadj, and A. Bali, Eds. ACM, 2020, pp. 139–142.

[106] P. Bresciani, A. Perini, P. Giorgini, F. Giunchiglia, and J. Mylopoulos, "Tropos: An agent-oriented software development methodology," *Autonomous Agents and Multi-Agent Systems*, Vol. 8, No. 3, 2004, pp. 203–236.

[107] P. Giorgini, J. Mylopoulos, E. Nicchiarelli, and R. Sebastiani, "Reasoning with goal models," in *International Conference on Conceptual Modeling*. Springer, 2002, pp. 167–181.

[108] J.O. Kephart and W.E. Walsh, "An artificial intelligence perspective on autonomic computing policies," in *Fifth IEEE International Workshop on Policies for Distributed Systems and Networks*. IEEE, 2004, pp. 3–12.

[109] R. de Lemos, H. Giese, H.A. Müller, M. Shaw, J. Andersson et al., "Software engineering for self-adaptive systems: A second research roadmap," in *Software Engineering for Self-Adaptive Systems II*. Springer, 2013, pp. 1–32.

[110] Z. Yang, Z. Li, Z. Jin, and Y. Chen, "A systematic literature review of requirements modeling and analysis for self-adaptive systems," in *Requirements Engineering: Foundation for Software Quality*, C. Salinesi and I. van de Weerd, Eds. Springer, 2014, pp. 55–71.

[111] R. Feldt and A. Magazinius, "Validity threats in empirical software engineering research – An initial survey," in *SEKE*, 2010, pp. 374–379.

[112] A. Ampatzoglou, S. Bibi, P. Avgeriou, M. Verbeek, and A. Chatzigeorgiou, "Identifying, categorizing and mitigating threats to validity in software engineering secondary studies," *Information and Software Technology*, Vol. 106, 2019, pp. 201–230.

[113] B. Combemale, J.A. Kienzle, and G. Mussbacher et al., "A hitchhiker's guide to model-driven engineering for data-centric systems," *IEEE Software*, Vol. 38, No. 4, 2021, pp. 71–84.

A. Quality assessment

This appendix complements Section 2.2.4 by presenting, in Tables A1 and A2, the result of the quality assessment against the criteria explained in Table 2. The color coding reflects how positive a result is (green is positive, yellow is neutral, and red is negative).

Table A1. Assessment of the studies on Goal/SysML against the identified quality criteria (Y = Yes, N = No, P = Partially, ? = Not provided)

Research study	C1	C2	C3	C4	C5	C6	C7	C8	C9
Amyot et al. 2016 [20]	Y	Y	Y	N	N	Y	Y	Y	N
Ahmad 2010 [59]	Y	Y	Y	Y	N	N	?	Y	P
Ahmad et al. 2012 [56]	Y	Y	N	Y	Y	Y	Y	Y	Y
Ahmad et al. 2013 [57]	Y	?	?	?	?	Y	?	P	N
Ahmad et al. 2013 [58]	Y	Y	N	Y	Y	Y	Y	P	P
Ahmad et al. 2015 [55]	Y	Y	Y	Y	Y	Y	Y	Y	Y
Ahmad and Bruel 2013 [60]	Y	Y	N	Y	Y	Y	Y	P	P
Ahmad and Bruel 2014 [61]	Y	?	?	?	?	N	?	Y	Y
Anda and Amyot 2020 [31]	Y	Y	Y	Y	Y	Y	Y	N	N
Anda and Amyot 2020 [73]	Y	Y	Y	Y	Y	N	?	Y	Y
Anda 2018 [75]	Y	Y	Y	Y	Y	N	?	Y	Y
Anda and Amyot 2019 [74]	Y	Y	Y	Y	Y	N	?	Y	Y
Apvrille and Roudier 2013 [63]	Y	Y	Y	Y	Y	N	?	N	N
Apvrille and Roudier 2015 [64]	Y	Y	N	Y	Y	Y	Y	N	N
Badreddin et al. 2016 [68]	Y	Y	Y	P	P	Y	Y	Y	N
Belloir et al. 2014 [62]	Y	Y	N	Y	Y	Y	Y	Y	P
Bousse 2012 [54]	Y	P	P	N	N	N	?	N	N
Cui and Paige 2012 [50]	Y	Y	Y	Y	Y	Y	P	N	N
Fan et al. 2016 [69]	Y	Y	Y	Y	Y	Y	Y	N	N
Gnaho et al. 2013 [51]	Y	Y	N	Y	Y	Y	Y	N	N
Gnaho et al. 2013 [52]	Y	Y	P	Y	Y	N	?	N	N
Ingram et al. 2014 [45]	Y	Y	Y	Y	Y	Y	Y	Y	Y
Laleau et al. 2014 [49]	Y	Y	Y	Y	Y	Y	Y	N	N
Lee et al. 2019 [71]	Y	Y	Y	Y	Y	N	?	N	N
Mammar and Laleau 2016 [53]	Y	Y	N	Y	Y	Y	P	N	N
Maskani el al. 2018 [72]	Y	Y	Y	Y	Y	Y	Y	N	N
Matoussi et al. 2011 [48]	Y	Y	N	Y	Y	N	?	N	N
Ozkaya 2007 [47]	Y	P	Y	N	N	N	?	N	N
Roudier and Apvrille 2015 [65]	Y	Y	N	Y	Y	Y	Y	N	N
Spyropoulos and Baras [67]	Y	Y	Y	Y	Y	Y	Y	P	N
Tsadimas et al. 2012 [66]	Y	Y	Y	Y	Y	Y	Y	N	N
Vanderperren and Dehaene 2005 [46]	Y	P	Y	P	N	N	?	P	N
Wang 2018 [70]	Y	Y	Y	Y	Y	Y	Y	N	N

Table A2. Assessment of the adaptation studies on Goal *or* SysML searches against the identified quality criteria (Y = Yes, N = No, P = Partially, ? = Not provided)

Research study	C1	C2	C3	C4	C5	C6	C7	C8	C9
Goals and Adaptation									
Baresi et al. 2010 [86]	Y	Y	Y	P	Y	Y	Y	Y	P
Baresi and Pasquale 2010 [87]	Y	Y	P	Y	Y	Y	P	Y	Y
Baresi and Pasquale 2010 [88]	Y	Y	P	P	Y	Y	Y	Y	P
Horkoff et al. 2014 [97]	Y	Y	Y	Y	Y	N	?	N	N
Morandini et al. 2008 [82]	Y	Y	N	N	Y	Y	Y	P	N
Morandini et al. 2017 [83]	Y	Y	Y	Y	Y	Y	Y	Y	Y
Qian et al. 2015 [94]	Y	Y	Y	Y	Y	N	?	Y	Y
Ramnath et al. 2008 [95]	Y	Y	Y	Y	Y	Y	Y	Y	Y
SysML and Adaptation									
Akbas and Karwowski 2013 [18]	Y	Y	N	Y	Y	Y	Y	N	N
Akbas et al. 2014 [19]	Y	Y	Y	Y	Y	Y	Y	N	N
Bareiß et al. 2016 [84]	Y	Y	Y	Y	Y	Y	Y	Y	Y
Ginigeme and Fabregas 2018 [96]	Y	Y	Y	Y	Y	N	?	N	N
Hussein et al. 2017 [89]	Y	Y	Y	Y	Y	Y	Y	N	N
Lopes et al. 2011 [92]	Y	Y	Y	Y	Y	Y	P	P	P
Meacham 2017 [90]	Y	Y	Y	Y	Y	Y	Y	Y	N
Parri et al. 2020 [85]	Y	Y	Y	Y	Y	Y	Y	Y	P
Ribeiro et al. 2013 [91]	Y	Y	Y	Y	Y	Y	Y	P	N
Soyler and Sala-Diakanda 2010 [17]	Y	Y	Y	P	P	Y	Y	P	P
Souza et al. 2020 [93]	Y	Y	Y	P	P	Y	P	P	N

e-Informatica Software Engineering Journal, Volume 16, Issue 1, 2022, pages: 220102, DOI: 10.37190/e-Inf220102

Analysis of Factors Influencing Developers' Sentiments in Commit Logs: Insights from Applying Sentiment Analysis

Rajdeep Kaur*, Kuljit Kaur Chahal*, Munish Saini**

*Department of Computer Science, Guru Nanak Dev University, Amritsar, India

**Department of Computer Engineering and Technology, Guru Nanak Dev University, Amritsar, India

rajdeep.rsh@gndu.ac.in, kuljitchahal.cse@gndu.ac.in, munish.cet@gndu.ac.in

Abstract

Background: In the open source software paradigm, software development depends upon efforts of volunteer members that are geographically dispersed and collaborate with each other over the Internet. Communication artifacts like mailing lists, forums, and issue tracking systems are used by developers for communication. The way they express themselves through these communication channels greatly influences their productivity, efficiency of development activities, and survival of the project as well. Therefore, it is essential to understand affective state of developers' contributions to make software engineering more effective.

Aim: This study examined commit logs of seven GitHub projects to analyze developers' sentiments. This study also investigated the relationship of developers' sentiments in commit logs with team size of project, type of change activity, and contribution volume.

Method: Sentiments of developers are calculated using SentiStrength-SE tool that is specialized in software engineering domain.

Results: Our findings revealed that the majority of sentiments conveyed by developers in commit logs were neutral. Furthermore, we found that team size, change activity, and commit contribution volume influenced sentiments conveyed in commit logs.

Conclusion: Our findings will help project managers to better understand developer sentiments while performing different software development tasks/activities. It will be beneficial in improving developer productivity and retention.

Keywords: human factors in software development teams, software developer, developers' sentiment, sentiment analysis, commit logs, developer activity type, and team size

1. Introduction

Sentiments of software developers greatly influence the quality and productivity of developed software [1]. Prior studies confirm that emotions impact task quality, productivity, creativity, group rapport, and job satisfaction [2]. Due to advancements in Natural Language Processing (NLP) and significance of human computer interaction, research associated with sentiments and emotional aspects of software developers' communication is gaining more traction in the software engineering domain.

Submitted: 16 Jun. 2021; Revised: 10 Sep. 2021; Accepted: 23 Sep. 2021; Available online: 15 Dec. 2021

Sentiment analysis is an opinion mining method used to identify people's sentiments, views, evaluations, feelings, attitudes, and appraisals about products, organizations, services, topics, events, issues, individuals, and their attributes [3]. It is basically used to classify opinion in written text into positive, negative, and neutral. Sentiment analysis was first introduced by Liu et al. [3]. Originally, sentiment analysis was used to detect the polarity of small text posted in product reviews, movie reviews, tweets, and microblogs [4]. In recent times, this technique is widely adopted by software engineering community and applied to various software artifacts like commit logs [4–7], mailing lists messages [8], issue comments [9, 10], code reviews [11], bug reports [12]. In order to better support developers during software development activities and understanding the social factors that affect productivity and retention, it is necessary to understand their sentiment in various software development tasks. This information may help managers of OSS software projects to better support developers with tools during software development and resolve the issues related to various tasks. Thus, it will help in improving developers' productivity as well as retention.

In the present field of study, we observed significant work done by different researchers to examine developers' sentiments in commit messages of OSS (Open Source Software) [4–7], etc. However to the best of our knowledge, none of them analyzed the relationship of type of change activity performed by developers, their commit contribution, and team size of a project (Large, Medium, and Small) with sentiments expressed by developers in commit logs. Our work also looks into the evolution of sentiments with respect to time. Thus lack of research in the domain motivated us to conduct this research work.

In this work, we investigated the sentiments of developers conveyed in commit logs. Sinha et al. [5] also examined the developers' sentiments in commit logs and relate the sentiments in commit messages with the day of week and number of changed files but our study has a different objective. We studied the developers' sentiments across seven well-known GitHub projects to examine the impact of project team size on developers' sentiments. Furthermore, type of change activity executed by developers was considered and then analyzed the impact of Type-1 (add + modify), Type-2 (delete + modify), and Type-3 (add + delete + modify) activity on the sentiment of developers projected in the commit logs. The existing literature reported three types of change activity viz. addition, deletion, and modification [4]. We grouped the individual change activity into combinations of two or three file change activities to create our own classification scheme. Apart from this, the authors also investigated the association between commit contribution and volume of sentiment. Sentiment volume is percentage of sentiments (positive, negative, and neutral) conveyed by individual developer in the commit log and commit contribution size is percentage of commits made by individual developer. Besides, our work also examined the evolution of sentiment in the project with respect to commits that is not taken into account by Sinha et al. [5]. To achieve the aforementioned objective, we formulated the following research questions:

RQ1: What are the overall developers' sentiments in the commit logs?

– Developers' inactivity in the project is associated with their negative and positive mood value [12]. Thus an understanding of developers sentiment attached to commit activity might be helpful for project managers in introducing measures to manage developers' sentiments that may ensure the stability of developers.

RQ2: Is there any relation between sentiments and team size of a project?

– The accomplishment of the large project relies on a large number of developers and a long development period. Developers working with a large code base may lead to negative emotions in the project due to workload and stress in managing a large code base. Moreover, staffing and task allocation is a complex task in large projects. Thus,

this makes it difficult to manage projects, and the decision of managers largely influences the mood of developers. Thus an understanding of impact of project team size can be used to effectively manage developers' emotions in the project that may lead to high productivity and improved job satisfaction [13].

RQ3: Does the type of change activity performed by a developer impact their sentiments in commit messages?

– Developers who convey positive emotions while executing a particular development task might be more efficient and fast in accomplishing a task [14] that will reduce cost of software. Thus understanding developers' sentiments attached to a particular task can be helpful in effective task allocation. For example, making tasks (read issues in an issue tracking system) simple to understand, and easy to solve by decomposing complex issues into smaller ones can improve developer productivity, and sentiment in commit logs.

RQ4: Is there any relation between developer sentiment volumes and commit contribution size?

– Understanding emotional state of developers involved in high or low commit activity may help project managers to effectively distribute workload among developers and increasing development activity as well as boosting neutral or positive sentiments.

RQ5: How has sentiment in the commit logs evolved over the period of time?

– Analyzing the evolution of sentiments, we can identify trends in sentiment expression in commit logs. Is it getting negative or positive? A particular time slot when sentiments in commit logs are shifting direction e.g. becoming more positive, we can identify the reasons and try to maintain that state. For example, it has been observed in this study that reduced negativity in commit logs coincides with launch of the Github platform in 2008. Managers can take motivational steps to boost developers that may increase their retention in the project.

RQ1 aims to identify general developers' sentiments conveyed in commit messages. RQ2 aims to discover the impact of team size on the sentiments expressed by developers in commit messages. RQ3 identifies the association between three types of changes activities (Type-1, Type-2, Type-3) performed by developers and their expressed sentiments. Type-1, Type-2, and Type-3 are combinations of two or more individual file change types (addition, deletion, and modification). RQ4 intends to ascertain the impact of developers commit contribution on sentiment volume. Sentiment volume is defined as a percentage of Positive, Negative, and Neutral sentiments conveyed by each developer in the commit log, and commit contribution is a percentage of commits made by each contributor in the project. RQ5 examined the evolution of sentiments with respect to the number of commits made by developers over the period of time.

Our study uses the Sentistrength-SE [15] tool to detect polarity of sentiments conveyed in commit logs messages. This tool used lexical approach and domain dictionary and specially designed for software engineering text.

The remainder of the paper is organized as following: Existing work related to current study is discussed in Section 2. Description of data collection methodology along with detail of sentiment analysis approach used to detect sentiments of developers in commit logs is presented in Section 3. The results of study are discussed in Section 4. Discussion is presented in Section 5. Some Threats to Validity are described in Section 6. Conclusions along with some future directions are presented in Section 7.

2. Related literature

Many studies have been conducted by researchers and practitioners in the past to analyze the developers' sentiments in OSS code repositories and related artifacts. They examined developers' sentiments in different software artifacts such as commit logs, commit comments, mailing list messages, and GitHub security debates. A summary of the related literature is presented in Table 1.

Some researchers evaluated the performance of SE-specific sentiment analysis tools, compared them in terms of accuracy, and proposed techniques to improve existing sentiment analysis tools. Novielli et al. [16] in 2021 presented a replication study to evaluate the performance of SE-specific tools. Sun et al. [17] proposed sentence structure to improve sentiment analysis in software engineering text. Biswas et al. [18] in 2020, investigated the effectiveness of a customized language representation model known as BERT and Novielli et al. [19] assessed the performance of four SE domain specific tools viz. Senti4SD, SentiCR, SentiStrength-SE, and DEVA in cross-platform. M. R. Wrobel [20] investigated the influence of adoption of lexicons on emotion mining in SE artifacts.

In the year 2021, Martin Obaidi and Jil Klünder [21] presented a systematic literature review of sentiment analysis tools designed for and applied in a software engineering context. This study explored sentiments analysis tools used in the software engineering field, utilized data sets, application areas of sentiment analysis tools, and problems faced at the time of developing such kinds of tools.

Some researchers explored the sentiment variation based on different factors and also examined the association of sentiments with various factors. Huq et al. [22] in 2020 examined the relation of sentiments with software bugs. In the same year, Kaur and Chahal presented investigation of developers' sentiments in commit comments [23]. In the year 2019, Paul et al. [11] analyzed the code review data of five open source projects to investigate the difference in expression of sentiments based on the gender of developers during various software development tasks. In the year 2018, Bharti and Singh [24] surveyed 20 software professionals to examine the developers' sentiments associated with code cloning practices. Islam and Zibran [7] studied the variance in emotion in commit messages that are related to bug introduction and bug fixing activities. Singh et al. [5] have analyzed the 3,171 commit messages that are related to refactoring activities to investigate the impact of 15 different code refactoring tasks on developers' sentiments. This study identified that the developers' sentiments are more negative during refactoring activities. Souza and Silva [25] examined the relationship between sentiments of developers and build breakage in a Travis CI (continuous integration). Sinha et al. [5] investigated the developer sentiment in the commit logs of GitHub projects and studied the association among developer sentiment and day of the week. They also examined the correlation between developer sentiment and the number of files changes performed by the developer in the commits. This study demonstrates that most of the sentiments projected by developers in the commit log were neutral. The negative sentiments are 10% higher than the positive and the majority of the negative sentiment was detected on Tuesday.

Islam and Zibran [13] investigated sentiments variation based on different types of tasks executed by developers, development period, in different size projects, and impact of emotions on software artifacts (i.e., length of commit message). Garcia et al. [12] analyzed the data of the bug tracking system and mailing list to examine the association between emotions and contributor activity.

Guzman et al. [4] examined commit comments of GitHub projects to investigate the relation of developer sentiment with the programming language used by the project, time

Table 1. Summary of related studies

Author and year	Scenario of motivation	Possible extension
Huq et al. (2020)	Examined the correlation between sentiments and software bugs	The relationship between sentiments and three types of file change activity can be explored.
Paul et al. (2019)	Examined the sentiments of developers in code review comments.	Developers' sentiments can be explored in commit logs messages.
Sinha et al. (2018)	Investigated the relation between the number of file changes and developers sentiments.	Relation between different combinations of file change can be explored.
Singh et al. (2017)	Examined the impact of software code refactoring activities on the sentiments of developers.	The impact of commit contribution on developers' sentiments can be explored.
Tourani et al. (2014)	Explore the existence of positive and negative emotions in user and developer mailing lists.	Commit logs can be explored to detect developer sentiments and various factors influencing sentiments.
Guzman et al. (2014)	Explored the association of emotion with team geographical location and day and time of the week.	Relation of sentiments with team size can be explored. The evolution of sentiments with respect to the number of commits over time can be explored.
Garcia et al. (2013)	Ascertain the association between emotions and contributor activity.	The relation of commit contribution with developers' sentiments can be explored.
Md Rakibul Islam and Minhaz F. Zibran, (2016)	Examined the impact of project and team size and length of commit message on emotional states of developers.	The impact of large, medium, and small team size projects on the sentiments of developers can be explored.
Pletea et al. (2014)	Explored the emotional expression in security discussions by analyzing commits and pull request comments.	Commit logs messages can be analyzed to explore sentiment expressed in different combinations of change activities.
Khan et al. (2010)	Analyzed the effect of emotions on software developers' debugging performance.	The impact of sentiments on commit contribution can be investigated.
Muller and Fritz, (2015)	Investigated developers' emotions and progress on change tasks by conducting lab study.	Emotions conveyed in software artifacts such as commit log can be explored.
Graziotin et al. (2014)	Explored the connection between developer emotion and their ability to solve analytical problems.	The association of sentiments with different file change activities can be explored.
Michal R. Wrobel, (2013)	Conducted a survey to investigate developers' emotions in the software development process and impact of emotions on performance.	Software artifacts such as commit logs can be examined to investigate developers' conveyed emotions.

and day of the week when the comment was written, team dispersal, and project approbation. This work revealed that java projects have more negative comments. The more positive comments are detected in projects having distributed teams and Monday was the most negative day for sentiments.

Tourani et al. [8] presented a study to investigate the presence of positive and negative emotions in user and developer mailing lists. This study found that both types of mailing lists have positive as well as negative sentiments and have a different focus.

Pletea et al. [26] examined sentiments associated with security discussions in commits and pull requests. This study identified that negative emotions are higher in security debates in comparison to non-security discussions. Khan et al. [27] have analyzed the impact of emotions on the debugging performance of software programmers. Müller and Fritz [28] presented a study on developers' emotions and progress on change tasks. Graziotin et al. [29] examined the association between developer emotion and their ability to solve analytical problems. They found that happy software developers are better at solving analytical problems. In the year 2013, Wrobel [30] presented a study on developers' emotions in the software development process by conducting a survey.

To the best of our knowledge, the work presented in the past does not explore the impact of team size of the project, type of change activity, and commit contribution on sentiments of developers. The work presented in this paper is motivated by Sinha et al. [5]. This study investigated the relation of the day of week and number of changed files with developers' sentiments. But this study does not explore the association of combinations of change activity type and commit contribution with developers' sentiments. One another study presented by Guzman et al. [4] examined the sentiments expressed by developers in commit comments and investigate their association with different factors like time and weekday, project approval, coding language, and team geographical distribution. But this study does not consider the team size of the project and its association with developers' sentiments [4]. Thus lack of research in the field motivated us to conduct this research work. Our work examined the whole commit logs of seven GitHub projects to analyze sentiments of software developers projected in commit logs and investigate the effect of team size, type of change activity, and commit contribution on the developers' sentiments. Furthermore, we also look into the evolution of sentiments to identify how these changes across the years along with the number of commits. We utilized the SentiStrength-SE tool to perform sentiment analysis. We selected this tool because it is the first domain-specific tool specially designed to detect sentiments in a software engineering context and provides better accuracy in comparison to the existing domain-independent sentiment analysis tools/toolkits [31].

3. Analysis methodology

In this section, we provide a description of the dataset along with details of the approach used to conduct sentiment analysis.

3.1. Data collection

GitHub is a popular version control and project management system that provides multiple collaborative artifacts viz commits, issues, and pull requests to contributors [32]. We extracted the data of seven GitHub projects. The projects were selected based on popularity, size, number of commits, number of contributors involved, long project history (more than 10 years), and having a valid Git (distributed version control system) repository. The projects have creation dates from 1972 to 2007. Table 2 describes the quantitative details of the projects. An overview of the selected projects is given below.

Table 2. Detail description of projects

Sr. No.	Name	Project size (in lines of code)	Number of stars	Number of commits	Number of developers	Start date	End date
1.	PostgreSQL	1,113,634	8,406	66329	51	Jul. 1996	Feb. 2019
2.	Glibc	1,305,634	547	49216	538	Jan. 1992	Feb. 2019
3.	Eclipse-CDT	1,498,813	141	30651	260	Jun. 2002	Feb. 2019
4.	GNUCash	2,361,864	1923	25372	185	Nov. 1997	Feb. 2019
5.	WordPress	1,549,456	15,135	44388	96	Apr. 2003	Feb. 2019
6.	Firebug	492,078	1,289	13060	47	Aug. 2007	Oct. 2017
7.	Rhino	806,709	2,896	3903	82	Apr. 1999	Feb. 2019

PostgreSQL is an open source RDBMS (relational database management system). Glibc is a GNU C library most commonly used by GNU/Linux system. Eclipse-CDT is an IDE (integrated development environment) for developing programs in C and C++. GNUCash is accounting software developed for individual and small businesses. WordPress is a PHP and MySQL based content management software. Firebug is a web browser extension for Mozilla Firefox. Rhino is an open source JavaScript implementation that is completely written in Java. Generally, scripting for end users is implemented in java application. We accessed the repositories of the projects from GitHub[1] or git[2]. The Git Bash tool was utilized to clone project repositories to the local machine. The commit logs of the projects were retrieved using the git log command. Commit logs of all selected projects were analyzed from their beginning to till February 2019. In case of Firebug ending period is October 2017.

3.2. Sentiment analysis

There are a variety of sentiment analysis tools viz. SentiStrength [33], StafordNLP [34], and NLTK [35], while most of them do not focus on technical text. As these tools are designed for non-technical text such as movie reviews or blogs posted on social networking sites such as twitter, their results are erroneous for technical artifacts in the Software Engineering (SE) domain [36]. Therefore, domain-specific techniques provide better accuracy to detect sentiments in software engineering text.

We used sentiment analysis tool SentiStrength-SE proposed by Islam and Zibran [15] to perform sentiment analysis on commit logs. Similar choice is made by Md Rakibul Islam and Minhaz F. Zibran in Software engineering domain to extract emotional score from commit messages [7]. Using SentiStrength as the baseline, this tool implements a lexical based approach and domain specific dictionary. We selected this tool because it is a first SE specific tool specially designed for Software Engineering to conduct sentiment analysis and it outperforms the existing domain-independent tools/toolkits [31]. SentiStrength-SE tokenizes the text into words and assigns a score to each word that conveys the underlying sentiment. The words with positive sentiment receive a score between +1 to +5 and words with negative score range between −1 to −5. The neutral score of words ranges between +1 to −1. The scoring is generated using a sentiment dictionary that includes the predetermined polarity score of sentiment words and phrases [32]. Sentistrength-SE

[1]https://github.com
[2]https://git-scm.com

provides maximum positive and maximum negative score of each sentence. The final score of sentence is calculated by adding maximum positive and maximum negative score by following the approach used by jongjelling et al. [37]. The methodology used for sentiment analysis is illustrated in Figure 1.

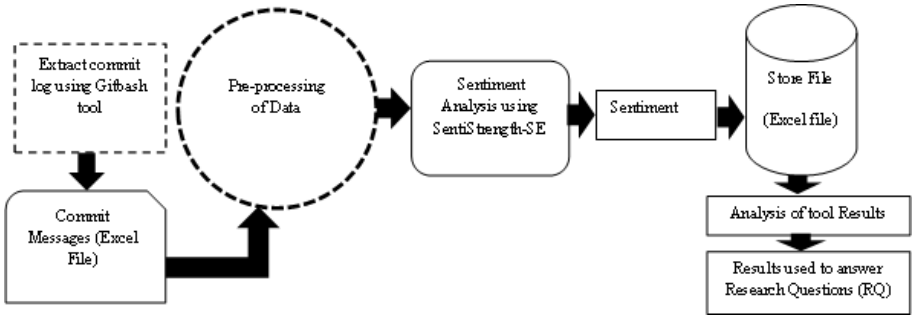


Figure 1. Methodology used for sentiment analysis

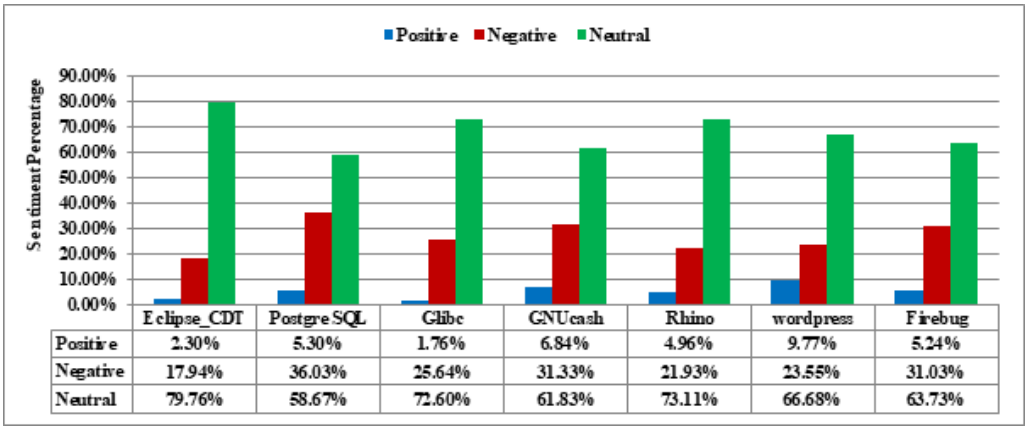
Firstly, we extracted the commit log using the git log command available in the Git Bash tool and saved the commit log data in CSV format. In the next step, extracted the commit messages and pre-processed the collected data to remove stop words, white spaces, non-alphanumeric symbols/characters, and punctuation marks from the text. In addition, also removed code, URLs, and system generated messages, e.g., error messages. Then sentiment analysis is performed using SentiStrength-SE tool. Finally, we get the sentiment score of each commit message.

4. Results and analysis

In this section, we report the results of each research question formulated in Section 1.

RQ1: What is the general developer sentiment in the commit logs?

We examined a total of 86,515 commit messages of seven OSS projects to analyze developers' sentiments in commit logs. Commit logs of all selected projects were analyzed from their beginning until the last observation date set by this study (Refer Table 2). Results of sentiment analysis using SentiStrength-SE are illustrated in Figure 2. Table 3



	Eclipse_CDT	PostgreSQL	Glibc	GNUcash	Rhino	wordpress	Firebug
Positive	2.30%	5.30%	1.76%	6.84%	4.96%	9.77%	5.24%
Negative	17.94%	36.03%	25.64%	31.33%	21.93%	23.55%	31.03%
Neutral	79.76%	58.67%	72.60%	61.83%	73.11%	66.68%	63.73%

Figure 2. Sentiments across all projects

Table 3. Commit messages with positive, negative, and neutral sentiment

Sentiment	Commit message	Final sentiment score
Positive	Add test case for pthread_sg etname_np	1
	some fixes to project description manager and build system to allow EFS hosted projects to function better	2
	Generic implementation of red-black binary tree It's planned to use in several places	2
Negative	Oops did inadvertent branch	–2
	Bugzilla 218654 This commit shows some files contain errors This is because they are being compiled against M4 I will rebuild against M5 shortly I did diff of the files and changes are exactly what I wanted They will compile against HEAD and M5 when that is resolved	–1
	Oops Removing unneeded System.err.println foo	–1
Neutral	Build/TestToolsMove WP_UnitTestCase_BaseassertPostConditions to more appropriate place	0
	New ScannerInfoProvider extension point allowing providers to be associated with build commands in the project description	0

presents some examples of positive, negative, and neutral commits from GitHub dataset. As noted in Figure 2, all projects (Eclipse-CDT, PostgreSQL, Glibc, GNUCash, Rhino, Firebug, and WordPress) have a higher proportion of the neutral sentiment as compared to the negative and positive ones. Eclipse-CDT has the highest neutral (79.77%) sentiments, and lowest negative sentiments in comparison to other projects. PostgreSQL logs have the most negative (36.03%) sentiments. The proportion of positive sentiment is the lowest in all projects as compared to neutral and negative sentiments. WordPress logs have the highest positive (9.77%) sentiments and Glibc has least positive sentiments (1.76%). Our findings clearly indicate that the overall sentiments expressed in commit logs were neutral.

Our findings clearly indicate that majority of commits in commit logs are neutral in comparison to negative and positive. There is lowest percentage of positive commits than negative and positive ones. The main reason for high neutrality in the commits may be that commits are different from online reviews and tweets. However, a small amount of commit messages in commit logs have different types of affective states than review comments posted online. People express their satisfaction and dissatisfaction about a product by writing reviews whereas software developers write commit messages when they submit their work output in the form of code in a repository. The submission may include some code and URLs while writing commit messages without mentioning any affective involvement that makes the sentiments conveyed in commits more neutral. Moreover, commit messages include many technical terms that do not have any sentiment manifestation. Therefore, it could be another reason for the neutral sentiments in commit logs. Moreover, commit messages include many technical terms that do not have any sentiment manifestation. Therefore, it could be another reason for the neutral sentiments in commit logs.

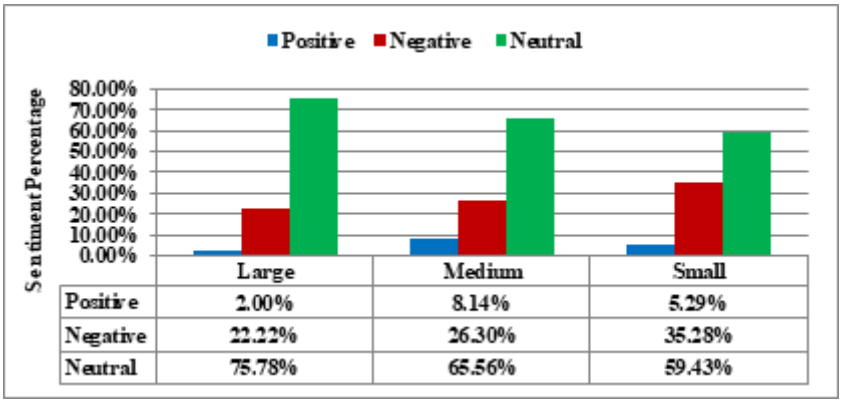
RQ2: Is there any relation between sentiments and team size of a project?

In this research question, our objective is to ascertain if size of the team in a project has any impact on sentiments expressed by developers in commit logs. We categorize the projects into large, medium, and small based on the number of contributors involved in each project (see Table 4) as recommended by Becher et al. [38]. We consider participant as developer who made at least one commit in the project. The projects having 40 to

Table 4. Project size boundaries

Parameters	Minimum developers	Maximum developers	Project Name
Small	40	60	PostgreSQL, Firebug
Medium	61	200	GNUCash, WordPress, Rhino
Large	201	∞	Glibc, Eclipse-CDT

60 developers are classified as small, projects with 61 to 200 developers as medium, and projects comprising more than 201 developers as large projects (see Table 4). Becher et al. present a study to analyze number of contributors in a random sample of projects included in the GNU/Linux distribution [38]. We followed the partition proposed by Becher et al. [38] to construct project size boundaries that are presented in Table 4 and categorize the projects into small, medium, and large based on these size boundaries [38]. The sentiment score of large, medium, and small projects is presented in Figure 3.



	Large	Medium	Small
Positive	2.00%	8.14%	5.29%
Negative	22.22%	26.30%	35.28%
Neutral	75.78%	65.56%	59.43%

Figure 3. Sentiment in projects with Large, Medium, and Small number of contributor

As shown in Figure 3, all three categories (Large, Medium, and Small) of projects have high count for neutral sentiments than negative and positive sentiments. The Projects with a large number of contributors have more neutral sentiments (75.78%) as compared to projects having medium and small number of contributors. But, we see an opposite trend in projects having medium and small number of contributors. In projects with medium team size, the percentage of positive sentiments (8.14%) expressed is higher and the percentage of negative sentiments is lower than projects with small team size. Lastly, negative sentiment is maximum in projects with small team size.

Due to the fluctuating number of team members in an OSS project over a period of time, it is worthy to relate sentiments in commit logs with the number of active developers in a smaller unit of time. For this, we identified active developers in the projects in each year of their lifetimes. To identify active developers, observation period is chosen for each project is January 2018 whereas for Firebug it is September 2016. Developers those show any activity after January 2018 is considered active. In case of Firebug developers having any activity after September 2016 considered active. Sentiments are mapped to number of active developers in each year to determine the relation of sentiment with active developers (team size). Results are presented in Figure 4. As shown in Figure 4, findings of Eclipse-CDT, PostgreSQL, Rhino and firebug indicate that neutral sentiments are high with large or small team. In case of negative sentiments large team indicate low negativity whereas small team indicate high negativity in sentiments. Results of Eclipse-CDT, PostgreSQL, Glibc,

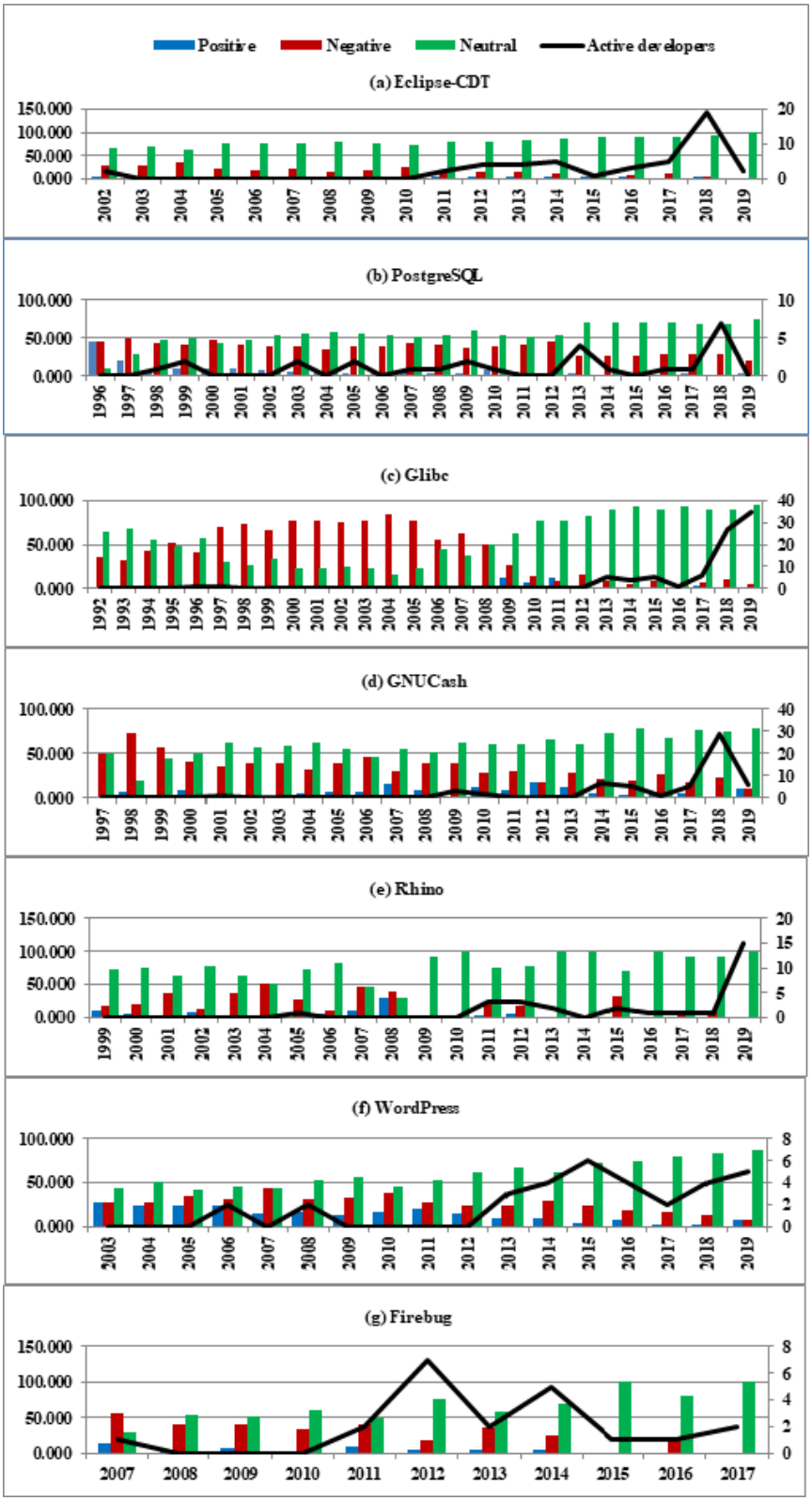


Figure 4. Sentiments and number of active contributors
(a) Eclipse-CDT, (b) PostgreSQL, (c) Glibc, (d) GNUCash, (e) Rhino, (f) WordPress, and (g) Firebug

WordPress, and Firebug indicate that positive sentiments are low with large team and high with small team. When we compared these results with Figure 3, we observed similar trend.

To confirm our results, we applied Pearson Correlation between number of active developers in each year and sentiments. The results of Pearson Correlation are presented in Table 5. In case of Glibc, Eclipse-CDT, GNUCash, and WordPress, we found strong positive correlation (>0.47) between neutral sentiments and number of active developers. In Glibc, Eclipse-CDT, and WordPress, we found strong negative correlation between number of active developers and negative sentiments. No significant correlation is found between number of active developers and positive sentiments. Only Wordpress shows significant correlation between number of active developers and positive sentiments.

Table 5. Pearson co-relation between active developers and sentiment
(* means correlation is significant at the 0.05;
** means correlation is significant at 0.01 level)

Project	Positive	Negative	Neutral
Glibc	–.117	–.446*	.482**
Eclipse-CDT	.036	–.533*	.498*
GNUCash	–.218	–.366*	.474*
WordPress	–.737**	–.634**	.769**
Rhino	–.186	–.296	.309
PostgreSQL	–.273	–.340	.349*
Firebug	.010	–.278	.228

Our findings clearly indicate that projects with large team size have more neutral sentiments. One main reason for high neutrality in the sentiments may be that developers in a large team are more formal and used many technical terms while writing commits that do not have any affective state. Moreover large teams may have laid down some formal coding guidelines. Therefore, it makes the sentiments more neutral. In small team size setup, projects have more negative sentiments in commit logs. It may be developers are less formal in a small team, or it could also be due to work pressure. There is need to look at it in the future work.

Our finding confirms that projects with different team size show different trends in the sentiments. Hence, team size of a project influences the sentiments expressed by developers in its commit logs.

RQ3: Does the type of change activity performed by a developer impact their sentiments in commit messages?

In this research question, we intended to recognize the relation between type of change activity performed by developers and sentiments expressed by them in the commit messages. There are three types of code change activities, i.e., addition, deletion, and modification [4], which can be combined in various ways to change a program. For example, some change may require adding new code along with modification of the existing lines of code. Based on these three types of file changes, we create our own classification by making the following combinations of file change types: add + modify, delete + modify, add + delete + modify. The motivation for these combinations is the evidence in the Software Engineering literature that modification of existing code is more difficult than adding new or deleting existing code. Creating new code is fun, but changing the existing one is hard.

We select five projects (PostgreSQL, Eclipse-CDT, Firebug, GNUCash, and WordPress) out of seven projects based on three types of activities performed by developers. For our analysis, we classify developers according to three types of change activities such as Type-1 (add + modify), Type-2 (delete + modify), and Type-3 (add + delete + modify) and analyze developers' sentiments based on the type of change performed by them.

The results of sentiment analysis based on three types of change activities are presented in the Figure 5. From these results, we observed that neutral sentiments have minimum occurrences for Type-3 activity. Also this is the activity which involves the most negative sentiments. Type-2 activity indicates high neutral sentiments (see results of PostgreSQL,

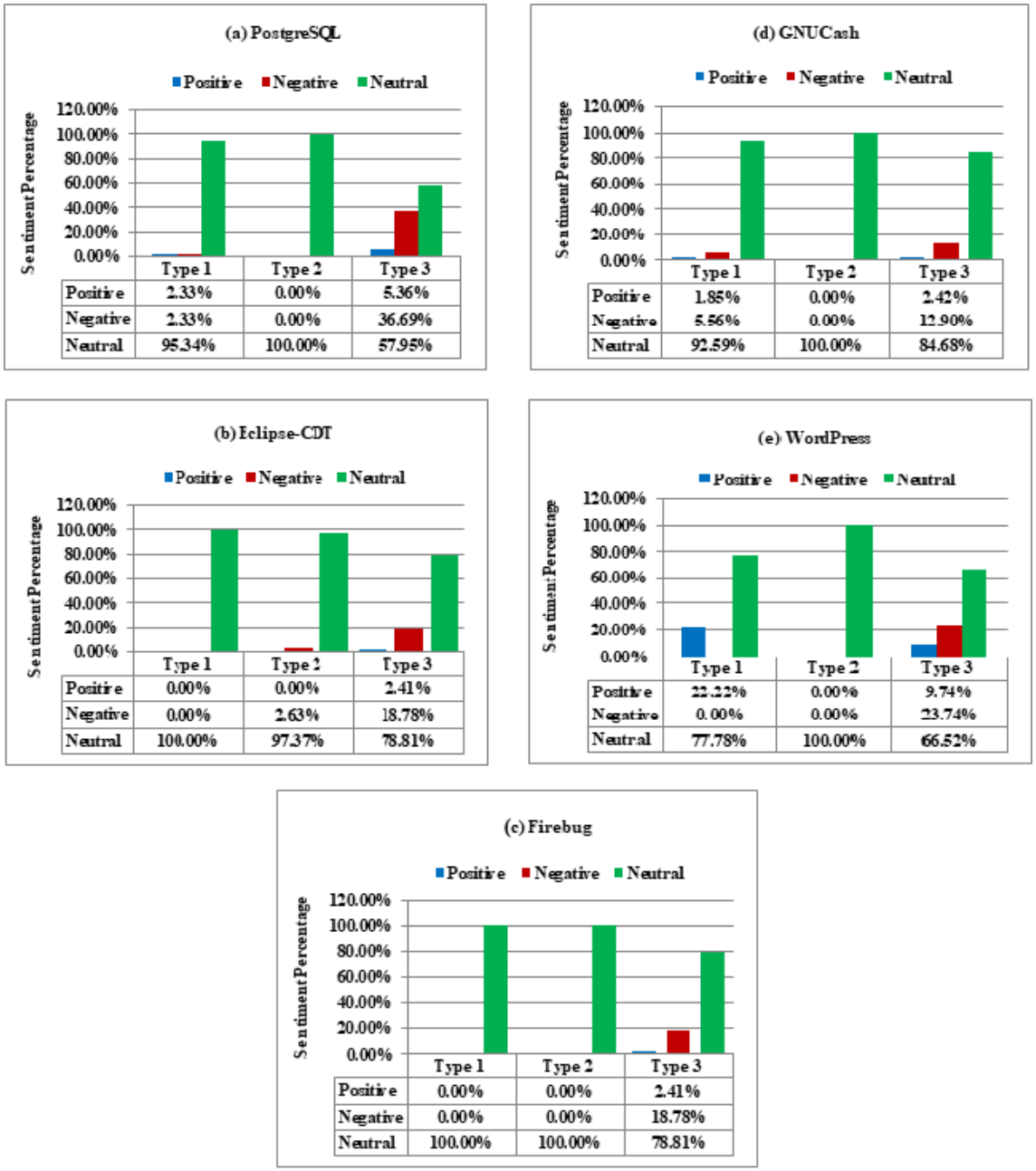


Figure 5. Sentiment and type of change activity:
(a) PostgreSQL, (b) Eclipse-CDT, (c) Firebug, (d) GNUCash, and (e) Wordpres

GNUCash, and WordPress indicated in Figure 5) in comparison to Type-1 and Type-3 activity.

To conclude, RQ3 results, the sentiments conveyed by developers in commit messages are influenced by the type of change activity performed by them. It shows that more negative and less neutral expression is put with Type-3 (add + delete + modify) activity. The reason for this could be that in Type-1 and Type-2 activities developers perform two operations in each while in Type-3 activity they perform 3 operations that means more complex work and it may make the sentiments more negative in comparison to Type-1 and Type-2. From these results, we inferred that when developers are involved in more than two activities, they express more negative expressions in the commit messages.

RQ4: Is there any relation between developer sentiment volume and commit contribution?

In this research question, we want to determine the association between sentiment volume and commit contribution. In order to achieve this goal, we analyzed developers' sentiments in commit logs and calculate the commit contribution of the top ten contributors. Commit contribution is the percentage of commits made by each individual contributor in a project. We calculate the commit contribution by dividing the total commits of each individual contributor by total number of commits made in the project. Sentiment volume, formulized in the same way as commit contribution size, is percentage of sentiments (Positive, Negative, and Neutral) conveyed by each individual developer in the commit log. We also compute the sentiment volume of each contributor by dividing individual contributor total sentiment (Positive, Negative, and Neutral) by total sentiments of the project. The formulas used for calculation of commit contribution and sentiment volume are as mentioned below:

$$Commit\ Contribution = \frac{Total\ Commits\ of\ Individual\ Contributors}{Total\ Number\ of\ Commits\ in\ the\ Project}$$

$$Sentiment\ Volume = \frac{Contributor\ Sentiment}{Total\ Sentimentin\ the\ Project}$$

$$Positive\ Sentiment\ Volume = \frac{Contributor\ Total\ Positive\ Sentiment}{Total\ Positive\ Sentiments\ of\ Project}$$

$$Negative\ Sentiment\ Volume = \frac{Contributor\ Total\ Negative\ Sentiment}{Total\ Negative\ Sentiments\ of\ Project}$$

$$Neutral\ Sentiment\ Volume = \frac{Contributor\ Total\ Neutral\ Sentiment}{Total\ Neutral\ Sentiments\ of\ Project}$$

We map sentiment volume to the commit contribution. The sentiment volume (positive, negative, and neutral) of top ten contributors along with their commit contribution is presented in Figure 6. Negative sentiment can be attributed to lead contributors in almost every project. It may be due to the project deadlines or other challenges such as quality evaluation that core contributors evoke more negative sentiment. It could also be due to the status in the team or control over the project that invites negative sentiments. For some projects such as PostgreSQL, and Firebug, neutral sentiment is clearly higher for developers with least contributions. But there is no such clear pattern for other sentiments and for

other projects. Rather, the sentiment is negative irrespective of the contribution size. So there is no trend that indicates any relation between contribution size and sentiment volume except that large contributors elicit more negative sentiment.

To conclude RQ4 results, we observed that the commit contribution of the developer influences their sentiments in the commit log. We noticed different trends in sentiments with respect to commit contribution. In most of the projects, the developers have more negativity in sentiment when their contribution is large and contributors with small commit contributions have a more positive sentiment. This implies that high commit activity causes negative sentiments in the project.

We applied Pearson Correlation to identify the correlation between commit contribution and sentiments (positive, negative, and neutral). In WordPress and firebug, we found a strong positive correlation (Pearson's correlation test above 0.70) between commit contributions and the positive sentiments whereas Eclipse-CDT has a strong negative correlation (>–0.70). The GNUcash and Rhino have a strong correlation (>0.70) and WordPress, Glibc, PostgreSQL, and Firebug have a very strong correlation (Pearson's correlation test >0.90) between commit contribution and negative sentiments. We do not find a strong correlation between commit contribution and neutral sentiments.

RQ5: How has sentiment in the commit logs evolved over the period of time?

In this research question, our aim is to analyze the evolution of sentiments across time along with the number of commits made by developers. To achieve this goal all selected projects are considered. We group the sentiments (Positive, Negative, and Neutral) by each individual year to show how sentiments change across years along with the number of commits made by developers. Figure 7 shows the evolution of sentiments along with the number of commits across the years.

There is an increase in the neutral sentiment over the period of time in all the projects, which is a good sign for technical communication. Also, negative sentiment has decreased. Positive sentiment has stayed at the bottom throughout with small variations. Looking at the commit activity along with the sentiment evolution, it is evident that there is no relation between change in commit activity and sentiment evolution. One can observe a high percentage of negative sentiment irrespective of whether commit activity is high or low (as throughout in PostgreSQL and GNUCash, or Glibc from 1994 to 2009). On the other hand, negative sentiment remains low when commit activity is high in case of Eclipse, WordPress, Firebug, and Rhino.

Positive sentiment is the least kind of expression in the commit logs. There is more or less interplay between negative and neutral sentiments in all the projects. When neutral sentiment decreases, negative sentiment replaces it. So we can say developers are either negative or neutral while expressing themselves in commit logs. It is good to see a trend of improvement in neutral sentiment over the period of time.

Moreover, we also perceive that in most of the projects (PostgreSQL, WordPress, Eclipse-CDT, Firebug, and Rhino) the sentiments seem to be more positive in the starting years as compared to the ending years of observation. The reason for it could be that when the project is in its initial stage, it is less complex, having few issues. But as the project progresses it becomes more complex, more developers join the project with time and more issues need to be resolved.

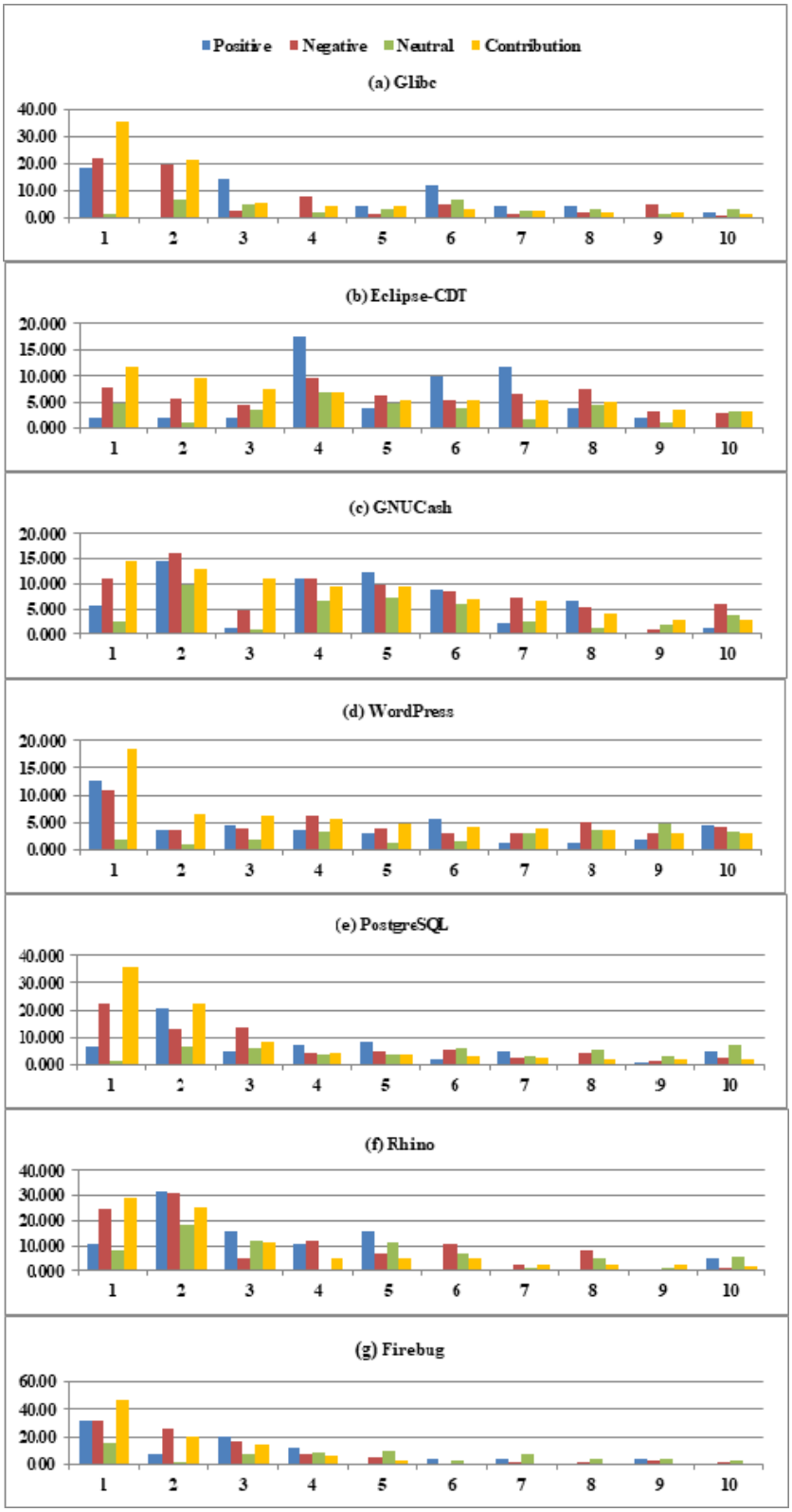


Figure 6. Commit contribution and sentiment volume: (a) Glibc, (b) Eclipse-CDT, (c) GNUcash, (d) WordPress, (e) PostgreSQL, (f) Rhino, and (g) Firebug

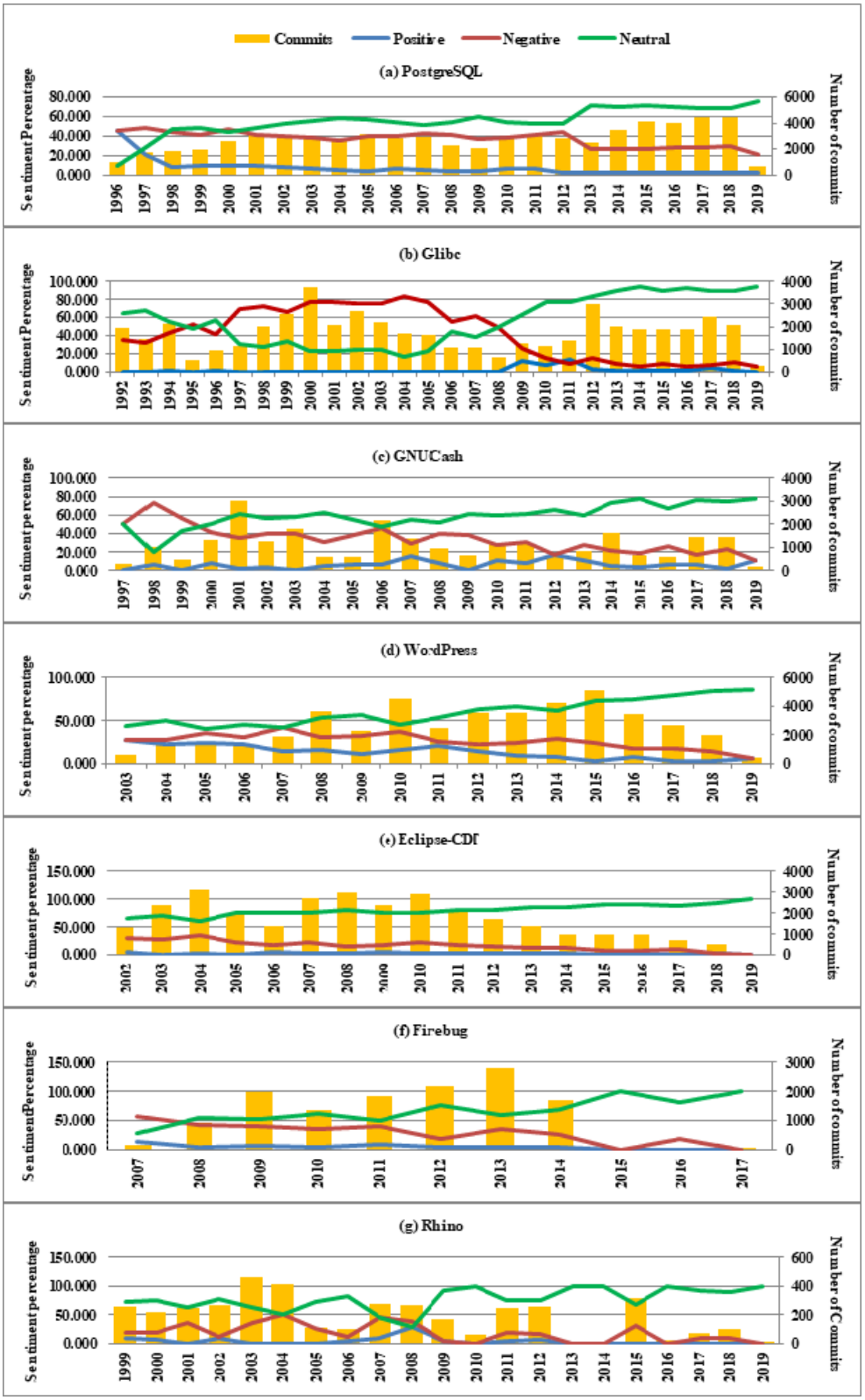


Figure 7. Evolution of sentiments by year:
(a) PostgreSQL, (b) Glibc, (c) GNUcash, (d) WordPress, (e) Eclipse-CDT, (f) Firebug, (g) Rhino

5. Discussion

In this work, we have examined 86,515 commit messages of seven well known GitHub projects to analyze the sentiments expressed by developers in the commit logs. Our main objective was to investigate the relation of team size, type of change activity, and commit contribution with sentiments in the commit logs. In addition to this, we also look into the evolution of sentiment in these projects. We found that most of the projects had high neutral sentiments in comparison to negative and positive ones. PostgreSQL indicates more negativity (36.03%) in sentiments and most of the negativity in the commit logs can be attributed to leading code contributors.

The majority of the commits in the commit logs are neutral. Our findings revealed that the team size of a project, type of change activity, and developers' commit contribution have an impact on the sentiment expressed in the commit logs. Furthermore, during the evolution of the project sentiments have different trends. We noticed that the commit logs have more positive/negative and less neutral sentiments in the initial years of the project in comparison to later years. The main reason behind it may be that in the starting years, a project is less complex and have a small number of issues but as the project progresses, more issues need to be resolved and large size of a team makes it a more formal platform and developers express themselves in a neutral way. Noticing the trend in Figure 7(b) indicates that expression in the glibc project, prior to 1990, was positive. It started getting negative after that. Age or maturity of a project does not influence developer sentiment expression in commit logs. But taking 2008 as the reference point, when Github was launched and most of the projects might have shifted to Github then, negative sentiment has decreased over the period of time. So it may be due to availability of the commit logs in the public domain, that sentiment expression has become more positive.

The study presented by Sinha [5] also examined the developers' sentiments in the commit logs. They identified that the majority of GitHub commits (74.74%) have neutral sentiments. As we compare our findings of RQ1, with results presented by Sinha [5], we noticed that our work found similar results. We observed that most of the commits in the commit log had a neutral sentiment. To compare our results with Sinha [5], we combined the sentiment results of all observed projects and found that in our analysis percentage of positive, negative, and neutral sentiments are 4.73%, 26.98%, 68.29%, respectively. In the case of our analysis positive sentiments are 2.47% and neutral sentiments are 6.45% less than Sinha's study. Negative sentiments are 8.93% higher than Sinha's study. This analysis shows that this result is very similar for a dataset different from the one studied in this research. They started with 28,466 OSS projects but considered only 5 projects for an in-depth sentiment analysis. So far detailed analysis, more work in this direction is required to confirm the findings for OSS projects of different domains and different sizes.

After this analysis, some actionable advice for the OSS community can be as follows:

- A project, large or small, should have a code of conduct mentioning the desired contribution quality in commit logs.
- In the issue tracking system, issues involving complex changes should be decomposed into multiple simple issues involving only two activities i.e. modify activity should be clubbed with either add or delete activity.
- Lead developers need to be aware of their sentiment expressions.
- Developers, looking for projects to contribute, can expect better commit logs, from sentiments point of view, in mature projects.

6. Threat to validity

The authors examined developers' sentiments in subject line of commit message but body of commit message may have different sentiments. For example, subject line may be neutral, but message body may be negative or vice versa. This aspect is missing in this study.

Same developer may have registered with multiple names. Multiple aliases related to same developers is not resolved that may influence the findings.

Moreover, the selection of the projects is biased as we included the projects having a valid Git repository while projects hosted on other platforms like Gitlab and Bitbucket are not taken into consideration. A subset of the research questions explored on a large Github dataset in [5] also gives results similar to the ones obtained here. In the future, we will extend our dataset to include more projects that are hosted on other software repositories. Furthermore, the result presented in this study only applies to OSS projects. In RQ4, the authors included the data of the top ten developers with very high commit activity while developers with very low commit (commit activity less than 1%) activities are not included in our analysis. In the future, we will extend our study to include developers with low levels of contribution.

Also combinations of file change viz. add + modify file, delete + modify file and add + delete + modify file are considered by authors to conduct analysis whereas impact of individual file change (add, delete, and modify) is not explored. Further research is required to examine the impact of individual file change like addition, deletion, and modification on sentiments.

Another limitation of our study is that we considered only a few factors to study the impact of developers' sentiments while there are many other factors such as code quality, gender, project age, and popularity that may influence sentiments expressed in the commit logs.

7. Conclusions

In this paper, the authors have analyzed the developers' sentiments in the commit logs of OSS projects. We examined 86,515 commit messages of the seven most popular OSS projects to analyze the sentiments expressed by developers in the commit logs. The authors investigated the impact of team size of the project, type of change activity (Type-1, Type-2, and Type-3) performed by developers, and code contribution volume to the sentiments expressed in the commit logs. Moreover, we analyzed the evolution of sentiments across years with respect to the number of commits made in each year.

Our study reveals that the majority of projects have neutral sentiments. This indicates that while creating commit log messages developers are more neutral. But when we compared negative with positive sentiments, we found that in case of three projects, percentage of negative sentiment is more than 10% greater than positive in all the projects, and negative sentiment is more than 23% higher than positive in four projects. In this study, we perceived that sentiments in the commit logs are influenced by team size. Neutral expressions are high with large team size and negative expressions are high with small team size. The type of change activity performed by developers also influences their sentiments expressed in the commit logs. Type-3 activity involving all the three change actions of addition, deletion, and modification, indicates more negative sentiments and low neutral sentiments. Furthermore, we also noticed that contribution size also impacts the volume of sentiment.

The developers with large commit contributions have more negativity in sentiments and developers having small commit contributions express more positive sentiments in commit logs. Besides, sentiments show different trend across years with respect to the number of commits made by developers. The developers have more positive sentiment in the initial years in comparison to the ending years. The neutral expression has increased over the period of time.

Our study results provide an understanding regarding developers' sentiments related to various software development team and project related concerns such as team size, contributor role, task complexity, and project evolution that will be helpful for OSS community in developing strategies to improve developer productivity and retention.

In the future, we intend to expand our research work by including more projects hosted on other platforms such as GitLab and Bitbucket. Large data sets and the complex interplay of various variables in this context demand to employ machine learning or deep learning techniques to identify the association.

We also want to look into why expression in small teams is negative and explore it from the perspectives of informal interactions as well as work pressure. This study can also be extended to include specific type of developers, e.g., lead or occasional, to study the difference in their sentiment expressions in the commit logs.

Acknowledgments

The research work presented in this paper is sponsored by UGC, Government of India. The authors are appreciative of UGC to provide funding under Rajiv Gandhi National Fellowship scheme to the first author. The authors are also grateful to the Department of Computer Science, Guru Nanak Dev University Amritsar, and Punjab for infrastructure and scholastic aid towards the ongoing research.

References

[1] D. Graziotin and F. Fagerholm, "Happiness and the productivity of software engineers," in *Rethinking Productivity in Software Engineering*. Springer, 2019, pp. 109–124.

[2] M. De Choudhury and S. Counts, "Understanding affect in the workplace via social media," in *Proceedings of the Conference on Computer Supported Cooperative Work*, 2013, pp. 303–316.

[3] B. Liu et al., "Sentiment analysis and subjectivity," *Handbook of Natural Language Processing*, Vol. 2, No. 2010, 2010, pp. 627–666.

[4] E. Guzman, D. Azócar, and Y. Li, "Sentiment analysis of commit comments in GitHub: An empirical study," in *Proceedings of the 11th Working Conference on Mining Software Repositories*, 2014, pp. 352–355.

[5] V. Sinha, A. Lazar, and B. Sharif, "Analyzing developer sentiment in commit logs," in *Proceedings of the 13th International Conference on Mining Software Repositories*, 2016, pp. 520–523.

[6] N. Singh and P. Singh, "How do code refactoring activities impact software developers' sentiments? – An empirical investigation into GitHub commits," in *24th Asia-Pacific Software Engineering Conference (APSEC)*. IEEE, 2017, pp. 648–653.

[7] M.R. Islam and M.F. Zibran, "Sentiment analysis of software bug related commit messages," *Network*, Vol. 740, 2018, p. 740.

[8] P. Tourani, Y. Jiang, and B. Adams, "Monitoring sentiment in open source mailing lists: exploratory study on the apache ecosystem," in *Proceedings of 24th Annual International Conference on Computer Science and Software Engineering*, Vol. 14, 2014, pp. 34–44.

[9] J. Ding, H. Sun, X. Wang, and X. Liu, "Entity-level sentiment analysis of issue comments," in *Proceedings of the 3rd International Workshop on Emotion Awareness in Software Engineering*, 2018, pp. 7–13.

[10] F. Jurado and P. Rodriguez, "Sentiment analysis in monitoring software development processes: An exploratory case study on GitHub's project issues," *Journal of Systems and Software*, Vol. 104, 2015, pp. 82–89.

[11] R. Paul, A. Bosu, and K.Z. Sultana, "Expressions of sentiments during code reviews: Male vs. female," in *26th International Conference on Software Analysis, Evolution and Reengineering (SANER)*. IEEE, 2019, pp. 26–37.

[12] D. Garcia, M.S. Zanetti, and F. Schweitzer, "The role of emotions in contributors activity: A case study on the gentoo community," in *International Conference on Cloud and Green Computing*. IEEE, 2013, pp. 410–417.

[13] M.R. Islam and M.F. Zibran, "Exploration and exploitation of developers' sentimental variations in software engineering," in *Research Anthology on Recent Trends, Tools, and Implications of Computer Programming*. IGI Global, 2021, pp. 1889–1910.

[14] A. Murgia, P. Tourani, B. Adams, and M. Ortu, "Do developers feel emotions? An exploratory analysis of emotions in software artifacts," in *Proceedings of the 11th Working Conference on Mining Software Repositories*, 2014, pp. 262–271.

[15] M.R. Islam and M.F. Zibran, "Leveraging automated sentiment analysis in software engineering," in *14th International Conference on Mining Software Repositories (MSR)*. IEEE, 2017, pp. 203–214.

[16] N. Novielli, F. Calefato, F. Lanubile, and A. Serebrenik, "Assessment of off-the-shelf SE-specific sentiment analysis tools: An extended replication study," *Empirical Software Engineering*, Vol. 26, No. 4, 2021, pp. 1–29.

[17] K. Sun, H. Gao, H. Kuang, X. Ma, G. Rong et al., "Exploiting the unique expression for improved sentiment analysis in software engineering text," *arXiv preprint arXiv:2103.13154*, 2021.

[18] E. Biswas, M.E. Karabulut, L. Pollock, and K. Vijay-Shanker, "Achieving reliable sentiment analysis in the software engineering domain using BERT," in *International Conference on Software Maintenance and Evolution (ICSME)*. IEEE, 2020, pp. 162–173.

[19] N. Novielli, F. Calefato, D. Dongiovanni, D. Girardi, and F. Lanubile, "Can we use SE-specific sentiment analysis tools in a cross-platform setting?" in *Proceedings of the 17th International Conference on Mining Software Repositories*, 2020, pp. 158–168.

[20] M.R. Wrobel, "The impact of lexicon adaptation on the emotion mining from software engineering artifacts," *IEEE Access*, Vol. 8, 2020, pp. 48 742–48 751.

[21] M. Obaidi and J. Klünder, "Development and application of sentiment analysis tools in software engineering: A systematic literature review," *Evaluation and Assessment in Software Engineering*, 2021, pp. 80–89.

[22] S.F. Huq, A.Z. Sadiq, and K. Sakib, "Is developer sentiment related to software bugs: An exploratory study on GitHub commits," in *27th International Conference on Software Analysis, Evolution and Reengineering (SANER)*. IEEE, 2020, pp. 527–531.

[23] R. Kaur and K.K. Chahal, "Analysis of developers' sentiments in commit comments," in *International Conference on Advanced Informatics for Computing Research*. Springer, 2020, pp. 3–12.

[24] S. Bharti and H. Singh, "Investigating developers' sentiments associated with software cloning practices," in *International Conference on Advanced Informatics for Computing Research*. Springer, 2018, pp. 397–406.

[25] R. Souza and B. Silva, "Sentiment analysis of Travis CI builds," in *14th International Conference on Mining Software Repositories (MSR)*. IEEE, 2017, pp. 459–462.

[26] D. Pletea, B. Vasilescu, and A. Serebrenik, "Security and emotion: sentiment analysis of security discussions on GitHub," in *Proceedings of the 11th Working Conference on Mining Software Repositories*, 2014, pp. 348–351.

[27] I.A. Khan, W.P. Brinkman, and R.M. Hierons, "Do moods affect programmers' debug performance?" *Cognition, Technology and Work*, Vol. 13, No. 4, 2011, pp. 245–258.

[28] S.C. Müller and T. Fritz, "Stuck and frustrated or in flow and happy: Sensing developers' emotions and progress," in *37th International Conference on Software Engineering*, Vol. 1. IEEE, 2015, pp. 688–699.

[29] D. Graziotin, X. Wang, and P. Abrahamsson, "Happy software developers solve problems better: Psychological measurements in empirical software engineering," *PeerJ*, Vol. 2, 2014, p. e289.

[30] M.R. Wrobel, "Emotions in the software development process," in *6th International Conference on Human System Interactions (HSI)*. IEEE, 2013, pp. 518–523.

[31] M.R. Islam and M.F. Zibran, "SentiStrength-SE: Exploiting domain specificity for improved sentiment analysis in software engineering text," *Journal of Systems and Software*, Vol. 145, 2018, pp. 125–146.

[32] S.F. Huq, A.Z. Sadiq, and K. Sakib, "Understanding the effect of developer sentiment on fix-inducing changes: an exploratory study on GitHub pull requests," in *26th Asia-Pacific Software Engineering Conference (APSEC)*. IEEE, 2019, pp. 514–521.

[33] M. Ortu, A. Murgia, G. Destefanis, P. Tourani, R. Tonelli et al., "The emotional side of software developers in JIRA," in *13th Working Conference on Mining Software Repositories (MSR)*. IEEE, 2016, pp. 480–483.

[34] M.M. Rahman, C.K. Roy, and I. Keivanloo, "Recommending insightful comments for source code using crowdsourced knowledge," in *15th International Working Conference on Source Code Analysis and Manipulation (SCAM)*. IEEE, 2015, pp. 81–90.

[35] R. Jongeling, P. Sarkar, S. Datta, and A. Serebrenik, "On negative results when using sentiment analysis tools for software engineering research," *Empirical Software Engineering*, Vol. 22, No. 5, 2017, pp. 2543–2584.

[36] F. Calefato, F. Lanubile, F. Maiorano, and N. Novielli, "Sentiment polarity detection for software development," *Empirical Software Engineering*, Vol. 23, No. 3, 2018, pp. 1352–1382.

[37] R. Jongeling, S. Datta, and A. Serebrenik, "Choosing your weapons: On sentiment analysis tools for software engineering research," in *International Conference on Software Maintenance and Evolution (ICSME)*. IEEE, 2015, pp. 531–535.

[38] K. Beecher, C. Boldyreff, A. Capiluppi, and S. Rank, "Evolutionary success of open source software: An investigation into exogenous drivers," *Electronic Communications of the EASST*, 2008.

e-Informatica Software Engineering Journal, Volume 16, Issue 1, 2022, pages: 220103, DOI: 10.37190/e-Inf220103

How Good Are My Search Strings? Reflections on Using an Existing Review As a Quasi-Gold Standard

Huynh Khanh Vi Tran*, Jürgen Börstler*, Nauman bin Ali*, Michael Unterkalmsteiner*

*Department of Software Engineering, SE-37179, Karlskrona, Sweden, Blekinge Institute of Technology

huynh.khanh.vi.tran@bth.se, jurgen.borstler@bth.se, nauman.ali@bth.se, michael.unterkalmsteiner@bth.se

Abstract

Background: Systematic literature studies (SLS) have become a core research methodology in Evidence-based Software Engineering (EBSE). Search completeness, i.e., finding all relevant papers on the topic of interest, has been recognized as one of the most commonly discussed validity issues of SLSs.
Aim: This study aims at raising awareness on the issues related to search string construction and on search validation using a quasi-gold standard (QGS). Furthermore, we aim at providing guidelines for search string validation.
Method: We use a recently completed tertiary study as a case and complement our findings with the observations from other researchers studying and advancing EBSE.
Results: We found that the issue of assessing QGS quality has not seen much attention in the literature, and the validation of automated searches in SLSs could be improved. Hence, we propose to extend the current search validation approach by the additional analysis step of the automated search validation results and provide recommendations for the QGS construction.
Conclusion: In this paper, we report on new issues which could affect search completeness in SLSs. Furthermore, the proposed guideline and recommendations could help researchers implement a more reliable search strategy in their SLSs.

Keywords: search string construction, automated search validation, quasi-gold standard, systematic literature review, systematic mapping study

1. Introduction

Systematic literature studies (SLS), including systematic literature reviews, systematic mapping studies and tertiary studies, have become core methods for identifying, assessing, and aggregating research on a topic of interest [1]. The need for completeness of search is evident from the quality assessment tools for SLS with questions like: "was the search adequate?", "did the review authors use a comprehensive literature search strategy?" or "is the literature search likely to have covered all relevant studies?" [2–4]. Several guidelines and recommendations have been proposed to improve the coverage of search strategies employed in SLS, e.g., using multiple databases [1], or using an evaluation checklist for

Submitted: 12 Sep. 2021; Revised: 20 Nov. 2021; Accepted: 07 Dec. 2021; Available online: 15 Dec. 2021

assessing the reliability of an automated search strategy [3]. While these guidelines and assessment checklists can be used to design a search string with a higher likelihood of good coverage, these are mostly subjective assessments.

During the design phase of an SLS, the main instrument researchers have for assessing the likely coverage of their search strings is using a known set of relevant papers that a keyword-based search ought to find [5, 6]. Such a set of known relevant papers is referred to as the quasi-gold standard (QGS) for an SLS. Thus, a QGS is a subset of a hypothetical gold standard, the complete set of all relevant papers on the topic.

Ali and Usman [3] suggest the following for identifying a known set of relevant papers: a) the researchers' background knowledge and awareness of the topic, b) general reading about the topic, c) papers included in related secondary studies, d) using a manual search of selected publication venues. Kitchenham et al. [1] suggest guidelines regarding the size of a QGS for a typical systematic review or a mapping study. The quality of QGS, as a representative sample of the actual population, is critical for deciding how good is a search string. Nevertheless, the QGS size alone is not sufficient for assessing the QGS quality. The diversity of studies in a QGS is also an important quality criterion as it increases the likelihood of being a representative subset of actual related papers. However, to the best of our knowledge, we have not found any related work on validating QGS quality or specific issues relating to using an existing SLS as a source for a QGS.

In a recent tertiary study [7] on test artifact quality, as suggested by Kitchenham et al. [1], we constructed a QGS by collecting relevant papers from an earlier tertiary study with a related broader topic [8] (software testing). Our assumption was that a tertiary review of software testing research, in general, would also cover secondary studies on the relatively narrower topic of test artifact quality.

While validating the search in this tertiary study, we have identified issues with the subject area filter in Scopus, the usage of the generic search term "software" as a limiting keyword in search, and issues with the search validation approach using a QGS. Based on our experience from constructing and validating search strings using a QGS, we have derived recommendations on validating automated search and constructing the QGS. Together with the existing guidelines in the literature for the search process, our recommendations help researchers construct a more reliable search strategy in an SLS.

The remainder of the paper is structured as follows: Section 2 provides an overview of guidelines for search validation. Section 3 presents the related work and our contribution. Section 4 summarizes the search process and search validation in our tertiary study [7]. Section 5 presents our findings when comparing the search results between the two tertiary studies [7, 8]. Section 6 details the found issues related to search string construction and search validation using QGS. Section 7 presents our proposed guidelines for validating the automated search and constructing the QGS for researchers undertaking large scale SLSs. Lastly, Section 8 concludes the paper.

2. Guidelines for search validation

Several guidelines exist for implementing SLSs with instructions on how to perform the search process [1, 2, 9]. Kitchenham et al. [1] provided detailed instructions on each step of a systematic review procedure. In particular, regarding the study search process, Kitchenham et al. [1] discussed the search completeness concept and different strategies to validate search results. Accordingly, a search strategy should aim to achieve an acceptable level of

search completeness while considering the time constraint and limit in human resources. Ultimately, the level of completeness depends on the type of the review (qualitative or quantitative) [1]. The completeness could be assessed subjectively based on expert opinion or objectively based on precision and recall [5, 6]. The recall of a search string, also called sensitivity, is the proportion of all the relevant papers found by the search. The precision is the proportion of the papers found by the search which are relevant to the study. By calculating the precision of a search, researchers could estimate the effort required to analyze the search result.

To compute recall and precision, ideally, researchers need to know the number of all relevant papers on the review topic, which is also called the gold standard. However, it is not easy to acquire the gold standard [1, 5], especially when the review domain is not limited. Hence, a quasi-gold standard, a subset of the gold standard, could be used instead. There are several approaches listed by Kitchenham et al. [1] to acquire a quasi-gold standard. They include asking experts in the review topic, using a literature review in the same or overlapping topic, conducting an informal automated search, or performing a manual search in specific publication venues within a certain period. Proposed by Zhang et al. [5], the last approach is claimed to be more objective and systematic in assessing automated search results than building the quasi-gold standard based solely on researchers' knowledge. In general, Zhang et al. proposed search strategy could be summarized as follows:

1. Identify publication venues (conferences, journals), databases and search engines. The venues are for manual search to establish a quasi-gold standard. The databases and search engines are for the automated search for relevant papers to answer the research question(s). It is worth noting that the selection of venues is still based on the researchers' domain knowledge; hence, this approach could potentially introduce as much bias as the approach of building a QGS by asking domain experts.
2. Establish the QGS. The QGS is built by conducting a manual search on the selected publication venues. All papers published in the given venues within a predefined time frame should be assessed based on the defined inclusion/exclusion criteria.
3. Construct search strings for the automated search. There are two ways to construct the search strings: (1) based on researchers' domain knowledge and experience; (2) based on word/phrase frequency analysis of the papers in the QGS.
4. Conduct automated search. The automated search is conducted using the search strings on the selected search engines/databases identified in the previous steps.
5. Evaluate search performance. The search performance is evaluated based on two criteria, quasi-sensitivity (recall) and precision. Depending on the predefined threshold (70%–80% as suggested by Zhang et al.), the search result could be either accepted and merged with the QGS or search strings should be revised until the automated search performance reaches the threshold.

3. Related work

Besides the general guidelines for the search process and search validation described in Section 2, various issues related to search strategies that could affect the search completeness have been discussed in the literature [6, 10–13]. We organized the reported issues into three groups.

The most common issue is the inadequacy of a search strategy in finding relevant publications [6, 10–13], which directly affects the search completeness. Ampatzoglou et

al. [10, 11] discussed the issue via one of their proposed validity categories, namely *study selection validity*. In this category, the threat "adequacy of relevant publication" [10, 11], which the authors quote, is about "has your search process adequately identified all relevant primary studies?". The authors did not provide further explanation or description of this threat. Still, they presented a list of mitigation actions such as conducting snowball sampling, conducting pilot searches, selecting known venues, comparing to gold standards. Based on these mitigation actions, we could see that this validity threat is about whether a search process has identified a representative set of relevant studies. It is noteworthy that our tertiary study [7] has applied all their proposed mitigation actions related to this threat except having an external expert review our search process. Bailey et al. [12] conducted three searches on three different topics to analyze the overlaps between search engines in the domain of software engineering. They reported that the selection of search engines and search terms could influence the number of found papers. One relevant finding is that for the topic *Software Design Patterns*, their general search terms ("software patterns empirical" and "software design patterns study") offered the highest recall, especially in Google Scholar. It is worth noting that they define the recall as a percentage of included papers found by a search engine out of the total number of included papers. To cope with the adequacy of relevant publication in the domain of software engineering experiment, Dieste et al. [6] discussed the trade-off between high recall and high precision in search. They proposed criteria for selecting databases and also reported lessons learned when building search terms. They also noted that using any synonyms of *experiment* alone would omit a huge set of relevant papers when searching articles reporting software engineering experiments. Imitiaz et al. [13], in their tertiary study, discussed different issues which could affect the adequacy of relevant publication in SLRs. These issues are search terms with many synonyms and unknown alternatives, the trade-off between generic and specific search string, search approaches (automated, manual, snowball sampling) selection, search level (title, abstract, keywords) and abstract quality.

The second most common issues which could impact the search completeness are inconsistencies and limitations of search engines and databases [3, 12, 14, 15]. Bailey et al. [12] identified two main issues with search engines: inconsistent user interfaces and limitations of search result display. They concluded that search engines do not provide good support for conducting SLRs due to these two issues. The inconsistencies in databases and search engines' internal search procedures and their output are also reported by Ali and Usman [3] and by Krüger et al. [14]. As reported in Krüger et al.'s study [14], API search results in databases could vary even within the same day. On top of that, databases and search engines evolve over time, which could lead to changes in their search API [3, 14]. Due to the identified limitations, the selection of search engines and databases becomes essential as it could impact search completeness. Chen et al. [15] proposed three metrics (overall contribution, overlap, exclusive contribution) to characterize search engines and databases which they called electronic data sources (EDS). These metrics could help researchers to choose EDS for their literature reviews. According to the authors, the *overall contribution*, which is about the percentage of relevant papers returned by an EDS, is the dominant factor in selecting EDS. Meanwhile, the *exclusive contribution* is about papers that could be found by one EDS only. This information helps researchers to decide which EDS could be omitted. The *overlap metric* (the papers returned by multiples EDS) could be used to determine the order of EDS in the search process.

The third most common issue is search terms standardization in software engineering [12, 16]. Bailey et al. [12] pointed out that there is a lack of standardization of terms used in

software engineering, which could influence the search result adequacy. They raised the need to have up-to-date keywords and synonyms to mitigate the risk of missing relevant papers. This standardization issue has also been reported by Zhou et al. [16] as one of the main validity threats in SLRs.

In summary, we have found several studies that reported issues with the search process and the importance of adequate search string construction and validation to achieve search completeness. In a tertiary study [7], we have encountered all of these issues and applied different strategies to mitigate validity threats related to the search process. These include systematically constructing search strings, piloting searches, selecting well-known digital search engines and databases, and using a relevant tertiary study's search results to build a QGS for search validation. Nevertheless, we have not identified any related work discussing the quality assessment of QGSs or issues related to the construction of QGSs from existing SLSs. Hence, based on our experience with evaluating the searches using the QGS, we propose guidelines for automated search and QGS validation, which could help researchers construct a more reliable search strategy in SLSs.

4. Analysis of using another SLS as QGS

This study is based on two recent tertiary studies [7, 8] conducted independently. Both articles were published in the Journal of Information and Software Technology. The first study [8] on software testing was undertaken by Garousi and Mäntylä, while the second one [7] with a narrower topic, test artifact quality, was published five years later by the authors of this study. A high-level overview of both tertiary studies can be found in Table 1.

Table 1. Overview of Tran et al.'s [7] and Garousi and Mäntylä's [8] tertiary studies

	Tran et al. [7] *(TAQ study)*	Garousi and Mäntylä [8] *(ST study)*
Title	Assessing test artifact quality – A tertiary study	systematic literature review of literature reviews in software testing
Publication year	2021	2016
Focus	Quality attributes of test artifacts and their measurement.	Mapping of research in software testing.
Research goals	To investigate how test artifact quality has been reported in secondary studies in the following aspects: (1) quality attributes of test artifacts; (2) quality measurements of test artifacts; (3) testing-specific context where the quality attributes and quality measurements have been studied.	To provide a systematic mapping of secondary studies in software engineering to investigate the following aspects: (1) different areas in software testing; (2) research questions types; (3) demographics of secondary studies; (4) citation of secondary studies.
Automated search	Yes	Yes
Snowballing	No	Yes
Search String	Iterative, see Figure 2 for details	See Figure 2 for details
Include SLRs & SMS	Yes	Yes
Include other reviews/surveys	No	Yes

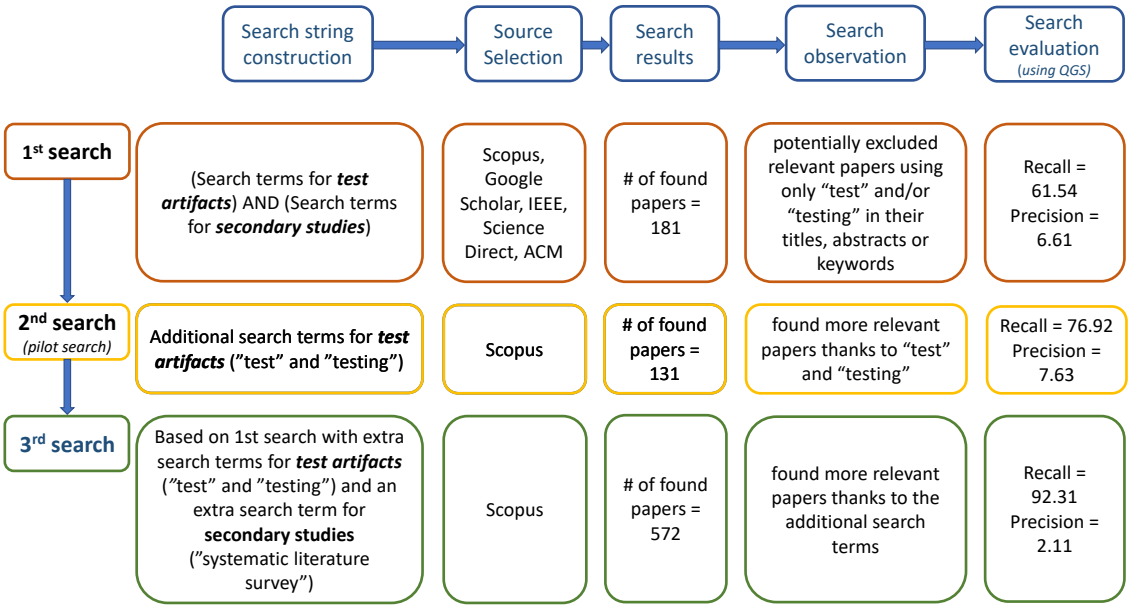


Figure 1. Overview of the search steps in the tertiary study on test artifact quality (TAQ study) [7]

For convenience, we refer to the tertiary study [8] on software testing as the ST study and the tertiary study on test artifact quality [7] as the TAQ study in this paper.

In the TAQ study [7], to evaluate the search performance, we constructed a QGS by extracting relevant papers from the ST study [8]. A summary of the resulting search strategy and search evaluation outcomes is illustrated in Figure 1. More details about the search process and the search evaluation using QGS are presented in Section 4.1 and Section 4.2 respectively. To better understand the result of the search performance evaluation, we also analyzed the differences in search results between the two tertiary studies. The analysis of these differences is described in Section 5.

4.1. Search process

In the TAQ study, test artifact refers to test case, test suite, test script, test code, test specification, and natural language test. The overview of the study's three searches is illustrated in Figure 1, and the search results are presented in Table 2. We used a visual analysis tool [17] called InteractiVenn[1] to analyze the overlaps in the search results. The TAQ study's search terms and their differences with the ST study's search terms are shown in Figure 2.

Since the TAQ study's search goal was to identify systematic secondary studies discussing test artifact quality, the search strings needed to capture two aspects: (A) systematic secondary studies and (B) test artifact quality. Hence, the search strings were constructed as (A AND B). To address aspect B (test artifact quality), we included search terms to describe test artifact such as "test case", "test script" while excluding the search term "quality" as this latter search term is too common to be useful as a separate component of a search string.

The first search was conducted in April 2019 and returned 181 papers (see Table 2). The initial set of 58 SLRs/SMSs found by the ST study was used to validate the completeness

[1]http://www.interactivenn.net/

Table 2. Search Results of the tertiary study on test artifact quality (TAQ study) [7]

Search	Database/ Search Engine	# of papers	Search Level
1st	Scopus	100	Title, abstract, keywords
	Google Scholar	27	Title
	IEEE	16	Title, abstract, keywords
	Science Direct	23	Title, abstract, keywords
	ACM	15	Title, abstract
	Total	181	
	Excl. duplicates	121	
	Excl. duplicates & clearly irrelevant studies	82	
2nd	Scopus	131	Title
	Excl. duplicates	131	
	Excl. duplicates & clearly irrelevant studies	114	
3rd	Scopus	572	Title, abstract, keywords
	Excl. duplicates	569	
	Excl. duplicates & clearly irrelevant studies	340	

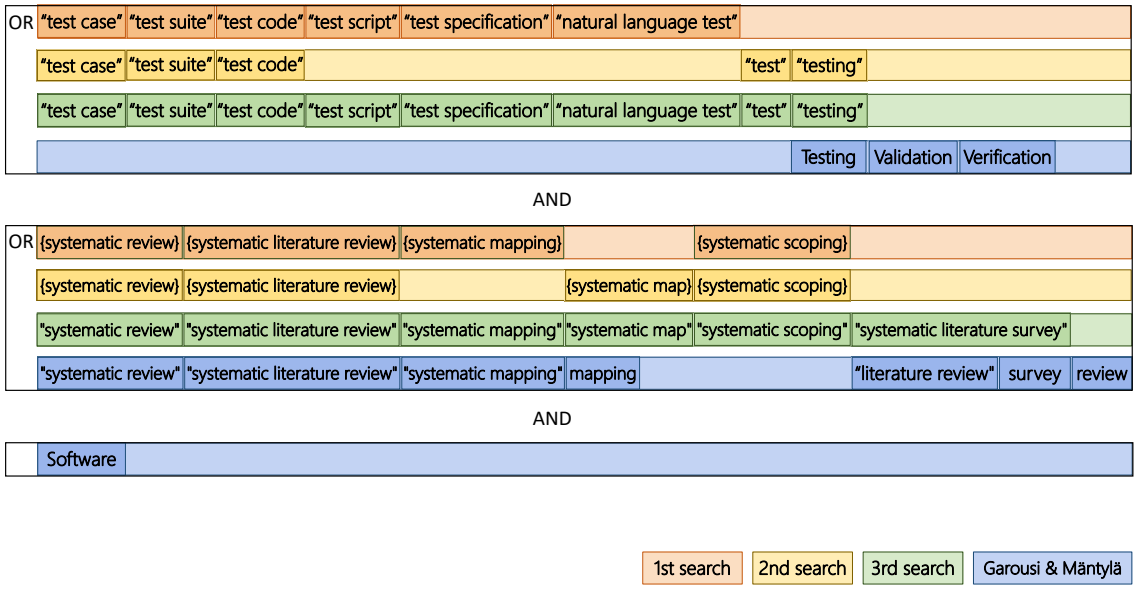


Figure 2. Comparison of the search terms used in the search strings of the two tertiary studies, the TAQ study [7] and the ST study [8]

of the searches (explanation on how these 58 papers were collected is in Section 4.2). Hence, to verify if the first search was adequate, we screened the titles and abstracts of the 39 SLRs/SMSs, which were not found by the first search but by the ST study only.

Among the 39 SLRs/SMSs, several are on different topics such as software product line testing, testing of web services, mutation testing techniques, etc. These papers used "test" and "testing" but no term for test artifact in titles and abstracts. Since these papers could potentially discuss test artifact quality but were not found by the first search, we considered it as a potential issue of the first search. In other words, the first search might

exclude relevant papers having "test" or "testing" but no term for test artifact in their titles, abstracts or keywords.

To verify the above hypothesis, we conducted a second search, which is a pilot search in Scopus in October 2019, including the additional search terms "test" and "testing". As a result, the second search returned 131 papers (see Table 2), which contained more relevant papers than the first search. Hence, we added the additional search terms "test" and "testing" in the third search to reduce the risk of missing relevant papers. Also, the third search included another search term, "systematic literature survey", which was inspired by the ST study's search terms. In other words, the third search was built based on the first search and the confirmed hypothesis from the second search (pilot search). The third search was conducted in Scopus in October 2019 and restricted to one subject area, "Computer Science", to reduce the search noise. The third search returned 572 papers, as shown in Table 2.

The overlaps between the search results are presented in Figure 3. All the numbers in the figure refer to papers after deletion of duplicates and obviously irrelevant papers, i.e., papers that are not about software engineering or computer science based on their titles, abstracts and keywords. The red box shows the distribution of 48 out of the complete set of 49 selected papers among the searches. One of the 49 selected papers was extracted from the ST study' search result (the decision on selecting papers from the ST study' search result is explained in Section 4.2); hence, it is not shown in the figure.

As shown in Figure 3, out of the 82 papers returned by the first search, 8 $(1 + 7)$ papers were included in the QGS, and 26 $(3 + 7 + 16)$ eventually turned out relevant. By considering the first search and the third search only (since the second search result is a subset of the third search result), the third search returned 276 $(8 + 14 + 55 + 199)$ additional papers, of which a further 4 $(1 + 3)$ were included in the QGS, and a further 22 $(8 + 14)$ turned out as relevant. Based on the above observation, we could see that most of the QGS papers were found by the first and third search (in total, 12 out of 13 QGS papers). It also turned out that we almost doubled the number of relevant papers with the third search. Therefore, we consider including the first and third search as a fair trade-off for this study in terms of the effort required to read papers and the returned benefit (identified relevant papers plus QGS papers). Nevertheless, the trade-off between

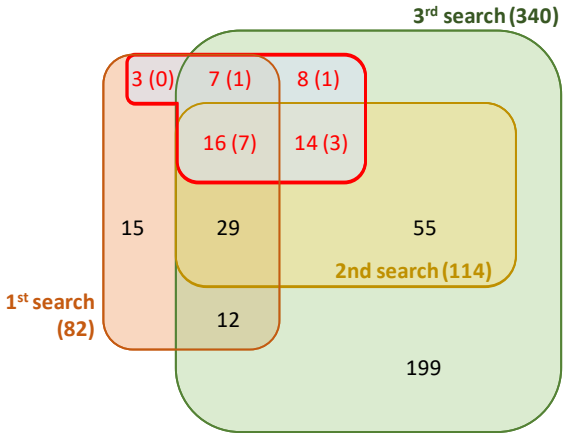


Figure 3. Overlaps between three searches in the tertiary study on test artifact quality (TAQ study) [7]. The red box illustrates the distribution of the selected papers among searches, and the numbers in parentheses show the number of papers belonging to the QGS

Table 3. Recall and Precision of searches in the tertiary study on test artifact quality (TAQ study) [7]

Considering all 58 SLRs and SMSs from the ST study's initial set as the QGS		
	1st search	3rd search
Recall	32.76	75.86
Precision	15.70	7.73
Considering only the 13 relevant SLRs and SMSs from the ST study's initial set as the QGS (see also the last column in Figure 1)		
	1st search	3rd search
Recall	61.54	92.31
Precision	6.61	2.11
Considering the 20 relevant SLRs and SMSs found by the 1st search but not by the ST study as the QGS		
	First-step forward snowballing	
Recall	50.00	
Precision	1.20	

recall and precision could be different depending on the goal of the targeting SLS. For example, if researchers aim to compare different techniques in software engineering, a high recall might be more desired than a high precision [1].

4.2. Search performance evaluation using a QGS

In this section, we describe how a QGS was constructed in the TAQ study. We then explain how the recall and precision of the first and third searches in this tertiary study were computed based on the QGS. In this evaluation process, we focused on the first search and third search only as the second search was actually a pilot search, and its result is a subset of the third search's (more details in Section 4.1).

It is worth emphasizing that we did not follow the instructions for constructing the QGS given by Zhang et al. [5] (more details on their instructions could be found in Section 2). Overall, the key difference is that we extracted relevant papers from the ST study [8] to build the QGS, while Zhang et al. suggested constructing a QGS by conducting a manual search in some publication venues with a specific time span. Our decision on how to construct the QGS is motivated by the fact that the ST study is a peer-reviewed tertiary study conducted by the domain experts and its topic (software testing) is related to and broader than the TAQ study's topic (test artifact quality). Using another literature review to collect known relevant papers for search validation is also one of the suggestions by Kitchenham et al. [1].

It is also necessary to mention that, although there is no information regarding the complete set of found papers, the ST study has provided access to its initial set of 123 papers which is the result of the ST study's authors removing clearly irrelevant papers from their search result [8]. By analyzing the 123 papers, we found two duplicate papers (having the same title, authors and abstract). Of the remaining 121 papers, 63 are *informal/regular* surveys, i.e., reviews without research questions as stated in the ST study. Hence, we focused on the remaining 58 $(121 - 63)$ papers, which are SLRs/SMSs as the TAQ study considered systematic reviews and mappings only.

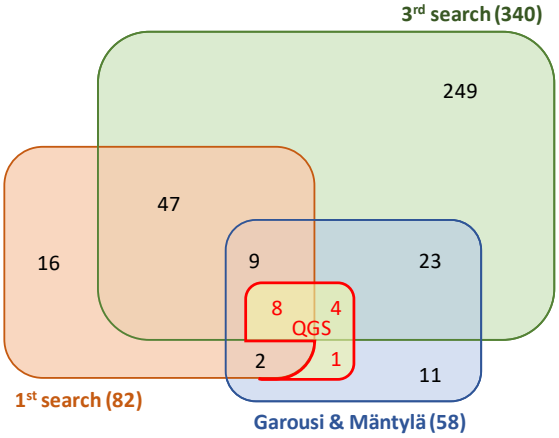


Figure 4. Overlaps between the first and third searches and the 58 SLRs/SMSs papers from the initial set of papers in the ST study [8]. The red box illustrates the distribution of the papers of the QGS

When considering all the 58 SLRs/SMSs papers from the initial set of papers in the ST study [8] as the QGS, the first and third searches found 18 and 44 papers from the QGS, respectively. The recall and precision of the two searches are relatively low, as shown in Table 3. Since these 58 papers might contain irrelevant papers to the scope of the TAQ study, we updated the QGS with the 13 relevant papers from the set of 58 SLRs/SMSs papers. The 13 papers were selected according to the TAQ study's study selection criteria (explained in Appendix A).

The distribution of the updated QGS over the first and third searches is shown in Figure 4. We need to note that all numbers in Figure 4 refer to papers after deletion of duplicates, obviously irrelevant papers and informal reviews. On the one hand, the two searches' precision decreased as the number of QGS papers found by the searches decreased (from 18 and 44 to 8 and 12 papers by the first and third search respectively). On the other hand, with this more accurate QGS, the recall of the two searches increased by a significant margin. Also, as shown in Table 3, even though the third search returns a higher reading load than the first search, it is still superior to the first one in terms of identifying relevant papers.

We considered two directions at this point: (1) select relevant papers from the first and third search for data extraction; or (2) do forward snowball sampling on the 13 relevant papers found by the ST study then select relevant papers from there. To pick an appropriate direction, we first conducted a first-step forward snowball sampling in Scopus on the 13 papers and calculated its recall and precision using the relevant papers found by the first search only as the QGS. We found 946 papers citing the 13 papers. The set reduced to 832 papers after removing duplicates (same title, abstracts, and authors). This set of 832 papers includes the ST study itself. Among these 832 papers, 10 of them met the TAQ study's study selection criteria (explained in Appendix A). Since the 13 papers were published between 2009 and 2015, our assumption was that forward snowball sampling on these 13 papers should help us identify relevant papers published from 2009 onward. Hence, we selected the 20 relevant papers published from 2009 found by the first search but not by the ST study as the QGS. As shown in Table 3, the recall and precision of the forward snowball sample were much lower than the ones of the third search. We might have found more relevant papers and improved the recall if conducting a more extended snowball

sampling on the 13 papers. However, considering the low possibility of getting a higher recall than the third search and yet much more effort required for the more extended snowball sampling, we decided to use the results of the first and third search and the initial set of 58 SLRs/SMSs papers from the ST study for the paper selection.

5. Findings

While evaluating the performance of the first and third searches in the TAQ study (described in Section 4.2), we also analyzed the differences in search results between the evaluated searches and the ST study's search. The purpose of the search result comparison is to understand better why the searches in the TAQ study achieve certain recall and precision and if these searches have any issues that we could fix or mitigate to improve their recall and precision. In this section, we report our findings from this search results comparison. The overlaps in search results between the two tertiary studies are shown in Figure 4.

Regarding the ST study's search result, there are two things we need to remark. First, in this search result comparison, the ST study's search result refers to its initial set of 58 SLRs/SMSs. These 58 papers do not contain informal/regular surveys, duplicate or clearly irrelevant papers to their study's topic (software testing) (more details on how these 58 papers were collected are in Section 4.2). Hence, before comparing the search results, we also removed duplicate and clearly irrelevant papers found in the first and third searches. As a result, there were 82 and 340 remaining papers, respectively, from the first and third search. Second, there is no information regarding when the ST study concluded its search. As the latest publication date of the papers found by the ST study's search is October 2015, we assume that the search found papers published until October 2015.

5.1. The first search and the ST study's search

As shown in Figure 4, the first search found 63 $(16 + 47)$ papers not included in the ST study's search result. Among those 63 papers, 26 papers were published before October 2015, which are within their assumed search period. The first possible explanation is that the first search included five search engines and databases (see Table 2), while the ST study searched on Scopus and Google Scholar. Indeed, six out of those 26 papers are from ACM and Science Direct. Second, the first search did not include the search term "software", which was mandatory in the ST study's search. Due to this difference in the search string construction, out of the 26 papers, the first search found 11 more papers. One interesting note is that the remaining nine papers $(26 - 6 - 11)$ could be found when applying the ST study's search string on Scopus and Google Scholar. It is possible that those papers were not indexed by Scopus or Google Scholar by the time the ST study's search was conducted.

The ST study found 39 papers $(4 + 23 + 1 + 11)$ (as shown in Figure 4) that were not included in the first search's result. Among these 39 papers, 33 of them did not have any terms for *test artifact* in title, abstract and keywords which is required by the first search. The remaining six papers $(39 - 33)$ did not use the term "systematic" in title, abstract and keywords; hence, they were also excluded by the first search, which only looked for systematic reviews.

5.2. The third search and the ST study's search

As shown in Figure 4, the third search found 296 $(249 + 47)$ papers that were not in the ST study's search result. Among these 296 papers, the first search found 47 of them. The possible reasons for the ST study's search result not containing those 47 papers are explained in Section 5.1. For the remaining 249 $(297 - 47)$ papers, 84 were published before October 2015, which meets their assumed search period. Out of these 84 papers, 31 did not use the term *software* in their titles, abstracts or keywords, which is one of the required search terms of the ST study. However, the other 53 papers $(84 - 31)$ meet the ST study's search string. We suspect that Scopus did not index these 53 papers by the time the ST study conducted its search.

The ST study found 14 papers $(2 + 1 + 11)$ (as shown in Figure 4) which the third search's result did not include. Six out of the 14 papers were not peer-reviewed; hence, they are out of the scope of this comparison. Among the other eight papers $(14 - 6)$ which were peer-reviewed, three of them did not use "systematic" in their titles, abstracts or keywords, and two of them [18, 19] were included under the subject area "Engineering" in Scopus. The third search did not find these five papers as the search accepted only systematic reviews and was limited to the subject area "Computer Science" in Scopus. The other three papers $(8 - 3 - 2)$ are not indexed in Scopus but other search engines/databases (Google Scholar, INSPEC, ACM), and two of them were found by the first search, which included those databases and search engines. We discuss the differences between the two searches next.

5.3. The first search and the third search

The first search found 18 papers $(16 + 2$ as shown in Figure 4) which the third search's result did not contain. Among those 18 papers, five of them were not categorized under the subject area "Computer Science" but different subject areas (three papers [20–22] under "Engineering"; one paper [23] under "Business, Management and Accounting/Decision Sciences/Social Sciences"; and one paper [24] under "Multidisciplinary"). The other 13 papers $(18 - 5)$ were found in other databases/search engines by the first search (six papers in Google Scholar, four papers in ACM, one paper each in IEEE, Wiley, and Web of Science). Hence, the main reasons are the databases/search engines selection and the subject area(s) selection in Scopus.

The third search found 276 papers $(249 + 23 + 4)$ (as shown in Figure 4) which the first search missed. It could be due to the more inclusive search strategy of the third search as it had extra search terms ("test", "testing", and "systematic literature survey", as shown in Figure 2).

6. Discussion

In this section, we first discuss issues relating to search string construction, then issues relating to using a QGS for search evaluation that we have discovered while evaluating the searches' performance in the TAQ study [7].

6.1. Issues in search string construction

Based on our findings described in Section 5, we identified the following issues with search string construction in SLSs.

The first issue is about using generic search terms in SLSs. Based on the differences in search results between the TAQ study [7] and the ST study [8], we found that adding generic terms (*software* in the case of the TAQ study) with the Boolean operator **AND** to a search string increases the risk of missing relevant papers. The problem is that in research areas where certain contexts are assumed, some keywords might not be explicitly stated since they are implied. It is the term *software* in the case of research in software development/quality/engineering. Hence, "AND software" just narrows down the search result as not all papers in software engineering use the term *software* in title-abstract-keywords. This also supports our decision of not including "AND quality" to the search strings. Oppositely, if generic terms are added to search strings with the Boolean operator **OR**, researchers likely have more noise in their search results. We, therefore, regard "AND software" and "AND quality" as unnecessary excluders due to their threat of excluding relevant papers, while we consider "OR software" an unnecessary includer due to its risk of retrieving non-relevant material.

The second issue we have identified is about search filters in Scopus. Search filters can be applied to various meta-data of a publication, such as language, document type, publication year, etc. By using search filters, researchers can limit their search results, for example, to papers written in English and published in the year 2021 only. In the case of the TAQ study case, we focus on the subject area filter in Scopus. We found that some papers were not categorized correctly according to their subject areas. For example, the ST study found two papers [18, 19] that could not be found by the third search (as discussed in Section 5.2). These papers were classified under the subject area *Engineering* instead of *Computer Science*. Likewise, the first search found five papers [20–24] that were not found by the third search. These five papers were classified wrongly in different subject areas (see Section 5.3) instead of *Computer Science*. This misclassification could be origin in the algorithm for detecting papers' subject areas in Scopus, inappropriate classification and keywording by the papers' authors, or a combination of both.

The third issue is search repeatability. We could not replicate the search result by the ST study in Scopus using their search string. The search repeatability issue has been well discussed in the literature [3, 11, 14, 16, 25]. We referred to the checklist proposed by Ali and Usman [3] for evaluating the search reliability of the ST study's search process. As a result, we found that some details the ST study could have reported to increase their search repeatability. Those details include search period, database-specific search strings, additional filters, deviation from the general search string, and database-specific search hits. The missing information and the potential inconsistencies in the API search of the search engine (Scopus in this case) could be the reasons for issues in search repeatability.

6.2. Issues related to using Quasi-gold standards

We have identified two issues related to using a quasi-gold standard (QGS) for search validation.

The first issue is about the QGS characteristics. To the best of our knowledge, several aspects have not been discussed sufficiently in the literature [1, 5]. Kitchenham et al. [1] described different approaches to constructing a QGS followed by a discussion on QGS size.

Zhang et al. [5] proposed a detailed guideline on building a QGS using a manual search on specific publication venues for a certain time span. We argue that QGS size is not the only aspect on which researchers should focus. We discuss this further and propose some suggestions to overcome this issue in Section 7.1.

The second issue with using the QGS for search validation is about the quality of the QGS itself. By its nature, the QGS is only an approximation of a complete set of relevant papers. However, by conducting more than one search, we could triangulate issues in the QGS and make informed decisions about modifying our search string. Comparing our search results to the ST study's search result (the basis for our QGS), we could identify the root causes for not finding certain relevant papers included in the QGS. This helped us establish whether our searches were simply not good enough with respect to the QGS or whether there were acceptable reasons for missing a paper. Additionally, the search result comparisons helped us to understand why the QGS did not contain certain relevant papers found by our searches. Thus, it allows us to identify shortcomings of the QGS and have more confidence in the quality of the QGS than relying solely on the recall and precision results.

7. Recommendations for QGS construction and search validation

As discussed in Section 6.2, we argue that recall and precision are important for assessing a search result but that they should not be the only criteria. It is also critical to analyze the root causes for not finding papers that the search should have found by looking into those papers of the QGS that the search missed. It might turn out that these papers did not use any of the search terms in the title, abstract or keywords or that they used different terminologies. The search can then be modified to ensure that one or more of those papers can be found. However, which root causes are addressed (and how) depends on the potential return on investment, i.e., the number of additional relevant papers that may potentially be found in relation to the total increase in the size of the search result. We recommend playing through various scenarios and assessing their potential return on investment with the help of precision and recall.

To address the root causes originating in the QGS, we first describe the desirable characteristics of a good QGS in Section 7.1 and then propose recommendations for constructing a QGS in Section 7.2. For root causes originating in the obtained search results, we propose an additional analysis step in Section 7.3. These suggestions are based on our findings (reported in Section 5 and Section 6) when evaluating the searches' performance in the TAQ study [7].

7.1. QGS desirable characteristics

Fundamentally, a QGS needs to be a good "representative" of the gold standard, and having a good QGS is vital for search validation in SLSs. In this section, we describe desirable characteristics of a good QGS. The characteristics are based on our experience from using QGS [7, 26] and Wohlin et al. [27] discussion on search as a sampling activity when the entire population (i.e., the set of all relevant papers) is unknown. Moreover, we draw inspiration from the snowball sampling guidelines for a good initial set to propose recommendations for arriving at a good QGS [28].

The main characteristics of a QGS discussed in the SE literature are *relevance* and *size* [1, 5]. For example, Kitchenham et al. [1] suggest indications for acceptable QGS sizes for various SLS types. However, as it is impossible to have true gold standards for most SLSs in SE [5] and the overall population of relevant papers is unknown [27], we argue that size alone is insufficient as an indicator of the quality of a QGS. We, therefore, introduce a third desirable characteristic, diversity, and present the complete list of QGS desirable characteristics as follows:

1. **Relevance**
   Each paper in the QGS should be relevant to the targeted topic. Any paper added to the QGS should meet the inclusion criteria of the ongoing SLS. In the TAQ study [7], we used the selected papers from a related SLS as a QGS after confirming that those papers met the selection criteria of the study.
2. **Size**
   Unlike *relevance* and *diversity*, where general suggestions have been provided, giving a recommendation for the size for a QGS is difficult since the "target population" is unknown. The number of relevant papers for an SLS can vary widely. The SLSs in Kitchenham et al. SLR of SLRs in software engineering [29] included 6–1485 relevant papers with a median of 30.5 papers. The tertiary study by da Silva et al. [30] lists a range from 4–691 (median: 46). Since the focus of an SLS can be general or narrow, depending on the topic of interest and the type of research questions, providing a general recommendation for the minimal size of a QGS seems impossible.
3. **Diversity**
   Diversity entails that a good QGS should comprise papers extracted from independent clusters representing different research communities, publishers, publication years and authors. This is important as even a large, and relevant QGS will be ineffective to objectively assess a search strategy if it is limited in its coverage.

7.2. QGS construction

There are neither fixed thresholds for quality indicators nor a deterministic way of arriving at a good QGS. However, the following recommendations[2] for identifying and selecting suitable papers for inclusion in a QGS provide heuristics that will increase the likelihood of creating a diverse QGS that can help determining 'is my search good enough' more objectively.

1. **Identification:** There are several approaches researchers could consider to locate relevant papers for their QGS construction:
   - Conduct manual search. Researchers first manually identify relevant venues (conferences, workshops, and journals) and researchers. After that, researchers can manually search for relevant papers by reading titles of papers in the selected venues (most common sources are Google Scholar, Scopus, DBLP) and of the selected authors.
   - Conduct informal search in electronic data sources. We recommend that persons conducting the informal search should be independent researchers. An independent researcher here is not involved in the study and has not participated in the design of the search strategy for the study. We recommend these additional considerations because the search terms used in the informal search might compromise the effectiveness of the QGS as a validation mechanism. For example, if the same search

[2]The recommendations in Section 7.2 are a synthesis of existing guidelines [1, 3, 11, 28] and our own experience as reported in this study and from using QGS in other systematic literature reviews [26].

terms are used for the informal search and the actual systematic search, then the recall is likely 100% since the actual search will probably find the same relevant papers as the informal search but not more than that. Hence, the 100% recall cannot guarantee that researchers achieve an acceptable level of search completeness. We further recommend that researchers should use citation databases like Scopus and Google Scholar in this step to avoid publisher bias.
   - Use expert's recommendation. Researchers could have an expert in the field (not involved in the search strategy design) recommend papers for a QGS for the current study. The experts should have access to the research questions and the selection criteria of the study.
   - Use an existing SLS. An existing SLS could be selected as a source of papers for the QGS. Since existing SLSs have been peer-reviewed, and their study selections typically have been validated, researchers will save time using this approach compared to the above approaches. However, the topics of existing SLSs will usually differ at least slightly from the topic of the new SLS (otherwise, a new SLS would not be necessary). The QGS might, therefore, not cover the research questions in the new SLS sufficiently. Hence, researchers should critically review the search and selection strategy of the selected SLS. We recommend using the checklist provided by Ali and Usman [3] to assist this evaluation. If the SLS had major weaknesses in search, we suggest supplementing the construction of the QGS with the above approaches.
2. **Selection:** The researchers should evaluate the potentially relevant papers identified through the above sources for relevance. We suggest using the selection criteria of the targeted SLS to select papers that should comprise the QGS.
3. **When to stop:** An exhaustive search of the potential sources listed above is impractical. After all, this is not the actual search but rather an attempt to create a good validation set for the search strategy. We, therefore, recommend that consulting a combination of sources and selection should be done iteratively until a sufficiently large, relevant and diverse QGS is obtained. What is sufficiently large will depend on the research questions and the breadth of the target research area. Due to the reasons discussed above (in Section 7.1), we do not recommend any range here and leave it to the subjective judgment of the researchers. Nevertheless, we argue that the more diverse a research area is, the larger a QGS is needed. As an indication of size, researchers should investigate the numbers of selected studies in existing SLSs in the area or the sizes of QGSs in related SLSs. Furthermore, if the QGS will be split for both search string formation and validation, a larger QGS will be required. Overall, a good QGS should be diverse, not too small, and relevant for answering the research questions. Primarily, the resulting QGS should have papers from different research communities, publishers, publication years, and authors.

7.3. Additional recommendation for search validation using QGSs

Kitchenham et al. [1] have discussed two approaches to validate a search strategy via search completeness (more details in Section 3). Researchers could use the personal judgment of experienced researchers to evaluate the search completeness. Since this approach is subjective, it might be challenging to quantify the search completeness level. The other approach is to measure the completeness level by calculating the precision and recall of searches based on a pre-established QGS. With the second approach, the completeness assessment becomes objective within the limits of the quality of the QGS. This means that

the quality assessment of the search string is connected with the quality assessment of the QGS. In other words, if the QGS was not constructed properly, even a high recall cannot guarantee that the search result is good.

Following the above guidelines will increase our confidence in the precision and recall values. While meeting certain search recall and precision thresholds (see [1, 5]) are necessary, it is also essential to understand how the search achieves these recall and precision scores. Hence, we suggest researchers perform the additional step of analyzing the differences between the search results and the QGS. This allows researchers to identify reasons for missing relevant papers with the automated search that are included in the connected QGS, and consequently improve their search strategy or document the limitations. For example, we found that it is necessary to be aware that subject areas categorization in some search engines might not categorize papers adequately. When comparing the search results with the QGS, we noticed that we could not find several papers as they were assigned to the wrong categories.

To facilitate this additional step, we suggest that researchers should use tools to analyse the search overlap. The metadata in search results is not consistently formatted across various data sources and often has minor differences like inconsistent capitalization and differences in encoding of special characters. Therefore, care must be taken to clean the data. Reference management tools like Zotero[3] or EndNote[4] can be used to compare the search results. Furthermore, the use of visualizations like Figures 3 and 4 helps to get a better understanding of comparative performance of various search strings. There are tools that can assist researchers in analyzing and visualizing lists intersections, such as one developed by the Bioinformatics and Evolutionary Genomics Group[5] or InteractiVenn[6] by Heberle et al. [17] that we used in this study.

7.4. Potential limitations

The recommendations and additional search validation steps proposed in this study are closely based on our experience while performing automated database searches in a tertiary study on test artifact quality [7]. In this tertiary study, we used another related tertiary study [8] to construct a QGS for the search validation. There could be other issues if we had used another search strategy or a different QGS construction approach. Therefore, the list of issues is not exhaustive, and the recommendations in this paper may need to be supplemented further.

For example, our recommendations for search validation using QGSs might not apply for SLSs with the traditional snowball sampling approach, i.e., all known relevant papers are used as the initial set. In other words, the QGS and the initial set are the same. Hence, the recall will always be 100% but not useful to validate the search completeness. However, the recommendations could become applicable if researchers split the whole set of known relevant papers into two subsets. In this case, one subset of known relevant papers will be used as the initial set for snowballing search, while the other will be used to validate the snowballing search results as the QGS.

[3]Zotero, a free and open-source reference management tool https://www.zotero.org/
[4]EndNote, a commercial reference management tool https://endnote.com/
[5]http://bioinformatics.psb.ugent.be/webtools/Venn/
[6]http://www.interactivenn.net/

8. Conclusions and lessons learned

Search incompleteness, i.e., the absence of relevant papers in the results produced by the employed search strategy, has been recognized as one of the most commonly discussed validity threats of systematic literature studies (SLSs). This study reports our experience with mitigating this validity threat while performing searches in a tertiary study on test artifact quality [7]. We constructed a quasi-gold standard (QGS) by extracting relevant papers from another relevant tertiary study [8] published several years before ours. While evaluating the tertiary study's searches using the QGS, we have found new issues with the search string construction and the search validation approach using a QGS. The issues could affect search completeness in SLSs. They relate to using generic search terms with the Boolean operator **AND**, the subject area filter in Scopus, and the QGS quality.

Consequently, we proposed extending the current search validation approach by the analysis step of the automated search validation results as well as recommendations on the QGS construction. The main argument of the analysis step of the search validation results is that recall and precision is not enough to validate an automated search. Researchers should analyze reasons for the automated search to miss relevant papers included in the QGS. Likewise, addressing the concern of QGS quality that has not been well studied in the literature, our recommendations on the QGS construction step helps researchers construct a high-quality QGS, i.e., a good "representative" of the gold standard. Ultimately, the extended guideline and recommendations can support researchers achieve a more reliable search process. To validate and improve the extended guidelines for search validation, we will collect feedback from the software engineering research community via interviews and surveys.

Acknowledgment

This work has been supported by ELLIIT, a Strategic Area within IT and Mobile Communications, funded by the Swedish Government. The work has also been supported by research grant for the VITS project (reference number 20180127) from the Knowledge Foundation in Sweden.

References

[1] B.A. Kitchenham, D. Budgen, and P. Brereton, *Evidence-Based Software Engineering and Systematic Reviews*. Chapman and Hall/CRC, 2015.

[2] N.B. Ali and M. Usman, "A critical appraisal tool for systematic literature reviews in software engineering," *Information and Software Technology*, Vol. 112, 2019, pp. 48–50.

[3] N.B. Ali and M. Usman, "Reliability of search in systematic reviews: Towards a quality assessment framework for the automated-search strategy," *Information and Software Technology*, Vol. 99, Jul. 2018, pp. 133–147. [Online]. https://linkinghub.elsevier.com/retrieve/pii/S0950584917304263

[4] M. Usman, N.B. Ali, and C. Wohlin, "A quality assessment instrument for systematic literature reviews in software engineering," *CoRR*, Vol. abs/2109.10134, 2021.

[5] H. Zhang, M. Babar, and P. Tell, "Identifying relevant studies in software engineering," *Information and Software Technology*, Vol. 53, No. 6, 2011, pp. 625–637.

[6] O. Dieste and A.G. Padau, "Developing search strategies for detecting relevant experiments for systematic reviews," in *Proceedings of the 1st International Symposium on Empirical Software Engineering and Measurement*, 2007, pp. 215–224.

[7] H.K.V. Tran, M. Unterkalmsteiner, J. Börstler, and N. bin Ali, "Assessing test artifact quality – A tertiary study," *Information and Software Technology*, Vol. 139, 2021.

[8] V. Garousi and M. Mäntylä, "A systematic literature review of literature reviews in software testing," *Information and Software Technology*, Vol. 80, 2016, pp. 195–216.

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

[10] A. Ampatzoglou, S. Bibi, P. Avgeriou, M. Verbeek, and A. Chatzigeorgiou, "Identifying, categorizing and mitigating threats to validity in software engineering secondary studies," *Information and Software Technology*, Vol. 106, 2019, pp. 201–230.

[11] A. Ampatzoglou, S. Bibi, P. Avgeriou, and A. Chatzigeorgiou, "Guidelines for managing threats to validity of secondary studies in software engineering," in *Contemporary Empirical Methods in Software Engineering*, M. Felderer and G.H. Travassos, Eds. Springer, 2020, pp. 415–441.

[12] J. Bailey, C. Zhang, D. Budgen, M. Turner, and S. Charters, "Search Engine Overlaps: Do they agree or disagree?" in *Proceedings of the 2nd International Workshop on Realising Evidence-Based Software Engineering*, May 2007, p. 2.

[13] S. Imtiaz, M. Bano, N. Ikram, and M. Niazi, "A tertiary study: experiences of conducting systematic literature reviews in software engineering," in *Proceedings of the 17th International Conference on Evaluation and Assessment in Software Engineering*, 2013, pp. 177–182.

[14] J. Krüger, C. Lausberger, I. von Nostitz-Wallwitz, G. Saake, and T. Leich, "Search. review. repeat? an empirical study of threats to replicating SLR searches," *Empirical Software Engineering*, Vol. 25, No. 1, 2020, pp. 627–677.

[15] L. Chen, M.A. Babar, and H. Zhang, "Towards an evidence-based understanding of electronic data sources," in *Proceedings of the 14th International Conference on Evaluation and Assessment in Software Engineering*. BCS, 2010, pp. 1–4.

[16] X. Zhou, Y. Jin, H. Zhang, S. Li, and X. Huang, "A map of threats to validity of systematic literature reviews in software engineering," in *Proceedings of the 23rd Asia-Pacific Software Engineering Conference*, 2016, pp. 153–160.

[17] H. Heberle, G.V. Meirelles, F.R. da Silva, G.P. Telles, and R. Minghim, "Interactivenn: A web-based tool for the analysis of sets through venn diagrams," *BMC bioinformatics*, Vol. 16, No. 1, 2015, pp. 1–7.

[18] J.R. Barbosa, A.M.R. Vincenzi, M.E. Delamaro, and J.C. Maldonado, "Software testing in critical embedded systems: A systematic review of adherence to the do-178b standard," in *Proceedings of the 3rd International Conference on Advances in System Testing and Validation Lifecycle*, 2011, pp. 126–130.

[19] A. Sharma, T.D. Hellmann, and F. Maurer, "Testing of web services – A systematic mapping," in *Proceedings of the 8th IEEE World Congress on Services*, 2012, pp. 346–352.

[20] T.K. Paul and M.F. Lau, "Redefinition of fault classes in logic expressions," in *Proceedings of the 12th International Conference on Quality Software*, 2012, pp. 144–153.

[21] I.U. Munasinghe and T.D. Rupasinghe, "A supply chain network design optimization model from the perspective of a retail distribution supply chain," in *Proceedings of the Manufacturing and Industrial Engineering Symposium: Innovative Applications for Industry*, 2016.

[22] J. Ahmad and S. Baharom, "A systematic literature review of the test case prioritization technique for sequence of events," *International Journal of Applied Engineering Research*, Vol. 12, No. 7, 2017, pp. 1389–1395.

[23] S. Pradhan, M. Ray, and S. Patnaik, "Coverage criteria for state-based testing: A systematic review," *International Journal of Information Technology Project Management*, Vol. 10, No. 1, 2019, pp. 1–20.

[24] P.K. Arora and R. Bhatia, "A systematic review of agent-based test case generation for regression testing," *Arabian Journal for Science and Engineering*, Vol. 43, No. 2, 2018, pp. 447–470.

[25] B. Kitchenham, P. Brereton, Z. Li, D. Budgen, and A. Burn, "Repeatability of systematic literature reviews," in *Proceedings of the 15th Annual Conference on Evaluation and Assessment in Software Engineering*, 2011, pp. 46–55.

[26] N.B. Ali, E. Engström, M. Taromirad, M.R. Mousavi, N.M. Minhas et al., "On the search for industry-relevant regression testing research," *Empir. Softw. Eng.*, Vol. 24, No. 4, 2019, pp. 2020–2055.

[27] C. Wohlin, P. Runeson, P. Da Mota Silveira Neto, E. Engström, I. Do Carmo Machado et al., "On the reliability of mapping studies in software engineering," *Journal of Systems and Software*, Vol. 86, No. 10, 2013, pp. 2594–2610.

[28] 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.

[29] B. Kitchenham, O. Pearl Brereton, D. Budgen, M. Turner, J. Bailey et al., "Systematic literature reviews in software engineering – A systematic literature review," *Information and Software Technology*, Vol. 51, No. 1, 2009, pp. 7–15.

[30] F. Da Silva, A. Santos, S. Soares, A. Frana, C. Monteiro et al., "Six years of systematic literature reviews in software engineering: An updated tertiary study," *Information and Software Technology*, Vol. 53, No. 9, 2011, pp. 899–913.

A. Study Selection Criteria

Our study selection inclusion/exclusion criteria are described as follows:

1. Phase 1: applied on authors, title and abstract
   - Exclude papers that:
     (E1) are duplicate papers;
     (E2) are not systematic studies[7];
     (E3) are not peer reviewed;
     (E4) are outside computer science or software engineering.
2. Phase 2: applied on title and abstract
   - Exclude papers that:
     (E5) are not about software testing.
   - Include papers that fulfil all of the following:
     (I1) are systematic literature reviews (SLR), quasi-SLRs, Multi-vocal literature reviews, or systematic mappings;
     (I2) discussed or potentially discussed quality of test artifacts
3. Phase 3: applied on full text
   - Exclude studies that:
     (E6) Are duplicate studies (two different studies using the same data)
   - Include studies which discussed any of the following:
     (I3) definition of the quality of test artifacts;
     (I4) quality characteristics of test artifacts;
     (I5) quality attributes of test artifacts;
     (I6) quality metrics of test artifacts;
     (I7) tools, methods, approaches, frameworks to assess test artifacts' quality;
     (I8) guidelines, checklists to write test artifacts.

[7]Garousi and Mäntylä's [8] initial set of 121 papers contained 63 informal surveys without research questions. Since we only were targeting systematic studies, these were excluded.

e-Informatica Software Engineering Journal, Volume 16, Issue 1, 2022, pages: 220104, DOI: 10.37190/e-Inf220104

Examining the Predictive Capability of Advanced Software Fault Prediction Models An Experimental Investigation Using Combination Metrics

Pooja Sharma*, Amrit Lal Sangal*
*Dr. B R Ambedkar National Institute of Technology, Jalandhar, India
poojanitjal@gmail.com, sangalal@nitj.ac.in

Abstract

Background: Fault prediction is a key problem in software engineering domain. In recent years, an increasing interest in exploiting machine learning techniques to make informed decisions to improve software quality based on available data has been observed.
Aim: The study aims to build and examine the predictive capability of advanced fault prediction models based on product and process metrics by using machine learning classifiers and ensemble design.
Method: Authors developed a methodological framework, consisting of three phases, i.e., (i) metrics identification (ii) experimentation using base ML classifiers and ensemble design (iii) evaluating performance and cost sensitiveness. The study has been conducted on 32 projects from the PROMISE, BUG, and JIRA repositories.
Result: The results shows that advanced fault prediction models built using ensemble methods show an overall median of F-score ranging between 76.50% and 87.34% and the ROC(AUC) between 77.09% and 84.05% with better predictive capability and cost sensitiveness. Also, non-parametric tests have been applied to test the statistical significance of the classifiers.
Conclusion: The proposed advanced models have performed impressively well for inter project fault prediction for projects from PROMISE, BUG, and JIRA repositories.

Keywords: product and process metrics, classifiers, ensemble design, software fault prediction, software quality

1. Introduction

Software fault prediction has been an important research topic in the software engineering field for more than three decades, increasingly catching the interest of researchers [1, 2]. According to IEEE terminology [3] the term fault is used to indicate an incorrect step, process, or data definition in a computer program (i.e., a BUG). In the literature, authors have addressed the software fault prediction (SFP) problem with two viewpoints, i.e., in the first viewpoint, they proposed new method or method combinations to increase fault prediction performance. In the second viewpoint, they used new parameters to present the most influential metrics for fault prediction. Based on first perspective many fault prediction approaches have been proposed in literature and most of these papers

Submitted: 27 Oct. 2020; Revised: 02 Dec. 2021; Accepted: 07 Dec. 2021; Available online: 14 Feb. 2022

categorize a software module faulty or non-faulty. Unfortunately, fault-proneness of software components classification remains a largely unsolved problem [2]. In order to address this issue, researchers have been increasingly using sophisticated techniques and we can say that the fault prediction is going towards novel and more attractive directions, like the use of machine learning, deep learning or unsupervised techniques [4–6]. The usage of machine learning algorithms has increased in the last decade and is still one of the most popular methods for defect prediction [4, 6–10]. According to Lessmann et al. [11] there is a need to develop more reliable research procedures before we can have confidence in the conclusion of comparative studies of software prediction models. Thus, in the present study we aim to consider and evaluate the performance of different classifier models and not any particular classifier. Further, application of ensemble techniques has been reported by the researchers [4, 8, 12] for improving the accuracy of fault prediction. Moreover, the diversity of classifiers, while building the ensemble model, should also be investigated to improve the effectiveness of the ensemble designs [9]. This motivated us to design ensembles for improving predictive capability of classifiers.

As regards to the second viewpoint, considerable amount of the research has been undertaken in which authors have used software metrics extracted from the code to unveil whether a software component is fault prone or not. It has been observed that fault estimation models are designed mainly based on product metrics in literature [13–16], but the models which are build using a combination of product and process metrics are little known [17, 18]. Though some authors [19, 20] has emphasized about the usage of both product and process metrics in their works. Madeyski and Jureczko [18], in their research, determined that process metrics provide information for fault proneness. The usage of process metrics to ascertain the faults possibly results in superior outcomes than only with the product metrics. They emphasized the need to conduct further studies and establish evidence for developing such advanced models. Radjenovic et al. [19] in their SLR, stressed finding ways to measure and evaluate process-related information for fault proneness. Wan et al. [19] in their study on perceptions, expectations, and challenges in defect prediction, concluded that software practitioners prefer rational, interpretable, and actionable metrics for defect prediction. It is also observed from the literature studies that not only process metrics have been shown to be superior to product metrics, but also alternative features have been proposed on the basis of developer-related factors, code smells, etc. [21–24]. This calls for further studies to examine the association between metrics and fault proneness to provide meaningful insights for making informed decisions. To this effect, the authors in the present study aimed to develop advanced models for software fault prediction, which utilises combination metrics. After finding a suitable set of product metrics, advanced fault prediction models are created using process metrics one at a time approach.

Thus, to motivate the need for development of advanced models for fault prediction authors in the present study developed a research framework which consists of three phases. In Phase-I, the metrics were identified after performing pre-processing and feature extraction on the datasets. In phase-II, experimentation is carried out by training and testing various models using machine learning classifiers, i.e., Naive Bayes (NB), Decision Tree (DT), Multilayer Perceptron (MLP), Random Tree (RT), and Support Vector Machine (SVM). To estimate the performance of the advanced models, an assessment criterion based upon accuracy, root mean square error (RMSE), F-score, and the area under curve AUC(ROC) has been applied. In phase-III, rather than relying on the outcome of base classifiers, authors used the ensemble approach to combine multiple classifiers to further improve the performance, particularly fault-detection abilities. Also, the cost sensitiveness

of the proposed best models is examined. The comparison of results confirms the predictive capability of proposed classifiers for developing advanced fault prediction models.

Thus, the significant contributions of the work are as follows:

1. Development of learning scheme consisting of both base and ensemble learning classifiers.
2. Building and examining the predictive capability of advanced fault prediction models.
3. Evaluating the cost sensitiveness of the proposed ensemble-based classifier using a cost evaluation framework.

The work presented in the study is reported as follows. Section 2 offers related research. Section 3 presents a description of the proposed framework, research questions, dataset selection, feature extraction, selection, normalization procedure, classifier selection, and performance measurement indices. Section 4 presents the experimental design and Section 5 presents the results. Section 6 presents threats to validity, and Section 7 presents the conclusions.

2. Related Work

Over the preceding two decades, software researchers have shown great prominence in fault prediction studies, as evident from work dealing with the development of fault prediction models. Table 1 presents the state of the art and proposed benchmark solutions. The contributions provided by the researchers in recent years are summarised based on the software metrics (product, process, change) and techniques used to tackle the software fault prediction problem. Malhotra and Jain [8] provided empirical comparison of software defect prediction models developed by using various boosting based ensemble methods on three open source JAVA projects. Ghotra et al. [25] studied the impact of classification techniques on the performance of defect prediction models. Yucalar et al. [26] conducted experiments using 15 software projects from the PROMISE repository to demonstrate that ensemble predictors might improve fault detection performance to some extent. Qiao et al. [16] proposed deep learning techniques to predict defects in a software system. The study by Malhotra [15] uses a logistic regression-based classifier on object-oriented metrics data set to predict the software fault proneness. Laradji et al. [27] demonstrated the positive effects of combining feature selection and ensemble learning on the performance of defect classification.

Comprehensive surveys on fault prediction were presented by Catal and Diri [28], Li Zhiqiang et al. [1]; Matloob et al. [9] and Radjenovic et al. [19] in the context of prediction models, modelling techniques and the metrics used. According to Radjenovic et al. [19], in the literature on fault prediction studies, process metrics account for 24%, source code accounts for 27%, and object-oriented accounts for 49%. Future studies shall apply the ways to measure and evaluate process-related information for fault proneness along with product metrics. Madeyski and Jureczko [17] performed an empirical study using industrial and open-source software datasets to ascertain the process metrics, which noticeably improved results. At the same time, they stressed upon replicating the study using machine learning approaches, as it is unclear whether the features that work fine in one method will also be useful in other approaches. Hence, experimentation can be conducted to investigate the usage of the product and process-related metrics. Khoshgoftaar et al. [29] build software quality models with majority voting using multiple training datasets. The work can be extended using data from various software project repositories and analyse the predictive capability of ensembles as compared to base classifiers for advanced models. Chen et al.

[30] in work investigated whether different crossproject defect prediction methods identify the same defective modules? The result can be extended by using learning approaches based on ensemble design to further improve crossproject defect prediction performance. In the study Zhang et al. [31], investigated the use of various algorithms that integrate ML predictors for cross-project defect prediction. However, for examining the predictive capability of advanced algorithms, additional experimentation is required.

Studying the presented works above, it is clear that using a pre-processing technique on the dataset significantly affected the performance of learning algorithm. Most of the studies lack the processing on a larger dataset so that the generalized model will be formed. Also, class imbalance problem, needs to be addressed to improve the performance of fault prediction [9]. The parameter combinations are often less investigated in literature studies. Hence, it is observed that the work can be replicated by including more datasets with focus on product and process software metrics and experimenting different scenarios or combinations of models (simple and advanced models) to achieve the reliability and robustness.

Further investigations shall include the use of more classifiers or classifier ensembles and the development of advanced defect prediction models with datasets from various projects written in different programming languages, and commercial projects from industry can also be considered for experimentation. In the proposed work, authors presented a three phase framework consisting of dataset pre-processing, feature extraction and selection; learning classifiers along with cost evaluation to predict the fault-prone components.

Table 1. Literature review

Authors	Metrics considered	Study outcomes and proposed benchmark solutions
Song et al. [2]	Product metrics	Authors proposed and evaluated a general framework for software defect prediction using different learning schemes for different data sets. The future work shall include process attributes for fault estimation. Experiments with the various available techniques can be undertaken for generalization.
Yang et al. [5]	Product metrics	Authors proposed a deep learning technique to predict defect-prone changes. The experiments can be replicated on more datasets using other classifiers to reduce the threats to external validity.
Yibiao et al. [6]	Change metrics	Authors investigated the predictive power of simple unsupervised models in effort-aware JIT defect prediction using commonly used change metrics. The work can be checked with closed-source software systems.
Yang et al. [4]	–	Authors hybridized various ensemble learning methods to examine performance of just-in-time defect prediction. Experiments on more datasets can be performed to reduce the threats due to external validity.
Matloob et al. [9]	–	This research provides a systematic literature review on the use of the ensemble learning approach for software defect prediction and stressed for further analysis and comparison of results.
Pascarella et al. [10]	Change metrics	Authors proposed a novel fine-grained just-in-time defect prediction model to predict the specific files, contained in a commit, that are defective. Future work can replicate the results on a larger set of systems in an industrial context by including other independent variables too.

Table 1 continued

Malhotra, Jain [8]	Product metrics	Authors provided empirical comparison of software defect prediction models developed by using various boosting based ensemble methods on three open source JAVA projects. The future work shall investigate more attributes for fault estimation with more datasets for replication.
Li et al. [14]	Code metrics	Authors summarised the defect prediction studies focusing on emerging topics, e.g., ML-based algorithms, data manipulation, and effort-aware prediction. They stressed overcoming the class imbalance problem and the development of models in defect prediction.
Ghotra et al. [25]	Product metrics	Authors studied the impact of classification techniques on the performance of defect prediction models using NASA dataset and the Promise dataset. Further experiments with the various available techniques can be undertaken for generalization.
Yucalar et al. [26]	Product metrics	The authors conducted experiments using 15 software projects from the Promise repository to demonstrate that ensemble predictors might improve fault detection performance to some extent. The future work shall investigate more attributes for fault estimation to provide help in successive releases.
Rathore and Kumar [12]	Product metrics	Authors performed an investigation on ensemble techniques for SFP by using 21 object-oriented software metrics. Future work can assess the ensemble techniques for the fault datasets from other software systems and shall include additional software metrics for generalization.
Qiao et al. [16]	Product metrics	The authors proposed deep learning techniques to predict defects in a software system. In future work, more investigations by including more projects are written in different programming languages, and commercial projects from industry can be carried out.
Malhotra [15]	Product metrics	The study uses a logistic regression-based classifier on object-oriented metrics data set to predict the software fault proneness. Future investigations shall include the use of more classifiers or classifier ensembles and the development of advanced defect prediction models with cross project defect prediction datasets from various projects.
Madeyski and Jureczko [18]	Product and Process	They performed an empirical study using industrial and open source software datasets to ascertain the process metrics, which noticeably improved results. At the same time, they stressed upon replicating the study using ML approaches, as it is unclear whether the features that work fine in one method will also be useful in other approaches. Hence, experimentation can be conducted to investigate the usage of the product and process-related metrics.
Radjenovic et al. [19]	Process and Product	According to the authors, in the literature on fault prediction studies, process metrics account for 24%, source code accounts for 27%, and object-oriented accounts for 49%. Future studies shall apply the ways to measure and evaluate process-related information for fault proneness along with product metrics.
Rahman, and Devanbu [24]	Product and process	Authors analysed the applicability and efficiency of process and code metrics. The future work shall replicate the findings with more data sets from several different perspectives.
Bird, Christian et al. [21]	Change metrics	Authors examined the relationship between different ownership measures and software failures in two large software projects: Windows Vista and Windows 7.

Table 1 continued

Nucci et al. [22]	Product and change metrics	Provided a developer centred bug prediction model. Work can be extended to analyse the role of developer related factors along with product metrics in the bug prediction field using different base line predictors.
Palomba et al. [23]	Process and Product	Authors evaluated the code smell intensity by adding it to existing bug prediction models based on both product and process metrics. Future work shall be devoted to the analysis of the contribution of smell-related information in the context of local-learning bug prediction models.
Laradji et al. [27]	Product metrics	Authors demonstrated the positive effects of combining feature selection and ensemble learning on the performance of defect classification. The work can be replicated by including more datasets with focus on product and process software metrics.
Lee et al. [32]	Process and Product	They proposed micro-interaction metrics to study developers interaction by experimenting with Mylyn dataset. More experiments need to be conducted to show that MIMs considerably improve software defect prediction.
Juneja [33]	Product	Authors proposed Neuro-fuzzy framework to predict the fault in software system based on feature based evaluation of inter-project and intra-project modules. The effectiveness of models can be compared using process metrics.
Wang et al. [34]	Product metrics	The authors performed a study using seven classifiers ensemble methods on MDP datasets from real software projects of NASA. The use of classifiers ensemble on multiple datasets can be experimented.
Petric et al. [35]	Product metrics	They used explicit diversity technique with stacking ensemble to investigate improvement in defect prediction. The work can be extended and the experiments should be conducted using more classifiers and applying full parameter search in order to build models with superior performances.
Pecorelli and Nucci [36]	Product metrics	Authors compared the performance of seven ensemble techniques on 21 open-source software projects to verify how ensemble techniques perform in cross and local project settings. The work can be replicated using cross-project and within-project strategies in larger contexts, using a richer set of independent variables.
Nucci et al. [37]	Product metrics	An empirical study conducted on 30 software systems indicates that ASCI exhibits higher performances than five different classifiers used independently and combined with the majority voting ensemble method. Work can be extended to analyse how the proposed model works in the context of cross-project bug prediction.
Bowles et al. [38]	Product metrics	Authors investigated difference in the individual defects and prediction stability using RPart, SVM, Naive Bayes, and Random Forest classifiers. They used NASA, open-source, and commercial datasets. The work can be extended by developing advanced models using ensemble-based classifiers.
Abaei and Selamat [39]	Product metrics	They proposed fuzzy clustering and probabilistic neural network to study defect prediction accuracy. The use of machine learning approaches can be investigated to analyze advanced defect prediction models.
Erturk and Sezer [40]	CK Product metrics	In their work, the authors concluded that ANFIS outperforms NN and SVM approaches for predicting faults. The future work may include the process metrics or develop advanced defect prediction models.

Table 1 continued

Zhang et al. [31]	Process and Product metrics	In the study authors, investigated the use of various algorithms that integrate ML predictors for cross-project defect prediction. However, for examining the predictive capability of advanced algorithms, additional experimentation is required.
Khoshgoftaar et al. [29]	Product and Process	Authors build software quality models using majority voting using multiple training datasets. The work be extended using data from various software project repositories and analyze the predictive capability of ensembles as compared to base classifiers for advanced models.
Yong Hu et al. [41]	Product (all CK metrics)	This study provides a research framework that combines cost-sensitive learning with the ensemble method. Future work can examine the use of ensembles trained on different datasets. Such solutions may not only enhance the prediction accuracy but also address the defect prediction problems.
Elish et al. [42]	product metrics	The authors used product metrics to investigate and empirically validate ensemble methods for software maintenance effort and change proneness. However, future studies shall use the proposed ensemble approaches to investigate defect prediction using combination metrics.
Chen et al. [30]	Process and Product	The authors in work investigated whether different cross project defect prediction methods identify the same defective modules. The result can be extended by using learning approaches based on ensemble design to further improve cross project defect prediction performance.
Peng He et al. [43]	Static code metrics	The authors provided guidelines for the selection of training data, classifier, and metric subset. They conducted an empirical study on software defect prediction with a simplified metric set. The guidelines can further be used to develop advanced models for defect prediction in different scenarios.
Wasiur R et al. [44]	Change metrics	Authors conducted an empirical study for defect prediction using software change metrics. The application of hybrid algorithms used in the task can be used to develop advanced models.
Kaur and Kaur [45]	Product metrics	Authors used statistical and machine learning techniques for predicting the quality of the software. For experimentation, they used five open source software projects. Further experiments can be conducted using product-process or combination metrics using cross project defect data.

3. Research Framework

The proposed framework consists of three phases, as shown in Figure 1. Phase-I deals with dataset pre-processing, feature extraction and experimental setup; Phase-II is classification methods, ensemble design and performance measurement and Phase-III is cost evaluation framework. Briefly, the phases shown in Figure 1 are discussed as:

Phase-I deals with identifying the metrics suite from metric datasets available in PROMISE, BUG, and JIRA dataset repository. Further, various pre-processing methods such as feature ranking methods and feature subset selection methods and normalization have been applied to select a minimal subset of features from the original dataset so that the features are reduced based on a specific evaluation criterion. It also reduces the dimensionality of feature

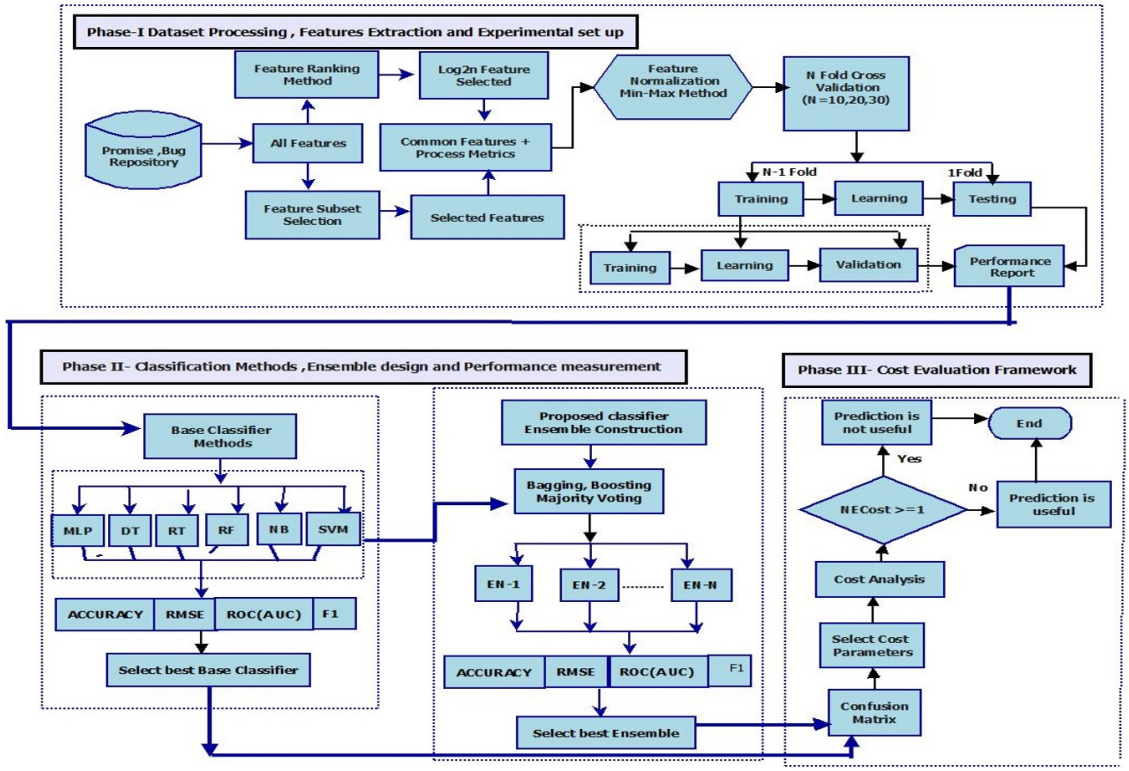


Figure 1. A framework of Proposed ensemble model with cost analysis

space, removes redundant, irrelevant information and improves the data quality, thereby improving the algorithm performance. An experimental design with N-fold cross-validation is used to train, test and replicate the experiment using various datasets.

Phase-II deals with the evaluation of simplified dataset representing different scenarios, i.e., scenario-1: simple model (product metrics); scenario-2: Advanced model-1 (Product metrics + NR process metric); scenario-3: Advanced model-2 (Product metrics + NDC process metric); scenario-4: Advanced model-3 (Product metrics + NML process metric); scenario-4: Advanced model-4 (Product metrics + NDPV process metric) using various base ML classifiers. The performances of proposed models are evaluated using performance indices, i.e., accuracy, AUC (ROC), RMSE, and F-score. Further, to improve base ML classifiers performance, the classifier ensembles were designed by following Bagging, AdaBoostM1 (which is the most popular version of boosting), and Voting algorithms.

Phase-III deals with examining the cost sensitiveness of the proposed ensemble classifiers. It is achieved by developing a cost analysis framework to compare the best ensembles cost with the best base classifier by finding normalized fault removal cost.

Research Questions

Based on the literature studies and potential research gaps, the research questions framed are as follows:

RQ1: How does the advanced defect prediction models proposed in the study perform using various machine learning classifiers?

RQ2: How does the ensemble design improve classification performance when compared to individual machine learning classifiers?

RQ3: Whether there exist any statistically significant performance difference among the base classifiers and ensemble classifiers?

RQ4: For a given software system, whether the proposed ensembles are cost sensitive?
The rationale behind the selection of the research questions RQ1 and RQ2 is to investigate the effectiveness of advanced models representing different scenarios of combination of software product and process metrics. These models are trained using base learning and ensemble based classifiers. The model performances are tested with measures such as accuracy, RMSE, ROC(AUC) and F-score. The rationale behind the usage of statistical test was to find the empirical evidence regarding the performance of predictors, i.e., to answer RQ3. Cost-based evaluation framework has been adopted to examine the cost-sensitiveness of proposed predictors in RQ4.

3.1. Selection of Dataset

In software engineering, Tera-Promise [46], Bug Prediction Dataset [47], Promise [48] and NASA and repositories contain versioned datasets of different software projects that can be assessed for fault prediction. In the present study, authors examined versioned datasets of (i) Ant, Camel, J-edit, Lucene, Synapse, Xalan, Xerces projects from the Promise repository, (ii) Equinox, Eclipse-JDT, Eclipse-PDE, MYLYN projects from the Bug dataset and (iii) ActiveMQ 5.0.0, Derby-10.5.1.1, Groovy1_6_BETA_1, Hbase-0.94.0, Hive-0.9.0, Jruby-1.1, Wicket-1.3.0beta2 from Jira repository, respectively. Table 2 presents the data related to versions, total modules, faulty modules, and defect rates of different projects with their interpretations. To improve the quality of software datasets, we performed data pre-processing following the guidelines provided by Shepperd et al. [49] in order to remove noisy data. To make the training set uniform for the fault-prone and non-fault prone classes to handle data imbalance, in the study, we have applied the synthetic minority over-sampling technique proposed by Chawla et al. [50]. In literature, researchers too have considered class imbalance learning techniques to improve the predictors performance [8, 29, 51].

Table 2. Project dataset versions

	Project	Total modules	Faulty modules	Defect rate
Promise dataset	ant 1.4	178	40	22.47
	ant 1.5	293	32	10.92
	ant 1.6	351	92	26.21
	ant 1.7	745	166	22.28
	camel 1.2	608	216	35.53
	camel 1.4	872	145	16.63
	camel 1.6	965	188	19.48
	jedit 4.0	306	75	24.51
	jedit 4.1	312	79	25.32
	jedit 4.2	367	48	13.07
	jedit 4.3	492	11	2.24
	Lucene 2.2	247	144	58.3
	Lucene 2.4	340	203	59.7
	synapse 1.1	222	60	27.03
	synapse 1.2	256	86	33.59
	xalan 2.5	803	387	48.19
	xalan 2.6	885	411	46.44

Table 2 continued

Promise dataset	xalan 2.7	909	897	98.79
	xerecs 1.2	440	71	16.14
	xerecs 1.3	453	69	15.23
	xerecs 1.4	588	437	74.32
Projects from Bug repository	Equinox	324	129	39.81
	Eclipse-JDT	997	206	20.06
	Eclipse-PDE	1497	209	13.96
	MYLYN	1862	245	13.15
Projects from Jira repository	ActiveMQ 5.0.0	1884	293	15.55
	Derby-10.5.1.1	2705	383	14.15
	Groovy1_6_BETA_1	821	70	8.52
	Hbase-0.94.0	1059	218	20.58
	Hive-0.9.0	1416	283	19.98
	Jruby-1.1	731	87	11.9
	Wicket-1.3.0beta2	1763	130	7.5

3.2. Feature Extraction, Selection and Normalization

Feature selection is categorised as feature ranking and feature subset selection, or be classified as filters and wrappers. In filter based algorithms, a subset of features is selected without involving any learning algorithm and in wrapper based algorithms feedback from a classification learning algorithm is used to determine the feature(s) to be included in development of a classification model. The more refined a feature subset becomes, the more stable a feature selection algorithm is [42]. It reduces the dimensionality of feature space, removes redundant, irrelevant information and improves the quality of the data thereby improving the performance of the algorithm. In the literature [42, 52, 53] numerous methods have been proposed to discard features which are least important to improve defect prediction.

3.2.1. Feature Ranking Methods

It is the process of ordering the features based upon the value of some scoring function, which generally measures feature relevance. In this study, authors have used Information Gain (IG) attribute estimation which is the frequency driven observation in which the information explicit to a particular metric is considered on the class value. The available information is corresponding to the fault proneness of specific modules. Similar feature ranking methods has been applied by various authors in their work on software fault prediction [7, 19, 26, 31]. Gain Ratio (GR) attribute estimation is an alternative of IG and is used to rank the attributes present in the datasets to reduce its bias [19, 54]. Gain Ratio is used for the proliferation of nodes when data is evenly distributed and small while choosing an attribute when all data belong to one branch.

3.2.2. Feature Subset Selection Methods

Instead of using all metrics of the dataset, a subset of features is used as input in the study. These methods are used to generate a subset of attributes that jointly have excellent predictive ability. The classifier subset evaluation method uses a classifier method to

estimate the merit of the possible subsets of features in the project. It evaluates the worth of a subset of attributes by considering the individual predictive ability of each feature along with the degree of redundancy between them [50, 53]. Correlation-based feature selection (CFS) evaluates values of the subset of attributes according to correlation with the class label and individual features along with the degree of redundancy between them [55]. Filtered subset evaluation is a random subset of the evaluator made to run on a class through an arbitrary filter using data [50, 56]. These filters do not change the order, and the numbers of attributes entirely depend on training data. In literature CFS based feature selection technique has been applied by various authors [7, 45, 57].

3.2.3. Feature extraction and selection

In the study, feature ranking and feature subset selection techniques such as IG, GR attribute evaluation, Classifier subset evaluation, CFS subset evaluation and Filtered subset evaluation were used in the experiments. The common sets of features extracted are shown in Table 3, respectively. A total of 15 features are selected and used in the experiments. The simple defect prediction model is constructed using the 15 product metrics, and advanced defect prediction models are built using 15 product metrics and single process metric with one at a time approach, as discussed in Section 3. Table A1 in appendix provides the definitions for the selected features based on product and process metrics.

Table 3. Selection of metrics

Feature Ranking Methods	Selected Metrics
Information Gain (IG)	AMC, LOC, CAM, LCOM3, LOC, AVG-CC, RFC, MFA, WMC, CBO, DAM, NPM, CE, MAX-CC, MOA, CA, NOC, CBM, IC, DIT
Gain Ratio (GR)	AMC, LCOM3, LOC, LCOM, CAM, AVG-CC, DAM, MFA, MOA, RFC, WMC, MAX-CC, CE, CBO, NPM, NOC, CA, CBM, IC, DIT
Feature Subset Methods	**Selected Metrics**
Subset evaluation Classifier	AMC, LCOM3, LOC, LCOM, CAM, AVG-CC, DAM, MFA, MOA, RFC, WMC, MAX-CC, CE, CBO, NPM, NOC, CA, CBM, IC, DIT
CFS subset evaluation	MOA, DAM, MAX-CC, LCOM, NOC, LCOM3, CE, IC, NPM, CBO, WMC, DIT, CA, RFC, MFA, AMC, LOC
Filtered subset evaluation	WMC, DIT, NOC, LCOM, NPM, MOA, CA, RFC, CE, LOC, DAM, AMC, CBO, AVG-CC, MAX-CC
Common Selected Features	LCOM, CA, LOC, AMC, CBO, RFC, DAM, WMC, DIT, NOC, MOA, CAM, MAX-CC, CE, NPM
Process Metrics	NR, NDC/NAUTH, NML/NREF, NDPV

3.2.4. Normalization of selected features

The performance of prediction models can also be affected by the different levels of design complexity metrics [58–60]. Various software metrics values which are obtained from the dataset have different ranges or magnitude; to make the data in a similar series or format,

we have applied data normalization. For the data normalization process, a simple min-max normalization method is used [61]. After the data is normalized, the values are transformed between intervals of 0–1.

3.3. Selection of classifiers

The main aim of the study is to demonstrate the predictive capability of advanced software defect perdition models. The well-known ML classifiers, i.e., Naive Bayes (NB); Decision tree (DT); Random tree (RT); Support Vector Machine (SVM) and Multilayer Perceptron (MLP) are used in the study to build defect prediction models. We used Catal et al. [28] review to determine frequency of base predictors in the software fault prediction literature. Authors performed comparative experimentation by taking one classifier from each category to achieve the balance between different classification models (statistical approaches, neural networks and tree-based methods) as proposed by various researchers [42–44, 61]. Also, to get an enhanced learning algorithm, classifiers ensembles have been designed. The names of the classifiers, classifiers ensembles and their references with brief description are presented in Table A2 in Appendix.

3.4. Performance measurement indices

For the assessment of defect predictors performance, various measures have been used in literature by researchers [11, 19, 27, 62, 63]. In the study, the performance indices, i.e., accuracy, RMSE, ROC (AUC) and F-score are used to measure the performance of fault prediction models. The brief details are presented in Table A3 in Appendix. Table A4 in Appendix presents the confusion matrix for fault prediction models, which is used to compute all the parameters. It contains actual and predicted classification information using various prediction techniques.

3.5. Framework for cost evaluation

Cost-based evaluation framework is necessary to assess the usability of designed fault prediction models. The analysis of cost evaluation is very important because misclassification of faulty prone (fp) modules is more costly as compared to the misclassification of non-faulty prone (nfp) modules. Some researchers [14, 41, 53] have adopted a cost evaluation criterion in their study. In this section, we discussed the cost evaluation framework, proposed by Wagner [64]. He has designed the cost-based evaluation framework based on certain constraints, as mentioned below:
(i) Different phases (unit, integration and testing phases) of testing account for different fault removal cost.
(ii) None of testing phase can detect 100% faults.
(iii) It is not practically feasible to perform unit testing on all modules, so a limited number of important logical paths should be selected to ensure proper working of the delivered software.
Since different projects are developed on varying platforms and in varying organization standards, the cost varies. The normalized fault removal cost for test techniques, i.e., unit, integration, system and field are presented in Table 4 with min, max and median values. The fault detection efficiency values for different test phases are taken from study by Jones [65]are summarized in Table 5. Wilde and Huitt [66] stated that more than fifty percent of

modules are usually very small in size; hence performing unit testing on these modules is not fruitful.

3.5.1. Estimated fault removal cost (E_{cost})

The estimated fault removal cost (E_{cost})is the sum of cost of unit testing, cost for integration test system test and the cost for field test. The number of faulty modules recognized by the predictor is the sum of true positive and false positive values. Hence, it is important to calculate testing and verification cost at the module level, which means that this cost is equal to the cost of unit testing ($Cost_{\text{unit}}$). Equation (1) shows the total unit testing cost.

$$Cost_{\text{unit}} = (TP + FP) * Cost_u \tag{1}$$

The fault removal cost for integration test ($Cost_{\text{integration}}$) is obtained as

$$Cost_{\text{integration}} = \delta_i * C_i * (FN + TP(1 - \delta_u)) \tag{2}$$

The left out faulty modules which are not predicted by integration testing are predicted by system test. Equation (3) gives the fault removal cost for system test

$$Cost_{\text{system}} = \delta_s * C_s * (1 - \delta_i) * (TP(1 - \delta_u) + FN) \tag{3}$$

For the left out faulty modules which were not predicted in system testing, field-testing is done. The fault removal cost for field test ($Cost_{\text{field}}$) is given by Eq. (4) as

$$Cost_{\text{field}} = (1 - \delta_s) * C_f * (1 - \delta_i)(TP(1 - \delta_u) + FN) \tag{4}$$

So, the value of the overall estimated fault removal cost can be determined by adding Eq. (1) to (4), as shown by Eq. (5)

$$E_{\text{cost}} = Cost_{\text{unit}} + Cost_{\text{integration}} + Cost_{\text{system}} + Cost_{\text{field}} \tag{5}$$

3.5.2. Estimated testing cost (T_{cost})

The steps followed to calculate estimated testing cost are:
The cost of unit testing on all the modules is given by Eq. (6)

$$Cost_{\text{unit}} = M_p * C_u * TM \tag{6}$$

The testing cost for faulty modules that are not detected during unit testing and may be detected in integration, system, and field testing are calculated as follows.

$$Cost_{\text{integration}} = \delta_i * C_i * (1 - \delta_u) * FM \tag{7}$$

$$Cost_{\text{system}} = \delta_s * C_s * (1 - \delta_i) * (1 - \delta_u) * FM \tag{8}$$

$$Cost_{\text{field}} = (1 - \delta_s) * (1 - \delta_i) * (1 - \delta_u) * FM \tag{9}$$

The overall value of estimated testing cost (T_{cost}) is given by adding the Eq. (6)) to (9), as represented by Eq. (10)

$$\begin{aligned} T_{\text{cost}} = (\{M_p * C_u * TM\} + \{\delta_i * C_{\text{i}} * (1 - \delta_u) * FM\} + \\ \{\delta_s * C_s * (1 - \delta_i) * (1 - \delta_u) * FM\} + \{(1 - \delta_s) * (1 - \delta_i) * (1 - \delta_u) * FM\} \end{aligned} \tag{10}$$

3.5.3. Normalized fault removal cost (NE_{cost})

The normalized fault removal cost is obtained as ratio of estimated fault removal cost to estimated testing cost, as shown by Eq. 11

$$NE_{\text{cost}} = \frac{E_{\text{cost}}}{T_{\text{cost}}} = \begin{cases} < 1 & \text{application of proposed fault prediction is useful} \\ \geq 1 & \text{application of testing methods is useful} \end{cases} \tag{11}$$

Where: E_{cost} and T_{cost} is the estimated fault removal cost of the software with and without using the fault prediction approach.

Table 4. Removal cost for test techniques (staff hours per defect)

Testing Type	Min	Max	Median
Unit (C_{u})	1.5	6	2.5
Integration (C_{i})	3.06	9.5	4.55
System (C_{s})	2.82	20	6.2
Field (C_{f})	3.9	66.6	27

Table 5. Fault identification efficiencies for different test phases

Testing Type	Min	Max	Median
Unit(δ_{u})	0.1	0.5	0.25
Integration(δ_{i})	0.25	0.60	0.45
System(δ_{s})	0.25	0.65	0.5

4. Experiment design

For conducting the experiments, we designed five scenarios, based on the research questions. In scenario1, we collected all the product metrics after data-processing and normalization. This is called Simple model. The detail of selected metrics is shown in Table 3. In scenario-2: the Advanced model-1 is constructed by using product metrics and one process metric, i.e., Product + NR metric. Similarly, in scenario-3: Advanced model-2 is formed by using Product + NDC metric, scenario-4 is constructed by using Advanced model-3 using Product + NML metric and in scenario-5 Advanced model-4 is built by using with Product + NDPV metric. All the designed models are tested on various project datasets repositories, i.e., Promise, Bug, and Jira using different classifiers such as DT, MLP, SVM, RT, NB and classifiers ensembles, as discussed in Section 3.3, respectively. The performance of various models Simple model; Advanced model-1; Advanced model-2; Advanced model-3, and Advanced model-4 are measured using accuracy, RMSE, ROC(AUC), and F-score.

The metrics used in the base classifiers are obtained after performing feature selection and feature ranking. N-fold cross-validation technique [51, 52] is used to evaluate the performance of the base classifiers, which makes use of both training and testing. Cross-validation technique splits the dataset into N parts each of which contains an equivalent number of samples in the dataset. While conducting the experiments algorithm is made to run

N times; and in each run, training is achieved through $(N-1)$ parts, and the testing is performed with the leftover part. N fold are usually selected as 10, 20, 30, 40, 50, 60, 70, 80, and 90. Authors tried with 10 fold for the cross-validation. This approach is carried out on different versions of datasets for different base classifiers.

To answer RQ2, i.e., to evaluate and compare the performance of various ensemble methods presented in Section 3, the library of the said algorithms was installed using the pip Python installer, e.g., (*sudo pip install xgboost*) to conduct the experiments. The algorithm packets used in the study are Bagging, AdaBoostM1 (which is the most popular version of boosting), and Voting [67]. Heterogeneous classifier ensembles applied the majority voting method, whereas the homogenous ones applied both bagging and boosting methods. For ensembles with boosting and bootstrap aggregating, the weak learners selected in the study are Decision Stump and REPTree, as these are widely used in literature studies [67–70]. In AdaBoosting, a training set is modified by repeatedly applying a basic learning device, i.e., classifier, under a pre-specified number of iterations. Initially, the training samples are equal in weight, and the first base classifier is trained to test the training set. Thus, at each iteration, a weight is assigned to each instance of the training set, and the weights of misclassified instances are increased so that their chances to be correctly predicted by the new models get increased. The adjusted training set trains the second basic classifier, and this process is repeated until a good learning device is obtained. During bootstrap aggregating, in the training phase, m data sets of the same size are extracted by performing sampling with replacement (bootstrap) from the training set. Therefore, for each data set, a model is trained using a weak classifier. For each instance, the multiple classifiers utilize a majority voting to obtain the classification result in the test phase. Ensembles are designed using voting works by constructing two or more sub-models. Each sub-model gives a prediction, which is pooled either by taking the mean or the mode of the predictions, permitting each sub-model to vote on the possible outcome. The final output is the class label that attains the maximum number of votes from the predictors. Otherwise, the input is rejected, and the classifier ensembles make no prediction. In our case, the base learners for ensemble design chosen are the four best classifiers. From the pool of four base classifiers, all sets of classifiers of size three were chosen to design ensembles committee. This meant that there were a total of four classifier ensembles. The various constituent combinations, so obtained are defined as: VOT-E1 (DT + MLP + RT), VOT-E2 (DT + MLP + SVM), VOT-E3 (MLP + RT + SVM), and VOT-E4 (DT + RT + SVM). The ensembles performance is measured using the same metrics as used for base classifiers discussed in Section 4. Also, to check whether the ensemble design improves the classification performance compared to individual machine learning classifiers, the comparison of the best ensemble, i.e., VOT-E2, is made with other base classifiers in terms of AUC(ROC) values.

To answer RQ3, i.e., whether there exist any statistically significant performance difference among the base classifiers and ensemble classifiers? Authors tested the following hypothesis using Friedmans tests and Wilcoxon signed rank tests [71].

H_0: There is no significant difference between base classifier performance and ensemble classifier performance.

To answer RQ4, i.e., cost sensitiveness of proposed ensembles, the normalized fault removal cost approach has been used as discussed in Section 3.5. Further, to evaluate the cost sensitiveness of the best ensemble classifier for the misclassification of faults, we predicted the fault removal cost of the best ensemble, i.e., VOT-E2 strategy, and compared its performance with the best base classifier, i.e., MLP.

5. Results and discussions

The section presents the experimental results and discussions to all the research questions. Results related to examining the predictive capability of advanced models are discussed in Section 5.1 followed by discussion on results based on ensemble design in Section 5.2. Section 5.3 discusses the results related to statistical difference among the base classifiers and ensemble classifiers and Section 5.4 discusses the results related to the cost sensitiveness of the proposed ensembles.

5.1. Results for predictive capability of advanced models

For examining the predictive capability of proposed advanced models, we evaluated the performance of simple model, advanced model-1, advanced model-2, advanced model-3 and advanced model-4 using various base classifiers. For the simple model, the values of accuracy so obtained are presented in Table 6 for all the datasets. Also, the results are

Table 6. Simple model accuracy with ten-fold for various classifiers

Projects	DT	DT [33]	MLP	MLP [33]	RT	RT [33]	NB	NB [33]	SVM
ant 1.4	77.53%	76.40%	77.99%	77.52%	75.45%	73.3%	67.97%	67.1%	73.23%
ant 1.5	93.88%	94.88%	94.93%	95.90%	90.88%	100	80.54%	80.45%	90.98%
ant 1.6	73.79%	72.93%	78.89%	73.21%	71.11%	69.76%	59.50%	58.4%	70.43%
ant 1.7	77.72%	75.83%	81.02%	75.97%	76.65%	74.56%	61.98%	61.07%	73.69%
camel 1.2	64.30%	64.30%	68.87%	64.43%	66.60%	65.65%	63.99%	62.7%	65.95%
camel 1.4	82.45%	82.45%	87.76%	87.02%	76.43%	79.85%	79.84%	79.9%	79.41%
camel 1.6	79.68%	79.66%	81.05%	80.51%	80.01%	76.70%	74.66%	74.65%	78.56%
jedit 4.0	74.67%	74.12%	74.67%	74.78%	70.00%	73.1%	67.00%	67.3%	73.89%
jedit 4.1	75.32%	75.33%	75.99%	75.85%	69.95%	69%	69.00%	69.5%	71.87%
jedit 4.2	86.10%	86.10%	87.93%	85.98%	81.90%	80%	74.00%	73.3%	81.14%
jedit 4.3	95.12%	95.12%	96.13%	95.73%	89.95%	95.5%	80.00%	80.1%	89.44%
Lucene 2.2	66.98%		69.77%		68.42%		55.00%		67.87%
Lucene 2.4	69.04%		73.27%		70.27%		78.92%		65.63%
synapse 1.1	72.52%	72.53%	72.87%	72.07%	66.12%	66.8%	69.98%	69.8%	64.98%
synapse 1.2	66.40%	66.1%	66.63%	65.62%	64.92%	66.5%	65.89%	66%	68.04%
xalan 2.5	51.43%	51.42%	54.76%	51.76%	48.89%	51%	54.00%	54.87%	50.98%
xalan 2.6	53.89%	53.9%	60.43%	62.43%	53.34%	53%	61.00%	60.09%	50.76%
xalan 2.7	71.29%	71%	71.24%	70%	62.14%	64.2%	55.00%	57%	58.42%
xerecs 1.2	82.41%	83.40%	83.41%	83.4%	79.41%	80.21%	73.45%	73.14%	78.34%
xerecs 1.3	84.55%	84.54%	84.59%	84.5%	82.98%	83%	76.99%	77%	80.88%
xerecs 1.4	57.36%	28.84%	61.52%	61.12%	90.01%	94%	78.92%	78.5%	91.91%
Equinox	74.07%		73.15%		71.91%		71.60%		73.46%
Eclipse-JDT	82.65%		84.35%		81.44%		83.95%		85.06%
Eclipse-PDE	89.3%		85.64%		79.89%		82.77%		84.05%
MYLYN	84.91%		86.36%		81.68%		83.94%		86.84%
ActiveMQ 5.0.0	86.46%		88.09%		82.95%		85.03%		85.56%
derby-10.5.1.1	85.80%		87.88%		83.25%		83.84%		84.02%
Groovy-1	91.47%		91.01%		92.73%		86.84%		91.59%
Hbase-0.94.0	82.43%		88.35%		77.71%		80.07%		81.11%
Hive-0.9.0	80.01%		86.81%		80.15%		82.52%		81.64%
Jruby-1.1	85.49%		90.02%		88.46%		89.09%		84.95%
Wicket-1	95.03%		95.98%		93.12%		93.42%		83.55%

compared with [33] for classifiers DT, MLP, RT and NB classifiers for the projects from Promise data set. Similarly, for all models, the values of accuracy are obtained. Table 7 shows the average accuracies of all the base classifiers for simple model, advanced model-1, advanced model-2, advanced model-3, and advanced model-4 with the standard deviation values after ten executions of the classifiers for all the datasets.

Table 7. Average accuracy for various models with standard deviation on different classifiers

	Projects	DT	MLP	RT	NB	SVM
PROMISE	Simple	$74 \pm 0.67\%$	$75 \pm 0.99\%$	$71 \pm 0.11\%$	$60 \pm 0.01\%$	$71 \pm 0.72\%$
	Advanced model-1	$81 \pm 0.09\%$	$80 \pm 0.74\%$	$75 \pm 0.37\%$	$73 \pm 0.07\%$	$76 \pm 0.94\%$
	Advanced model-2	$87 \pm 0.01\%$	$87 \pm 0.08\%$	$82 \pm 0.06\%$	$81 \pm 0.30\%$	$76 \pm 0.45\%$
	Advanced model-3	$83 \pm 0.03\%$	$85 \pm 0.16\%$	$79 \pm 0.08\%$	$72 \pm 0.42\%$	$74 \pm 0.36\%$
	Advanced model-4	$77 \pm 0.04\%$	$79 \pm 0.18\%$	$77 \pm 0.05\%$	$73 \pm 0.55\%$	$72 \pm 0.16\%$
BUG Dataset	Simple	$82 \pm 0.28\%$	$82 \pm 0.66\%$	$79 \pm 0.18\%$	$79 \pm 0.86\%$	$82 \pm 0.76\%$
	Advanced model-1	$83 \pm 0.90\%$	$84 \pm 0.70\%$	$80 \pm 0.66\%$	$80 \pm 0.85\%$	$83 \pm 0.59\%$
	Advanced model-2	$85 \pm 0.63\%$	$86 \pm 0.92\%$	$81 \pm 0.95\%$	$80 \pm 0.19\%$	$85 \pm 0.27\%$
	Advanced model-3	$84 \pm 0.14\%$	$84 \pm 0.36\%$	$81 \pm 0.04\%$	$79 \pm 0.59\%$	$84 \pm 0.38\%$
	Advanced model-4	$81 \pm 0.96\%$	$82 \pm 0.76\%$	$79 \pm 0.73\%$	$79 \pm 0.47\%$	$82 \pm 0.96\%$
JIRA	Simple	$86 \pm 0.53\%$	$89 \pm 0.59\%$	$85 \pm 0.34\%$	$85 \pm 0.69\%$	$84 \pm 0.54\%$
	Advanced model-1	$88 \pm 0.15\%$	$91 \pm 0.07\%$	$87 \pm 0.39\%$	$87 \pm 0.64\%$	$85 \pm 0.45\%$
	Advanced model-2	$89 \pm 0.76\%$	$91 \pm 0.85\%$	$87 \pm 0.32\%$	$88 \pm 0.32\%$	$86 \pm 0.18\%$
	Advanced model-3	$88 \pm 0.75\%$	$90 \pm 0.65\%$	$87 \pm 0.44\%$	$87 \pm 0.72\%$	$86 \pm 0.05\%$
	Advanced model-4	$86 \pm 0.89\%$	$89 \pm 0.72\%$	$85 \pm 0.97\%$	$86 \pm 0.30\%$	$84 \pm 0.81\%$

For promise dataset the average accuracy for the simple model in MLP is 75%, for Advanced model-1 is 80%, Advanced model-2 is 87%, Advanced model-3 is 85%, and Advanced model-4 is 79%. It is clear from the bar graph that average accuracy for MLP is higher for Advanced model-2, than for Advanced model-3, Advanced model-1 and simple model. From the bar graph Figure 2a it is observed that average accuracy is behaving well with advanced models as compared to simple models. The average accuracy for the DTs simple model is 74%, for advanced model-1 is 81%, advanced model-2 is 87%, advanced model-3 is 83%, and advanced model-4 is 77%. So, it is clear from the bar graph that the average accuracy for DT is higher for advanced model-2 then for advanced model-3, advanced model-1, and simple model. The average accuracy results achieved for all projects from Promise data set by Decision tree, Random Tree, Naive Bayes and Multilevel Perceptron classifiers for advanced models is 82.4%, 78.25%, 74.75% and 81.75%

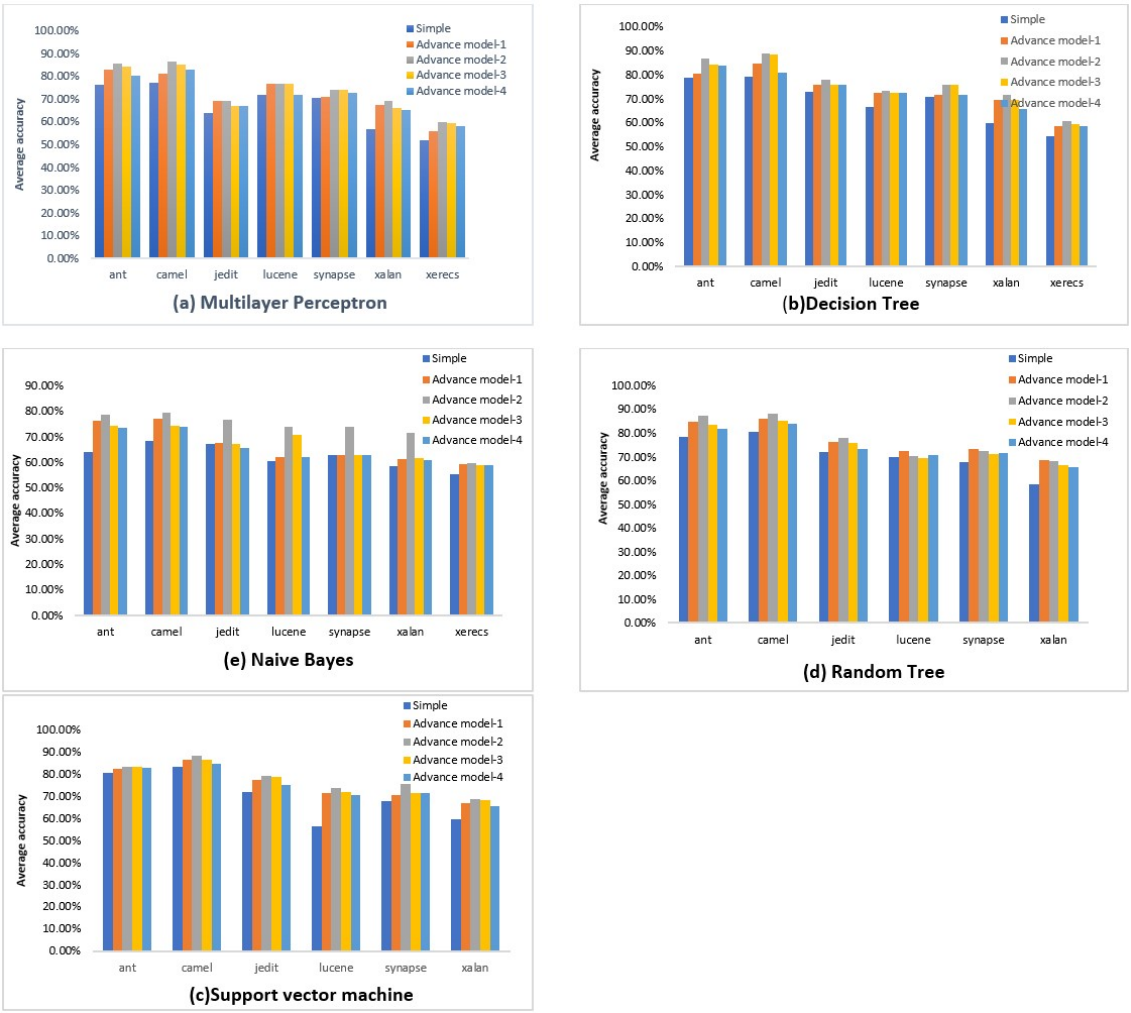


Figure 2. Average Accuracy for MLP, DT, SVM, RT, and NB using PROMISE data

as compared to 64.58%, 63.83%, 61.17% and 64.54%, reported by Juneja [33]. This shows that advanced models performed better.

As shown in the graph Figure 2b, the average accuracy is behaving well with advanced models compared to a simple model. The average accuracy is high in Camel projects and low for Xerces projects. The average accuracies for various classifiers like SVM, RT, and NB are calculated as shown in Figure 2c to e. The results show that in the advanced model-2, average accuracy for SVM, RT, and NB is 76%, 82%, and 81%, respectively. The model is behaving significantly good as the average accuracy is higher than 0.5. So, from Table 7 and Figures 2a–e, it is clear that the advanced model-2 (Product + NDC metric) is performing better as compared to other models.

For the projects from Bug repository, the results of average accuracy in the case of advanced model-2, for MLP is 86%, for DT is 85% , for SVM is 85%, for RT is 81%and for NB is 80%, respectively. The model is behaving significantly well as average accuracy is higher than 0.5. So, it is clear that the advanced model-2 (Product + NDC metric) performs better than other models.

For the projects from Jira repository, the results of average accuracy for advanced model-2, for MLP is 91%, for DT is 89% , for SVM is 86%, for RT is 87%, NB is 88%,

respectively. The model is behaving significantly well as average accuracy is higher than 0.5. So, it is clear that the Advance model-2 (Product + NDC metric) performs better than other models for Jira projects.

After presenting the accuracy-based evaluation, further analysis is conducted to examine the root mean square error for Promise, Bug and Jira dataset repositories. The average RMSE values for the Promise dataset in proposed advanced model-1, advanced model-2, advanced model-3 and advanced model-4 is low as compared to the simple model. Table 8 presents the details of the average RMSE with standard deviation. The Advanced model-2 has the error ratio 0.13, 0.12, 0.18, 0.19, and 0.16 for DT, MLP, RT, NB and SVM which is significantly lower than the simple model.

Table 8. Average RMSE for various models with standard deviation on different classifiers

	Projects	DT	MLP	RT	NB	SVM
PROMISE	Simple	0.21±0.0057	0.19 ± 0.006	0.21±0.0101	0.20±0.0059	0.17±0.0089
	Advanced model-1	0.16±0.0067	0.14 ± 0.007	0.19±0.0090	0.18 ± 0.006	0.17±0.0067
	Advanced model-2	0.13 ± 0.007	0.12 ± 0.005	0.18±0.0398	0.19 ± 0.046	0.16 ± 0.009
	Advanced model-3	0.15 ± 0.007	0.14 ± 0.008	0.19±0.0256	0.18 ± 0.025	0.15 ± 0.011
	Advanced model-4	0.16 ± 0.006	0.16 ± 0.005	0.19±0.0006	0.19 ± 0.005	0.17 ± 0.012
BUG Dataset	Simple	0.22 ± 0.084	0.20±0.0127	0.21±0.0317	0.21±0.0997	0.17 ± 0.038
	Advanced model-1	0.21 ± 0.067	0.18 ± 0.055	0.19±0.0672	0.21±0.0302	0.15 ± 0.037
	Advanced model-2	0.18 ± 0.09	0.15±0.0545	0.16±0.0995	0.17±0.0997	0.13 ± 0.035
	Advanced model-3	0.18 ± 0.075	0.16±0.0902	0.19±0.0215	0.19±0.0615	0.13 ± 0.068
	Advanced model-4	0.20±0.0387	0.18±0.0857	0.20±0.0727	0.20±0.0382	0.16 ± 0.052
JIRA	Simple	0.20 ± 0.099	0.15±0.0395	0.19±0.0265	0.18±0.0951	0.17 ± 0.048
	Advanced model-1	0.19 ± 0.088	0.13±0.0757	0.17±0.0742	0.17 ± 0.048	0.15 ± 0.097
	Advanced model-2	0.18 ± 0.037	0.12±0.0108	0.15 ± 0.085	0.15 ± 0.085	0.14 ± 0.019
	Advanced model-3	0.19 ± 0.014	0.13±0.0982	0.16±0.0334	0.16 ± 0.095	0.15 ± 0.067
	Advanced model-4	0.20 ± 0.038	0.14±0.0721	0.18±0.0295	0.17±0.0938	0.16±0.0308

For Bug dataset the average RMSE values for proposed advanced model-2, and advanced model-3 are significantly lower than the advanced model-4, Advance model-1 and simple model. The average RMSE values for DT, MLP, RT, NB and SVM are 0.18, 0.15, 0.16, 0.17 and 0.13, respectively for advanced model-2.

For Jira dataset the average RMSE values for proposed advance model-2 are significantly lower than the advanced model-3, advanced model-4, advanced model-1 and simple model. The RMSE values for advanced model-1 and advanced model-3 are almost similar for DT,

MLP and SVM. The average RMSE values for DT, MLP, RT, NB and SVM are 0.18, 0.12, 0.15, 0.15 and 0.14, respectively for advanced model-2.

AUC is the other performance measure considered in the study. Greater the AUC value better is the model performance [7, 20]. The ROC curves provide the trade-off between the TPR and FPR for a predictive model using different probability thresholds. These measures are good performance indicator for the classification of an imbalanced dataset.

Table 9 shows the aggregative AUC of all the base classifiers for simple, advanced model-1, advanced model-2, advanced model-3 and advanced model-4 with the standard deviation values after ten executions of the classifiers. The aggregative average AUC for Promise dataset achieved in advanced model-2 are 76%, 79%, 70%, 65% and 75% for DT, MLP, RT, NB and SVM respectively. As evident from literature studies [7] that the AUC value lying between 0.7 and 1 is considered significantly high and the accuracy value lying between 0.6 and 0.7 is considered significantly good. It is evident from the Table 9 that the advanced model-2 achieves and maintains high accuracy with respect to all classifiers. The advanced model-3 is followed by advanced model-1and advanced model-4.

For the Bug dataset the average ROC values for the advanced model-2 and advanced model-3 are significantly higher and for advanced model-1and advanced model-4 the average ROC values are good as compared to simple model as shown in Table 9. The average accuracy of advanced model-2 for DT, MLP, RT, NB, and SVM are 79%, 83%, 74%,

Table 9. Average ROC(AUC) for various Models with standard deviation on different classifiers

	Projects	DT	MLP	RT	NB	SVM
PROMISE	Simple	71 ± 0.009	74 ± 0.006	63 ± 0.014	61 ± 0.02	70 ± 0.014
	Advanced model-1	75 ± 0.001	78 ± 0.006	67 ± 0.017	63 ± 0.09	74 ± 0.004
	Advanced model-2	76 ± 0.002	79 ± 0.008	70 ± 0.001	65 ± 0.98	75 ± 0.009
	Advanced model-3	77 ± 0.012	78 ± 0.012	69 ± 0.019	62 ± 0.06	73 ± 0.009
	Advanced model-4	70 ± 0.013	77 ± 0.002	65 ± 0.016	60 ± 0.07	69 ± 0.008
BUG Dataset	Simple	73 ± 0.09025	77 ± 0.5775	66 ± 0.05	63 ± 0.955	74 ± 0.2425
	Advanced model-1	76 ± 0.058	81 ± 0.04	71 ± 0.37	67 ± 0.845	77 ± 0.0725
	Advanced model-2	79 ± 0.03375	83 ± 0.14	74 ± 0.7375	70 ± 0.5325	78 ± 0.1825
	Advanced model-3	78 ± 0.069	81 ± 0.0785	73 ± 0.515	72 ± 0.4375	76 ± 0.7725
	Advanced model-4	74 ± 0.079	77 ± 0.0665	67 ± 0.2975	66 ± 0.2475	75 ± 0.2325
JIRA	Simple	76 ± 0.0628	82 ± 0.11	72 ± 0.62857	81 ± 0.181	74 ± 0.7142
	Advanced model-1	78 ± 0.0732	83 ± 0.051571	74 ± 0.39429	83 ± 0.11	76 ± 0.5485
	Advanced model-2	79 ± 0.01429	84 ± 0.046	76 ± 0.27143	83 ± 0.82	77 ± 0.9171
	Advanced model-3	78 ± 0.0986	83 ± 0.0351	74 ± 0.92429	82 ± 0.75571	76 ± 0.8457
	Advanced model-4	77 ± 0.06	82 ± 0.0514	73 ± 0.88571	81 ± 0.97143	75 ± 0.7271

70%, and 78%, respectively. It shows that the proposed advanced models has performed impressively well for inter project fault prediction.

We also calculated the performance of simple and advanced models in terms of F-score. F-score values can range from (0–1) and accepted to be better as it approaches to one [41, 60]. Table 10 presents the average F-score for simple and advanced models for Promise, Bug, and Jira datasets on various classifiers. It is observed from Table 10; in the Promise dataset the advanced model-2 for MLP classifier has the highest $F1$-score value, i.e., 0.83 as compared to the advanced model-3 (0.82), advanced model-2 (0.80), and the simple model (0.79). The F-score value for DT in simple model (0.76), advanced model-1 (0.77), advanced model-2 (0.81), the advanced model-3 (0.80) and the advanced model-4 is (0.76). It is observed from the table that the advanced model-2 is behaving significantly well in all the classifiers. The MLP is behaving well than DT, DT is better than RT, RT is better than SVM, and SVM is better than NB.

Similarly, for the Bug dataset the advanced model-2 and the advanced model-3 have almost similar values for MLP and NB classifiers. The advanced model-2 having F-score MLP (0.88), DT (0.84), RT (0.84), NB (0.81) and SVM (0.85). For the simple model the F-score values are 0.82, 0.86, 0.81, 0.79, 0.78 for DT, MLP, RT, NB, and SVM classifiers, respectively. Similarly, for advanced model-1 the DT, MLP, RT, NB and SVM the values for F-score are 0.82, 0.87, 0.82, 0.80 and 0.82, respectively and; for the advanced model-3

Table 10. Average F-score for various models with standard deviation on different classifiers

	Projects	**DT**	**MLP**	**RT**	**NB**	**SVM**
PROMISE	Simple	0.76±0.012	0.79±0.012	0.79 ± 0.03	0.73 ± 0.06	0.75 ± 0.05
	Advanced model-1	0.77 ± 0.04	0.80 ± 0.18	0.77 ± 0.16	0.74±0.017	0.76 ± 0.04
	Advanced model-2	0.81 ± 0.05	0.83 ± 0.05	0.79 ± 0.07	0.78 ± 0.09	0.79 ± 0.09
	Advanced model-3	0.80 ± 0.03	0.82 ± 0.16	0.78±0.078	0.79 ± 0.02	0.78 ± 0.06
	Advanced model-4	0.76 ± 0.05	0.79 ± 0.06	0.75 ± 0.04	0.74±0.053	0.74 ± 0.08
BUG Dataset	Simple	0.82 ± 0.13	0.86 ± 0.07	0.81±0.024	0.79 ± 0.01	0.78 ± 0.05
	Advanced model-1	0.82±0.045	0.87 ± 0.1	0.82 ± 0.05	0.80±0.023	0.82 ± 0.09
	Advanced model-2	0.84±0.063	0.88 ± 0.12	0.84±0.063	0.81 ± 0.05	0.85 ± 0.02
	Advanced model-3	0.83 ± 0.05	0.88 ± 0.05	0.83±0.045	0.82 ± 0.06	0.85 ± 0.07
	Advanced model-4	0.81 ± 0.09	0.87 ± 0.03	0.81±0.098	0.79 ± 0.07	0.79 ± 0.04
JIRA	Simple	0.81±0.045	0.83±0.061	0.79 ± 0.07	0.79 ± 0.09	0.79 ± 0.06
	Advanced model-1	0.82 ± 0.08	0.84 ± 0.07	0.80±0.023	0.79 ± 0.06	0.80±0.078
	Advanced model-2	0.83±0.045	0.89 ± 0.01	0.82 ± 0.06	0.80 ± 0.05	0.83±0.087
	Advanced model-3	0.83±0.063	0.88 ± 0.08	0.81 ± 0.05	0.80 ± 0.16	0.83 ± 0.04
	Advanced model-4	0.81±0.092	0.78±0.065	0.79 ± 0.07	0.76 ± 0.01	0.81±0.045

and for the advanced model-4 the F-score values are 0.83, 0.88, 0.84, 0.81, 0.85 and 0.81, 0.87, 0.81, 0.79, 0.79, respectively.

For the Jira dataset, the average F-score has the highest values for advanced model-2 compared to advanced model-1, advanced model-3, advanced model-4, and simple model. The advanced model-2 has F-score values for MLP (0.89), DT (0.83), RT (0.82), NB (0.80) and SVM (0.83), respectively. The advanced model-2 with MLP has the highest F-score values as compared to other classifiers. Thus, it is observed that MLP performs best with values 0.83, 0.88, and 0.89 with advanced model-2 for Promise, Bug, and Jira datasets. Comparing the overall performance, the advanced model-2 with MLP performs best followed by advanced model-3.

So, it is concluded for the RQ1 that Advanced model-2 with MLP classifier having high predictive capability as compared to other models and classifiers. The advanced model-2 with MLP has high accuracy, ROC (AUC) and F-score, and small RMSE values.

5.2. Experiment results based on ensemble design

In this section, we have summarised the results and discussed the main findings of various ensemble methods. Table 11 presents the results for average Accuracy, average RMSE, average ROC(AUC), and average F-score. Diagrammatically, the results of the performance measures are shown with the help of Box plots. Figures 3a–l shows the box plot analysis results for proposed ensembles with respect to the average Accuracy, average AUC(ROC), average F-score, and average RMSE. The different regions of box plots in the figures present the maximum, median minimum, first quartile, and third quartile values of the dataset. The middle line of the box indicates the median value of the dataset. With respect to average accuracy, average AUC(ROC), average F-score, and average RMSE, the proposed ensembles VOT-E1 and VOT-E2 have high median value and high maximum value followed by AdaBoost and Random Forest with features based on Product + NDC

Table 11. Ensemble results for average Accuracy, average RMSE, average ROC(AUC), and F-score

Datasets	Bag	ADA	RF	VOT-E1	VOT-E2	VOT-E3	VOT-E4
Average accuracy							
PROMISE Dataset	87.35%	88.07%	87.41%	89.22%	88.14%	86.30%	82.94%
BUG Dataset	87.69%	88.02%	86.79%	89.15%	88.70%	87.37%	85.08%
JIRA Dataset	90.19%	90.26%	88.89%	91.25%	90.47%	89.15%	88.45%
Average ROC (AUC)							
PROMISE Dataset	78.51%	77.81%	79.32%	80.11%	83.92%	79.58%	77.17%
BUG Dataset	77.09%	79.23%	80.07%	80.53%	81.64%	79.25%	78.97%
JIRA Dataset	78.18%	79.01%	78.19%	81.60%	84.05%	79.82%	78.19%
Average F1 score							
PROMISE Dataset	77.91%	80.68%	78.64%	80.37%	83.29%	82.01%	78.54%
BUG Dataset	78.62%	81.12%	78.86%	85.09%	87.34%	84.10%	78.72%
JIRA Dataset	76.50%	80.09%	76.94%	85.26%	87.24%	81.06%	79.80%
Average RMSE							
PROMISE Dataset	0.1845	0.1785	0.1966	0.1693	0.1724	0.1843	0.199
BUG Dataset	0.1855	0.1765	0.20195	0.181615	0.18525	0.2036	0.2014
JIRA Dataset	0.170	0.1785	0.1966	0.1601	0.1679	0.1748	0.1992

Figure 3. Box plots for Ensemble Results for average accuracy, average RMSE, average ROC(AUC) and average F-score

metric data set for projects not only from Promise repository but also from Bug and Jira dataset repositories, which validates the results and makes the approach more reliable. From Figure 3a, it is observed that projects from Promise repository, VOT-E1 have the highest accuracy, i.e., 0.8922, and high median value, i.e., 0.8906.

Similarly, the box plot for VOT-E2 shows the median value of 0.8797 and a maximum value of 0.8814. Similar trends are observed for projects from Bug and Jira repositories, as

shown in Figure 3b and Figure 3c, respectively. Thus, VOT-E1 performs better in terms of accuracy as compared to other ensembles.

The average AUC(ROC) values are shown in Figures 3d–f. From Figure 3d, it is observed that for projects from Promise repository, VOT-E2 have the highest AUC(ROC), i.e., 0.8392, and high median value, i.e., 0. 8402 followed by VOT-E1 with median value 0.7972 and a maximum value 0.8011. The values of ensembles constructed with Boosting and Bagging are 0.7851 and 0.7781, respectively. Similar trends are observed for Bug and Jira repositories projects as shown in Figures 3e and 3f. VOT-E2 performs better in terms of accuracy as compared to other ensembles. We also found that the difference in AUC(ROC) for the best performing ensembles, i.e., VOT-E1 and VOT-E2 is very minimal, ranging from 1% to 3%. Moreover, the shape of the box-plot for the projects is nipped, which signifies that the performance of the ensemble method is the same among all the releases of different projects from Promise, Jira and Bug repositories.

The performance for the average F-score is shown in Figure 3g–i, respectively. From Figure 3g, it is observed that for projects from Promise repository, VOT-E2 have the highest F-score, i.e., 0.8329 followed by VOT-E3 0.8201, AdaBoost 0.8068, and Random Forest 0.7864. The box plot for VOT-E3 shows the median value of 0.8134 and the maximum value of 0.8201. Similar tendency is observed for F-score values for projects from Bug and Jira repositories as shown in Figures 3h and 3i, where VOT-E2 has the highest F-score value followed by VOT-E1. The overall median of F-score ranges between 85% (VOT-E2) and 77.71% (Bagging). So, VOT-E2 performs better in terms of F-score as compared to other ensembles.

The average root mean square error values are shown in Figure 3j–l. From Figure 3j, it is observed that for the projects from Promise repository, the average RMSE is least for the ensembles VOT-E1, i.e., 0.1693 and VOT-E2, i.e., 0.1724 and high for VOT-E4 0.199, AdaBoost 0.1785, and Random Forest 0.1966. The VOT-E1 and VOT-E2 performed better, and it has the narrow box compared to the ensembles with bagging, adaboost, and RF, respectively. The overall difference of RMSE ranges between 0.1703 (VOT- E1) and 0.1998 (VOT- E4).

5.3. Results for examining the performance difference among the base classifiers and ensemble classifiers

The results shown in Table 12 in bold indicate the best performance. The proposed VOT-E2 produced the best results with advanced model-2, advanced model-3 and advanced model-1, while it gave second best results for advanced model-4 and simple model datasets. The values of ROC(AUC) for base classifiers were taken from Table 9. For comparison of classifiers, the average ranks were computed. The ranks for each classifier for each dataset were ascertained and later on summed up to get average ranks by dividing the average values by the number of datasets. The lower the average ranking value; the better is the performance of the model. The proposed ensemble-1 has a lower average rank of 1.2, followed by the classifier MLP with a rank 1.6. All other classifiers have ranks between 1.2 and 5.8 as shown in Table 13.

Thus, based upon the results, we can say that ensemble learning (i.e., AdaBoosting, Bagging, Random Forests, and Voting) works best as compared to base predictors. The ensemble algorithms combine signals from base classifiers in the committee to produce an enhanced fault prediction algorithm. While experimenting, we have noticed that ensemble techniques performed better among all the advanced models. With respect to average

Table 12. Comparisons of different classifiers in terms of ROC(AUC)

Classifiers	Simple	Advanced model-1	Advanced model-2	Advanced model-3	Advanced model-4	Avg Rank
DT	0.7338	0.7637	0.7801	0.7772	0.7371	3
MLP	0.7789	**0.8069**	**0.8206**	**0.8071**	**0.7871**	1.8
SVM	0.7029	0.7587	0.7703	0.7554	0.7332	4.0
RT	0.6723	0.7092	0.7366	0.7248	0.6873	5.8
NB	0.6871	0.7135	**0.740**	0.7241	0.6942	5.2
VOT-E2	**0.7801**	**0.8182**	**0.8320**	**0.8156**	0.7780	1.2

Table 13. Friedman test comparison

	J (base classifiers)	Pair wise differences
VOT- E2	DT	1.6 ($P < 0.01$)
	MLP	0.4
	SVM	2.6 ($P < 0.001$)
	RT	3.5 ($P < 0.001$)
	NB	4.6 ($P < 0.001$)

accuracy, average AUC(ROC), average F-score, and average RMSE, the proposed ensembles VOT-E1 and VOT-E2 have high median value and high maximum value followed by AdaBoost and Random Forest with features based on (Product + NDC metric data set) for projects not only from Promise repository but also from Bug and Jira dataset repositories, which validates the results and makes the approach more reliable. In general, the ensemble methods show an overall median of F1 score ranging between 76.50% and 87.34% and the ROC (AUC) between 77.09% and 84.05%. Base classifiers instead, reach an overall average F-score ranging between 73% (simple model) and 83% (Advanced model-2) for Promise data set and the ROC(AUC) between 60% (Advanced model-4) and 79% (Advanced model-2). Thus, we can say that the ensemble design enhances the strengths of multiple predictors and supplements to state of art in fault prediction problem [35, 72].

Furthermore, to examine whether the measured average ranks are significantly different from the mean rank 3.5, the Friedman test has been applied. The results of the test show below the significance level ($p < 0.01$), which means that at least two of the predictors are significantly different from each other. When the scores differ significantly, the researchers in the literature recommended follow-up pair-wise comparisons [68, 70]. For pair-wise comparisons, Wilcoxon signed ranks test had been applied. The results of pair-wise comparisons are presented in Table 13. It is observed that the performance of ensemble classifier is considerably dissimilar than other classifiers, apart from the MLP based classifier. Thus, the null hypothesis is rejected, which states that there is no significant difference between base classifiers performance and ensemble classifier performance. The results of Friedmans tests and Wilcoxon signed rank tests illustrate that the ensemble method exhibits statistically significant performance differences.

5.4. Results for examining the cost sensitiveness

Table 14 presents the predicted values of estimated fault removal cost (E_{cost}) for various projects for both the best ensemble and the best base classifier. The unit, integration, system, and field-testing values are used to obtain estimated values for fault removal cost

using Equations (1) to (4) presented in Section 3.5.1. Table 15 shows the estimated testing cost (T_{cost}) values for various projects. These cost values are obtained using equations (6) to (9) presented in Section 3.5.1. The values of fault removal cost (staff hours per defect) for test techniques are shown in Table 4, and values of testing phase efficiencies are presented in Table 5 in Section 3.5. Further, normalized cost values (N_{cost}) of VOT-E1 ensemble and best base classifier for projects from datasets (Promise, Bug and Jira) are obtained using Equation (11). The values so obtained are presented in Table 16. The

Table 14. Estimated Fault Removal cost (E_{cost}) for best base classifier and best ensemble classifier

Projects	Best base classifier (MLP)			VOT-E2 ensemble classifier		
	Min	Max	Median	Min	Max	Median
ant	1042	3862	1852	935	3712	2320
camel	534	2144	1167	654	2356	1367
Jedit	102	147	276	69	155	287
lucene	854	2393	3000	844	2382	2976
synapse	400	1018	1650	444	968	1601
xalan	3506	9803	11653	3498	9800	11650
xerecs	777	3912	1821	662	2461	1641
Equinox	596.06	2296.8	1502.42	562	2283.43	1407.7
Eclipse-JDT	921.753	4037.02	2419.48	918.45	4041.12	2389.04
Eclipse-PDE	1726.65	6565.25	3711.75	1710.12	6500.98	3645
MYLYN	2124.93	7548.55	4361.03	2023.34	7481.16	4243.03
ActiveMQ 5.0.0	1390.19	6510.64	3776.14	1390	6511	3768
derby-10.5.1.1	2041.09	9432.96	5607.88	2100	9456	5704
Groovy1_6_BETA_1	336.57	1309.85	857.90	321.34	1296	823
Hbase-0.94.0	1039.84	4907.87	2835.09	1109	4987	2965
Hive-0.9.0	1286.61	5661.904	3392.029	1261	5673	3753
Jruby-1.1	376.61	1547.42	962.66	376	1654	997
Wicket-1.3.0-beta2	1129.77	5998.59	3519.73	1127	5974	3678

Table 15. Estimated Testing cost (T_{cost}) for various projects

Projects from PROMISE repository										
	Unit	Integ.	System	Ant	Camel	Jedit	Lucene	Synapse	Xalan	Xerecs
Min	0.1	0.25	0.25	1079.78	1313.84	403.53	892.17	461.93	3500.35	1812.63
Max	0.5	0.6	0.65	3913.59	4796.06	1587.23	3072.74	1637.63	11807.5	6128.94
Median	0.25	0.45	0.5	2322.85	2782.28	707.21	2126.17	1040.95	8664.3	4398.43
Projects from BUG repository										
	Unit	Integ.	System	Equinox	Eclipse-JDT	Eclipse-PDE	MYLYN			
Min	0.1	0.25	0.25	647.89	1394.33	1778.74	2165.49			
Max	0.5	0.6	0.65	2276.44	5074.07	6604.41	8063.44			
Median	0.25	0.45	0.5	1486.42	2973.17	3623.32	4381.36			
Projects from Zira repository										
	Unit	Integ.	System	ActiveMQ	derby	Groovy1	Hbase	Hive	Jruby	*Wicket*
Min	0.1	0.25	0.25	2332.65	3230.89	835.46	1478.49	1950.26	821.32	1730.28
Max	0.5	0.6	0.65	8614.81	11987.9	3170.84	5381.41	7109.69	3072.74	6603.56
Median	0.25	0.45	0.5	4811.25	6591.98	1613.06	3151.27	4142.42	1643.08	3293.55

Table 16. Normalized fault removal cost (NE_{cost})

Projects	Best base classifier (MLP)			VOT-E2 ensemble classifier		
	Min	Max	Median	Min	Max	Median
ant	0.96	0.98	0.79	0.86	0.84	0.99
camel	0.40	0.44	0.42	0.49	0.50	0.49
jedit	0.25	0.21	0.17	0.17	0.21	0.18
lucene	0.95	1.12	0.97	0.94	1.12	0.97
synapse	0.87	0.98	1.0	0.96	0.93	0.98
xalan	1.0	1.13	0.98	0.99	1.13	0.98
xerecs	0.72	0.99	0.78	0.61	0.62	0.70
Equinox	0.75	0.82	0.79	0.86	1.00	0.94
Eclipse-JDT	0.66	0.79	0.81	0.65	0.79	0.80
Eclipse-PDE	0.56	0.60	0.71	0.96	0.98	1.0
MYLYN	0.62	0.72	0.83	0.93	0.92	0.96
ActiveMQ 5.0.0	0.59	0.75	0.78	0.59	0.75	0.78
derby-10.5.1.1	0.63	0.78	0.85	0.64	0.78	0.86
Groovy-1_6_BETA_1	0.40	0.41	0.53	0.38	0.40	0.51
Hbase-0.94.0	0.70	0.91	0.89	0.75	0.92	0.94
Hive-0.9.0	0.65	0.79	0.81	0.64	0.79	0.90
Jruby-1.1	0.45	0.50	0.58	0.45	0.53	0.60
Wicket-1.3.0beta2	0.65	0.90	1.06	0.65	0.90	1.11

values > 1.0 show that the proposed best ensemble, i.e., VOT-E1 is cost-effective. It entails that if the results of fault prediction are used with software testing, then overall testing cost and effort can be saved. At the same time, values greater than 1.0 demonstrate that the results of fault prediction do not help save overall testing cost and effort, and thus, it is suggested not to use fault prediction models in such cases. From the results presented in Table 16, it can be observed that for almost all projects, i.e., Ant, Camel, Jedit, Synapse, Xerecs, Equinox, Eclipse JDT, EclipsePDE, MYLYN, ActiveMQ 5.0.0, derby 10.5.1.1, Groovy-1_6_BETA1, Hbase-0.94.0, Hive-0.9.0 and Jruby-1.1 from the Promise, Bug and Jira repositories, N_{cost} values are lower or equal to the threshold value, i.e., 1.0 for both the proposed ensemble VOT-E2 and best base classifier MLP except in few cases, i.e., Xalan, Lucene and Wicket-1.3.0beta2 the normalized cost values are more than threshold value. Therefore, as observed from the results, it may not be beneficial to make use of SFP based on the suggested best ensemble and best base classifier. Thus, it is advisable to test all the modules at the unit level in place of using predictor for defect prediction for such projects. For all other datasets, the values of N_{cost} are lower than the threshold value, i.e., 1 and thus it is advantageous to utilize fault prediction approaches proposed in the study.

Table 17 presents the summary of research questions. Also, comparison of few related studies in literature with the proposed study is provided in Table 18.

6. Threats to validity

The section presents discussion on possible validity threats of the work presented in the paper along with possible measures how we mitigated them.

Construct validity: These types of threats are concerned with the relationship between theory and observations. In this work, we built advanced models for defect prediction using product and process metrics. To improve the quality of software datasets, we applied

Table 17. Summary of research questions

Research question	Discussion
RQ1: How does the advanced defect prediction models proposed in the study perform using various machine learning classifiers?	For conducting the experiments, we designed five scenarios, i.e., Simple model; Advanced model-1 (Product + NR metric); Advanced model-2 (Product + NDC metric); Advanced model-3 (Product + NML metric); and Advanced model-4 (Product + NDPV metric), respectively. All the designed models are tested on various project datasets repositories, i.e., PROMISE, BUG and JIRA using different classifiers such as DT, MLP, SVM, RT, NB. The advanced model-2 with MLP classifier having high predictive capability followed by advanced model-3. Results discussed in Section 5.1 shows that the proposed advanced models have performed impressively well for inter project fault prediction.
RQ2: How does the ensemble design improve classification performance when compared to individual machine learning classifiers?	In general, the ensemble methods show an overall median of F-score ranging between 76.50% and 87.34% and the ROC (AUC) between 77.09% and84.05%. Base classifiers instead, reach an overall average F-score ranging between 73% (simple model) and 83% (Advanced model-2) for PROMISE data set and the ROC (AUC) between 60% (Advanced model-4) and 79% (Advanced model-2). Thus, we can say that the ensemble design enhances the strengths of multiple predictors and supplements to state of art in fault prediction problem. While experimenting, we have noticed that ensemble techniques (i.e., AdaBoosting, Bagging, Random Forests, and Voting) performed better among all the advanced models as discussed in Section 5.2. With respect to average accuracy, average AUC (ROC), average F-score, and average RMSE, the proposed ensembles VOT-E1 and VOT-E2 have high median value and high maximum value followed by AdaBoost and Random Forest with features based on (Product + NDC) metric data set for projects not only from PROMISE repository but also from BUG and JIRA dataset repositories, which validates the results and makes the approach more reliable.
RQ3: Whether there exist any statistically significant performance difference among the base classifiers and ensemble classifiers?	For pair-wise comparisons, Wilcoxon signed ranks test had been applied. It is observed from the results (Table 13 in Section 5.3) that the performance of ensemble classifier is considerably dissimilar than other classifiers, apart from the MLP based classifier. The results of Friedmans tests and Wilcoxon signed rank tests illustrate that the ensemble method exhibits statistically significant performance differences.
RQ4: For a given software system, whether the proposed ensembles are cost sensitive?	From the results presented in Table 16 Section 5.4, it is observed that for almost all projects, i.e., Ant, Camel, Jedit, Synapse, Xerecs, Equinox, Eclipse JDT, Eclipse PDE, MYLYN, ActiveMQ5.0.0, derby 10.5.1.1, Groovy-1 _6 _BETA _1, Hbase-0.94.0, Hive-0.9.0 and Jruby-1.1 from the PROMISE, BUG and Zira repositories, Ncost values are lower or equal to the threshold value, i.e. 1.0 for both the proposed ensemble VOT-E2 and best base classifier MLP except in few cases, i.e. Xalan, Lucene and Wicket-1.3.0beta2 the normalized cost values are more than threshold value. Thus, for a given software system, the proposed ensembles are cost sensitive.

Table 18. Comparision summary of research techniques

Authors	Study objective	Software fault data sets used/ repository	Fault prediction techniques	Results
Song et al. [2]	Proposed and evaluated a general framework for software defect prediction	NASA, MDP and AR Data Sets	Three learning algorithms. Naive Bayes (NB), J48, one R and 12 learning schemes	Naive Bayes performs much better than J48, and J48 is better than OneR No learning scheme dominates, i.e., always outperforms the others for all 17 data sets.
Rathore and Kumar [26]	Empirical study of ensemble techniques for software fault prediction	28 software fault datasets (PROMISE repository)	7 ensemble techniques, i.e., Dagging, Decorate, Grading, MultiBoostAB, RealAdaBoost, Rotation Forest, and Ensemble Selection	Precision = 0.995 (Rotation Forest) Recall = 0.994 (Rotation Forest) AUC = 0.986 (Decorate) Cost-sensitiveness: proposed ensemble techniques saved software testing cost and effort for 20 out of 28 fault datasets.
Laradji et al. [27]	To demonstrate the positive effects of combining feature selection and ensemble learning on the performance of defect classification.	06 datasets: Ant-1.7, Camel-1.6, KC3, MC1, PC2, and PC4	Average probability ensemble (APE) consisting of 7 classifiers, i.e., random forests, gradient boosting, stochastic gradient descent, W-SVMs, logistic regression, multinomial naive Bayes, and Bernoulli naive Bayes	Higher AUC measures for each dataset, which were close to 1.0 in the case of PC2, PC4 and MC1 datasets.
Proposed approach	Study aims to develop advanced models for software defect prediction which uses both product and process metrics.	32 projects from PROMISE, BUG, and JIRA dataset repository.	5 Base learners and ensemble methods (i.e., AdaBoosting, Bagging, Random Forests, and Voting)	Ensemble methods: F-score (76.50% −87.34%) and the ROC (AUC) (77.09% −84.05%) for Product + NDC metric data for all data sets. Cost-sensitiveness: VOT-E2 ensemble saved software testing cost and effort for 29 out of 32 fault datasets.

dimensional reduction, which is achieved by using feature ranking and feature subset selection techniques [7]. In this way, we obtained a reduced set of 15 features for defect prediction. Thus, data preprocessing helps to avoid the creation of an unstable model [30]. The study results are replicated with various datasets from Promise, Bug and Jira repositories which makes the study reliable. We have used N fold cross-validation while conducting experiments to avoid bias due to sampling. To obtain the experimentation results, authors in the study used F-score and ROC(AUC) metrics which are considered to be more consistent measures for evaluation of classification algorithms [26].

Conclusion validity: It denotes the relation between treatment and outcome. During the study, our objective was to examine the overall predictive capability of proposed advanced models using various machine learning classifiers. We also examined whether the ensemble design improves classification performance as compared to base machine learning classifiers. For this, we designed ensembles using bagging, boosting and voting to examine the improvement in the performance of proposed defect prediction models. The

performance of ensembles is measured by constructing box plots for accuracy, precision and AUC(ROC). From the results, it is observed that fault prediction capability is increased using ensemble-based learning [29, 63]. Furthermore, we validated the statistical significance of performance differences among the base classifiers and the best ensemble classifier, using the non-parametric Friedman test and performed pair wise comparisons using the Wilcoxon Rank-Sum test to test the worst-performing classifiers.

External validity: It deals with the generalization of the results of our study to other settings. In this study, we considered thirty two releases from various datasets used for different application domains. To minimize the effect of a particular tool/technology, in our normalized dataset we have taken applications developed using different version control systems (CVS and SVN) and diverse bug tracking tools (Ckjm, Bug Info, Quality Spy) [17, 63]. We preprocessed the data related to the metrics and obtained a reduced set of thirteen features for defect prediction so that the generalized prediction models are formed. The comparative assessment performed using base and ensemble classifiers verified the significance of proposed advanced models in fault prediction and helps to minimize the threat due to external validity.

7. Conclusions

The work presents advanced models for software fault prediction, in which authors have used information related to product and process metrics. The models for investigation were built based on five different scenarios as discussed in Section 4. Scenario-1: simple model (consists of product metrics only); scenario-2: advanced model-1 (product metrics + NR process metric); scenario-3: advanced model-2 (product metrics + NDC process metric); scenario-4: advanced model-3 (product metrics + NML process metric); scenario-4: advanced model-4 (product metrics + NDPV process metric). The various base classifiers used to predict the performance of proposed models are DT, MLP, RT, SVM and NB. The study has been conducted on thirty-two open-source code projects extracted from the Promise, Jira and Bug repositories. The results show that among base classifiers the MLP based base classifier captures high average accuracy (87%), average ROC(AUC) (79%), average F-score (83%) and least RMSE error (0.12) for advanced model-2 constructed using (product + NDC metrics) from Promise repository as compared to other advanced models, i.e., advanced model-1, advanced model-3 and advanced model-4, respectively. Similar trend is observed for projects from Jira and Bug repositories too.

Furthermore, to examine whether the ensemble design improves classification performance as compared to individual machine learning classifiers we used the ensemble approach based on bagging, boosting and voting to combine multiple classifiers and conducted replication experiments. The comparison of results using average accuracy, average RMSE, average ROC(AUC) and average F-score confirms the predictive capability of proposed classifiers for developing advanced defect prediction models. The VOT-E2 (DT + MLP + SVM) ensemble produced the best results with advanced model-2, advanced model-3 and advanced model-1 followed by VOT-E1 classifiers (DT + MLP + RT), in terms of ROC(AUC) and F-measure. Further to validate the statistical significance of performance differences among the base classifiers and classifier ensemble, we also tested the hypothesis H_0, that there is no significant difference between base classifier performance and ensemble classifier performance using a non-parametric test. We also evaluated the fault removal estimation cost for the proposed ensemble and best base classifier. The normalized fault

removal cost is obtained for different projects from Promise, Jira and Bug repositories by calculating the ratio of estimated fault removal cost to estimated testing cost, which is below the threshold value, i.e., less than one.

Our results shows that the advanced fault estimation models constructed with a normalized and minimum subset of software metrics, which includes product metrics and one process metric at a time, provide satisfactory performance as compared to simple models constructed using product metrics alone. The proposed approach based on combination models may prove useful to software engineers for their new projects. Though in the study, authors have conducted experiments using projects from Promise, Jira and Bug repositories, still, to establish evidence and improve generalization of results, the investigations shall be replicated using more open-source and cross-project data sets. Several defect prediction models have been developed which use heterogeneous metric data from other projects [31, 63]. The investigation using more number projects would not only increase the variety of examined data but also helps to improve the external validity of the research outcomes.

References

[1] Z. Li, X.Y. Jing, and X. Zhu, "Progress on approaches to software defect prediction," *Iet Software*, Vol. 12, No. 3, 2018, pp. 161–175.

[2] Q. Song, Z. Jia, M. Shepperd, S. Ying, and J. Liu, "A general software defect-proneness prediction framework," *IEEE transactions on software engineering*, Vol. 37, No. 3, 2010, pp. 356–370.

[3] *IEEE Standard Glossary of Software Engineering Terminology*, IEEE Std. 610.12-1990, 1990. [Online]. https://ieeexplore.ieee.org/document/159342

[4] X. Yang, D. Lo, X. Xia, and J. Sun, "TLEL: A two-layer ensemble learning approach for just-in-time defect prediction," *Information and Software Technology*, Vol. 87, 2017, pp. 206–220.

[5] X. Yang, D. Lo, X. Xia, Y. Zhang, and J. Sun, "Deep learning for just-in-time defect prediction," in *International Conference on Software Quality, Reliability and Security*. IEEE, 2015, pp. 17–26.

[6] Y. Yang, Y. Zhou, J. Liu, Y. Zhao, H. Lu et al., "Effort-aware just-in-time defect prediction: simple unsupervised models could be better than supervised models," in *Proceedings of the 24th ACM SIGSOFT international symposium on foundations of software engineering*, 2016, pp. 157–168.

[7] Ö.F. Arar and K. Ayan, "Deriving thresholds of software metrics to predict faults on open source software: Replicated case studies," *Expert Systems with Applications*, Vol. 61, 2016, pp. 106–121.

[8] R. Malhotra and J. Jain, "Handling imbalanced data using ensemble learning in software defect prediction," in *10th International Conference on Cloud Computing, Data Science and Engineering (Confluence)*. IEEE, 2020, pp. 300–304.

[9] F. Matloob, T.M. Ghazal, N. Taleb, S. Aftab, M. Ahmad et al., "Software defect prediction using ensemble learning: A systematic literature review," *IEEE Access*, 2021.

[10] L. Pascarella, F. Palomba, and A. Bacchelli, "Fine-grained just-in-time defect prediction," *Journal of Systems and Software*, Vol. 150, 2019, pp. 22–36.

[11] 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, 2008, pp. 485–496.

[12] S.S. Rathore and S. Kumar, "An empirical study of ensemble techniques for software fault prediction," *Applied Intelligence*, Vol. 51, No. 6, 2021, pp. 3615–3644.

[13] R. Jabangwe, J. Börstler, D. Šmite, and C. Wohlin, "Empirical evidence on the link between object-oriented measures and external quality attributes: A systematic literature review," *Empirical Software Engineering*, Vol. 20, No. 3, 2015, pp. 640–693.

[14] Z. Li, X.Y. Jing, and X. Zhu, "Heterogeneous fault prediction with cost-sensitive domain adaptation," *Software Testing, Verification and Reliability*, Vol. 28, No. 2, 2018, p. e1658.
[15] R. Malhotra, "An empirical framework for defect prediction using machine learning techniques with Android software," *Applied Soft Computing*, Vol. 49, 2016, pp. 1034–1050.
[16] L. Qiao, X. Li, Q. Umer, and P. Guo, "Deep learning based software defect prediction," *Neurocomputing*, Vol. 385, 2020, pp. 100–110.
[17] I. Kiris, S. Kapan, A. Kılbas, N. Yılmaz, I. Altuntaş et al., "The protective effect of erythropoietin on renal injury induced by abdominal aortic-ischemia-reperfusion in rats," *Journal of Surgical Research*, Vol. 149, No. 2, 2008, pp. 206–213.
[18] L. Madeyski and M. Jureczko, "Which process metrics can significantly improve defect prediction models? An empirical study," *Software Quality Journal*, Vol. 23, No. 3, 2015, pp. 393–422.
[19] D. Radjenović, M. Heričko, R. Torkar, and A. Živkovič, "Software fault prediction metrics: A systematic literature review," *Information and software technology*, Vol. 55, No. 8, 2013, pp. 1397–1418.
[20] Y. Wu, Y. Yang, Y. Zhao, H. Lu, Y. Zhou et al., "The influence of developer quality on software fault-proneness prediction," in *EIghth International Conference on Software Security and Reliability (SERE)*. IEEE, 2014, pp. 11–19.
[21] C. Bird, N. Nagappan, B. Murphy, H. Gall, and P. Devanbu, "Don't touch my code! Examining the effects of ownership on software quality," in *Proceedings of the 19th ACM SIGSOFT symposium and the 13th European conference on Foundations of software engineering*, 2011, pp. 4–14.
[22] D. Di Nucci, F. Palomba, G. De Rosa, G. Bavota, R. Oliveto et al., "A developer centered bug prediction model," *IEEE Transactions on Software Engineering*, Vol. 44, No. 1, 2017, pp. 5–24.
[23] F. Palomba, M. Zanoni, F.A. Fontana, A. De Lucia, and R. Oliveto, "Toward a smell-aware bug prediction model," *IEEE Transactions on Software Engineering*, Vol. 45, No. 2, 2017, pp. 194–218.
[24] F. Rahman and P. Devanbu, "How, and why, process metrics are better," in *35th International Conference on Software Engineering (ICSE)*. IEEE, 2013, pp. 432–441.
[25] B. Ghotra, S. McIntosh, and A.E. Hassan, "Revisiting the impact of classification techniques on the performance of defect prediction models," in *37th IEEE International Conference on Software Engineering*, Vol. 1. IEEE, 2015, pp. 789–800.
[26] F. Yucalar, A. Ozcift, E. Borandag, and D. Kilinc, "Multiple-classifiers in software quality engineering: Combining predictors to improve software fault prediction ability," *Engineering Science and Technology, an International Journal*, Vol. 23, No. 4, 2020, pp. 938–950.
[27] I.H. Laradji, M. Alshayeb, and L. Ghouti, "Software defect prediction using ensemble learning on selected features," *Information and Software Technology*, Vol. 58, 2015, pp. 388–402.
[28] C. Catal and B. Diri, "Investigating the effect of dataset size, metrics sets, and feature selection techniques on software fault prediction problem," *Information Sciences*, Vol. 179, No. 8, 2009, pp. 1040–1058.
[29] T.M. Khoshgoftaar, K. Gao, and N. Seliya, "Attribute selection and imbalanced data: Problems in software defect prediction," in *22nd IEEE International conference on tools with artificial intelligence*, Vol. 1. IEEE, 2010, pp. 137–144.
[30] X. Chen, Y. Mu, Y. Qu, C. Ni, M. Liu et al., "Do different cross-project defect prediction methods identify the same defective modules?" *Journal of Software: Evolution and Process*, Vol. 32, No. 5, 2020, p. e2234.
[31] Y. Zhang, D. Lo, X. Xia, and J. Sun, "Combined classifier for cross-project defect prediction: An extended empirical study," *Frontiers of Computer Science*, Vol. 12, No. 2, 2018, p. 280.
[32] T. Lee, J. Nam, D. Han, S. Kim, and H.P. In, "Micro interaction metrics for defect prediction," in *Proceedings of the 19th ACM SIGSOFT symposium and the 13th European conference on Foundations of software engineering*, 2011, pp. 311–321.
[33] K. Juneja, "A fuzzy-filtered neuro-fuzzy framework for software fault prediction for inter-version and inter-project evaluation," *Applied Soft Computing*, Vol. 77, 2019, pp. 696–713.

[34] H. Wang, T.M. Khoshgoftaar, and A. Napolitano, "A comparative study of ensemble feature selection techniques for software defect prediction," in *Ninth International Conference on Machine Learning and Applications*. IEEE, 2010, pp. 135–140.

[35] J. Petrić, D. Bowes, T. Hall, B. Christianson, and N. Baddoo, "Building an ensemble for software defect prediction based on diversity selection," in *Proceedings of the 10th ACM/IEEE International symposium on empirical software engineering and measurement*, 2016, pp. 1–10.

[36] F. Pecorelli and D. Di Nucci, "Adaptive selection of classifiers for bug prediction: A large-scale empirical analysis of its performances and a benchmark study," *Science of Computer Programming*, Vol. 205, 2021, p. 102611.

[37] D. Di Nucci, F. Palomba, R. Oliveto, and A. De Lucia, "Dynamic selection of classifiers in bug prediction: An adaptive method," *IEEE Transactions on Emerging Topics in Computational Intelligence*, Vol. 1, No. 3, 2017, pp. 202–212.

[38] D. Bowes, T. Hall, and J. Petrić, "Software defect prediction: do different classifiers find the same defects?" *Software Quality Journal*, Vol. 26, No. 2, 2018, pp. 525–552.

[39] G. Abaei, A. Selamat, and H. Fujita, "An empirical study based on semi-supervised hybrid self-organizing map for software fault prediction," *Knowledge-Based Systems*, Vol. 74, 2015, pp. 28–39.

[40] E. Erturk and E.A. Sezer, "A comparison of some soft computing methods for software fault prediction," *Expert systems with applications*, Vol. 42, No. 4, 2015, pp. 1872–1879.

[41] Y. Hu, B. Feng, X. Mo, X. Zhang, E. Ngai et al., "Cost-sensitive and ensemble-based prediction model for outsourced software project risk prediction," *Decision Support Systems*, Vol. 72, 2015, pp. 11–23.

[42] M.O. Elish, H. Aljamaan, and I. Ahmad, "Three empirical studies on predicting software maintainability using ensemble methods," *Soft Computing*, Vol. 19, No. 9, 2015, pp. 2511–2524.

[43] P. He, B. Li, X. Liu, J. Chen, and Y. Ma, "An empirical study on software defect prediction with a simplified metric set," *Information and Software Technology*, Vol. 59, 2015, pp. 170–190.

[44] W. Rhmann, B. Pandey, G. Ansari, and D.K. Pandey, "Software fault prediction based on change metrics using hybrid algorithms: An empirical study," *Journal of King Saud University-Computer and Information Sciences*, Vol. 32, No. 4, 2020, pp. 419–424.

[45] A. Kaur and I. Kaur, "An empirical evaluation of classification algorithms for fault prediction in open source projects," *Journal of King Saud University-Computer and Information Sciences*, Vol. 30, No. 1, 2018, pp. 2–17.

[46] D. Cotroneo, A.K. Iannillo, R. Natella, R. Pietrantuono, and S. Russo, "The software aging and rejuvenation repository: http://openscience.us/repo/software-aging," in *International Symposium on Software Reliability Engineering Workshops (ISSREW)*. IEEE, 2015, pp. 108–113.

[47] M. D'Ambros, M. Lanza, and R. Robbes, "An extensive comparison of bug prediction approaches," in *Proceedings of MSR 2010 (7th IEEE Working Conference on Mining Software Repositories)*. IEEE CS Press, 2010, pp. 31 – 41.

[48] T. Menzies, B. Caglayan, E. Kocaguneli, J. Krall, F. Peters et al., "The promise repository of empirical software engineering data," *West Virginia University, Department of Computer Science*, 2012.

[49] M. Shepperd, Q. Song, Z. Sun, and C. Mair, "Data quality: Some comments on the NASA software defect datasets," *IEEE Transactions on Software Engineering*, Vol. 39, No. 9, 2013, pp. 1208–1215.

[50] N.V. Chawla, K.W. Bowyer, L.O. Hall, and W.P. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique," *Journal of artificial intelligence research*, Vol. 16, 2002, pp. 321–357.

[51] S. Wang and X. Yao, "Using class imbalance learning for software defect prediction," *IEEE Transactions on Reliability*, Vol. 62, No. 2, 2013, pp. 434–443.

[52] K. Gao, T.M. Khoshgoftaar, H. Wang, and N. Seliya, "Choosing software metrics for defect prediction: an investigation on feature selection techniques," *Software: Practice and Experience*, Vol. 41, No. 5, 2011, pp. 579–606.

[53] Z.H. Zhou and X.Y. Liu, "Training cost-sensitive neural networks with methods addressing the class imbalance problem," *IEEE Transactions on knowledge and data engineering*, Vol. 18, No. 1, 2005, pp. 63–77.

[54] L. Kumar, S. Misra, and S.K. Rath, "An empirical analysis of the effectiveness of software metrics and fault prediction model for identifying faulty classes," *Computer Standards and Interfaces*, Vol. 53, 2017, pp. 1–32.
[55] M. Dash and H. Liu, "Consistency-based search in feature selection," *Artificial intelligence*, Vol. 151, No. 1-2, 2003, pp. 155–176.
[56] S.S. Rathore and S. Kumar, "Linear and non-linear heterogeneous ensemble methods to predict the number of faults in software systems," *Knowledge-Based Systems*, Vol. 119, 2017, pp. 232–256.
[57] T. Menzies, J. Greenwald, and A. Frank, "Data mining static code attributes to learn defect predictors," *IEEE transactions on software engineering*, Vol. 33, No. 1, 2006, pp. 2–13.
[58] A.E.C. Cruz and K. Ochimizu, "Towards logistic regression models for predicting fault-prone code across software projects," in *3rd international symposium on empirical software engineering and measurement*. IEEE, 2009, pp. 460–463.
[59] J. Li, D.M. Witten, I.M. Johnstone, and R. Tibshirani, "Normalization, testing, and false discovery rate estimation for RNA-sequencing data," *Biostatistics*, Vol. 13, No. 3, 2012, pp. 523–538.
[60] J. Nam, S.J. Pan, and S. Kim, "Transfer defect learning," in *35th international conference on software engineering (ICSE)*. IEEE, 2013, pp. 382–391.
[61] S. Matsumoto, Y. Kamei, A. Monden, K.i. Matsumoto, and M. Nakamura, "An analysis of developer metrics for fault prediction," in *Proceedings of the 6th International Conference on Predictive Models in Software Engineering*, 2010, pp. 1–9.
[62] Y. Jiang, B. Cukic, and Y. Ma, "Techniques for evaluating fault prediction models," *Empirical Software Engineering*, Vol. 13, No. 5, 2008, pp. 561–595.
[63] X. Xuan, D. Lo, X. Xia, and Y. Tian, "Evaluating defect prediction approaches using a massive set of metrics: An empirical study," in *Proceedings of the 30th Annual ACM Symposium on Applied Computing*, 2015, pp. 1644–1647.
[64] S. Wagner, "A literature survey of the quality economics of defect-detection techniques," in *Proceedings of the ACM/IEEE international symposium on Empirical software engineering*, 2006, pp. 194–203.
[65] C. Jones and O. Bonsignour, *The economics of software quality*. Addison-Wesley Professional, 2011.
[66] N. Wilde and R. Huitt, "Maintenance support for object-oriented programs," *IEEE Transactions on Software Engineering*, Vol. 18, No. 12, 1992, p. 1038.
[67] T. Wang, W. Li, H. Shi, and Z. Liu, "Software defect prediction based on classifiers ensemble," *Journal of Information and Computational Science*, Vol. 8, No. 16, 2011, pp. 4241–4254.
[68] K. Bańczyk, O. Kempa, T. Lasota, and B. Trawiński, "Empirical comparison of bagging ensembles created using weak learners for a regression problem," in *Asian Conference on Intelligent Information and Database Systems*. Springer, 2011, pp. 312–322.
[69] G. Catolino and F. Ferrucci, "An extensive evaluation of ensemble techniques for software change prediction," *Journal of Software: Evolution and Process*, Vol. 31, No. 9, 2019, p. e2156.
[70] L. Reyzin and R.E. Schapire, "How boosting the margin can also boost classifier complexity," in *Proceedings of the 23rd international conference on Machine learning*, 2006, pp. 753–760.
[71] J. Petrić, D. Bowes, T. Hall, B. Christianson, and N. Baddoo, "Building an ensemble for software defect prediction based on diversity selection," in *Proceedings of the 10th ACM/IEEE International symposium on empirical software engineering and measurement*, 2016, pp. 1–10.
[72] A.T. Mısırlı, A.B. Bener, and B. Turhan, "An industrial case study of classifier ensembles for locating software defects," *Software Quality Journal*, Vol. 19, No. 3, 2011, pp. 515–536.
[73] J. Bansiya and C.G. Davis, "A hierarchical model for object-oriented design quality assessment," *IEEE Transactions on software engineering*, Vol. 28, No. 1, 2002, pp. 4–17.
[74] E. Shihab, Z.M. Jiang, W.M. Ibrahim, B. Adams, and A.E. Hassan, "Understanding the impact of code and process metrics on post-release defects: a case study on the eclipse project," in *Proceedings of the ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, 2010, pp. 1–10.

[75] R. Martin, "OO design quality metrics," *An analysis of dependencies*, Vol. 12, No. 1, 1994, pp. 151–170.

A. Appendix

Table A1. Description of selected features

Product metrics	Description	References
WMC	It is the weighted sum of methods implemented within a class.	[73]
NOC	It is defined as the number of instant sub classes (children) subordinated to a class (parent) in the class hierarchy.	[73]
CBO	It defines the number of other classes that are tied to a given class during method call or function call, abstraction etc.	[73]
RFC	It is a count of methods in a class or methods directly called by these.	[73]
LCOM	It is a number of private methods in a class which dont connect the class fields.	[73]
Ca	It is used to measure the number of classes that depends on a given class.	[74]
Ce	It is used to measure the number of classes on which a given class depends.	[74]
NPM	It is a number of public methods in a given class.	[58]
LOC	It is a number of lines of code in a given class.	[59]
DAM	It is the ratio of the number of private/protected attributes to the total number of attributes in a given class.	[65]
MOA	It is a number of classes whose declaration is user defined.	[65]
AMC	It is the average size of methods in a given class.	[75]
Max−*CC*	It is the maximum McCabes CC score for methods in a given class.	[75]
Avg−*CC*	It is the arithmetic mean of McCabes CC score for methods in a given class.	[75]
Process metrics	**Description**	**References**
NR	It represents the number of revisions of a given class because of bug/or some enhancements in a specific revision or version of a software system.	[17, 24, 44, 74]
NDC/*NAUTH*	It counts the number of different programmers /developers/authors who committed their changes in the given class during the improvement of the specific revision of the software.	[17, 24, 44, 74]
NML/*NREF*	It is the sum of all number of lines that are added or altered or number of times a file has been refactored.	[17, 24, 44, 74]
NDPV	The metric counts the number of defects in the previous version being corrected in the respective class while developing the previous releases or versions.	[24, 44, 74]

Table A2. Details of base classifiers and classifier ensembles

Base Classifiers	Description
Naive Bayes (NB) [29]	It is a probabilistic classification technique based on Bayes Theorem with an assumption, that each pair of features being classified is independent of each other.
Decision Tree (DT) [29]	The simplest supervised learning method which creates tree structure to consider target values as discrete set or decision rules known as classification tree and nodes denotes class labels.
Random Tree (RT) [29]	It is a collection of multiple trees which are relatively uncorrelated, operating as a committee, split out a class with the most votes for models prediction.
Multilayer Perceptron (MLP) [29]	A supervised feed-forward artificial neural network model which maps input data onto a set of appropriate outputs and in between these two, an arbitrary number of hidden layers which work as a computational engine of the MLP.
Support vector machines (SVMs) [41]	SVMs are a group of supervised learning methods which makes use of statistical learning theory for classification. The methods are proposed by Cortes and [69]. The basic idea of SVM is to identify a similarity distance between two entities (classes) by considering a distance metric between them. It could also be used to handle unbalanced classes.
Ensemble classifiers	**Description**
Random Forest (RF) [29]	It is an ensemble-based method used in classification which constitute multiple decision trees on randomly selected data at training time and get prediction from each tree and choose best solution by voting.
Boosting [61]	The method is proposed by Freund [71]. It modifies a training set by repeatedly applying a basic learning device (i.e., classifier) under a pre-specified number of iterations. Adaptive Boosting (AdaBoost) is a well-known Boosting technique.
Bootstrap aggregating [29]	Bagging is a bootstrap method proposed by Breiman [72] that mainly extracts a training sample from a training set by returning them to each extraction. It allocates equal weight to developed models, thereby reduces the variance related with classification, which in turn improves the classification process.
Voting [29]	It represents the simplest ensemble algorithm used for classification or regression problems. Each sub model in the algorithm makes use of votes or algebraic combinations (mean or the mode) of heterogeneous predictors to make predictions.

Table A3. Details of performance measures

Measures	Defined as	Description
Pd/Recall/TP	$TP/(TP + FN)$	It is defined as the ratio of the number of defective instances that are correctly classified as defective to the total number of defective instances.
Pf	$FP/(FP + TN)$	It is defined as the ratio of the number of non-defective instances that are wrongly classified as defective to the total number of non-defective instances.
Precision	$TP/(TP + FP)$	Precision is defined as the number of correctly identified positive results to the total number of all positive outcomes, including those not recognized correctly.
$F1$-score	$(2 \times precision \times Pd)/(precision + Pd)$	It is a measure for harmonic mean of Pd and $precision$.
Accuracy	$(TP + TN)/(TP + TN + FP + FN) \times 100$	It denotes the percentage of correctly predicted instances.
Root mean square error (RMSE)	$\sqrt{1/N \sum_{i=1}^{N} (A_i - P_i)^2}$	It defines the square root of the mean of squared differences of actual and expected predictions.
ROC(AUC)	Receiver operating Characteristics (Area under curve)	ROC(AUC) measures the performance of the classification problems at various thresholds in the imbalanced data-set. ROC is a probability curve, and AUC represents the measure of separability. If AUC value is high, the model is predicting better. It ranges from 0 to 1
False-positive rate (FPR)	$FPR = FP/(TP + TN)$	The expectancy of the false-positive ratio to the total of actual negative.
False-negative rate (FNR)	$FNR = FN/(TP + FN)$	The ratio of the individuals with an identified positive instance for which the classified test result is negative.

Table A4. Confusion matrix for classifying data as faulty or non-faulty

		Actual	
		Positive	Negative
Predicted	Positive	True-positive (TP)	False-positive (FP)
	Negative	False-negative (FN)	True-negative (TN)

Under the cost evaluation framework, the notations used to formulate various costs are:

- E_{cost}: Estimated fault removal cost of the software when software fault prediction results are used
- T_{cost}: Estimated fault removal cost of the software without the use of software fault prediction results
- NE_{cost}: Normalized fault removal cost of the software when software fault prediction is used
- C_u: Normalized value of fault removal cost when unit testing is done
- C_i: Normalized value of fault removal cost when integration testing is done
- C_s: Normalized value of fault removal cost when system testing is done
- C_f: Normalized value of fault removal cost when field testing is done
- M_p: Percentage of modules when unit tested
- FM: Total number of faulty modules, and
- TM: Total number of modules in software projects
- FP, FN: Number of false positives, number of false negatives
- TP: Number of true positives

e-Informatica Software Engineering Journal, Volume 16, Issue 1, 2022, pages: 220105, DOI: 10.37190/e-Inf220105

A Systematic Review of Ensemble Techniques for Software Defect and Change Prediction

Megha Khanna*

*Department of Computer Science, Sri Guru Gobind Singh College of Commerce, University of Delhi

meghakhanna86@gmail.com

Abstract

Background: The use of ensemble techniques have steadily gained popularity in several software quality assurance activities. These aggregated classifiers have proven to be superior than their constituent base models. Though ensemble techniques have been widely used in key areas such as Software Defect Prediction (SDP) and Software Change Prediction (SCP), the current state-of-the-art concerning the use of these techniques needs scrutinization.
Aim: The study aims to assess, evaluate and uncover possible research gaps with respect to the use of ensemble techniques in SDP and SCP.
Method: This study conducts an extensive literature review of 77 primary studies on the basis of the category, application, rules of formulation, performance, and possible threats of the proposed/utilized ensemble techniques.
Results: Ensemble techniques were primarily categorized on the basis of similarity, aggregation, relationship, diversity, and dependency of their base models. They were also found effective in several applications such as their use as a learning algorithm for developing SDP/SCP models and for addressing the class imbalance issue.
Conclusion: The results of the review ascertain the need of more studies to propose, assess, validate, and compare various categories of ensemble techniques for diverse applications in SDP/SCP such as transfer learning and online learning.

Keywords: Ensemble learning, Software change prediction, Software defect prediction, Software quality, Systematic review

1. Introduction

Technology has ensured that software is a fundamental part of every activity. This necessitates the development and maintenance of good quality software products. However, rigid deadlines, limited budgets, and scarce resources often impede the development of competent software products. Thus, it is essential to perform Software Quality Assurance (SQA) activities so that the quality of software products is not compromised. Software Defect Prediction (SDP) and Software Change Prediction (SCP) models, which predict defect prone and change prone parts of software in its early stages of development are popular means of prioritizing effort for SQA activities. Defect prone and change prone parts, though few, account for a majority of the defects and changes in a software [1, 2]. Thus, SQA efforts should be focused on these parts as they need to be meticulously designed and

Submitted: 23 Jul. 2021; Revised: 09 Jan. 2022; Accepted: 21 Feb. 2022; Available online: 04 Mar. 2022

carefully verified [3–5]. Software practitioners may design and verify these parts in such a manner so that future occurrences of defects can be minimized and the effect of changes may be localized [2, 6–9]. These activities would assure the timely delivery of cost-effective and maintainable software products.

Over the years, the research community has extensively explored various algorithms for developing SDP and SCP models. Amongst the various categories, the "ensemble" techniques are a key category that have been widely investigated by the researchers [10–13]. These techniques are an assembly of diverse base models, where each base model attempts to resolve the original problem at hand [12], which in our case is the determination of defect prone and change prone classes. Ensemble Techniques (ET) output the result of the aggregated base models as "aggregation" provides a more stable and reliable estimate with an improved predictive ability [14–17]. Combining several diverse base models is analogous to consulting a committee of experts thereby resulting in more accurate predictions [18].

Given the unique nature of ET and its improved performance over single models, it is vital to systematically summarize and analyze the empirical evidence for its use in SDP and SCP literature. Previous studies have comprehensively evaluated the use of ET for feature selection [19], effort estimation [12], and class imbalance problem [20]. Also, there have been several efforts by researchers that systematically summarize the SDP and SCP studies from various aspects. Radjenovic et al. [21] examined various software metrics in the context of SDP and found object-oriented metrics to be most prevalent. Catal [22] investigated SDP studies in the period 1990–2009 to summarize the metrics, performance measures, methods, datasets, and experimental results used in the studies. Hosseini et al. [23] synthesized the state of the art concerning the use of cross-project models in SDP studies. Malhotra [10] evaluated the use of several machine learning techniques for SDP. Amongst other findings, she reported that ET were used in 18% of 65 primary studies. Wahono [11] conducted a systematic literature review of 71 SDP studies to investigate the methods, datasets, frameworks, and research trends in SDP. An interesting result of the review pointed out that researchers have suggested the use of ET and the use of the boosting algorithm for improving the performance of existing machine learning classifiers. Two other previous reviews have also scrutinized SDP and SCP studies [24, 25], but with respect to the use of search-based algorithms and validity threats specific to its usage. A recent review by Malhotra and Khanna [13] assessed the various predictors, techniques and their predictive performance, experimental settings and validity threats in 38 SCP studies. Amongst other results, the review study encouraged the use of ET as they were found to be popular as well as effective (when evaluated in terms of accuracy and AUC measures) in the SCP domain. This study complements the previous work as we investigate the use of ET in both SCP as well as SDP domain (a related area of SCP). We analyze the several categories of ET, their rules, predictive capability and their possible application in aiding the SDP/SCP problems. Certain other researchers [26–28] have also reviewed SDP and SCP literature. However, to the best of the author's knowledge, there has been no study till date which has focused on a comprehensive evaluation of the use of ET for SDP and SCP, which is the primary aim of this study.

To facilitate an extensive analysis of ET used in SDP and SCP literature we examine the following Research Questions (RQ):

– RQ1: What is the categorization of ET? Which is the most popular category of ET in SDP/SCP literature?
– RQ2: What are the various applications of ET in SDP/SCP literature?

- RQ3: Which rules/mechanisms are used for combining base models to form ET in SDP/SCP literature?
- RQ4: What is the performance of ET for various tasks in the domain of SDP/SCP? How does the performance of ET compare to other non-ensemble techniques and amongst each other?
- RQ5: What are the various reported threats to validity specific to the use of ET in SDP/SCP literature?

The objective of the study is to systematically collect and rigorously evaluate literature studies that develop classification models using ET to predict defect prone and change prone parts of a software. This would help in summarizing the current trends for the use of ET in SDP/SCP literature and further determine gaps in existing research. The study is structured into five further sections, which includes research methodology followed to conduct the review (Section 2), review results (Section 3), discussion and proposed future work (Section 4), threats to validity of the review (Section 5) and conclusions (Section 6).

2. Review methodology

To accomplish our goals, we performed a systematic literature review in three stages according to the guidelines stated by Kitchenham et al. [29]. The first stage was planning which included identifying the review objectives and the protocol for conducting the review. As discussed in the previous section, we evaluated existing systematic reviews on the topic. However, these reviews did not focus on the application of ET in SDP and SCP. Thus, the primary objective of this review was to study the existing literature and provide a critical overview of the use of ET in the domain of SDP and SCP. Thereafter, the research questions were formulated and the review protocol was defined. The review protocol characterizes the search strategy for extracting relevant studies from literature, criteria for including and excluding the collected studies, a benchmark for quality assessment of candidate studies, processes for data extraction from primary studies, and the method for synthesis of extracted data. The second stage of the review involves conducting the review according to the procedures decided in the planning stage. This stage collects the relevant studies and scrutinizes them if they are fit to be primary studies of the review. Thereafter, data pertaining to the formulated RQ's is extracted and synthesized. The last stage of the review concerns itself with reporting of the findings of the review. Here, we report crisp answers to the investigated RQ's and document research gaps in the form of future work to interested researchers.

2.1. Search strategy

To search for relevant studies, we need to prepare a search string by combining appropriate search terms. These search terms were determined by selecting "key" terms from the RQ's of the review [21]. Furthermore, equivalent terms and other possible spellings were examined for the identified search terms. Thereafter, the search string was defined by combining all synonymous terms using Boolean "OR" and all distinct terms by Boolean "AND". The following search string was used:

("software") AND ("Defect" OR "Fault" OR "Error" OR "Bug" OR "Change" OR "Evolution") AND ("proneness" OR "prone" OR "predict*" OR "probability" OR "classification" OR "empirical") AND ("Ensemble" OR "Bagging" OR "Boosting"

OR “Machine learning” OR “Soft Computing” OR “Random Forest” OR “Bootstrap Aggregating” OR “Adaboost” OR “Combin*” OR “Stack*” OR “Meta*” OR “Rotation Forest” OR “Voting” OR “Logitboost”).

We conducted the search in five well-known literature sources namely ScienceDirect, ACM Digital Library, IEEEXplore, Wiley Online Library, and SpringerLink. These sources were chosen based on our previous knowledge of conducting reviews in the SDP and SCP domains [13, 24]. Moreover, most of the primary studies in previously conducted systematic reviews in SDP and SCP are indexed in these sources [10, 11]. The search string was modified suitably according to the requirements of each literature source. It examined the title, abstract, and keywords of the studies in the literature databases. The period of the search was limited from January 2000 to December 2020. We also removed the duplicate studies which were extracted from more than one source. To avoid missing a relevant study, we also scanned the reference lists of recent reviews on SDP and SCP [10, 13] and those of the already collected candidate studies. These efforts resulted in the collection of 182 relevant studies. These studies were further scrutinized using the inclusion and exclusion criteria stated in the next section.

2.2. Inclusion and exclusion criteria

Before stating the criteria for inclusion and exclusion, we first define “defect proneness” and “change proneness” attributes of a software entity. Both these attributes are dichotomous. A software entity is designated as defect-prone if a defect is likely to occur in a subsequent release of the software. Most of the studies in literature, label a class/module as defect-prone if one or more bugs have occurred in the class [3, 4]. On the other hand, a software entity is termed as change-prone if it is likely to change in a future released version of the software product. Majority of studies labeled software entities with a threshold value of one or more changes as change-prone [13]. Certain other studies in literature use “median-based” [30] or “boxplot-based partition method” [31] for labeling change-prone classes [13]. Keeping these definitions in mind we state the following criteria for inclusion and exclusion of the collected studies.

2.2.1. Inclusion criteria

- Empirical studies that use ET for SDP or SCP.
- Empirical studies that compare different ET with each other or with other non-ensemble techniques for SDP or SCP.
- Empirical studies that propose new ET for SDP or SCP.

2.2.2. Exclusion criteria

- Studies that use ET for dependent variables other than defect proneness and change proneness, such as the number of defects/changes, class stability, just in time defect prediction, bug assignment, code churn, etc.
- Studies including ET just to compare or demonstrate their proposed models/performance measures. These studies were excluded as they included ET without any discussion and did not perform or focus on their empirical evaluation.
- Similar studies that were conducted by the same authors. In case a conference paper is extended in a journal, the conference paper is excluded.

– Studies that used clustering or clustering ensembles for prediction.
– Review studies, poster papers, Ph.D. dissertations, and studies with little or no empirical analysis.
– Studies that were not written in English language.

Study inclusion and exclusion was done in two steps. We first applied the mentioned criteria on the title, abstract, and keywords. Thereafter, the remaining studies were adjudged based on their full text. After applying the above discussed criteria, we obtained 106 candidate studies.

2.3. Quality assessment

Each candidate study obtained after application of the inclusion and exclusion criteria was further subject to quality assessment. This step ensures the selection of only those studies, which are capable of effectively answering the investigated research questions. Quality assessment was done by two researchers by formulating a checklist shown in Table 1. According to the table, criteria (i) evaluates whether the study clearly states its aims, while criteria (ii) assesses whether ET and its uses have been clearly mentioned in the study. Criteria (iii) assesses whether the study mentions the base learners and aggregation mechanism used by the ET. Criteria (iv) allocates lower score to a study if it basis it's results on less than five datasets. While criteria (v) focus on selection of an appropriate validation method such as ten-fold, cross-project or others, criteria (vi) evaluates whether robust and appropriate performance measures such as Area Under the Receiver operating characteristic Curve (AUC), Balance, Mathews Correlation Coefficient (MCC) etc., have been used. Studies that base their results only on biased performance measures like accuracy are given less scores. Criteria (vii) evaluates whether models developed using ET are compared with other models, while criteria (viii) allocates a higher score to studies that have performed statistical validation of their results. Criteria (ix) checks if the study has mentioned its probable threats. Finally, Criteria (x) gives higher score to a study that add value to existing literature on ensembles at the time of its publication. As this was hard to evaluate, due to temporal aspect of relevance of the work, the authors allocated lesser score to similar studies that were published in the same year. Similar checklists were used in previous reviews [12, 13, 32]. Each of the two researchers conducting the quality assessment independently assessed each study on the ten questions stated in the checklist.

Table 1. Quality questions

(i)	Does the study state its objectives in a clear and precise manner?
(ii)	Is the use of ET and its application clearly defined?
(iii)	Are the base learners clearly stated? Are the rules/mechanisms for combining base learners to form ET clearly described?
(iv)	Is the experiment conducted on an appropriate number of datasets?
(v)	Are the models developed using ET validated appropriately using effective validation methods?
(vi)	Are the models developed using ET effectively assessed using suitable performance measures?
(vii)	Are the models developed using ET compared with models developed using other non-ensemble techniques or amongst each other?
(viii)	Do the results of the study map to its objectives? Are the results statistically validated?
(ix)	Does the study state possible threats to validity specific to the use of ET?
(x)	Does the study add value to the existing work on ensembles in SDP/SCP literature?

The questions could have three possible responses: Yes (score of +1), Partly (score of +0.5), and No (score of 0). In case the researchers disagreed on the allocated score, a discussion ensued to allocate a reasonable score. The cumulative score (CS) of the study was a sum of all the scores of the questions mentioned in the checklist. A study obtaining a score of 5 or more (50% or more) was considered to be of acceptable quality [12, 13, 32]. We rejected 29 studies on the basis of quality assessment. The remaining 77 studies were termed as primary studies.

2.4. Data extraction and synthesis

For each of the study selected after quality assessment, we extracted the relevant data to answer the RQ's. The extracted data consisted of basic details of a paper such as a title, authors, year of publication, etc. as well as details specific to the experiment that is required to answer the RQs (mentioned in Table 2). After extracting the data, we need to synthesize it to appropriately answer the investigated RQ's. Table 2 mentions the manner in which the data was analyzed and synthesized with respect to each RQ and the expected result after the synthesis and analysis.

Table 2. Data extraction and synthesis

RQ	Data extracted from primary study	Data analysis	Result
RQ1: What is the categorization of ET? Which is the most popular category of ET in SDP/SCP literature?	ET Used, Base learners of ET	Categorization of ET used on the basis of base models, i.e., their similarity, course of aggregation, cooperative or competitive relationship, means of diversity, dependency amongst themselves and the type of base learner used	Pie-charts depicting percentage of SDP and SCP studies using a specific categorization of ET, Bar chart of primary studies using different categories of base learners
RQ2: What are the various applications of ET in SDP/SCP literature?	Application or stage where ET was used	Listing and analyzing the percentage of primary studies that utilized ET for a specific stage/application while developing SDP/SCP models	Finding the most common and sporadic applications of ET while developing SDP/SCP models
RQ3: Which rules/mechanisms are used for combining base models to form ET in SDP/SCP literature?	Rules for formulating the ET used in the study	Categorizing the ET in accordance with aggregation mechanism (Weighing or Meta-learning)	A table listing the various combination rules, the various ET using the specific rules and the number of primary studies using the rule

Table 2 continued

RQ	Data Extracted from Primary Study	Data Analysis	Result
RQ4: What is the performance of ET for various tasks in the domain of SDP/SCP? How does the performance of ET compare to other non-ensemble techniques and amongst each other?	Other non-ensemble techniques used, datasets used, validation method used, performance measures used (such as accuracy, recall, AUC), dataset-wise value of performance measures for the developed SPD/SCP model in the study	Computing boxplots and various descriptive statistics of the extracted performance measure values, Dataset-wise comparison (vote count method) and statistical analysis using Wilcoxon signed-rank test of ET with other non-ensemble techniques based on their application in SDP/SCP domain, Pairwise comparisons using wilcoxon test amongst ET based on their application	Evaluation of performance of ET based on computed statistics and boxplots, Graphs representing the comparative performance of ET with other non-ensemble techniques, Tables representing ET comparison amongst themselves, Superior ET for specific applications
RQ5: What are the various reported threats to validity specific to the use of ET in SDP/SCP literature?	Threats specific to use of ET (only extracted from studies containing "Threats to Validity" or "Limitations" section)	Listing and categorizing (Construct/External/Internal) various threats specific to the scenario when ET were used in SDP/SCP	Recommendations to researchers for planning future studies that minimize the commonly found threats.

3. Results

This section discusses the review results. We first state the overview of the selected studies followed by an analysis to answer the various investigated RQ's. We also discuss and analyze the review results to determine research gaps. This aids in proposing directions for future work in the domain.

3.1. Overview of primary studies

The various steps followed to collect the primary studies have already been mentioned in Sections 2.1–2.3. Figure 1 states the number of studies collected in each step from the various sources (A–F). The year-wise distribution of primary studies is depicted in Figure 2. It may be seen from the figure that there has been a consistent increase in the number of studies using ET in recent years (2016–2020).

Table 3 states all the 77 primary studies along with their study identifier (SI) and cumulative quality score (CS). ES12, ES37, and ES40 are top-scoring primary studies with a CS of 9.5. Amongst the primary studies, the most popularly cited study in accordance with citation count normalized with respect to year was ES3, followed by ES18, ES8, ES26,

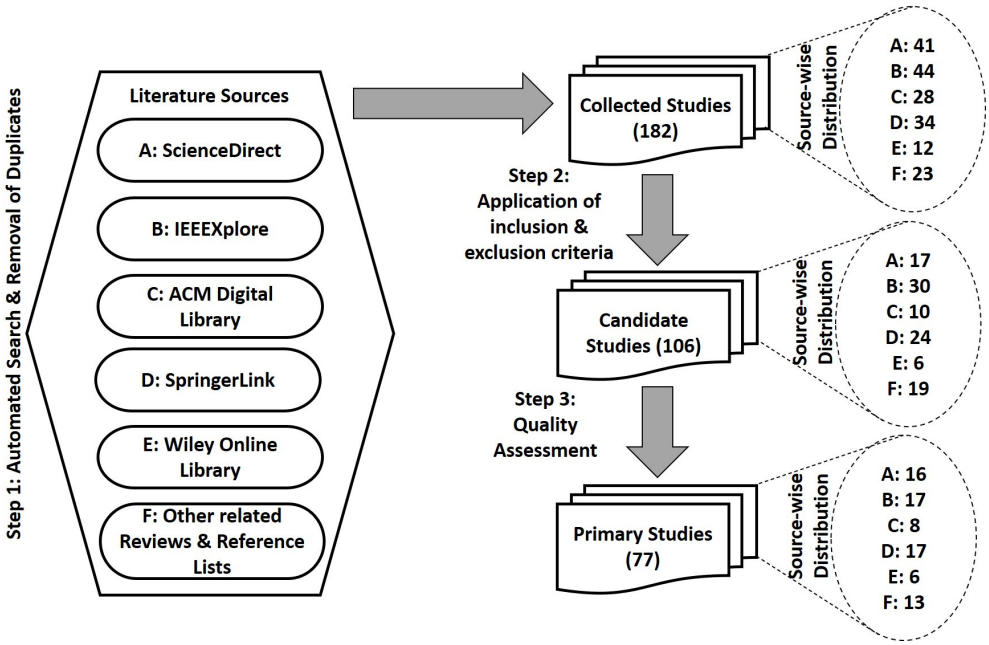


Figure 1. Primary studies collection and source-wise distribution

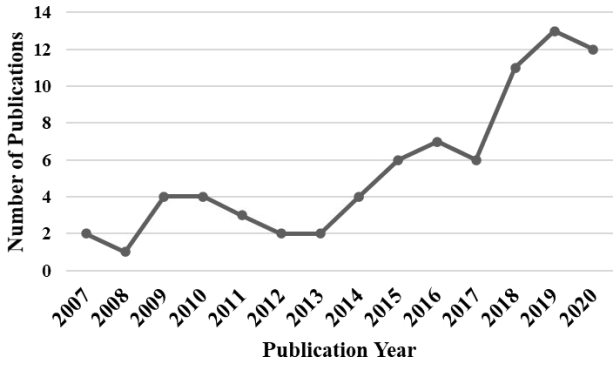


Figure 2. Year-wise distribution of primary studies

and ES35. We also classified the primary studies based on their publication venue. 33% of the primary studies were published in conferences, while 63% of the studies were published in various reputed journals. Three studies were published as chapters. The most popular publication venues were "Information and Software Technology" journal and "Software Quality" journal with six and five studies, respectively. Thereafter, "IEEE Transactions on Software Engineering" was the source of four primary studies. No conference was the source of more than one primary study. It was also noted that 67 primary studies used ensembles for SDP, while only 10 primary studies investigated the use of ensembles for SCP. A similar trend was also observed if we accounted for the rejected studies. Out of the 29 rejected studies, only four studies developed SCP models, all other rejected studies developed SDP models.

Table 3. Primary studies with quality score

SI	Study Name	CS	SI	Study Name	CS	SI	Study Name	CS
ES1	Jiang et al. 2007 [33]	5.5	ES27	Rubinic et al. 2015 [34]	5.5	ES53	Ali et al. 2019 [35]	7.5
ES2	Ma et al. 2007 [36]	7	ES28	Siers and Islam 2015 [37]	6.5	ES54	Campos et. al. 2019 [38]	6.5
ES3	Lessmann et al. 2008 [3]	8	ES29	Li and Wang 2016 [39]	6.5	ES55	Catolino and Ferrucci 2019 [30]	8.5
ES4	Jia et al. 2009 [40]	5.5	ES30	Malhotra 2016 [41]	8	ES56	Gong et al. 2019 [42]	8.5
ES5	Khoshgoftaar et al. 2009 [43]	6	ES31	Ryu et al. 2016 [44]	8.5	ES57	He et al. 2019 [45]	8.5
ES6	Mende and Koschke 2009 [46]	6.5	ES32	Petric et al. 2016 [47]	9	ES58	Kumar et al. 2019 [48]	6
ES7	Seiffert et al. 2009 [49]	8	ES33	Wang et al. 2016a [50]	8.5	ES59	Li et al. 2019a [51]	10
ES8	Arisholm et al. 2010 [52]	6.5	ES34	Wang et al. 2016b [53]	8.5	ES60	Li et al. 2019b [54]	6
ES9	Liu et al. 2010 [55]	8	ES35	Xia et al. 2016 [56]	8.5	ES61	Malhotra and Kamal 2019 [57]	6
ES10	Zheng 2010 [58]	6	ES36	Alsawalqah et al. 2017 [59]	6	ES62	Malhotra and Khanna 2019b [60]	9
ES11	Seliya et al. 2010 [4]	8.5	ES37	Di Nucci et al. 2017 [61]	9.5	ES63	Tong et. al. 2019 [62]	9
ES12	Misirh et al. 2011 [63]	9.5	ES38	Kumar et al. 2017 [64]	7	ES64	Tran et al. 2019 [65]	6.5
ES13	Peng et al. 2011 [66]	6	ES39	Malhotra and Khanna 2017b [67]	7	ES65	Zhou et. al. 2019[68]	9
ES14	Seliya and Khoshgoftaar 2011 [69]	8.5	ES40	Ryu et al. 2017[70]	9.5	ES66	Abbas et al. 2020 [71]	6
ES15	Gao et al. 2012 [72]	5	ES41	Yohannese et al. 2017 [73]	6	ES67	Aljamaan and Alazba 2020 [74]	8.5
ES16	Sun et al. 2012 [75]	8.5	ES42	Agarwal and Singh 2018 [76]	5.5	ES68	Ansari et al. 2020 [77]	7
ES17	Wang et al. 2013 [78]	7.5	ES43	Bowes et al. 2018 [79]	7.5	ES69	Banga and Bansal 2020 [80]	5.5
ES18	Wang and Yao 2013 [81]	8	ES44	Chen et al. 2018 [82]	8.5	ES70	Elahi et al. 2020 [83]	7.5
ES19	Kaur and Kaur 2014 [84]	8.5	ES45	El-Shorbagy et al. 2018 [85]	6	ES71	Goel et al. 2020 [86]	5.5
ES20	Panichella et al. 2014 [87]	9	ES46	Malhotra and Bansal 2018 [88]	8	ES72	Khuat and Le 2020 [89]	7.5
ES21	Rodriguez et al. 2014 [90]	8	ES47	Malhotra and Khanna 2018a [17]	8.5	ES73	Malhotra and Jain 2020 [91]	6
ES22	Suma et al. 2014 [92]	5.5	ES48	Mousavi et al. 2018 [93]	7.5	ES74	Pandey et al. 2020 [94]	9
ES23	Chen et al. 2015 [95]	9	ES49	Moustafa et al. 2018 [96]	6	ES75	Rathore and Kumar 2020 [97]	9
ES24	Elish et al. 2015 [98]	6.5	ES50	Tong et al. 2018 [99]	8.5	ES76	Saifan and Abuwaridh 2020 [100]	7.5
ES25	Hussain et al. 2015 [101]	7	ES51	Zhang et al. 2018 [102]	7.5	ES77	Yucular et al. 2020 [103]	9
ES26	Laradji et al. 2015 [104]	7.5	ES52	Zhu et al. 2018 [31]	8.5			

3.2. Categorization of ET

We first list the various ET used in primary studies along with the study identifier using the specific ET (Appendix, Table A1). An analysis of the Appendix indicates that the most popular ET were Random Forests (RF), Bagging (BAG), AdaBoost (AB) and Voting amongst Heterogenous Base Learners (VHetBL) used in 42%, 32%, 32% and 23% of primary studies, respectively. Thereafter, we categorize the various ET according to five different criteria [105–107] on the basis of base models as shown in Figure 3. We also evaluated the percentage of SDP and SCP primary studies according to the ET used by them corresponding to various categories (Figures 4 and 5). The various categorizations are explained as follows:

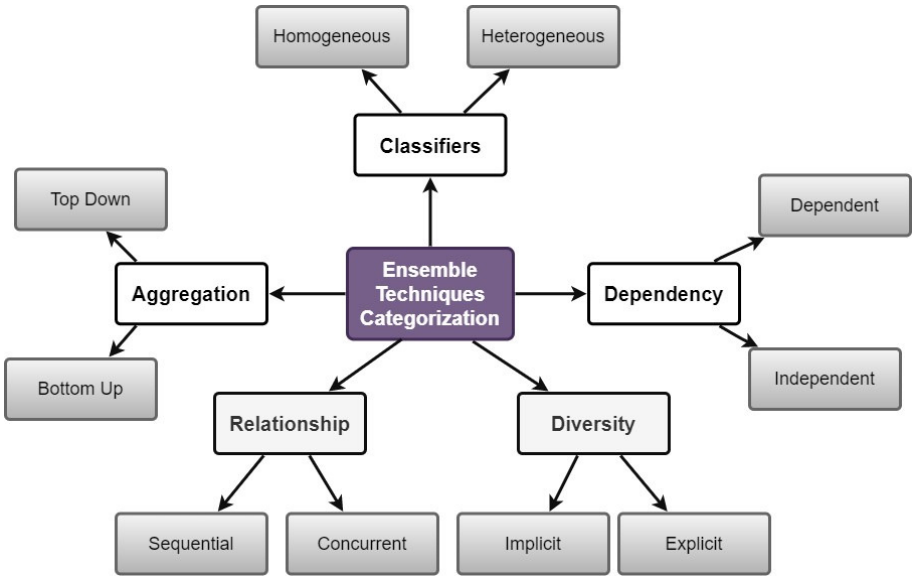


Figure 3. Categorization of ET

1. *Similarity of base models*: This categorization indicates the similarity of learners used to construct the base models in an ensemble (homogeneous or heterogeneous). A homogeneous ensemble combines base models developed using the same data analysis technique. Some examples of the homogeneous ensemble include RF, BAG, LB, AB, Rotation Forest (ROT), Dagging (Dag), Random Subspace (RS), MultiBoost (MBoost), DECORATE, Logit Model Trees (LMT), Double Transfer Boosting (DTB), Adaptive Selection of Optimum Fitness (ASOF), Multiple Kernel Ensemble Learning (MKEL), various other cost-sensitive ensembles like AdaCost, MetaCost (MC), SMOTEBoost (SMBoost), AdaBoost.NC (BNC), Voting amongst Homogeneous Base Learners (VHomBL), etc. On the other hand, a heterogeneous ensemble combines base models developed using diverse data analysis techniques. Examples of heterogeneous ET used in primary studies are voting based ensembles with varied learners for base models, Stacking, Two Stage Ensemble (TSE), Non-Linear Decision Tree Forest (NDTF), Best Training Ensemble (BTE), Combined Defect Predictor (CODEP), Adaptive Selection of Classifiers (ASCI), etc. It may be noted that the Validation and Voting (VV) ensemble is homogeneous in ES9 but heterogeneous in ES36. Also, Omni Ensemble Learning (OEL) used by ES48 is a special category of ET which combines the concept of both homogeneity and heterogeneity. It uses bagging approach along with random oversampling for

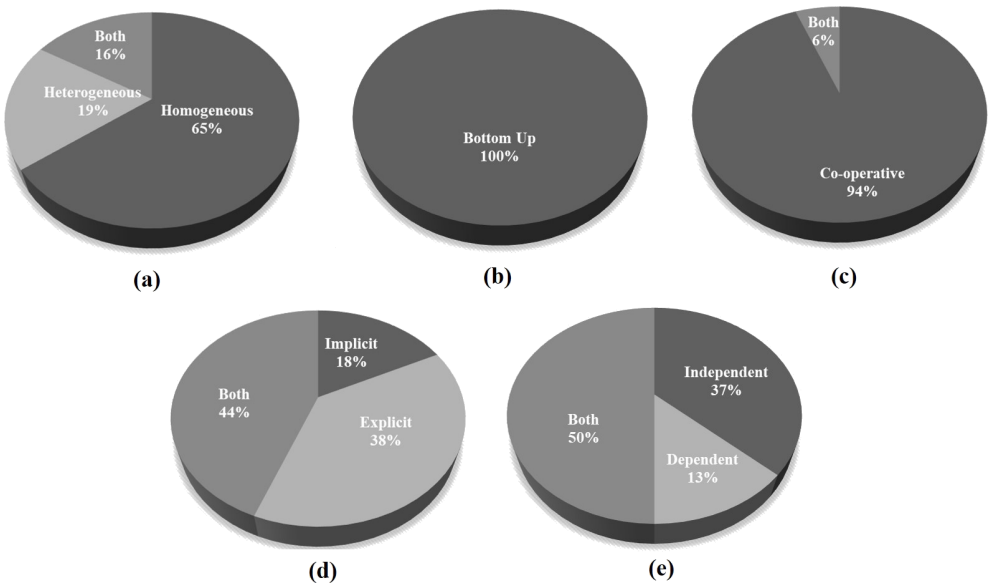


Figure 4. Percentage of SDP primary studies in each category according to: (a) learner similarity, (b) aggregation, (c) relationship, (d) diversity, and (e) dependency categorizations

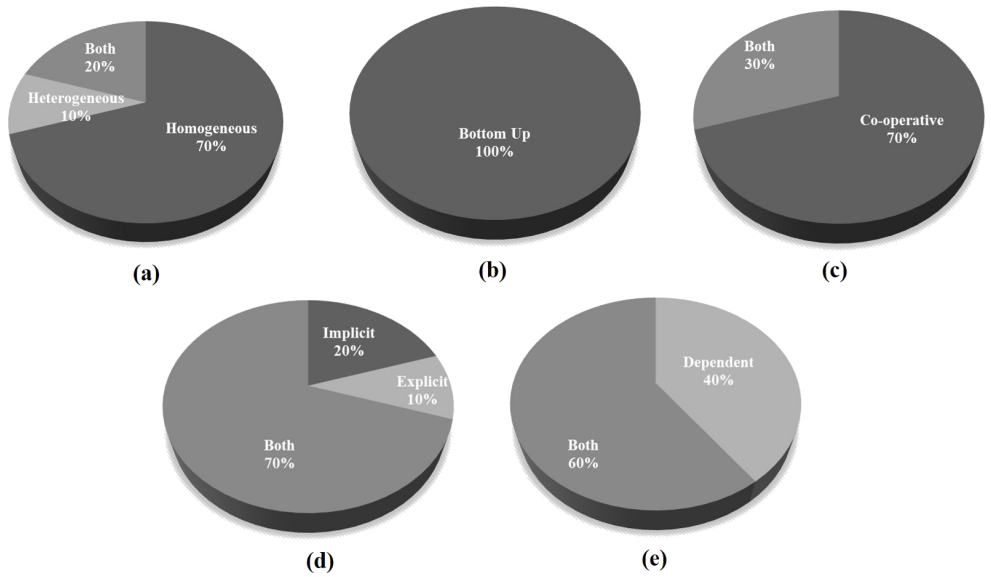


Figure 5. Percentage of SCP primary studies in each category according to: (a) learner similarity, (b) aggregation, (c) relationship, (d) diversity, and (e) dependency categorizations

handling the class imbalance issue. This can be categorized as homogeneous. However, in the next stage for SDP, genetic algorithm is used for ensemble selection amongst 34 different base classifiers. This stage of the OEL is heterogeneous. According to Figure 4a, 65% of the SDP primary studies used homogeneous ET, while 19% of the studies used heterogeneous ET. The other remaining 16% used both heterogeneous and homogeneous ET. On the other hand, as depicted in Figure 5a, 70% of the SCP primary studies used homogeneous ET, 10% used only heterogeneous ET, while 20% used both homogeneous and heterogeneous categories of ET.

2. *Aggregation of base models*: This categorization indicates whether the ensemble aggregation is top-down or bottom-up. A top-down ensemble combines its base models without taking into account the outputs generated by them. On the other hand, in a bottom-up ensemble, the outputs of base models are crucial for aggregating them [108]. Ensembles that use voting or stacking are a subset of bottom-up ensembles [105]. All the ET used in primary studies (both SDP and SCP) are bottom-up ensembles as depicted in figures (Figure 4b and 5b).
3. *Relationship amongst base models*: This categorization suggests the relationship amongst different base models for producing the ensemble output. The base models of an ET could be competitive or cooperative. An ensemble is termed as competitive if only one of the constituent base model is selected to produce the final output [105]. Examples of such ET used in primary studies are ASOF, ASCI, BTE, NDTF, Omni Ensemble Learning, and Multischeme. A co-operative ensemble is the one in which the output of all the constituent base models is combined to produce the final output [106]. All other ET mentioned in Table A2 (Appendix) such as RF, AB, LB, CODEP, BAG, etc. are co-operative ensembles. As shown in Figures 4c and 5c, 94% and 70% of the primary studies used co-operative ET in SDP and SCP, respectively, while 6% of the SDP primary studies and 30% of the SCP primary studies used both co-operative and competitive ET. There was no primary study that used only competitive ET.
4. *Diversity of base models*: This categorization dictates the means for the diversity of base models, i.e., implicit or explicit. Implicit ET employs mechanisms for assuring that the constituent models are diverse [106]. These mechanisms could be random subsamples of the training data or random selection of features etc. Implicit techniques do not measure if diversity is introduced or not. Examples of implicit ET are RF, BAG, SysFor, Multischeme, MC, ROT, RS, Dag, Cost-sensitive Forest, MBoost, Roughly Balanced Bagging (RBBag), Balanced RF, Sampling based Online Bagging (SOB), Oversampling based Online Bagging (OOB) and Undersampling based Online Bagging (UOB). Explicit ET employs a measurement to ensure that the constituent models are different from each other [106]. Examples include AB, LB, DECORATE, AdaCost, etc. Explicit ensembles may also use different base learners to ensure diversity among base learners such as in ASCI, BIT, CODEP, TSE or may use the same learner but with a significant difference in basic configurations (such as different kernels or different fitness variant) as in MKEL and ASOF. Figure 4d depicts 38% of the SDP primary studies used explicit ET, 18% used implicit ET, and the other studies (44%) evaluated both implicit and explicit ET. Amongst the primary studies that use ET in SCP (Figure 5d), majority of the studies (70%) investigated both implicit and explicit ET.
5. *Dependency amongst base models*: This categorization dictates how the various base models interact with each other (dependent or independent). In dependent ET the various base models or consequent iterations of an ensemble interact with each other. A base model constructed later may benefit from the guidance provided by a base model (iteration) constructed earlier [107]. Some of the dependent ET used in primary studies are AB, LB, MKEL, AdaCost, DECORATE, TransferBoost, SMBoost, and DTB. On the other hand, in an independent ET, several base models are constructed in parallel, which are independent of the other base models (iterations). BTE, NDTF, CODEP, Stacking, RF, BAG, MC, ROT, VHomBC, and VHetBC are examples of independent ET used in primary studies [107]. It was observed (Figure 4e) that 37% of the SDP primary studies used independent ET, 13% used dependent ET and 50% investigated both dependent and independent ET. However, there was no SCP primary

study (Figure 5e) that investigated the use of only independent ET, 40% of the studies used dependent ET, while the other 60% used both independent and dependent category of ET.

It may be noted that Coding based Multiclassifier (CEL) proposed by ES15 was a very different type of ET and could not be categorized into the discussed categories.

We also assessed the various categories of base learners used by the ET in each primary study. The base learners were categorized into various families as suggested by [10, 13, 43]. These categories were tree-based learners, support vector machine, Bayesian learners, rule-based learners, instance-based learners, search-based algorithms, artificial neural networks, ensemble learners, logistic regression, and other miscellaneous learners. The base learners which were included in each category are mentioned in Appendix (Table A2). While analyzing the data for base learners we found that 14 primary studies did not mention the base learners used by them. Figure 6 depicts the number of the remaining 63 primary studies which use base learners from a specific family. According to the figure, tree-based learners are the most popular category used by 70% of the studies (both SDP and SCP). Thereafter, Logistic Regression (LR) was used by 56% of the studies. Also, Bayesian learners were used as base learners in 42% of the primary studies. It was interesting to note that ET were themselves used as base learners for constructing other ET in 42% of the studies. We term such techniques as an ensemble of ensembles. Rule-based learners and instance-based learners were less popular as base learners of ET (used in 20% and 30% of studies, respectively). Search based algorithms and miscellaneous learners were the least popular categories as they were each used by only 14% and 19% of the studies, respectively.

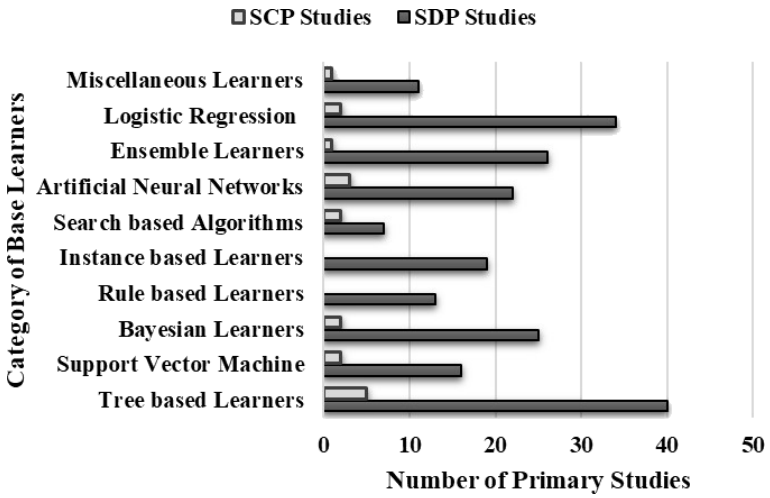


Figure 6. Number of primary studies with specific category of base learners

3.3. Application of ET

All the studies collected in the review develop either SDP or SCP models using ET. However, there are multiple factors that make the task of developing SPD/SCP model difficult and challenging. ET in the collected studies were not only used for model development but for also handling these other critical factors which include existence of large number of features, lack of defect-prone or change-prone instances making the training data imbalanced, evaluating prediction in realistic online scenarios or unavailability of appropriate training

data. We investigated the primary studies to ascertain the various applications of ET, i.e., what was the underlying use of ET in SDP/SCP. The various applications are listed as under along with the percentage of primary studies that utilized the ET for the particular application.

- As a learning algorithm for developing the SDP model (65%).
- As a learning algorithm for developing the SCP model (13%).
- Addressing the class imbalance issue (37%).
- Transfer learning (10%).
- Online Learning (3%).
- Feature selection (1%).

As indicated above, the majority of the studies (65%) used ET as learners for developing SDP models. On the other hand, only 13% of the primary techniques used ET as learning algorithms for developing SCP models. ET used for these two applications included RF, LMT, AB, BAG, LB, DECORATE, VHetBL, VHomBL, VV, ROT, CODEP, Stacking, NDTF, BTE, RS, Dag, XGBoost, ASCI, ASOF, etc. It may be noted that 85% of the primary studies developed within-project models for SDP/SCP using validation techniques such as hold-out validation, k-fold cross-validation, or inter-release validation. However, there may be a scenario where previous data related to the same project may be unavailable or difficult to collect [52]. For such a situation, researchers suggest the use of cross-project models [86, 87, 102]. A critical issue in developing these models is that varied projects may have different data distributions or different metric sets. To overcome such issues, a transfer learning mechanism, which derives common observations and expertise from the available projects and transfers it to the target project [56, 70] is proposed by the research community. This is a relatively recent application of ET as the first primary study which used ET for transfer learning was published in 2014 (ES32). As mentioned above, 10% of the primary studies used ET for transfer learning. These ET were TransferBoost, Improved Transfer Adaptive Boosting (ITrAdaBoost), Kernel Spectral Embedding Transfer Ensemble (KSETE), TSE, Value Cognitive Boosting with Support Vector Machine (VCB-SVM), DTB, VHetBL, and TransferCostSensitive Boosting (TCSBoost).

Another critical issue while developing SDP/SCP models is the presence of imbalanced training data [20, 57, 67, 109]. In general, a standard classifier assumes that each class is present in equal proportion, i.e., there is an equal number of defect-prone/change-prone and not defect prone/not change prone classes in a dataset. This assumption hinders the development of an effective SDP/SCP model as the class distributions in actual datasets are biased. This results in erroneous identification of the minority class instances. Therefore, a popular application of ET in the primary studies was using them for addressing the class imbalance issue (37% of primary studies). As proposed by Galar et al. [20], we categorized the ET used for class imbalance in primary studies into Cost-sensitive ET, Boosting-based ET, Bagging-based ET, and Hybrid ET. Boosting based and Bagging based ensembles combine Bagging and Boosting with data preprocessing techniques such as random undersampling, SMOTE, oversampling, etc. Hybrid ensembles combine both bagging and boosting techniques along with data preprocessing techniques. Furthermore, we also mention a category of ET namely "Novel" ET, which are proposed by primary studies but could not be categorized into the above categories.

- *Cost-Sensitive ET*: MetaCost, AdaCost, Csb2, Adc2, Dynamic Adaboost.NC (DNC), Adaboost.NC (BNC), Cost-Sensitive Forest, Cost-sensitive Boosting Neural Networks, MKEL, TCSBoost.

- *Boosting based ET*: SMBoost, RUSBoost, WeightedSmoteBoost, SelectRUSBoost, Non-negative Sparse based Semiboost, DataBoost, MSMOTEBoost.
- *Bagging based ET*: SOB, OOB, UOB, RBBag, OEL.
- *Hybrid ET*: MBoost, Ensemble Random Undersampling
- *Novel ET*: CEL, KSETE, TSE, Bug Prediction using Deep representation and Ensemble learning.

It may be noted that there were five studies (ES32, ES40, ES56, ES59, ES63) which proposed ET (VCB-SVM, TCSBoost, ITrAdaBoost, TSE, KSETE) dealing with both the important issues, i.e., handling class imbalance and transfer learning for cross-project defect prediction. However, only one study ES14 proposed the SelectRUSBoost technique, which incorporates feature selection and class imbalance learning. Feature selection involves choosing a subset of features (independent variables) that develops SDP/SCP models with good predictive capability. Two studies (ES17 and ES54) used ET for online learning. ES54 used ET for online failure prediction, where past data is correlated with the existing state of the system to predict the occurrence of faults in near future. The study by Wang et al. [78], i.e., ES17, deals with the scenario where data continuously arrives in streams, and the training data is constantly updated with new data. There are multiple runs of time sensitive prediction which leads to better prediction models that are less biased [110]. ES17 also addressed the issue of class imbalance and proposed the SOB technique. As there are very few studies that propose comprehensive ET which deal with multiple issues simultaneously, more such techniques should be proposed and validated in future studies.

3.4. Rules/mechanisms for combining base models

As ET are a combination of different base models, this RQ concerns itself with the mechanism of aggregating the outputs of constituent base models. We found that the base models were combined either by giving appropriate weights to the output of constituent base models or through the process of meta-learning. The construction of a meta-learning ensemble generally involves multiple learning stages. The outputs of constituent base models act as inputs to the meta-learner, which is responsible for producing the final ensemble output. The ET which use meta-learning as a combination mechanism were ASCI, ASOF, MKEL, MC, Ensemble Selection, Grading, OEL, Stacking, CODEP, GcForest, DeepForest and NDTF. All other ET used the weighing mechanism for combining the base models. However, there were two exceptions, which we could not categorize properly into weighing or meta-learning mechanisms. These were: a) CEL (ES16), which used a specific coding mechanism for aggregation, and b) LMT, though many LB iterations were performed in LMT, there was only one resultant tree at the end. Table 4 lists the various combination rules used by the ET, the number of ET, and the number of primary studies using the specific combination rule. According to the Table, the most popular combination rule was “majority voting” amongst the base models, which was used in 78% of the primary studies. The ET which used this rule were BAG, RF, UOB, OOB, SOB, RS, SysFor, Balanced RF, VHetBL, VHomBL, VV, Extra Trees and Dag. The next popular combination rule was “weighing based on misclassification error of the base model” used in 56% of the primary studies. This is a combination rule generally used by several ET using the Boosting mechanism such as AB, LB, VCB-SVM, SMBoost, WeightedSmoteBoost, SelectRUSBoost, MBoost, etc. and some others like MKEL. Another popular combination rule was “Average Probability”, which was used in 14 primary studies.

Table 4. Combination rules

Combination rule	Number of ET	Number of primary studies
Average Probability	9	14
Selection of Best	2	4
Majority Vote	13	60
Maximum Confidence Score	1	1
Weighing based on misclassification error and cost adjustment	4	4
Weighing based on misclassification error of the base model	22	45
Weighted adjustment based on misclassification error and penalty for ambiguity	2	4
Weighing based on MCC obtained by the base model	1	1
Weighing based on data distribution of target data, misclassification error, and cost adjustment	2	2
Weighing based on error on the prediction of instances in the training target data	1	2
Voting based on cost-sensitive labeling of records	1	1
Weighing of base models that lead to objective function (inconsistency between labels and similarities) minimization	1	1
Weighted adjusted probabilities and probability adjustment	1	3
Weighing based on predictive performance on other data	1	2
Weighing based on the predictive capability to predict hard instances	1	2
Weighing based on predictive performance on other data and ability to predict hard instances	1	2

3.5. Performance of ET

It is crucial to evaluate the effectiveness of ET for SDP and SCP. To do so, we analyze the performance measures of the developed SDP and SCP models by various ET in the primary studies. An analysis of the performance measures used in the primary studies indicates AUC as the most widely used performance measure (65%) in these studies. Moreover, the use of AUC for assessing the performance of predictive models has also been propagated in literature studies as it is capable of handling skewed datasets with disparate class distributions and the dissimilar cost of various classification errors [67, 111]. AUC is computed by plotting recall and 1-specificity on the y-axis and x-axis, respectively and estimating the area under the plotted curve. Apart from AUC, Recall and F-measure performance measures were found to be popularly used in the primary studies. While recall states the percentage of correctly identified defect-prone/change-prone classes, it gives us no insight into the number of incorrectly identified defect-prone/change-prone classes. However, this measure has been used in many previous review studies [10, 13, 23, 24, 27] to assess the performance of the developed SDP/SCP models. Thus, we include it for assessing the predictive capability of ET. On the other hand, F-measure, which is computed as the harmonic mean of precision and recall has not been included in our analysis. Menzies et al. [112] have criticized the use of precision due to its unstable nature, thus, raising

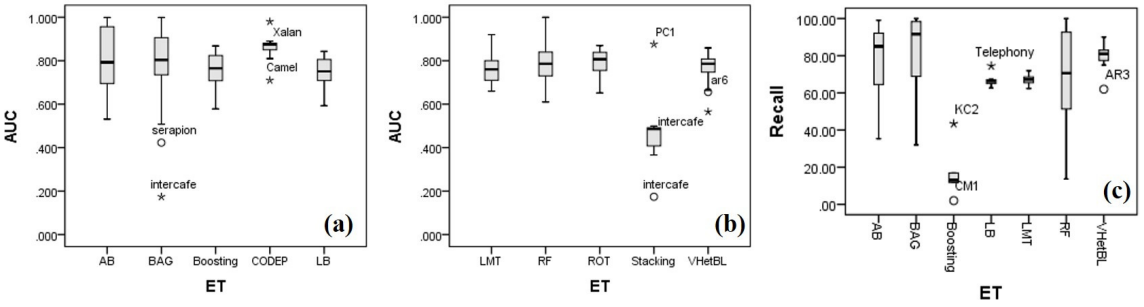


Figure 7. Dataset-wise boxplots when ET is used as a learning algorithm for developing SDP models (a), (b) AUC (c), recall

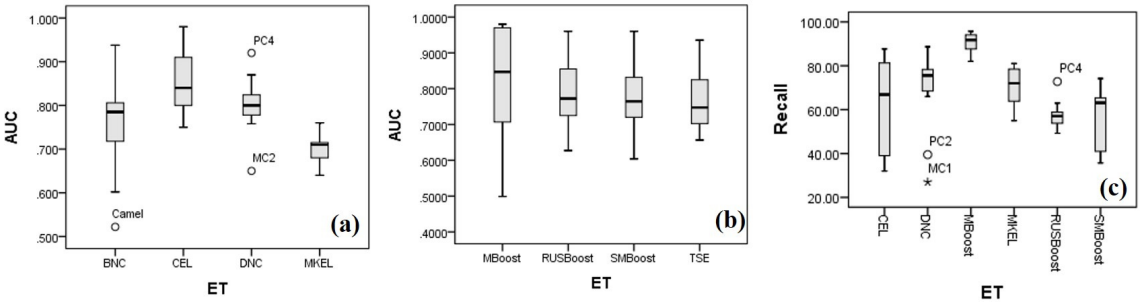


Figure 8. Dataset-wise boxplots for ET that handle class-imbalance issue (a), (b) AUC (c), recall

concerns over use of F-measure as a performance evaluator [10]. He and Garcia [109] have also doubted the capability of F-measure while "comparing the performance of different classifiers over a range of sample distributions". Therefore, we analyze two performance measures, AUC, and Recall for determining the predictive performance of ET.

The performance of ET was analyzed dataset wise. Similar to [10], we found that the most popular datasets used in SDP were NASA datasets (CM1, JM1, KC1, KC2, KC3, KC4, MC1, MC2, MW1, PC1, PC2, PC3, PC4, PC5) and Promise repository datasets such as AR5, AR6, Jedit (http://promise.site.uottawa.ca/SERepository/datasets-page.html). Some other open-source datasets (Gate, Intercafe, Lucene, Xalan, Tomcat, Synapse, Velocity, etc.) and application package datasets (Bluetooth, Contacts, Email, Calendar, Telephony, etc.) from the Android operating system were also used by primary studies for SDP. Similar to SDP, various open-source software datasets were used by SCP studies such as ArgoUML, FreeMind, Eclipse, Ant, Lucene, Gate, KolMafia, etc. and datasets from the Android operating system.

We analyze the performance of only those ET whose performance measures (AUC and Recall) could be extracted from at least two or more studies and have been validated on at least three or more datasets. This was done to yield generalized results and comparisons across studies. The performance of ET was assessed with respect to datasets and outlier values (Figs 7–9) were disposed off. Thereafter, various statistics were reported. By using such rules, an ET that might have shown exceptional performance in just one study or on specific datasets will not be designated as a good performer in the SDP/SCP domain. As ET have several parameter values such as the number of base models, different base learners, etc., their performance values may vary a lot. However, we are interested in finding the best values of ET. Thus, we use the following rules while extracting the values of performance measures:

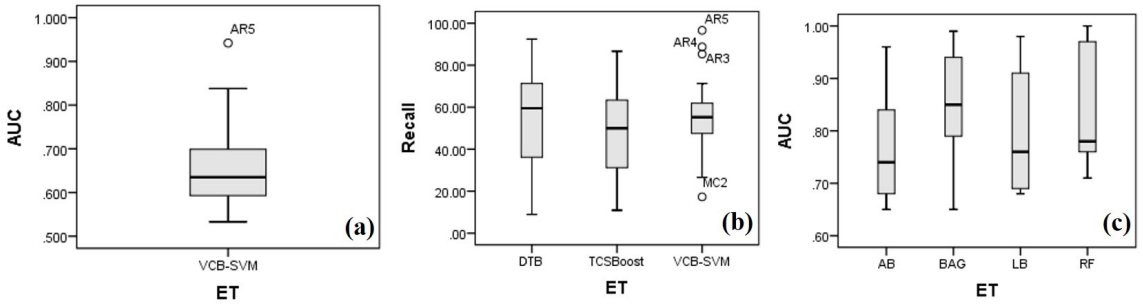


Figure 9. (a), (b) Dataset wise AUC and recall boxplots for ET used for transfer learning, (c) dataset wise AUC boxplot for ET used as a learning algorithm for developing SCP models

– If an ET has been evaluated on the same dataset, more than once in a study (maybe with different internal parameter settings), we report the highest values.
– If two studies have evaluated an ET on the same dataset, we again report the highest values amongst the two studies for the particular dataset.

The rules for extracting the performance measures are similar to the ones used by [13, 24].

3.5.1. Assessment of performance statistics of ET

We grouped the ET according to the various applications they were used for (Section 3.3). Figures 7–9 depict the boxplots of the ET for AUC and Recall performance measures. The outliers are labeled with their corresponding dataset names. As some ET were used for more than one application, we segregated the studies according to the application the ET was used for to construct the boxplots. For instance, AB, BAG, LB, and RF were used both as learning algorithms for developing SDP and SCP models. Figures 7a and 7b depicts the boxplots when they were used for developing SDP models and Figure 9c depicts the boxplot when they were used for developing SCP models. TCSBoost has two applications (addressing class imbalance and transfer learning). However, we include it in the Transfer learning application for simplicity. The performance measure values for VHetBL could be extracted for two applications: as a learning algorithm for developing SDP models and for transfer learning but since we could extract the values for transfer learning application in only one study for VHetBL, we only reported the values for its application as a learning algorithm for developing SDP models. Similarly, the performance measures values for TSE could be extracted for two applications, i.e., for transfer learning and for addressing the class imbalance issue. But since only one study reported values for the transfer learning application, we include TSE as an algorithm for addressing the class imbalance issue.

Table 5 reports the AUC and Recall statistics of ET for various applications. These statistics were computed after removing the outliers depicted in Figures 7–9. The table reports the minimum, maximum, mean, median, standard deviation, and the count of the number of datasets from which the statistics are computed for each ET. According to the table, apart from Stacking and MKEL all ET depicted a mean AUC score of 0.75 or above for three of the discussed applications namely as a learning algorithm for developing SDP and SCP models and for addressing the class imbalance issue. This indicates the favorability of ET for these applications. Though, a little lower, but ET for transfer learning showed AUC mean values of 0.65 indicating they could be effective for it. However, we could analyze the AUC values for just one ET in the transfer learning domain. CODEP,

Table 5. Performance measure statistics of ET for various applications

ET	Performance measure	Dataset count	Mini-mum	Maxi-mum	Mean	Median	Standard deviation
Learning algorithm for developing SDP models							
AB	AUC	40	0.53	0.99	0.81	0.79	0.13
	Recall	20	35.34	99.00	78.63	85.00	16.43
BAG	AUC	42	0.51	0.99	0.81	0.81	0.11
	Recall	24	32.00	100.00	84.16	91.65	17.40
Boosting	AUC	18	0.58	0.87	0.76	0.77	0.07
	Recall	3	11.80	16.90	13.90	13.00	2.17
CODEP	AUC	8	0.81	0.89	0.87	0.88	0.02
LB	AUC	28	0.60	0.84	0.75	0.75	0.07
	Recall	6	62.70	67.22	65.54	66.09	1.61
LMT	AUC	17	0.66	0.92	0.76	0.76	0.07
	Recall	7	62.30	71.95	67.05	67.14	2.95
RF	AUC	55	0.61	1.00	0.80	0.79	0.10
	Recall	46	13.70	100.00	68.06	70.60	26.41
ROT	AUC	20	0.65	0.87	0.79	0.81	0.06
Stacking	AUC	13	0.37	0.50	0.46	0.49	0.05
VHetBL	AUC	14	0.66	0.86	0.78	0.79	0.05
	Recall	6	75.00	90.00	82.00	81.00	4.61
Handling class imbalance issue							
BNC	AUC	12	0.60	0.94	0.78	0.79	0.08
CEL	AUC	17	0.75	0.98	0.86	0.84	0.07
	Recall	14	32.00	87.67	61.21	66.86	20.69
DNC	AUC	9	0.76	0.87	0.81	0.80	0.03
	Recall	10	66.00	88.70	76.71	76.42	5.78
MKEL	AUC	8	0.64	0.77	0.71	0.71	0.04
	Recall	12	55.00	81.00	70.59	72.07	8.93
MBOOST	AUC	18	0.50	0.98	0.83	0.85	0.13
	Recall	8	82.05	95.74	90.59	91.73	4.56
RUSBoost	AUC	16	0.63	0.96	0.77	0.79	0.09
	Recall	8	49.20	63.00	55.98	56.35	4.11
SMBoost	AUC	16	0.60	0.96	0.78	0.77	0.09
	Recall	9	35.70	74.20	55.53	63.10	13.44
TSE	AUC	12	0.66	0.94	0.76	0.75	0.08
Transfer learning							
DTB	Recall	33	8.90	92.40	55.56	59.50	21.61
TCSBoost	Recall	33	10.90	86.60	48.05	50.00	21.73
VCB-SVM	AUC	33	0.53	0.84	0.65	0.63	0.08
	Recall	30	26.60	71.30	53.23	54.75	9.53
A learning algorithm for developing SCP models							
AB	AUC	13	0.65	0.96	0.77	0.74	0.09
BAG	AUC	15	0.65	0.99	0.85	0.85	0.09
LB	AUC	13	0.68	0.98	0.79	0.76	0.11
RF	AUC	13	0.71	1.00	0.85	0.78	0.11

BAG and ROT depicted the best median AUC values 0.88, 0.81 and 0.81, respectively for their use as a learning algorithm for developing SDP models. BAG also attained the best median AUC value (0.85) for its use as a learning algorithm for SCP models. MBoost depicted the best median AUC values (0.85) for handling imbalanced datasets.

An analysis of the mean Recall values stated in Table 4 indicates that the values range from 60%–90% in the majority of the cases except for Boosting and some of its variants such as DTB, RUSBoost, SMBoost, VCB-SVM, and TCSBoost. These ET depicted poor recall values which were in the range of 48%–55%. On the other hand, other ET based on boosting mechanism such as AB and MBoost depicted exceptionally good median recall values (AB: 85.00%, MBoost: 91.73%). Thus, future studies should continue to explore and validate various ET based on the boosting mechanism. The statistics of the AUC and Recall values reported in Table 5 confirm the capability of ET for the various discussed applications. Moreover, the use of ET should be encouraged for these applications.

3.5.2. Comparative performance of ET with other techniques

To ascertain the effectiveness of ET, it is important to compare them with other non-ensemble techniques. The rules for extracting data for comparison were similar to the ones stated in Section 3.5.1. As discussed previously, the comparison was done with respect to the application the ET is used for. In order to perform the comparison, we used only the AUC performance measure due to its robustness and stability. We do not compare the techniques based on recall values as analyzing recall may not give a comprehensive picture in the case of imbalanced datasets [67] and since our comparison is dataset wise we should select the performance criteria for comparison wisely. It may be noted that we could extract relevant data for comparing only two applications of the ET, i.e., as a learning algorithm for developing SDP models and as a learning algorithm for developing SCP models.

We compared 10 ET (AB, BAG, LB, LMT, RF, Boosting, CODEP, ROT, Stacking, VHetBL) with 10 non-ensemble techniques (Artificial Neural Network (ANN), Bayesian Network (BN), Decision Tree C4.5, Classification and Regression Tree (CART), Decision Table (Dec.T), K-Nearest Neighbor (KNN), LR, Naïve Bayes (NB), Support Vector Machine (SVM), Voting Feature Intervals (VFI)) to assess their capability as a learning algorithm for developing SDP models. The non-ensemble techniques that were chosen for comparison were based on two criteria: a) the data for comparison could be extracted so that ET could be compared on at least 3 or more datasets, which were used in at least 2 or more studies b) they should represent the various categories of learners as depicted in Figure 6. The chosen non-ensemble techniques were representative of support vector machine, i.e., SVM, artificial neural networks, i.e., ANN, tree-based learners (C4.5 and CART), Bayesian learners (NB and BN), rule-based learner, i.e., Dec.T, instance-based learner, i.e., KNN, statistical learner, i.e., LR and miscellaneous learner, i.e., VFI. The comparison was performed using vote count method (dataset wise), i.e., we computed the number of datasets (votes) on which a specific ET is better than a specific non-ensemble technique and the number of datasets (votes) on which a specific ET is worse than a specific non-ensemble technique in terms of AUC value. These results are depicted in Figure 10. For instance, in Figure 10a, the AUC value of AB was better than BN in 17 datasets, while the AUC value of AB was worse than BN in 9 datasets. This means 17 votes favor AB and 9 votes are against AB, when compared with BN. Similarly, in Figure 10b, the AUC value of Boosting was better than C4.5 by 12 votes and there were 6 votes against Boosting when compared with C4.5.

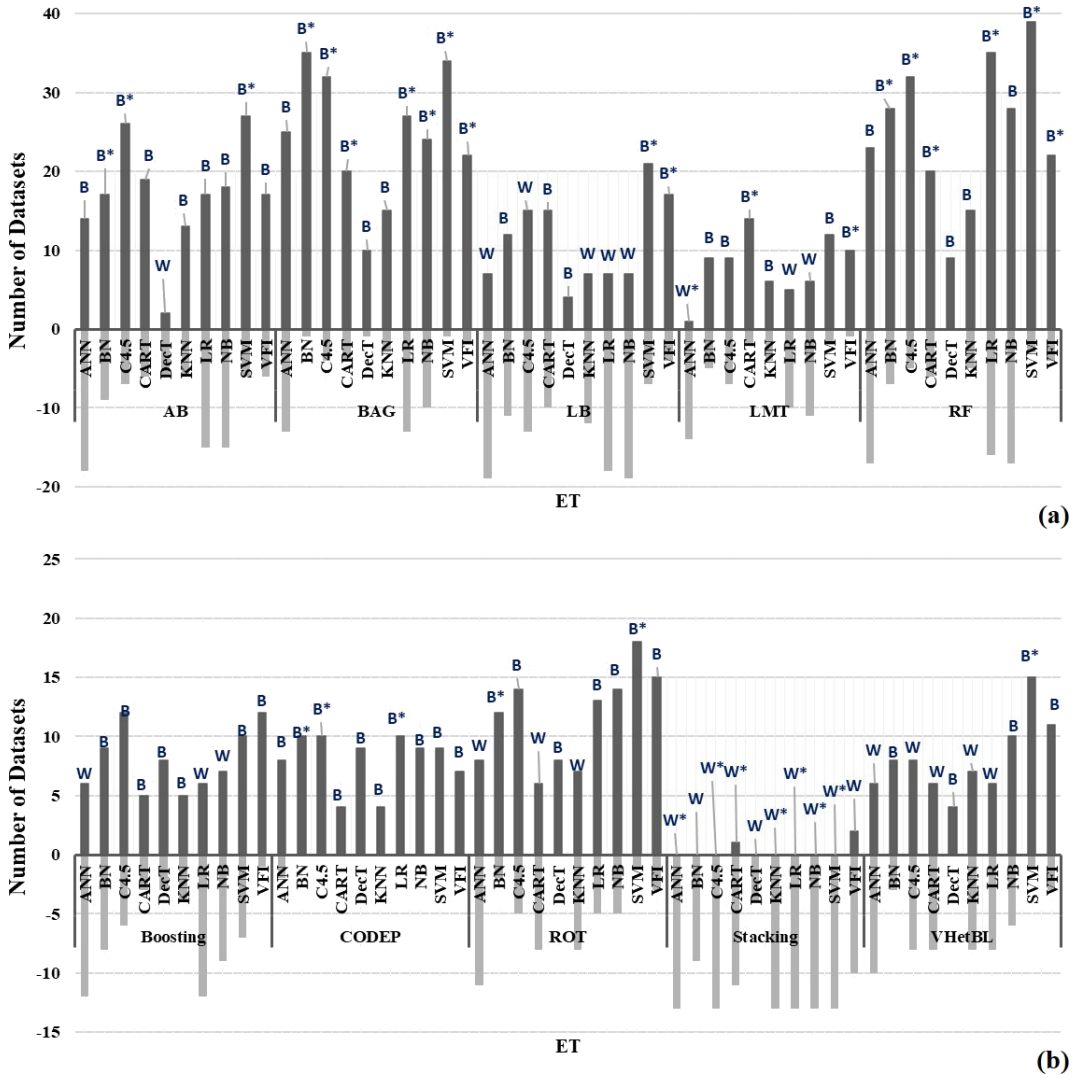


Figure 10. Comparison results of ET with non-ensemble techniques when they are used as a learning algorithm for developing SDP models

Apart from comparing the results dataset wise, we also performed a statistical analysis of the comparison between an ET and the various chosen non-ensemble techniques. Wilcoxon signed-rank test with Bonferroni correction was conducted to pairwise compare the AUC values of an ET and the various other compared techniques. Though, while conducting pairwise comparisons paired t-test is a common choice, Wilcoxon signed-rank test is considered safe as it does not require the underlying data to follow normal distribution. Moreover, in case of Wilcoxon signed rank test, its outcome is generally less influenced by exceptionally superior or inferior performance of a technique corresponding to a dataset (i.e., an outlier) [113]. These reasons favor the use of the test for the comparison. The test was conducted at an α value of 0.05. The results of these pairwise comparisons are also depicted in Figure 10 (at the top of data columns). If an ET fared significantly better than the compared non-ensemble technique it was depicted as B*, however, if the ET was better but the results were not significant it was depicted as B at the top of the data column (in Figure 10). Similarly, if the ET turned out to be significantly poorer than the compared non-ensemble technique, it was depicted as W* and if it was worse but not significantly,

it was depicted with the symbol W. For instance, according to Figure 10a, RF is better than ANN, Dec.T, KNN and NB. But these results were not significant. However, RF was significantly better than BN, C4.5, CART, LR, SVM, and VFI according to the Wilcoxon signed-rank test. As depicted in Figure 10b, the AUC values of ROT on various datasets were found to be worse than ANN, CART, and KNN, but not significantly.

According to the results shown in Figure 10, BAG, RF and CODEP exhibited the best AUC values as they were better than all the compared non-ensemble techniques, and more so significantly better than 7, 6 and 3 non-ensemble techniques, respectively. Thereafter, the results of AB, Boosting, and ROT were also good as they were better than the majority of the compared non-ensemble techniques and were only not significantly worse than a maximum of three compared techniques. The results of Stacking were quite poor as it was found worse than the majority of the compared techniques. It was interesting to note while comparing different ET that Stacking, a heterogeneous ET showed the worst results. However, VHetBL and CODEP other heterogeneous techniques showed encouraging results. Though, VHetBL, uses the weighing mechanism for aggregation, both CODEP and stacking use meta-learning as combination mechanisms.

We also compared the performance of ET as a learning algorithm for developing SCP models. However, we could only extract the AUC results of three non-ensemble techniques (ANN, LR, and NB) to be compared with four ET namely AB, BAG, LB, and RF. Similar to the comparison performed for the application as a learning algorithm for developing SDP models, we compared the AUC values dataset wise and performed Wilcoxon signed-rank test with Bonferroni correction. The results of the comparison and the statistical test are indicated in Figure 11. According to the figure apart from the case when AB was compared with ANN, all other ET were found better than the compared non-ensemble techniques. Wilcoxon test results indicated BAG to be the best as it was significantly superior than all the compared techniques.

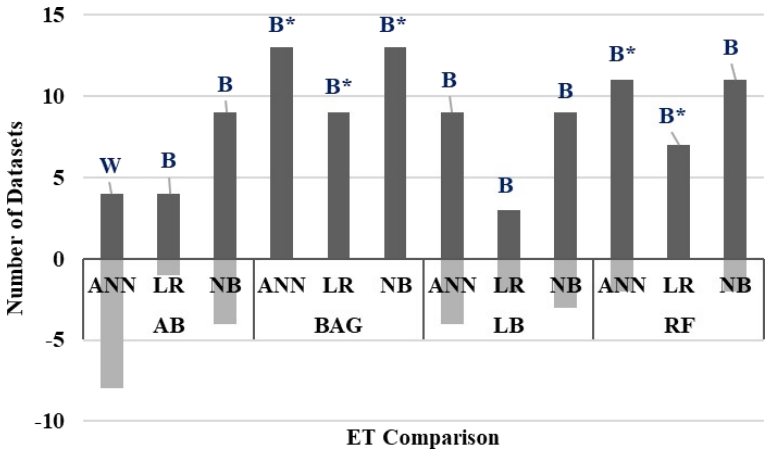


Figure 11. Comparison results of ET with non-ensemble techniques when they are used as a learning algorithm for developing SCP models

The results discussed in the section indicate that the AUC values of the majority of the analyzed ET were found better than the non-ensemble techniques. The primary reason for the good performance of ET is that they combine multiple learners and give stable results as compared to single learners. Also, the various base models of ET are of diverse nature, i.e., several base classifiers are combined that explore a "set of hypotheses" as an

alternative to a single model that searches the "best hypothesis" [114]. This mechanism thereby improves the performance of ET as compared to non-ensemble techniques that are single classifiers. The results discussed in the section favor the use of ET for the explored applications and in other related domains.

3.5.3. Comparative performance amongst ET

As indicated in the previous two sections, ET have been found effective in the SDP/SCP domain for doing various tasks. Furthermore, we intend to evaluate if a specific ET outperforms others in the various applications where they are used. We pairwise compare the AUC values (dataset wise) of all the ET amongst each other for each application and analyze whether a specific ET is significantly better than the majority of other ET for a particular application. The rules for comparing different ET and extracting the AUC values are similar to the ones mentioned in the previous sections. For comparison amongst ET we use Wilcoxon signed-rank test with Bonferroni correction at an α value of 0.05.

Tables 6–8 state the Wilcoxon test results when we compare ET amongst each other, which are used for applications, i.e., as a learning algorithm for developing SDP models, for addressing the class imbalance issue, and as a learning algorithm for developing SCP models respectively. The tables use the following symbols:

- B: When the results of ET depicted in the row is found better than the results of ET depicted in the column. However, not significantly.
- B*: When the results of ET depicted in the row are significantly better than the results of the ET depicted in the column.
- W: When the results of ET depicted in the row are found worse than the results of ET depicted in the column. However, not significantly.
- W*: When the results of ET depicted in the row are significantly worse than the results of the ET depicted in the column.
- EQ: When both the compared ETs get equivalent results, i.e., neither worse nor better.
- ND: When the data to compare the ETs could not be extracted from the primary studies.

It may be noted that we could not compare ET for the application of transfer learning as we could extract AUC statistics for only one technique (VCB-SVM) for this application.

Table 6. Comparison amongst ET for use as a learning algorithm for developing SDP models (Wilcoxon test results)

	AB	BAG	Boost-ing	CODEP	LB	LMT	RF	ROT	Stack-ing	VHetBL
AB	–	W	B	W	B	B	W	B	B*	B
BAG	B	–	W	W*	B	B	W	B	B*	B
Boost-ing	W	B	–	W	B	W	W	W	B	W
CODEP	B	B*	B*	–	B	ND	B	B	B	B
LB	W	W	W	W	–	W	W	W	B*	W
LMT	W	W	B	ND	B	–	W	W	B	W
RF	B	B	B	W	B	B	–	W	B*	B
ROT	W	W	B	W	B	B	B	–	B*	B
Stacking	W*	W*	W	W	W*	W	W*	W*	–	W*
VHetBL	W	W	B	W	B	B	W	W	B*	–

Table 6 compares the ET that were used as a learning algorithm for developing SDP models. The Wilcoxon test results in Table 6 depict that the Stacking technique was found worse than all the compared ET. On the other hand, the results of CODEP, another heterogenous ensemble that uses the meta-learning combination mechanism was found better than all the other compared ET. Apart from Stacking and CODEP, all the other compared ET use the weighing mechanism for combining the base models. Amongst the ET that used weighing as a mechanism for combination, LB exhibited the worst results. It was only found better than Stacking. The best results were shown by the RF technique as it fared better than seven of the other compared ET. However, it may be noted that the results of RF was significantly better in only one case (when compared with stacking). After CODEP and RF, ROT, AB and BAG showed good results as they were found better than 5 or more compared ET.

Table 7 depicts the comparison results (Wilcoxon test) amongst ET which were used for specifically handling the class imbalance issue. According to the Table, the MBoost technique was the best as its AUC results were better than seven of the compared ET, moreover significantly better than three ET (MKEL, RUSBoost, SMBoost). The next best results were shown by the CEL technique, which was found better than all the other compared ET except MBoost. The worst ET according to the table was MKEL, as its AUC values were poorer than all the compared ET. The poor performance of MKEL could be due to the random selection strategy used to initialize the training set. It is a possibility that no defective samples were selected in the initial training set, thus leading to poor results [50]. RUSBoost was found better than only MKEL and TSE while SMBoost was found better than three ET (MKEL, TSE, and RUSBoost). It is interesting to note that MBoost, a combination of wagging and boosting gives exceptionally good results however, just boosting when combined with a sampling technique (such as SMOTE (SM) or Random Undersampling (RUS)), though handles the class imbalance issue, but fares poorer than most of the other explored ET in the domain.

Table 7. Comparison amongst ET for handling class imbalance issue (Wilcoxon test results)

	BNC	CEL	DNC	MKEL	MBoost	RUS-Boost	SM-Boost	TSE
BNC	–	W	B	B	W	EQ	B	B
CEL	B	–	B	B*	W	B*	B*	B
DNC	W	W	–	B	W	B	B	B
MKEL	W	W*	W	–	W*	W	W	W
MBoost	B	B	B	B*	–	B*	B*	B
RUS-Boost	EQ	W*	W	B	W*	–	W	B
SMBoost	W	W*	W	B	W*	B	–	B
TSE	W	W	W	B	W	W	W	–

Table 8 states the Wilcoxon test results of the comparative performance of ET when used as a learning algorithm for developing SCP models. According to the table, the best ET was BAG as it was better than all the other compared ET, moreover, the results were significant in two out of three cases. The AB technique exhibited poor AUC values and was found significantly worse in all the comparisons. Similarly, the LB technique also showed poor results than most of the other compared ET. The RF technique also exhibited good results for this application.

Table 8. Comparison amongst ET for use as a learning algorithm for developing SCP models (Wilcoxon test results)

	AB	BAG	LB	RF
AB	–	W*	W	W*
BAG	B*	–	B*	B
LB	B	W*	–	W*
RF	B*	W	B*	–

According to the results, we find that RF and BAG turned out to be a superior technique as a learning algorithm for developing both SDP and SCP models. BAG creates multiple bootstrap training samples for developing diverse base models. RF combines the bootstrap samples used in bagging along with random features to create diverse base models. Though, CODEP technique also showed good results, it was evaluated in only three primary studies. However, both BAG and RF have been widely used in literature studies for various applications in the SDP/SCP domain. A key reason for their popularity is their effective performance and ease of availability, i.e., open-source tools such as WEKA [115], etc. have efficient implementations of BAG and RF. On the other hand, ET like CODEP, ROT, MBoost, and CEL though exhibited good results in different applications are rarely used in SDP/SCP literature. This could be possibly because of the lack of tools that provide their implementation. These techniques should be widely explored in future studies. It may also be noted that though the comparison results indicate that Stacking, TSE and MKEL performed worse than the majority of the other compared ET, but there were very few studies that could provide data for comparing these ET. Thus, these results are not necessarily true in all scenarios. Researchers must perform more experiments that investigate different ET and compare different ET for a specific application.

3.6. Threats specific to the use of ET

While using ET for SDP/SCP, it is essential to understand the possible threats one needs to address for the effective application of these techniques. This would allow the computation of effective and realistic results. We extracted threats specific to the use of ET from the "Threats to Validity" or the "Limitations" section of the primary studies. However, we found that 42% of the primary studies did not report their threats (i.e., did not have any "Threats to Validity" or "Limitations" section). Another section of primary studies (25%) though stated their corresponding threats but did not specify any threats on the use of ET. Only 33% of studies stated threats specific to the use of ET.

The threats extracted from the primary studies were further categorized into 'Construct validity', 'Internal Validity', and 'External Validity' threats [116]. We state only those threats which could be extracted from two or more primary studies. This was done to eliminate threats that are specific to the experimental designs of a corresponding study. The various threats extracted from primary studies are listed in Table 9.

The extracted 'Construct Validity' threats in Table 9 state that the various internal parameter settings, base learners, and combination mechanisms are not experimented by the primary studies. However, it may be noted that though parameter tuning mechanisms [117, 118] may produce effective internal parameter settings, it is very difficult for a researcher to account for a change in base learners and combination mechanisms. In fact, researchers may

Table 9. Threats to validity specific to the use of ET

Threat	Supporting studies
Construct validity	
Does not experiment with various internal parameter settings of the ET or ensemble size	ES8, ES9, ES25, ES31, ES32, ES40, ES44, ES54, ES56, ES59, ES65, ES67
Does not experiment with different base learners for a specific ET	ES11, ES12, ES20, ES52, ES54, ES59, ES75
Does not account for the variation of results with the change in combination mechanism	ES12, ES20
Internal validity	
Threat concerning proper re-implementation of ET proposed by other studies for comparison with other ET	ES22, ES37, ES59
Bias concerning the selection of ET used in the study	ES25, ES37, ES70
Use of random sample selection strategy for training the ET	ES33, ES34, ES57
External validity	
The number and nature of datasets used for validating the ET may not be appropriate to produce generalized results	ES25, ES77, ES65
Bias concerning the selection of baseline models for comparing ET to obtain generalized results	ES52, ES55, ES77

perform experiments just to evaluate various base learners and combination mechanisms of specific ET (such as ES18).

A critical threat to internal validity is the proper re-implementation of ET (proposed/explored by other studies) for comparing its results with the ET proposed in the corresponding primary study. This needs to be done very carefully, and the results of the re-implemented ET should be matched with base studies to ensure they have been properly replicated. However, we would like to add, as previously mentioned in Section 3.2, 18% of primary studies did not mention the base learners used by the ET, moreover, 17% did not mention (or partially mention) the ensemble size used by them. Such practice makes replication of ET impossible. Researchers must mention all the parameter settings, base learners, and ensemble size of the ET used by them. Other internal validity threats involve bias in the selection of ET used by a study, and use of random selection strategy used for training certain ET (such as MKEL).

One threat to "External Validity" (Table 9) states the bias in the selection of datasets for performing the experiment. However, this threat can only be mitigated by using datasets of varied domains, sizes, and programming languages. The other external validity threat concerns itself with the selection of baseline models for comparing the ET. A researcher should choose a representative set of baseline models that are widely used by researchers for a specific application or represent various categories of algorithms (such as while analyzing ET for the class imbalance issue, a researcher may select baseline models that are representative each from Cost-sensitive ET, Boosting based ET, Bagging based ET, Hybrid ET and Novel ET as discussed in Section 3.3). Moreover, a researcher should clearly state the reason behind his choice of baseline models for comparison.

4. Discussion of results and future work

This section discusses the results presented in Section 3 and analyzes the gaps in the literature. Out of the 77 primary studies, only 10 studies used ET for SCP, all other studies were focused on SDP. Both SDP and SCP are important key activities that aid in software quality improvement. Thus, researchers should compulsorily conduct more studies that analyze and compare the capabilities of ET for various tasks in SCP apart from SDP. Furthermore, on the basis of the analysis conducted in this paper, we propose future work to researchers, which is discussed in the following sections.

4.1. Discussions related to RQ1

RQ1 attempts to categorize various ET used in literature according to the base learners used and other criteria involving their fucntioning.

- As discussed in RQ1, ET were categorized on various parameters such as the similarity of learners used by base models, their aggregation, relationship, diversity, and dependency. Though the primary studies of the review investigated ET that corresponded to most of these categories, few categories were ignored by a majority of the studies. For instance, no study used an ET which used the top-down approach for aggregation of base models. Also, there were very few studies which investigated ET that had competitive relationship amongst base models. Future studies should evaluate these less commonly explored categorizations of ET.
- While analyzing the families of machine learning techniques used as base learners, we found that only 11% of primary studies used search-based algorithms as base learners. Researchers have ascertained the effectiveness of search-based algorithms in the domain of software quality predictive modeling [26]. Therefore, future studies should extensively explore ET that use search-based algorithms as base learners.
- Another interesting class of ET (explored by 35% of primary studies) were ensembles of ensembles that use ET such as RF, BAG, AB, and Gradient Boosting as base learners. Apart from these ET other techniques such as ROT, MC, Random Subspace or other ET should be investigated as base learners for forming new ensembles. Certain primary studies also proposed new ensemble of ensembles such as Deep Forest (ES65), Ensemble Random Undersampling (ES44) and others. However, it was also observed that only one primary SCP study evaluated ET as base learners. More studies which assess the use of ensemble of ensembles should be conducted in the domain of SCP.
- The essence of ET is aggregation of several base models to yield a more stabilized and reliable predictive outcome. However, all the base models of an ET use the original feature set of the training dataset. These original features primarily quantify the measurements in the process (such as evolution-based metrics [9]) or the code structure (represented by code metrics [7]). On the contrary, deep learning techniques in SDP generates new higher-level features from the original given feature set which symbolize the semantic attributes and have found to yield better predictive outcomes than models developed using original feature set [119, 120]. However, they do not generate multiple models to provide aggregated and stabilized results. A culmination of both these techniques, i.e., deep learning and ensemble learning is promising. Such a combination has been explored by two primary studies. ES65 uses Deep Forest for ensembles while ES74 uses deep representation of software metrics followed by two stage ensemble learning. The results of both the studies have ascertained that blend of deep learning

and ensemble learning is successful and would yield upscaling of current SDP models. Researchers in future should further explore the combination of these two paradigms for conclusive and generalized results.

4.2. Discussions related to RQ2

RQ2 investigates the various applications for which ET have been utilized in SDP/SCP literature.

- We found only six ET namely SOB, OOB, UOB and three heterogeneous learners based on plurality voting, soft voting and stacking, which were explored for online learning in SDP. Moreover, there were only two primary studies that evaluated ET for online learning. There is an urgent need for more studies that assess and evaluate ET for online learning not just for SDP but for SCP too.
- After analyzing the results of RQ2, we found that 65% of the studies used ET as a learning algorithm for developing SDP models. However, there are various other less commonly explored applications of ET such as transfer learning (explored by 10% of primary studies) which need more investigation by the research community. It may also be noted that we could not find sufficient data to assess and compare ET for these less explored applications. These observations necessitate a mandatory step by the research community in exploring ET for transfer learning and other less explored applications.
- Researchers in the future should propose ET that collectively deal with diverse issues such as handling imbalanced data and online learning together or other unified ET such as TCSBoost that deal with multiple issues simultaneously. Studies should also be conducted to extensively validate such proposed techniques and obtain generalized conclusions concerning their effectiveness.

4.3. Discussions related to RQ3

RQ3 analyzes the various mechanisms/rules which have been used to aggregate base models in ET used in SDP/SCP literature.

- Researchers may conduct studies where they experiment with different base learners for a specific ensemble technique. The results of such studies can be used to effectively choose base learners as there are a wide variety of options available in literature as discussed in RQ1. Studies should also be conducted to evaluate different combination rules, i.e., if they improve or deteriorate the performance of a specific ensemble technique.
- As we evaluated the various combination mechanisms used in ET, we found that there was a need to extensively validate the ET which are based on the meta-learning mechanism. The performance of such techniques (evaluated in RQ4) could not be generalized as though CODEP yielded exceptionally good results, the results of Stacking was found to be poor when compared with non-ensemble techniques and amongst each other. However, as the comparison data for these techniques could be extracted from very few studies, these techniques should be explored by large number of studies in both SDP as well as SCP.

4.4. Discussions related to RQ4

RQ4 evaluates the performance of ET for the various applications in SDP/SCP domain. It also attempts to compare the performance of ET amongst each other and with other non-ensemble techniques for the various applications.

- While conducting comparisons for RQ4, we were not able to effectively compare all the applications of ET. Thus, more studies should be conducted which provide comparisons of ET amongst each other and with different non-ensemble techniques for varied applications. Moreover, apart from AUC, other stable performance measures such as MCC [90] or Balance [63] should be widely used by researchers to report and compare the results of SDP/SCP models developed using various ET. Different ET that address the class imbalance issue may also be compared based on "cost-effectiveness" to provide a comprehensive picture to other researchers and software practitioners.
- Certain ET such as ROT, MBoost, and CEL, though exhibited promising results were not widely used by primary studies. Such ET should be thoroughly investigated for various applications. Also, various ET such as Multischeme, Non Negative Sparse-based Semiboost, RBBag, ASCI, DECORATE, ASOF, etc., were only investigated in one primary study each. More studies must be conducted which evaluate and compare such ET in the SDP/SCP domain.
- It was also observed that ET belonging to the same category may exhibit contrasting results. For instance, as Stacking, CODEP and VHetBL are heterogeneous ET, but exhibit very different results when they were compared as a learning algorithm for developing SDP models. Researchers in future should conduct comparisons amongst specific categories of ET such as comparison amongst heterogeneous learners for several applications to observe their capabilities and effectiveness.

4.5. Discussions related to RQ5

RQ5 scrutinizes the primary studies for the various threats specific to the use of ET in SDP/SCP literature.

- The threats specific to ET could be extracted from just a few studies. As a good practice, researchers should state all the possible threats to validity in their studies. Moreover, they should design their experiments so that possible threats can be minimized as far as possible.
- The review results indicated several primary studies that did not either state the base learners used (18% of primary studies) or did not mention the ensemble size (17% of primary studies). Such incomplete information hinders the replication of results by other researchers. Also, several researchers proposed new ET, however, they should be encouraged to provide tools for their proposed techniques. This would enable other researchers to validate and replicate their proposed techniques. If not tools, researchers should at least clearly state all the internal parameter settings, base learners used, and combination mechanisms so that others may replicate and repeat their results for comparison.

5. Threats to validity

This section discusses the various threats to the validity of this review. The search for relevant studies of the review included the formulation of a search string by choosing specific search terms from the research questions. The search string was thereafter used to retrieve studies from five electronic databases. However, it may be the case that certain relevant studies may not include the search terms in their titles, abstracts, or keywords. We might miss such studies. In order to address this threat, we manually scanned the reference lists of all the extracted studies so that we may not miss a relevant study. Furthermore, we also scanned the reference lists of two recent reviews [10, 13] conducted on SDP and SCP. We are positive that these steps reduce the risk of missing out on a relevant study.

Another possible threat to the review results occurs from our assumption that all the primary studies present their results in an unbiased manner. However, there could be publication bias, wherein there are higher chances that positive results of ET are reported rather than negative results [32]. There is a possibility that the authors of a study may incorrectly claim that their proposed ET is better than other ET prevalent in literature. In order to encounter this threat, we included "empirical studies which compare different ET with each other or with other non-ensemble techniques for SDP or SCP" as an inclusion criterion (mentioned in Section 2.2.1). Such studies only aim to compare various existing ET, and do not propose their own techniques. Therefore, such studies would report both favorable and unfavorable results of ET, mitigating the publication bias.

To evaluate the capability of ET for various applications related to SDP and SCP, we performed a comparison of the results of SDP/SCP models developed by various ET and non-ensemble techniques. For doing so, data was extracted from different primary studies. However, these studies use diverse experimental designs and settings (internal parameters of ET, size of ET, base learners of ET, independent variables, datasets, preprocessing techniques, etc.). This could be a possible threat to the review. This threat was mitigated by reporting the statistics dataset wise, after removing the outliers. This would ensure that exceptional values reported by a specific study due to its corresponding experimental design are removed. Moreover, we also state the median values to report the most common values rather than extreme results reported by a study. Another possible threat in comparing ET and non-ensemble techniques is that there could be certain bias in the dataset wise comparison performed in the review. As already pointed out, we collect only those studies that use ET or compare ET with each other and other non-ensemble techniques. Since, we do not collect and extract data from studies that have used only non-ensemble techniques on the compared datasets, the comparison may be biased and more favorable for ET. The only way to address this threat is to additionally collect and extract data from studies that have employed non-ensemble techniques on the said datasets. However, this is beyond the purview of the study.

The external validity of the review concerns itself with the appropriateness of the primary studies of the review, as per the review's objective, so that the review results are valid and generalizable. The review protocol is clearly defined so that we extract a valid set of primary studies, which are in line with the review objectives. Also, the study clearly states the review protocol, which supports the replicability and repeatability of the review.

6. Conclusion

This review systematically summarizes the use of ET in SDP and SCP studies. We analyzed the studies that used ET in SDP/SCP literature from five perspectives namely their category, application, combination mechanism, performance, and probable threats that could occur while using ET. We extensively explored 5 online libraries and extracted 77 primary studies in the period from January 2000 to December 2020. The primary findings of the review are summarized below:

- ET used in SDP/SCP literature can be categorized according to five criteria which includes: a) similarity of the base models (homogeneous and heterogeneous), b) aggregation mechanism of base models (top-down and bottom-up), c) relationship amongst base models (competitive or cooperative), d) diversity of base models (implicit and explicit), and e) dependency amongst base models (dependent and independent). Amongst the mentioned criteria, we found that homogeneous, bottom-up, cooperative, explicit, and independent ET are popular with respect to learner similarity, aggregation, relationship, diversity, and dependency categorizations. Tree-based learners were the most popular machine learning family which were used as base learners for ET.
- After analyzing the primary studies, we found six different applications of ET in SDP/SCP literature. The most common application of ET was its use as a learning algorithm for developing an SDP model. The other applications were addressing the issue of imbalanced training data, their use as a learning algorithm for developing an SCP model, transfer learning, online learning, and feature selection.
- Primarily, there are two mechanisms for combining base models, one is where the output of constituent base models are given specific weights to get an aggregated ensemble output, the second is when an ensemble is constructed through meta-learning. Only twelve ET used the meta-learning mechanism while all others used the weighing mechanism. We found sixteen combination rules for ET that used the weighing mechanism for aggregation. Amongst them, some of the popular ones were majority voting, providing weights according to misclassification error, and combining the base models according to average probability.
- The performance of ET was analyzed dataset wise by evaluating the AUC and recall performance metrics. A mean AUC value of 0.75 or above was depicted by a majority of the explored ET when used as a learning algorithm for developing SDP or SCP models or for addressing the imbalanced data issue. Majority of ET that were used as a learning algorithm for developing SDP models depicted median recall values in the range 70%–90%. A comparison of ET with other non-ensemble techniques (conducted using vote count method and Wilcoxon signed ranked test) indicated that RF and BAG were superior and popular ET as they exhibited better results than most of the other compared non-ensemble techniques when being used as learners for developing SDP or SCP models. The CODEP technique, a heterogeneous ET also exhibited favourable results. We also compared ET amongst each other and found CODEP, RF and BAG to be the best performing ET when used for developing SDP/SCP models and MBoost as the best technique for handling skewed data.
- Amongst 77 primary studies, only 33% of them reported the threats specific to the use of ET. The construct validity threats included the inability of the study to account for the change in parameter settings, base learners, and the combination mechanism of the ET. The internal validity threats need to address the biased selection of ET in a study, suitable replication of ET proposed by other studies, and accounting for the random

selection strategy for training certain ET. The reported external validity threats could be addressed by the selection of an appropriate number and nature of datasets for empirical validation and selection of appropriate baseline models for comparing the ET.

Acknowledgement

The author would like to acknowledge the contribution of Ms. Sugandha Gupta for helping in data extraction and quality analysis of the candidate studies.

References

[1] N.E. Fenton and N. Ohlsson, "Quantitative analysis of faults and failures in a complex software system," *IEEE Transactions on Software Engineering*, Vol. 26, No. 8, Aug. 2000, pp. 797–814.

[2] A.G. Koru and J. Tian, "Comparing high-change modules and modules with the highest measurement values in two large-scale open-source products," *IEEE Transactions on Software Engineering*, Vol. 31, No. 8, Aug. 2005, pp. 625–642.

[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, May 2008, pp. 485–496.

[4] N. Seliya, T.M. Khoshgoftaar, and J. Van Hulse, "Predicting faults in high assurance software," in *12th International Symposium on High Assurance Systems Engineering*. IEEE, Nov. 2010, pp. 26–34.

[5] R. Malhotra and M. Khanna, "An exploratory study for software change prediction in object-oriented systems using hybridized techniques," *Automated Software Engineering*, Vol. 24, No. 3, Sep. 2017, pp. 673–717.

[6] A.G. Koru and H. Liu, "Identifying and characterizing change-prone classes in two large-scale open-source products," *Journal of Systems and Software*, Vol. 80, No. 1, Jan. 2007, pp. 63–73.

[7] D. Romano and M. Pinzger, "Using source code metrics to predict change-prone java interfaces," in *27th International Conference on Software Maintenance (ICSM)*. IEEE, Sep. 2011, pp. 303–312.

[8] E. Giger, M. Pinzger, and H.C. Gall, "Can we predict types of code changes? An empirical analysis," in *9th Working Conference on Mining Software Repositories (MSR)*. IEEE, Jun. 2012, pp. 217–226.

[9] M.O. Elish and M. Al-Rahman Al-Khiaty, "A suite of metrics for quantifying historical changes to predict future change-prone classes in object-oriented software," *Journal of Software: Evolution and Process*, Vol. 25, No. 5, May 2013, pp. 407–437.

[10] R. Malhotra, "A systematic review of machine learning techniques for software fault prediction," *Applied Soft Computing*, Vol. 27, Feb. 2015, pp. 504–518.

[11] R.S. Wahono, "A systematic literature review of software defect prediction: research trends, datasets, methods and frameworks," *Journal of Software Engineering*, Vol. 1, No. 1, Apr. 2015, pp. 1–16.

[12] A. Idri, M. Hosni, and A. Abran, "Systematic literature review of ensemble effort estimation," *Journal of Systems and Software*, Vol. 118, Aug. 2016, pp. 151–175.

[13] R. Malhotra and M. Khanna, "Software change prediction: A systematic review and future guidelines," *e-Informatica Software Engineering Journal*, Vol. 13, No. 1, 2019, pp. 227–259.

[14] L.I. Kuncheva and C.J. Whitaker, "Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy," *Machine Learning*, Vol. 51, No. 2, May 2003, pp. 181–207.

[15] L. Jonsson, M. Borg, D. Broman, K. Sandahl, S. Eldh et al., "Automated bug assignment: Ensemble-based machine learning in large scale industrial contexts," *Empirical Software Engineering*, Vol. 21, No. 4, Aug. 2016, pp. 1533–1578.

[16] S.S. Rathore and S. Kumar, "Linear and non-linear heterogeneous ensemble methods to predict the number of faults in software systems," *Knowledge-Based Systems*, Vol. 119, Mar. 2017, pp. 232–256.

[17] R. Malhotra and M. Khanna, "Particle swarm optimization-based ensemble learning for software change prediction," *Information and Software Technology*, Vol. 102, Oct. 2018, pp. 65–84.

[18] M. Re and G. Valentini, "Ensemble methods: A review," in *Advances in Machine Learning and Data Mining for Astronomy, Data Mining and Knowledge Discovery*. Chapman-Hall, 2012, pp. 563–594.

[19] V. Bolón-Canedo and A. Alonso-Betanzos, "Ensembles for feature selection: A review and future trends," *Information Fusion*, Vol. 52, Dec. 2019, pp. 1–12.

[20] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, and F. Herrera, "A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, Vol. 42, No. 4, Aug. 2011, pp. 463–484.

[21] D. Radjenović, M. Heričko, R. Torkar, and A. Živkovič, "Software fault prediction metrics: A systematic literature review," *Information And Software Technology*, Vol. 55, No. 8, Aug. 2013, pp. 1397–1418.

[22] C. Catal, "Software fault prediction: A literature review and current trends," *Expert Systems With Applications*, Vol. 38, No. 4, Apr. 2011, pp. 4626–4636.

[23] 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, Nov. 2017, pp. 111–147.

[24] R. Malhotra, M. Khanna, and R.R. Raje, "On the application of search-based techniques for software engineering predictive modeling: A systematic review and future directions," *Swarm and Evolutionary Computation*, Vol. 32, Feb. 2017, pp. 85–109.

[25] R. Malhotra and M. Khanna, "Threats to validity in search-based predictive modelling for software engineering," *IET Software*, Vol. 12, No. 4, Jun. 2018, pp. 293–305.

[26] C. Catal and B. Diri, "A systematic review of software fault prediction studies," *Expert Systems With Applications*, Vol. 36, No. 4, May 2009, pp. 7346–7354.

[27] T. Hall, S. Beecham, D. Bowes, D. Gray, and S. Counsell, "A systematic literature review on fault prediction performance in software engineering," *IEEE Transactions on Software Engineering*, Vol. 38, No. 6, Oct. 2011, pp. 1276–1304.

[28] R. Malhotra and A.J. Bansal, "Software change prediction: A literature review," *International Journal of Computer Applications in Technology*, Vol. 54, No. 4, Nov. 2016, pp. 240–256.

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

[30] G. Catolino and F. Ferrucci, "An extensive evaluation of ensemble techniques for software change prediction," *Journal of Software: Evolution and Process*, Mar. 2019, p. e2156.

[31] X. Zhu, Y. He, L. Cheng, X. Jia, and L. Zhu, "Software change-proneness prediction through combination of bagging and resampling methods," *Journal of Software: Evolution and Process*, Vol. 30, No. 12, Oct. 2018, p. e2111.

[32] J. Wen, S. Li, Z. Lin, Y. Hu, and C. Huang, "Systematic literature review of machine learning based software development effort estimation models," *Information and Software Technology*, Vol. 54, No. 1, Jan. 2012, pp. 41–59.

[33] Y. Jiang, B. Cukic, and T. Menzies, "Fault prediction using early lifecycle data," in *The 18th International Symposium on Software Reliability (ISSRE'07)*. IEEE, Nov. 2007, pp. 237–246.

[34] E. Rubinić, G. Mauša, and T.G. Grbac, "Software defect classification with a variant of NSGA-II and simple voting strategies," in *International Symposium on Search Based Software Engineering*. Springer, Sep. 2015, pp. 347–353.

[35] A. Ali, M. Abu-Tair, J. Noppen, S. McClean, Z. Lin et al., "Contributing features-based schemes for software defect prediction," in *International Conference on Innovative Techniques and Applications of Artificial Intelligence*. Springer, Dec. 2019, pp. 350–361.

[36] Y. Ma, L. Guo, and B. Cukic, "A statistical framework for the prediction of fault-proneness," in *Advances in Machine Learning Applications in Software Engineering*. IGI Global, 2007, pp. 237–263.

[37] M.J. Siers and M.Z. Islam, "Software defect prediction using a cost sensitive decision forest and voting, and a potential solution to the class imbalance problem," *Information Systems*, Vol. 51, Jul. 2015, pp. 62–71.

[38] J.R. Campos, E. Costa, and M. Vieira, "Improving failure prediction by ensembling the decisions of machine learning models: A case study," *IEEE Access*, Vol. 7, Dec. 2019, pp. 177 661–177 674.

[39] G. Li and S. Wang, "Oversampling boosting for classification of imbalanced software defect data," in *35th Chinese Control Conference (CCC)*. IEEE, Jul. 2016, pp. 4149–4154.

[40] H. Jia, F. Shu, Y. Yang, and Q. Wang, "Predicting fault-prone modules: A comparative study," in *International Conference on Software Engineering Approaches for Offshore and Outsourced Development*. Springer, Jul. 2009, pp. 45–59.

[41] R. Malhotra, "An empirical framework for defect prediction using machine learning techniques with Android software," *Applied Soft Computing*, Vol. 49, Dec. 2016, pp. 1034–1050.

[42] L. Gong, S. Jiang, and L. Jiang, "An improved transfer adaptive boosting approach for mixed-project defect prediction," *Journal of Software: Evolution and Process*, Vol. 31, No. 10, Oct. 2019, p. e2172.

[43] T.M. Khoshgoftaar, P. Rebours, and N. Seliya, "Software quality analysis by combining multiple projects and learners," *Software Quality Journal*, Vol. 17, No. 1, Mar. 2009, pp. 25–49.

[44] D. Ryu, O. Choi, and J. Baik, "Value-cognitive boosting with a support vector machine for cross-project defect prediction," *Empirical Software Engineering*, Vol. 21, No. 1, Feb. 2016, pp. 43–71.

[45] H. He, X. Zhang, Q. Wang, J. Ren, J. Liu et al., "Ensemble multiboost based on ripper classifier for prediction of imbalanced software defect data," *IEEE Access*, Vol. 7, Aug. 2019, pp. 110 333–110 343.

[46] T. Mende and R. Koschke, "Revisiting the evaluation of defect prediction models," in *Proceedings of the 5th International Conference on Predictor Models in Software Engineering*, May 2009, pp. 1–10.

[47] J. Petrić, D. Bowes, T. Hall, B. Christianson, and N. Baddoo, "Building an ensemble for software defect prediction based on diversity selection," in *Proceedings of the 10th ACM/IEEE International symposium on empirical software engineering and measurement*, Sep. 2016, pp. 1–10.

[48] L. Kumar, S. Lal, A. Goyal, and N. Murthy, "Change-proneness of object-oriented software using combination of feature selection techniques and ensemble learning techniques," in *Proceedings of the 12th Innovations on Software Engineering Conference (formerly known as India Software Engineering Conference)*. ACM, Feb. 2019, p. 8.

[49] C. Seiffert, T.M. Khoshgoftaar, and J. Van Hulse, "Improving software-quality predictions with data sampling and boosting," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, Vol. 39, No. 6, Sep. 2009, pp. 1283–1294.

[50] T. Wang, Z. Zhang, X. Jing, and L. Zhang, "Multiple kernel ensemble learning for software defect prediction," *Automated Software Engineering*, Vol. 23, No. 4, Dec. 2016, pp. 569–590.

[51] Z. Li, X.Y. Jing, X. Zhu, H. Zhang, B. Xu et al., "Heterogeneous defect prediction with two-stage ensemble learning," *Automated Software Engineering*, Vol. 26, No. 3, 2019, pp. 599–651.

[52] E. Arisholm, L.C. Briand, and E.B. Johannessen, "A systematic and comprehensive investigation of methods to build and evaluate fault prediction models," *Journal of Systems and Software*, Vol. 83, No. 1, Jan. 2010, pp. 2–17.

[53] T. Wang, Z. Zhang, X. Jing, and Y. Liu, "Non-negative sparse-based semiboost for software defect prediction," *Software Testing, Verification and Reliability*, Vol. 26, No. 7, Nov. 2016, pp. 498–515.

[54] R. Li, L. Zhou, S. Zhang, H. Liu, X. Huang et al., "Software defect prediction based on ensemble learning," in *Proceedings of the 2019 2nd International conference on data science and information technology*, Jul. 2019, pp. 1–6.

[55] Y. Liu, T.M. Khoshgoftaar, and N. Seliya, "Evolutionary optimization of software quality modeling with multiple repositories," *IEEE Transactions on Software Engineering*, Vol. 36, No. 6, May 2010, pp. 852–864.

[56] X. Xia, D. Lo, S.J. Pan, N. Nagappan, and X. Wang, "Hydra: Massively compositional model for cross-project defect prediction," *IEEE Transactions on software Engineering*, Vol. 42, No. 10, Nov. 2016, pp. 977–998.

[57] R. Malhotra and S. Kamal, "An empirical study to investigate oversampling methods for improving software defect prediction using imbalanced data," *Neurocomputing*, Vol. 343, May 2019, pp. 120–140.

[58] J. Zheng, "Cost-sensitive boosting neural networks for software defect prediction," *Expert Systems with Applications*, Vol. 37, No. 6, Jun. 2010, pp. 4537–4543.

[59] H. Alsawalqah, H. Faris, I. Aljarah, L. Alnemer, and N. Alhindawi, "Hybrid SMOTE-ensemble approach for software defect prediction," in *Computer Science on-Line Conference*. Springer, Apr. 2017, pp. 355–366.

[60] R. Malhotra and M. Khanna, "Dynamic selection of fitness function for software change prediction using particle swarm optimization," *Information and Software Technology*, Vol. 112, Aug. 2019, pp. 51–67.

[61] D. Di Nucci, F. Palomba, R. Oliveto, and A. De Lucia, "Dynamic selection of classifiers in bug prediction: An adaptive method," *IEEE Transactions on Emerging Topics in Computational Intelligence*, Vol. 1, No. 3, May 2017, pp. 202–212.

[62] H. Tong, B. Liu, and S. Wang, "Kernel spectral embedding transfer ensemble for heterogeneous defect prediction," *IEEE Transactions on Software Engineering*, Vol. 14, No. 8, Sep. 2019, pp. 1–21.

[63] A.T. Mısırlı, A.B. Bener, and B. Turhan, "An industrial case study of classifier ensembles for locating software defects," *Software Quality Journal*, Vol. 19, No. 3, Sep. 2011, pp. 515–536.

[64] L. Kumar, S. Misra, and S.K. Rath, "An empirical analysis of the effectiveness of software metrics and fault prediction model for identifying faulty classes," *Computer Standards and Interfaces*, Vol. 53, Aug. 2017, pp. 1–32.

[65] H.D. Tran, L.T.M. Hanh, and N.T. Binh, "Combining feature selection, feature learning and ensemble learning for software fault prediction," in *11th International Conference on Knowledge and Systems Engineering (KSE)*. IEEE, Oct. 2019, pp. 1–8.

[66] Y. Peng, G. Kou, G. Wang, W. Wu, and Y. Shi, "Ensemble of software defect predictors: an AHP-based evaluation method," *International Journal of Information Technology and Decision Making*, Vol. 10, No. 01, Jan. 2011, pp. 187–206.

[67] R. Malhotra and M. Khanna, "An empirical study for software change prediction using imbalanced data," *Empirical Software Engineering*, Vol. 22, No. 6, Dec. 2017, pp. 2806–2851.

[68] T. Zhou, X. Sun, X. Xia, B. Li, and X. Chen, "Improving defect prediction with deep forest," *Information and Software Technology*, Vol. 114, Oct. 2019, pp. 204–216.

[69] N. Seliya and T.M. Khoshgoftaar, "The use of decision trees for cost-sensitive classification: an empirical study in software quality prediction," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, Vol. 1, No. 5, Sep. 2011, pp. 448–459.

[70] D. Ryu, J.I. Jang, and J. Baik, "A transfer cost-sensitive boosting approach for cross-project defect prediction," *Software Quality Journal*, Vol. 25, No. 1, Mar. 2017, pp. 235–272.

[71] R. Abbas, F.A. Albalooshi, and M. Hammad, "Software change proneness prediction using machine learning," in *International Conference on Innovation and Intelligence for Informatics, Computing and Technologies (3ICT)*. IEEE, Dec. 2020, pp. 1–7.

[72] K. Gao, T.M. Khoshgoftaar, and A. Napolitano, "A hybrid approach to coping with high dimensionality and class imbalance for software defect prediction," in *11th international conference on machine learning and applications*, Vol. 2. IEEE, Dec. 2012, pp. 281–288.

[73] C.W. Yohannese, T. Li, M. Simfukwe, and F. Khurshid, "Ensembles based combined learning for improved software fault prediction: A comparative study," in *12th International conference on intelligent systems and knowledge engineering (ISKE)*. IEEE, Nov. 2017, pp. 1–6.

[74] H. Aljamaan and A. Alazba, "Software defect prediction using tree-based ensembles," in *Proceedings of the 16th ACM international conference on predictive models and data analytics in software engineering*, Nov. 2020, pp. 1–10.

[75] Z. Sun, Q. Song, and X. Zhu, "Using coding-based ensemble learning to improve software defect prediction," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, Vol. 42, No. 6, Dec. 2012, pp. 1806–1817.

[76] A. Agrawal and R.K. Singh, "Empirical validation of OO metrics and machine learning algorithms for software change proneness prediction," in *Towards Extensible and Adaptable Methods in Computing*. Springer, Nov. 2018, pp. 69–84.

[77] A.A. Ansari, A. Iqbal, and B. Sahoo, "Heterogeneous defect prediction using ensemble learning technique," in *Artificial Intelligence and Evolutionary Computations in Engineering Systems*. Springer, 2020, pp. 283–293.

[78] S. Wang, L.L. Minku, and X. Yao, "Online class imbalance learning and its applications in fault detection," *International Journal of Computational Intelligence and Applications*, Vol. 12, No. 4, Dec. 2013, p. 1340001.

[79] D. Bowes, T. Hall, and J. Petrić, "Software defect prediction: Do different classifiers find the same defects?" *Software Quality Journal*, Vol. 26, No. 2, Jun. 2018, pp. 525–552.

[80] M. Banga and A. Bansal, "Proposed software faults detection using hybrid approach," *Security and Privacy*, Jan. 2020, p. e103.

[81] S. Wang and X. Yao, "Using class imbalance learning for software defect prediction," *IEEE Transactions on Reliability*, Vol. 62, No. 2, Apr. 2013, pp. 434–443.

[82] L. Chen, B. Fang, Z. Shang, and Y. Tang, "Tackling class overlap and imbalance problems in software defect prediction," *Software Quality Journal*, Vol. 26, No. 1, Jun. 2018, pp. 97–125.

[83] E. Elahi, S. Kanwal, and A.N. Asif, "A new ensemble approach for software fault prediction," in *17th International Bhurban Conference on Applied Sciences and Technology (IBCAST)*. IEEE, Jan. 2020, pp. 407–412.

[84] A. Kaur and K. Kaur, "Performance analysis of ensemble learning for predicting defects in open source software," in *international Conference on Advances in Computing, Communications and Informatics (ICACCI)*. IEEE, Sep. 2014, pp. 219–225.

[85] S.A. El-Shorbagy, W.M. El-Gammal, and W.M. Abdelmoez, "Using SMOTE and heterogeneous stacking in ensemble learning for software defect prediction," in *Proceedings of the 7th International Conference on Software and Information Engineering*, May 2018, pp. 44–47.

[86] L. Goel, M. Sharma, S.K. Khatri, and D. Damodaran, "Defect prediction of cross projects using PCA and ensemble learning approach," in *Micro-Electronics and Telecommunication Engineering*. Springer, 2020, pp. 307–315.

[87] A. Panichella, R. Oliveto, and A. De Lucia, "Cross-project defect prediction models: L'union fait la force," in *Software Evolution Week-IEEE Conference on Software Maintenance, Reengineering, and Reverse Engineering (CSMR-WCRE)*. IEEE, Feb. 2014, pp. 164–173.

[88] R. Malhotra and A. Bansal, "Investigation of various data analysis techniques to identify change prone parts of an open source software," *International Journal of System Assurance Engineering and Management*, Vol. 9, No. 2, Apr. 2018, pp. 401–426.

[89] T.T. Khuat and M.H. Le, "Evaluation of sampling-based ensembles of classifiers on imbalanced data for software defect prediction problems," *SN Computer Science*, Vol. 1, No. 2, Mar. 2020, pp. 1–16.

[90] D. Rodriguez, I. Herraiz, R. Harrison, J. Dolado, and J.C. Riquelme, "Preliminary comparison of techniques for dealing with imbalance in software defect prediction," in *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering*, May 2014, pp. 1–10.

[91] R. Malhotra and J. Jain, "Handling imbalanced data using ensemble learning in software defect prediction," in *10th International Conference on Cloud Computing, Data Science and Engineering (Confluence)*. IEEE, Jan. 2020, pp. 300–304.

[92] V. Suma, T. Pushphavathi, and V. Ramaswamy, "An approach to predict software project success based on random forest classifier," in *ICT and Critical Infrastructure: Proceedings of the 48th Annual Convention of Computer Society of India-Vol II*. Springer, 2014, pp. 329–336.

[93] R. Mousavi, M. Eftekhari, and F. Rahdari, "Omni-ensemble learning (OEL): utilizing over-bagging, static and dynamic ensemble selection approaches for software defect prediction," *International Journal on Artificial Intelligence Tools*, Vol. 27, No. 6, Sep. 2018, p. 1850024.

[94] S.K. Pandey, R.B. Mishra, and A.K. Tripathi, "BPDET: An effective software bug prediction model using deep representation and ensemble learning techniques," *Expert Systems with Applications*, Vol. 144, Apr. 2020, p. 113085.

[95] L. Chen, B. Fang, Z. Shang, and Y. Tang, "Negative samples reduction in cross-company software defects prediction," *Information and Software Technology*, Vol. 62, Jun. 2015, pp. 67–77.

[96] S. Moustafa, M.Y. ElNainay, N. El Makky, and M.S. Abougabal, "Software bug prediction using weighted majority voting techniques," *Alexandria Engineering Journal*, Vol. 57, No. 4, Dec. 2018, pp. 2763–2774.

[97] S.S. Rathore and S. Kumar, "An empirical study of ensemble techniques for software fault prediction," *Applied Intelligence*, Vol. 51, No. 6, Jun. 2021, pp. 3615–3644.

[98] M.O. Elish, H. Aljamaan, and I. Ahmad, "Three empirical studies on predicting software maintainability using ensemble methods," *Soft Computing*, Vol. 19, No. 9, Sep. 2015, pp. 2511–2524.

[99] H. Tong, B. Liu, and S. Wang, "Software defect prediction using stacked denoising autoencoders and two-stage ensemble learning," *Information and Software Technology*, Vol. 96, Apr. 2018, pp. 94–111.

[100] A.A. Saifan and L. Abu-wardih, "Software defect prediction based on feature subset selection and ensemble classification," *ECTI Transactions on Computer and Information Technology (ECTI-CIT)*, Vol. 14, No. 2, Oct. 2020, pp. 213–228.

[101] S. Hussain, J. Keung, A.A. Khan, and K.E. Bennin, "Performance evaluation of ensemble methods for software fault prediction: An experiment," in *Proceedings of the ASWEC 24th Australasian software engineering conference*, Sep. 2015, pp. 91–95.

[102] Y. Zhang, D. Lo, X. Xia, and J. Sun, "Combined classifier for cross-project defect prediction: an extended empirical study," *Frontiers of Computer Science*, Vol. 12, No. 2, 2018, pp. 280–296.

[103] F. Yucalar, A. Ozcift, E. Borandag, and D. Kilinc, "Multiple-classifiers in software quality engineering: Combining predictors to improve software fault prediction ability," *Engineering Science and Technology, an International Journal*, Vol. 23, No. 4, Aug. 2020, pp. 938–950.

[104] I.H. Laradji, M. Alshayeb, and L. Ghouti, "Software defect prediction using ensemble learning on selected features," *Information and Software Technology*, Vol. 58, Feb. 2015, pp. 388–402.

[105] L. Rokach, "Taxonomy for characterizing ensemble methods in classification tasks: A review and annotated bibliography," *Computational statistics & data analysis*, Vol. 53, No. 12, Oct. 2009, pp. 4046–4072.

[106] C. Sammut and G.I. Webb, Eds., *Encyclopedia of Machine Learning*. Springer Science & Business Media, 2011.

[107] O. Sagi and L. Rokach, "Ensemble learning: A survey," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, Vol. 8, No. 4, 2018, p. e1249.

[108] A.J. Sharkey, "Types of multinet system," in *International Workshop on Multiple Classifier Systems*. Springer, Jun. 2002, pp. 108–117.

[109] H. He and E.A. Garcia, "Learning from imbalanced data," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 21, No. 9, Jun. 2009, pp. 1263–1284.

[110] M. Tan, L. Tan, S. Dara, and C. Mayeux, "Online defect prediction for imbalanced data," in *37th International Conference on Software Engineering*, Vol. 2. IEEE, May 2015, pp. 99–108.

[111] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognition Letters*, Vol. 27, No. 8, Jun. 2006, pp. 861–874.

[112] T. Menzies, A. Dekhtyar, J. Distefano, and J. Greenwald, "Problems with precision: A response to "Comments on 'Data mining static code attributes to learn defect predictors"'," *IEEE Transactions on Software Engineering*, Vol. 33, No. 9, Aug. 2007, pp. 637–640.

[113] J. Derrac, S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm and Evolutionary Computation*, Vol. 1, No. 1, Mar. 2011, pp. 3–18.
[114] T.G. Dietterich, "Ensemble methods in machine learning," in *International Workshop on Multiple Classifier Systems*. Springer, Jun. 2000, pp. 1–15.
[115] F. Eibe, M.A. Hall, and I.H. Witten, "The WEKA workbench. online appendix for data mining: practical machine learning tools and techniques," in *Morgan Kaufmann*. Elsevier Amsterdam, The Netherlands, 2016.
[116] T.D. Cook, D.T. Campbell, and A. Day, *Quasi-experimentation: Design and analysis issues for field settings*. Houghton Mifflin Boston, 1979, Vol. 351.
[117] W. Fu, V. Nair, and T. Menzies, "Why is differential evolution better than grid search for tuning defect predictors?" *arXiv preprint arXiv:1609.02613*, 2016.
[118] C. Tantithamthavorn, S. McIntosh, A.E. Hassan, and K. Matsumoto, "The impact of automated parameter optimization on defect prediction models," *IEEE Transactions on Software Engineering*, Vol. 45, No. 7, Jan. 2018, pp. 683–711.
[119] S. Omri and C. Sinz, "Deep learning for software defect prediction: A survey," in *Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops*, Jun. 2020, pp. 209–214.
[120] E.N. Akimova, A.Y. Bersenev, A.A. Deikov, K.S. Kobylkin, A.V. Konygin et al., "A survey on software defect prediction using deep learning," *Mathematics*, Vol. 9, No. 11, Jan. 2021, p. 1180.

Glossary

AB: AdaBoost
ANN: Artificial Neural Network
AUC: Area Under the Receiver operating characteristic Curve
ASOF: Adaptive Selection of Optimum Fitness
ASCI: Adaptive Selection of Classifiers
BAG: Bagging
BN: Bayesian Network
BNC: AdaBoost.NC
BTE: Best in Training Ensemble
CART: Classification and Regression Tree
CEL: Coding based Multiclassifier
CS: Cumulative Quality Score
CODEP: Combined Defect Predictor
Dag: Dagging
Dec.T: Decision Table
Decorate: Diverse Ensemble Creation by Oppositional Relabeling of Artificial Training Examples
DNC: Dynamic Adaboost.NC
DTB: Double Transfer Boosting
ET: Ensemble Techniques
ITrAdaBoost: Improved Transfer Adaptive Boosting
KNN: K-Nearest Neighbor
KSETE: Kernel Spectral Embedding Transfer Ensemble
LB: LogitBoost
LMT: Logit Model Tree
LR: Logistic Regression
MBoost: MultiBoost
MC: MetaCost
MCC: Mathews Correlation Coefficient
MKEL: Multiple Kernel Ensemble Learning
NB: Naïve Bayes
NDTF: Non-Linear Decision Tree Forest
OEL: Omni Ensemble Learning
OOB: Oversampling based Online Bagging
RBBag: Roughly Balanced Bagging
RF: Random Forests
ROT: Rotation Forest
RQ: Research Question
RUSBoost: Random UnderSampling Boosting
RS: Random Subspace
SCP: Software Change Prediction
SDP: Software Defect Prediction
SI: Study Identifier
SMBoost: SMOTEBoost
SOB: Sampling based Online Bagging
SQA: Software Quality Assurance
SVM: Support Vector Machine
TCSBoost: TransferCostSensitive Boosting

TSE: Two Stage Ensemble
UOB: Undersampling based Online Bagging
VCB-SVM : Value Cognitive Boosting with Support Vector Machine
VFI: Voting Feature Intervals
VHetBL: Voting amongst Heterogenous Base Learners
VHomBL: Voting amongst Homogeneous Base Learners
VV: Validation and Voting
WEKA: Waikato Environment for Knowledge Analysis

Appendix

Table A1 lists all the primary studies that use a specific ET. Table A2 states the machine learning family of the techniques, which have been used as base learners for ET in the primary studies.

Table A1. List of ET used by primary studies

ET	Primary Studies
AdaBoost	ES7, ES8, ES13, ES21, ES24, ES25, ES29, ES30, ES36, ES39, ES41, ES44, ES46, ES47, ES50, ES55, ES56, ES60, ES61, ES62, ES67, ES73, ES74, ES76, ES77
AdaBoost.NC	ES18, ES73
AdaCost	ES14, ES40
Adc2	ES14
Adaptive Selection of Classifiers	ES37
Adaptive Selection of Optimum Fitness	ES62
Average Probability Ensemble	ES26, ES54, ES70
Average Voting	ES51
Bagging	ES6, ES13, ES16, ES19, ES21, ES24, ES30, ES31, ES36, ES39, ES41, ES42, ES46, ES47, ES50, ES51, ES52, ES55, ES60, ES61, ES62, ES69, ES74, ES76, ES77
Balanced Random Forests	ES2
Best in Training Ensemble	ES24, ES38, ES58
Boosting	ES2, ES16, ES19, ES31, ES32, ES51
Bug Prediction using Deep representation and Ensemble learning	ES74
Cascaded Weighted Majority Voting	ES49
Cascaded Randomized Weighted Majority Voting	ES49
Categorical Boosting	ES67
Combined Defect Predictor	ES20, ES35, ES51
Coding based Multi classifier	ES16, ES33, ES57
Cost-sensitive Forest	ES28
Cost-sensitive Boosting Neural Networks	ES10, ES33
Csb2	ES14
Dagging	ES75, ES77
Data Boost	ES73
DeepForest	ES65
Diverse Ensemble Creation by Oppositional Relabeling of Artificial Training Examples (DECORATE)	ES8, ES75
Double Transfer Boosting	ES23, ES56
Dynamic Adaboost.NC	ES18, ES33, ES44, ES57
Ensemble learning phase in HYDRA	ES35, ES56
Ensemble Random Undersampling	ES44
Ensemble Selection	ES75
Extra Trees	ES67
GcForest	ES65
Gradient Boosting	ES67
Grading	ES75

Table A1 continued

ET	Primary Studies
Histogram based Gradient Boosting	ES67
Improved Transfer Adaptive Boosting	ES56
Kernel Spectral Embedding Transfer Ensemble	ES63
Logistic Model Tree	ES3, ES30, ES46
LogitBoost	ES4, ES30, ES39, ES46, ES47, ES62, ES74, ES77
Maximum Voting	ES51
Metacost	ES14, ES21, ES39, ES61
Multiboost	ES57, ES66, ES75, ES77
Multiple Kernel Ensemble Learning	ES33, ES57
Multischeme	ES77
MSMOTEBoost	ES73
Non Negative Sparse based Semiboost	ES34
Non-linear Decision Tree Forest	ES24, ES38, ES58
Oversampling-based Online Bagging	ES17
Omni Ensemble Learning	ES48
Random Subspace method	ES60
Random Forest	ES1, ES2, ES3, ES4, ES6, ES14, ES16, ES18, ES22, ES23, ES26, ES30, ES32, ES36, ES39, ES42, ES43, ES44, ES46, ES47, ES48, ES50, ES51, ES53, ES55, ES60, ES61, ES62, ES65, ES66, ES67, ES71, ES74, ES77
Rotation Forest	ES19, ES21, ES48, ES75, ES77
Roughly balanced Bagging	ES11
Random Undersampling Boosting (RUSBoost)	ES15, ES21, ES44, ES73
RealAdaBoost	ES75
Randomized Weighted Majority Voting	ES49
Sampling based Online Bagging	ES17
SelectRUSBoost	ES15
SMOTEBoost	ES18, ES21, ES44, ES73
Stacking	ES13, ES25, ES31, ES45, ES54, ES66, ES70, ES77
SysFor	ES28
Transfer Adaptive Boosting	ES56
Transfer Cost-Sensitive Boosting	ES40, ES56
TransferBoost	ES35, ES40
Two Stage Ensemble	ES50, ES59, ES64
Undersampling based Online Bagging	ES17
Validation and Voting Classifier	ES9, ES37
Value Cognitive Boosting with Support Vector Machine	ES32, ES56, ES59
Voting amongst Homogeneous Base Learners (VHomBL)	ES27, ES47, ES62
Voting amongst Heterogeneous Base Learners (VHetBL)	ES5, ES9, ES12, ES13, ES24, ES25, ES38, ES48, ES54, ES55, ES58, ES60, ES66, ES68, ES70, ES72, ES77
Weighted Majority Voting	ES49
WeightedSmoteBoost	ES29
XGBoost	ES67, ES71

Table A2. List of techniques (used as base learners) belonging to each machine learning family

Tree-based Learners	C4.5, Random Tree, Decision Tree, Decision Stump, J48, CART, Alternating Decision Tree, Partial Decision Tree, Tree Disc Classification, Naïve Bayes Tree, REP Tree.
Support Vector Machine	Support Vector Machine, Sequential Minimal Optimization, Voted Perceptron, Pegasos.
Bayesian Learners	Naïve Bayes, Bayesian Network, Multinomial Naïve Bayes, Bernoulli Naïve Bayes, Parzen classifier with the Gaussian kernel, Uncorrelated normal densities based quadratic Bayes.
Rule-based Learners	One Rule, Lines-of-Code, Decision Table, Ripper Down Rules, Repeated Incremental Pruning to Produce Error Reduction.
Instance-based Learners	Locally weighted learning with decision stump, 1-Instance based Learning, K-Instance based Learning, K-Nearest Neighbor, Nearest Mean Classifier, Scaled Nearest Mean Classifier.
Search based Algorithms	Genetic Algorithm, Genetic Programming, Particle Swarm Optimization, Non-Dominated Sorting Genetic Algorithm-II (NSGA-II).
Artificial Neural Networks	Multilayer Perceptron, Radial Basis Function, Linear Perceptron classifier with Batch Processing, Levenberg–Marquardt feed-forward neural network, Automatic Levenberg–Marquardt feed-forward neural network.
Ensemble Learners	RF, BAG, AB, Boosting, XGBoost, Boosting, Gradient Boosting.
Miscellaneous Learners	Voting Feature Interval, KStar, KMeans, Random Subspace, Stochastic Gradient Descent, Minimum Least Square Linear Classifier, Subspace Classifier, Linear classifier based on Principal Component Analysis, Linear Discriminant Classifier, Quadratic Discriminant Classifier, Minimum Linear Least Square Classifier, Linear classifier based on Karhunen Loeve (KL) expansion of common covariance matrix.

e-Informatica Software Engineering Journal, Volume 16, Issue 1, 2022, pages: 220106, DOI: 10.37190/e-Inf220106

A Comparison of Citation Sources for Reference and Citation-Based Search in Systematic Literature Reviews

Nauman bin Ali*, Binish Tanveer*
*Blekinge Institute of Technology, Sweden
nauman.ali@bth.se, binish.tanveer@bth.se

Abstract

Context: In software engineering, snowball sampling has been used as a supplementary and primary search strategy. The current guidelines recommend using Google Scholar (GS) for snowball sampling. However, the use of GS presents several challenges when using it as a source for citations and references.
Objective: To compare the effectiveness and usefulness of two leading citation databases (GS and Scopus) for use in snowball sampling search.
Method: We relied on a published study that has used snowball sampling as a search strategy and GS as the citation source. We used its primary studies to compute precision and recall for Scopus.
Results: In this particular case, Scopus was highly effective with 95% recall and had better precision of 5.1% compared to GS's 2.8%. Moreover, Scopus found nine additional relevant papers. On average, one would read approximately 15 extra papers in GS than Scopus to identify one additional relevant paper. Furthermore, Scopus supports batch downloading of both citations and papers' references, has better quality metadata, and does better source filtering.
Conclusion: This study suggests that Scopus seems to be more effective and useful for snowball sampling than GS for systematic secondary studies attempting to identify peer-reviewed literature.

Keywords: Snowball sampling, snowballing, reference-based, citation-based, search strategy, systematic review, systematic mapping

1. Introduction

Systematic literature reviews and mapping studies [1] rely on a systematic and extensive search to identify the literature on a topic of interest. The two main search strategies in such secondary studies have been: (1) the use of keyword-based search and (2) supplementing the keyword-based results with snowball sampling [2]. However, others have proposed to use snowball sampling as the primary search method [3, 4]. Snowball sampling refers to the use of reference-of (for backward snowballing) and citations-to (for forward snowballing), a set of papers for identifying other relevant papers.

The indexing/citation database plays a critical role, whether using snowball sampling as the primary or supplementary search strategy. The coverage of the citation database may limit the snowball sampling strategy's effectiveness. Several alternative electronic data

Submitted: 28 Sep. 2021; Revised: 08 Mar. 2022; Accepted: 09 Mar. 2022; Available online: 17 Mar. 2022

sources for citation search exist, e.g., Elsevier Scopus (Scopus), Clarivate Analytics – Web of Science (WoS), and Google Scholar (GS). However, the current guidelines [4] recommend using GS.

For keyword-based search, where an automatic search is conducted using a combination of keywords, several studies have investigated the relevance and coverage of different electronic data sources [5–7]. However, no such investigation is reported of electronic data sources for snowball sampling in software engineering (SE) to the best of our knowledge.

In this study, we have compared Scopus and GS for use in the snowball sampling search strategy. We choose GS and Scopus, as these are among the most used citation databases in SE systematic reviews [2]. Similarly, the snowball sampling guidelines in SE recommend the use of GS [4]. The snowball sampling guidelines [4] already have over 2000 citations[1]. At least 1150 of these 2000 citing articles mention "google scholar"[2] indicating that GS is one of the sources used in these articles. This further justifies this study as the results could potentially have significant implications for future SE research employing snowball sampling in their search strategy.

The remainder of the paper is structured as follows: Section 2 describes the related work. Section 3 presents the approach used in this study for comparing the two sources. Section 4 presents the results. In Section 5 we further discussed the limitations of GS in light of related research. Section 6 presents our recommendations for future studies using snowball sampling search strategy. Section 7 highlights the validity threats and limitations of the paper. Section 8 concludes the paper.

2. Related work

We discuss the related work for this study in three complementary themes: (1) guidelines for the design, reporting, and evaluation of search strategies (2) the evaluation of electronic data sources used in SE, and (3) studies comparing citation databases without a focus on SE.

2.1. Search guidelines for systematic secondary studies

Several comprehensive guidelines for designing keyword-based search [1] and snowball sampling [4] are available. Furthermore, new improvements have been suggested to the design and assessment guidelines [8–11] based on the limitations identified in the repeatability of search in existing SLRs [2, 9].

Even when using keyword-based search as the primary search method, it is recommended to supplement the search using snowball sampling [1]. Thus, both search strategies will benefit from this study that assesses the usefulness and effectiveness of the currently recommended citation source.

2.2. Comparison of databases for keyword-based search in SE

There have been numerous studies investigating electronic data sources for keyword-based searches covering topics such as features required to support secondary studies ([7, 12, 13]) overlaps among sources ([5, 6, 14, 15]), and the value of Google and GS ([7, 13]). These

[1]On September 20, 2021, in GS, Wohlin's guidelines [4] had accumulated over 2000 citations.
[2]In GS, we searched for "Google Scholar" within the articles citing Wohlin's guidelines [4].

studies conclude that: (a.) multiple sources should be searched ([5, 6, 14, 15]), and (b.) GS and Google have good coverage of SE literature and SE secondary studies ([7, 13]).

However, we could not find a study in SE that compared citation databases for snowball sampling. In this study, we fill this gap by assessing the effectiveness and usefulness of the recommended citation database, i.e., GS, in the current guidelines and comparing it with other commonly used citation databases in SE research.

2.3. Studies comparing citation databases outside SE

Several comparisons of citations databases have been conducted, which are summarised in Table 1. The main sources compared include GS, WoS, Scopus, and Microsoft Academic Search (MAS). Some studies have indicated that the coverage (both in terms of indexing papers and citations) varies between different sources depending on the timeframe and research areas considered [16, 17]. However, one can conclude that overall, GS has the most indexed bibliographic records and citations [18–21]. GS also seems to have a faster indexing speed [22]. On the other hand, the quality of data and transparency of what is indexed is better in paid services like Scopus [22].

In this study, which is the first in SE literature, we compare GS and Scopus as sources for citations and references when performing snowball sampling. Unlike the studies discussed

Table 1. An overview of related work on comparing various data sources outside SE

ID	Data sources compared	Parameters	Main conclusions
[16]	GS, WoS	As a source for forward snowball sampling for public health literature.	WoS is recommended for public health guidance needs.
[17]	GS, WoS, Scopus	For citation tracking in two fields oncology and condensed matter physics.	Databases performance varied for different research areas and publication years.
[18]	GS, MAS	No of papers indexed and citation to those papers for several authors.	GS indexed more papers and citations for information and computing literature.
[22]	GS, Scopus	Indexed sources and indexing speed.	Scopus provides a clear documentation of what is indexed in its database. Scopus has higher accuracy and quality of data.The most important additional source indexed by GS is Google Books. GS has faster indexing speed.
[19]	GS, WoS, Scopus, MAS, and eight others	Number of bibliographic records indexed by the data source.	GS had the most number of bibliographic records.
[20]	GS, WoS, Scopus, MAS, and two others.	The number of citations to a set of 2515 highly-cited documents.	GS has the most number of citations.
[21]	GS, WoS, Scopus, MAS.	The number of citations to set of 150 articles from journals with high, low and no impact factors.	GS and Microsoft Academic had similar average number of citations, which were much higher than WoS and Scopus.

above, we take into consideration the relevance of the additional citations (not just the number of citations) found by a source.

3. Research method

In this study, we have only compared Scopus and GS. These two are the most often used [2] citation databases [6] in secondary studies in SE. Moreover, GS is the recommended source in snowball sampling guidelines in SE [4]. Hence, we attempt to answer the following research question:

RQ: *How effective and useful are GS and Scopus citation databases for implementing snowball sampling-based search strategy?*

To compare Scopus and GS, we have used a published systematic review [23] (from here on referred to as the case SLR) that has used GS for executing the snowball sampling search strategy. We did this for convenience as we had access to the intermediate and final results, which is hard to obtain for papers where one has not been a co-author [24]. In the future, we intend to replicate the analysis reported in this paper on more published papers reducing the bias that having a single case introduces. This limitation is further discussed in Section 7. The data used in the study is available for replication and further analysis by other researchers at this link.

3.1. Criteria for assessing the effectiveness and usefulness

We now present the criteria for evaluating the effectiveness and usefulness of citation databases as used in this study:

Effectiveness: The primary studies from the case SLR have allowed us to objectively assess the implication of using Scopus instead of GS using the following metrics (adapted from [1, 16]):

- **Recall** = *100 * (# of primary studies found in the search) / (total # of primary studies).*
- **Precision** = *100 * (total # of primary studies in the search results) / (total # of search results).*
- **Number needed to read for each relevant paper (NNR)** = *(total # of excluded papers) / (# of primary studies found in the search results)*

Usefulness: Usefulness [25] in this study is defined as a subjective measure of how well a source supports users performing snowball sampling. We consider the following features as indicators for the usefulness of a citation source for snowball sampling:

- Ability to easily download citations to a paper.
- Ability to easily download the references in a paper.
- Ability to easily filter citations and references (e.g., based on the publication language, venue, or whether they are peer-reviewed).

This is not an exhaustive list of features. However, these are essential for enabling the use of snowball sampling as a search approach.

3.2. Overview of the relevant aspects of the case SLR

The case SLR [23] was an attempt to find industrially relevant regression testing research. Existing SLRs on the topic of regression testing were identified and used as a start-set for

one iteration of forward and backward snowball sampling. Forward snowballing here refers to reviewing the citations to the papers in the start-set, and backward snowballing refers to checking the references used in the papers in the start-set. By one-iteration, we mean that no further snowballing was performed on the additionally included relevant papers found in the first iteration.

The search in the case SLR [23] was done in August 2016. Since the search was done in August (without a clear cut-off at a full year), it was impossible to recreate the citations list of Scopus in August 2016 at the time of the current study. Therefore, to have a relatively fair comparison, we have included the citations from both GS and Scopus up to and including the calendar year 2016.

Table 2 provides information about the start set, the number of citations and references in the start set. The case SLR had 38 primary studies. However, four papers were excluded from the comparison in the current study (i.e., making 34 primary studies in Table 2). Of the four papers not considered as primary studies in the current review, three were excluded as these were not identified by snowball sampling in the case SLR (these were added as a known-set of papers), and the remaining one paper was in pre-print in 2016, i.e., at the time of the search in the case SLR. The paper was eventually printed in 2017. This means that a search now will not find it as a publication in 2016 but as a publication from 2017. Furthermore, we had identified 12 of these primary studies through backward snowballing (i.e., through references in the seed set and do not represent the value of the data source, since they are listed in the full-text of the papers). Therefore, when assessing the effectiveness of GS and Scopus, we have used only 22 primary studies found by forward snowballing for comparison. For assessing the usefulness, we consider the features of the citation sources for both forward and backward snowball sampling.

Table 2. Details of the start set used in the case SLR [23]

No. of papers in seed set	11
No. of references in the seed set	877
No. of unique references in the seed set	506
No. of primary studies	34
Primary studies found through backward snowballing only	12

4. Results

After removing duplicate citations, we had 764 citations in GS and 415 in Scopus (see Table 3). Table 3 shows the number of citations and the objective measures of effectiveness for both sources. Please note that recall by definition will be 100% for the source used as a baseline. We also present the potential impact of having used Scopus in the Table 3.

The Venn diagram (see Figure 1) shows that 365 papers (that did not meet the inclusion criteria of the case SLR) and 21 (primary studies) are shared between Scopus and GS. At the same time, Scopus and GS each have 20 and 377 unique papers (that is papers that did not meet the inclusion criteria of the case SLR), respectively. Whereas nine potential primary studies are identified only by Scopus, and GS only identifies one unique primary study.

Table 3. Precision and recall for the two sources

	GS	Scopus
No. of citations to the seed papers	937	498
Unique citations to the seed papers (after removing duplicates)	764	415
Using only GS for forward snowballing, and its comparison with Scopus shows the following:[a]		
Of the 22 primary studies identified in GS	22	21
Precision	(22/764) * 100 = 2.8%	(21/415) * 100 = 5.1%
Recall	(22/22) * 100 = 100%	(21/22) * 100 = 95%
NNR	(377 + 365)/22 = 33.7	(29 + 365)/21 = 18.8
Using only Scopus for forward snowballing, and its comparison with GS shows the following:[b]		
Of the 30 potential primary studies identified in Scopus	21	30
Precision	(21/764) * 100 = 2.7%	(30/415) * 100 = 7.2%
Recall	(21/30) * 100 = 70.0%	(30/30) * 100 = 100%
NNR	(377 + 365 + 1)/21 = 35.3	(20 + 365)/30 = 12.8

[a] The nine potentially relevant papers only identified by Scopus are not considered in the analysis.
[b] One primary study only identified by GS is not considered in the analysis.

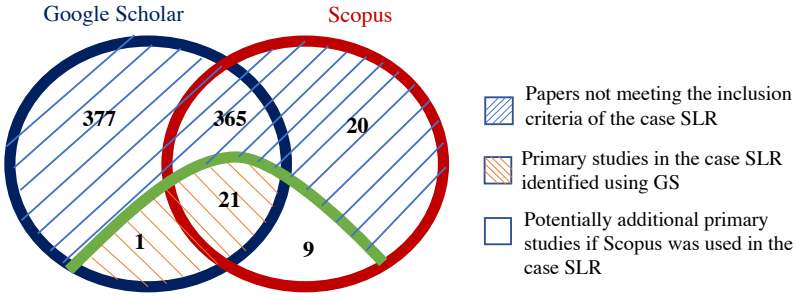


Figure 1. Data and results of the comparison between GS and Scopus

4.1. Effectiveness of GS and Scopus

Of the 22 primary studies identified through forward snowballing in GS, except for one paper, we found all the primary studies through Scopus as well (see Figure 1). While both the missing primary study and the referenced seed paper are indexed in Scopus, the citation is not recognized in Scopus. We reported the issue to Elsevier's support, and the papers are now correctly linked.

We further analysed the 29 unique citations in Scopus (see Figure 1) by applying the selection criteria of the case SLR. We found that nine of these papers meet the selection criteria and would be shortlisted for data extraction and synthesis. However, we have not done the data extraction and re-analysis of the entire data for the current paper as we do not consider it essential for the objective of this paper. The impact of these nine papers on the metrics of effectiveness used in the study is presented in Table 3. The numbers indicate that Scopus would have been a far superior choice. However, since we have not done data

extraction from these nine potential primary studies, we are not confident if they will all become primary studies. Therefore, for the remainder of the paper, we will focus on the numbers based on the case SLR where GS was used.

For the case SLR, the values for precision and NNR show that Scopus is more effective than GS. Scopus found 95% (21 out of the 22 papers) of the relevant papers identified by GS with considerably higher precision. The NNR value in (see Table 3) suggests that, on average, one would have to examine 15 extra papers in GS than Scopus to identify an additional relevant paper.

4.2. Usefulness of GS and Scopus

Table 4 summarises our assessment of GS and Scopus against the stated criteria for usefulness.

Table 4. Usefulness of GS and Scopus for snowball sampling

	GS	Scopus
Ability to easily download citations to a paper	No	Yes
Ability to easily download the references in a paper	No	Yes
Ability to easily filter citations and references	No	Partially yes[a]

[a] In Scopus, of the 14 fields of metadata to use for filtering citations only four fields are available to filter references in a paper. This was last confirmed in December 2021.

Downloading citations to a paper and references in a paper: In GS, it is difficult to download citations to papers. There is no native support for batch downloading of citations. Furthermore, to prevent denial of service attacks, GS blocks any attempt to automate the download. For example, one of the seed papers for the case SLR has over 1200 citations making it very difficult to download the citations manually. Furthermore, GS has no support for backward snowballing as references in the papers have to be manually extracted from the papers' full text.

On the other hand, we found that Scopus facilitates both forward and backward snowballing, by enabling batch download of citations and references.

Filtering citations and references: In GS, we found no means to exclude based on the publication language or whether they have been peer reviewed. For systematic studies that only include peer-reviewed literature published in certain languages (which is often the case in SE), we consider this a significant limitation of GS. Furthermore, due to the quality of metadata in GS, it was also difficult to remove duplicates. It took considerable effort to resolve minor differences in the titles and venues of the papers.

In Scopus, we can extract additional metadata about the publication, including the publication type and language that significantly aids in the selection process. Moreover, removing duplicates was reasonably straightforward in the citations retrieved through Scopus, as the data were relatively clean.

5. Discussion of GS in light of the related work

As discussed in Section 4, our study shows that GS does not have features that are necessary for its use in snowball sampling-based search. Furthermore, we found that Scopus was more

effective and useful for this purpose. To further strengthen our recommendation to use Scopus instead of GS, we now briefly discuss the limitations of GS in terms of the nature and quality of metadata indexed in it, transparency of what is indexed, and its support for snowball sampling. We base this section both on the results of our study and also on investigations of GS by others.

5.1. Lack of transparency in what is indexed

There is a lack of transparency regarding what is indexed in GS [16, 26] which may explain to a certain degree the changing citation numbers for the same period [27]. This is a serious threat to the reliability of search when using GS. Furthermore, Winter et al. [27] concluded in a longitudinal study that the number of citations substantially increased in GS for the same articles retroactively (i.e., when the search was repeated for the number of citations for a paper in the same time period on a later date, a larger number of citations was retrieved). They conclude that coverage seems to have stabilized over the more recent years [27]. However, in a recent investigation Martín-Martín and López-Cózar [26] found large fluctuations in coverage of literature by GS, which they conclude is a clear limitation of GS's use as a data source for bibliometrics.

5.2. Quality of metadata

GS does not facilitate automatic data collection (see Section 4), and researchers use custom web scrapers to extract the list of citing documents (e.g., see Martín-Mart í [20]). For the current study, we used Publish or Perish[3]. However, we noticed several shortcomings in the collected data, e.g., several entries were missing venues, abstract, or publication years. This is consistent with the observations by other researchers [22, 28–30]. For example, Adriaanse and Rensleigh [30] compared the content quality of WoS, GS, and Scopus and found that Scopus outperformed both WoS and GS [30]. They concluded that GS had the most inconsistencies, like mistakes in author spellings and order and the volume and issue numbers for the publications. Recent bibliometric studies using citation data in various disciplines including SE have also used Scopus [31, 32].

5.3. The quality of literature in GS

Aguillo [33] investigated the literature coverage by GS by analyzing which web domains are the sources for their records. GS indexes low-quality literature like low-impact journals, teaching material, unpublished reports. They concluded that GS lacks the quality to use in bibliometric studies, a conclusion shared by other studies [22].

GS may be a useful source for studies interested in both peer-reviewed and non-peer-reviewed literature, e.g., in multi-vocal literature reviews [34] or topics wherein insufficient scientific literature is available.

However, there is considerable and unavoidable noise in GS search results for other studies, where primarily peer-reviewed literature is of interest. For example, the citations analysis of a paper with 234 citations in GS [35] revealed that only 116 of the 234 citations were from journal and conference papers in English, and 54 of the remaining 118 citations were from Grey-Literature.

[3]Harzing, A.W. (2007) Publish or Perish, available from https://harzing.com/resources/publish-or-perish.

6. Recommendations when using snowball sampling

The studies using snowball sampling as the search strategy often conflate a systematic literature study's search and selection phase. We have observed at least two consequences of this: (1) the level of record-keeping is insufficient for cross-validation and replications (in particular for studies considering a large number of papers), (2) it is challenging to employ the best practices for study selection (e.g., using multiple reviewers or using text mining-based solutions).

SLR authors need to record the meta information for each citation and reference considered in various snowball iterations in an SLR. Another benefit of documenting the start set, the metadata of papers considered and the finally included primary studies list will be to enable comparison of various citation sources for snowball sampling. However, several current SLRs using snowball sampling as the primary search strategy do not document the data about intermediate references and citations considered in an SLR.

Suppose Scopus is used to operationalize the snowballing strategy. Then with some additional effort, one can automatically download the citations and references and other necessary metadata, including publication venues, language, abstracts, and keywords. Once these references and citations are collated, and duplicates are removed (as done in the keyword-based search), we can proceed with using state-of-the-art procedures, and tools [24, 36, 37]) to assist the selection process [1].

Furthermore, the metrics and indicators used in this study [1, 6, 16] can be used to assess the electronic data sources for snowball sampling. However, these metrics must be interpreted in relative terms, i.e., to compare two or more data sources, as the entire population of all primary studies is unknown, and we typically only identify a subset of the primary studies in our search [38].

7. Validity threats

The current study has used only one case SLR; therefore, we need similar comparative analysis of other secondary studies to gain more confidence in the value of using Scopus. However, the results of the study illustrate the need to evaluate the recommendation of using GS in the guidelines for performing snowball sampling.

In this study, surprisingly (as several studies as discussed in Section 2 considered GS more comprehensive than Scopus), we found that Scopus has 29 unique contributions that are not available in GS. After applying the inclusion-exclusion criteria from the case SLR, we identified that nine of these papers would have been included in the case SLR. It will be interesting to see what may have been the impact of these on the results of the case SLR. However, that analysis has not been done in the current study since we did not consider it essential for the objective of this paper.

Since we are doing the study in 2021 and looking at citations in 2016, this may be a disadvantage to the database that is more efficient in indexing new publications and updating the citations. This limitation of our study can be overcome by replicating the analysis on more recently concluded SLRs that have used GS for snowball sampling.

We have cleaned the data extensively to avoid any problems, e.g., hyphenation or case differences in the citing papers' titles. However, we may have still missed a few unique cases where the same papers are considered unique due to slight differences. However, due

to the measures taken and manual checking of some of the unique results, we are confident that this is not a significant threat to this study's validity.

Furthermore, the criteria used for usefulness are not very comprehensive. In the future, we should also collect additional data about the perceived usefulness of the two citation databases. However, we think that the criteria used for evaluation in this study indicate the usefulness of the databases for use in snowball sampling.

8. Conclusion

In this study, we have compared and empirically evaluated two leading alternative sources of citation data for snowball sampling. GS and Scopus have very different features and have different strengths, which will make them suitable for different use cases. However, based on the results of the current study, we conclude that Scopus is a superior source for snowball sampling in SE research when primarily peer-reviewed literature is targeted.

The results of this study suggest that by using Scopus instead of GS researchers can save substantial effort in data collection and reduce the effort spent on selection without a significant likelihood of missing relevant peer-reviewed literature. *Based on these findings, we recommend that the researchers employing a snowball sampling search strategy may use Scopus in the future.*

In the future, we would like to replicate the analysis reported in this study with other published secondary studies and with additional citation databases.

Acknowledgements

This work has been supported by ELLIIT, a Strategic Area within IT and Mobile Communications, funded by the Swedish Government and by research grants for the VITS project (reference number 20180127) and the SERT project from the Knowledge Foundation in Sweden.

References

[1] B.A. Kitchenham, D. Budgen, and P. Brereton, *Evidence-Based Software Engineering and Systematic Reviews*. Chapman & Hall/CRC, 2015.

[2] J. Krüger, C. Lausberger, I. von Nostitz-Wallwitz, G. Saake, and T. Leich, "Search. Review. Repeat? An empirical study of threats to replicating SLR searches," *Empir. Softw. Eng.*, Vol. 25, No. 1, 2020, pp. 627–677.

[3] M. Skoglund and P. Runeson, "Reference-based search strategies in systematic reviews," in *13th International Conference on Evaluation and Assessment in Software Engineering, EASE*, Workshops in Computing, D. Budgen, M. Turner, and M. Niazi, Eds. Durham University, UK: BCS, 2009, pp. 31–40. [Online]. http://ewic.bcs.org/content/ConWebDoc/25022

[4] C. Wohlin, "Guidelines for snowballing in systematic literature studies and a replication in software engineering," in *18th International Conference on Evaluation and Assessment in Software Engineering, EASE'14*, 2014, pp. 38:1–38:10.

[5] J. Bailey, C. Zhang, D. Budgen, M. Turner, and S. Charters, "Search engine overlaps: Do they agree or disagree?" in *2nd International Workshop on Realising Evidence-Based Software Engineering, REBSE'07*, 2007, p. 2.

[6] L. Chen, M.A. Babar, and H. Zhang, "Towards an evidence-based understanding of electronic data sources," in *14th International Conference on Evaluation and Assessment in Software Engineering, EASE*. BCS, 2010, pp. 135–138.

[7] A. Yasin, R. Fatima, L. Wen, W. Afzal, M. Azhar et al., "On using grey literature and Google Scholar in systematic literature reviews in software engineering," *IEEE Access*, Vol. 8, 2020, pp. 36 226–36 243.

[8] N. bin Ali and M. Usman, "A critical appraisal tool for systematic literature reviews in software engineering," *Inf. Softw. Technol.*, Vol. 112, 2019, pp. 48–50. [Online]. https://doi.org/10.1016/j.infsof.2019.04.006

[9] N. bin Ali and M. Usman, "Reliability of search in systematic reviews: Towards a quality assessment framework for the automated-search strategy," *Information and Software Technology*, Vol. 99, 2018, pp. 133–147. [Online]. https://linkinghub.elsevier.com/retrieve/pii/S0950584917304263

[10] M. Usman, N. bin Ali, and C. Wohlin, "A quality assessment instrument for systematic literature reviews in software engineering," *CoRR*, Vol. abs/2109.10134, 2021. [Online]. https://arxiv.org/abs/2109.10134

[11] H.K.V. Tran, J. Börstler, N. bin Ali, and M. Unterkalmsteiner, "How good are my search strings? Reflections on using an existing review as a quasi-gold standard," *e-Informatica Software Engineering Journal*, Vol. 16, No. 1, 2022. [Online]. https://doi.org/10.37190/e-inf220103

[12] P. Singh and K. Singh, "Exploring automatic search in digital libraries: A caution guide for systematic reviewers," in *21st International Conference on Evaluation and Assessment in Software Engineering*, EASE'17. New York, NY, USA: ACM, 2017, pp. 236–241. [Online]. http://doi.acm.org/10.1145/3084226.3084275

[13] R. Fatima, A. Yasin, L. Liu, and J. Wang, "Google Scholar vs. dblp vs. Microsoft Academic Search: An indexing comparison for software engineering literature," in *44th Annual Computers, Software, and Applications Conference (COMPSAC)*. Madrid, Spain: IEEE, 2020, pp. 1097–1098. [Online]. https://ieeexplore.ieee.org/document/9202826/

[14] T. Dybå, T. Dingsøyr, and G.K. Hanssen, "Applying systematic reviews to diverse study types: An experience report," in *Proceedings of the First International Symposium on Empirical Software Engineering and Measurement, ESEM*. ACM / IEEE Computer Society, 2007, pp. 225–234. [Online]. https://doi.org/10.1109/ESEM.2007.59

[15] J.A.M. Santos, A.R. Santos, and M.G. de Mendonça, "Investigating bias in the search phase of software engineering secondary studies," in *12th Workshop on Experimental Software Engineering*, 2015, pp. 488–501.

[16] P. Levay, N. Ainsworth, R. Kettle, and A. Morgan, "Identifying evidence for public health guidance: A comparison of citation searching with Web of Science and Google Scholar: Identifying Evidence for Public Health Guidance," *Research Synthesis Methods*, Vol. 7, No. 1, 2016, pp. 34–45.

[17] N. Bakkalbasi, K. Bauer, J. Glover, and L. Wang, "Three options for citation tracking: Google Scholar, Scopus and Web of Science," *Biomedical Digital Libraries*, Vol. 3, 2006.

[18] J. Ortega and I. Aguillo, "Microsoft Academic search and Google Scholar citations: Comparative analysis of author profiles," *Journal of the Association for Information Science and Technology*, Vol. 65, No. 6, 2014, pp. 1149–1156.

[19] M. Gusenbauer, "Google Scholar to overshadow them all? Comparing the sizes of 12 academic search engines and bibliographic databases," *Scientometrics*, Vol. 118, No. 1, 2019, pp. 177–214.

[20] A. Martín-Martín, M. Thelwall, E. Orduña-Malea, and E.D. López-Cózar, "Google Scholar, Microsoft Academic, Scopus, Dimensions, Web of Science, and OpenCitations' COCI: A multidisciplinary comparison of coverage via citations," *Scientometrics*, Vol. 126, No. 1, 2021, pp. 871–906. [Online]. https://doi.org/10.1007/s11192-020-03690-4

[21] M. Levine-Clark and E. Gil, "A new comparative citation analysis: Google Scholar, Microsoft Academic, Scopus, and Web of Science," *Journal of Business and Finance Librarianship*, Vol. 26, No. 1–2, 2021, pp. 145–163.

[22] H.F. Moed, J. Bar-Ilan, and G. Halevi, "A new methodology for comparing Google Scholar and Scopus," *Journal of Informetrics*, Vol. 10, No. 2, 2016, pp. 533–551. [Online]. https://www.sciencedirect.com/science/article/pii/S1751157715302285

[23] N. bin Ali, E. Engström, M. Taromirad, M.R. Mousavi, N.M. Minhas et al., "On the search for industry-relevant regression testing research," *Empirical Software Engineering*, Vol. 24, No. 4, 2019, pp. 2020–2055.

[24] Z. Yu and T. Menzies, "FAST^2: An intelligent assistant for finding relevant papers," *Expert Syst. Appl.*, Vol. 120, 2019, pp. 57–71. [Online]. https://doi.org/10.1016/j.eswa.2018.11.021

[25] F.D. Davis, "Perceived usefulness, perceived ease of use, and user acceptance of information technology," *MIS quarterly*, 1989, pp. 319–340.

[26] A. Martín-Martín and E.D. López-Cózar, "Large coverage fluctuations in Google Scholar: A ase study," *CoRR*, Vol. abs/2102.07571, 2021. [Online]. https://arxiv.org/abs/2102.07571

[27] J.C.F.d. Winter, A.A. Zadpoor, and D. Dodou, "The expansion of Google Scholar versus Web of Science: A longitudinal study," *Scientometrics*, Vol. 98, No. 2, 2014, pp. 1547–1565.

[28] E.D. López-Cózar, E. Orduña-Malea, and A. Martín-Martín, "Google Scholar as a data source for research assessment," in *Springer Handbook of Science and Technology Indicators*, Springer Handbooks, W. Glänzel, H.F. Moed, U. Schmoch, and M. Thelwall, Eds. Springer, 2019, pp. 95–127. [Online]. https://doi.org/10.1007/978-3-030-02511-3_4

[29] G. Halevi, H. Moed, and J. Bar-Ilan, "Suitability of Google Scholar as a source of scientific information and as a source of data for scientific evaluation – Review of the literature," *Journal of Informetrics*, Vol. 11, No. 3, 2017, pp. 823–834.

[30] L. Adriaanse and C. Rensleigh, "Web of Science, Scopus and Google Scholar a content comprehensiveness comparison," *Electronic Library*, Vol. 31, No. 6, 2013, pp. 727–744.

[31] J.P. Ioannidis, K.W. Boyack, and J. Baas, "Updated science-wide author databases of standardized citation indicators," *PLoS Biology*, Vol. 18, No. 10, 2020, p. e3000918.

[32] K. Petersen and N. bin Ali, "An analysis of top author citations in software engineering and a comparison with other fields," *Scientometrics*, Vol. 126, No. 11, 2021, pp. 9147–9183. [Online]. https://doi.org/10.1007/s11192-021-04144-1

[33] I. Aguillo, "Is Google Scholar useful for bibliometrics? A webometric analysis," *Scientometrics*, Vol. 91, No. 2, 2012, pp. 343–351.

[34] V. Garousi, M. Felderer, and M.V. Mäntylä, "Guidelines for including grey literature and conducting multivocal literature reviews in software engineering," *Infiormation Software Technology*, Vol. 106, 2019, pp. 101–121. [Online]. https://doi.org/10.1016/j.infsof.2018.09.006

[35] N. bin Ali, H. Edison, and R. Torkar, "The impact of a proposal for innovation measurement in the software industry," in *ESEM'20: ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, M.T. Baldassarre, F. Lanubile, M. Kalinowski, and F. Sarro, Eds. Bari, Italy: ACM, 2020, pp. 28:1–28:6. [Online]. https://doi.org/10.1145/3382494.3422163

[36] N. bin Ali and K. Petersen, "Evaluating strategies for study selection in systematic literature studies," in *ACM-IEEE International Symposium on Empirical Software Engineering and Measurement, ESEM'14*, M. Morisio, T. Dybå, and M. Torchiano, Eds. Torino, Italy: ACM, 2014, pp. 45:1–45:4. [Online]. https://doi.org/10.1145/2652524.2652557

[37] K. Petersen and N. bin Ali, "Identifying strategies for study selection in systematic reviews and maps," in *Proceedings of the 5th International Symposium on Empirical Software Engineering and Measurement, ESEM*. IEEE Computer Society, 2011, pp. 351–354. [Online]. https://doi.org/10.1109/ESEM.2011.46

[38] C. Wohlin, P. Runeson, P.A. da Mota Silveira Neto, E. Engström, I. do Carmo Machado et al., "On the reliability of mapping studies in software engineering," *J. Syst. Softw.*, Vol. 86, No. 10, 2013, pp. 2594–2610. [Online]. https://doi.org/10.1016/j.jss.2013.04.076

e-Informatica Software Engineering Journal, Volume 16, Issue 1, 2022, pages: 220107, DOI: 10.37190/e-Inf220107

Microservice-Oriented Workload Prediction Using Deep Learning

Sebastian Ştefan*, Virginia Niculescu*
*Faculty of Mathematics and Computer Science, Babeş-Bolyai University
stefansebii@gmail.com, virginia.niculescu@ubbcluj.ro

Abstract

Background: Service oriented architectures are becoming increasingly popular due to their flexibility and scalability which makes them a good fit for cloud deployments.
Aim: This research aims to study how an efficient workload prediction mechanism for a practical proactive scaler, could be provided. Such a prediction mechanism is necessary since in order to fully take advantage of on-demand resources and reduce manual tuning, an auto-scaling, preferable predictive, approach is required, which means increasing or decreasing the number of deployed services according to the incoming workloads.
Method: In order to achieve the goal, a workload prediction methodology that takes into account microservice concerns is proposed. Since, this should be based on a performant model for prediction, several deep learning algorithms were chosen to be analysed against the classical approaches from the recent research. Experiments have been conducted in order to identify the most appropriate prediction model.
Results: The analysis emphasises very good results obtained using the MLP (MultiLayer Perceptron) model, which are better than those obtained with classical time series approaches, with a reduction of the mean error prediction of 49%, when using as data, two Wikipedia traces for 12 days and with two different time windows: 10 and 15 min.
Conclusion: The tests and the comparison analysis lead to the conclusion that considering the accuracy, but also the computational overhead and the time duration for prediction, MLP model qualifies as a reliable foundation for the development of proactive microservice scaler applications.

Keywords: microservices, web-services, workload-prediction, performance-modeling, microservice-applications, microservice scaler

1. Introduction

Microservice architectures are considered to be the next step in the evolution of Service Oriented Architectures (SOA) that were popularised in the 90s [1]. Some particular aspects of the microservices are their fine granularity, focus on decoupling, scalability, usage of lightweight protocols, and strong DevOps integration [2]. They are currently seeing a huge adoption rate: a survey of Kong Inc. done in the summer of 2019 with 200 technology leaders at large U.S. companies has revealed that 84% of them have embraced microservices, and 40% believe that organizations will fail within 3 years if they do not keep up with these [3]. Furthermore, microservices are a good fit for cloud deployments, proven by the

Submitted: 17 Sep. 2021; Revised: 18 Feb. 2022; Accepted: 09 Mar. 2022; Available online: 25 Mar. 2022

large scale operations of companies like Amazon, Netflix and LinkedIn, and their reported improvements after switching from the monolithic model [4].

Workload prediction is important in order to ensure efficient scaling of these services and optimisation of cloud resource usage, which means starting up new services during periods of high traffic and stopping some of them when resources are not needed. An analysis of the application of microservices, described in [4], has shown that the use of tools designed to deploy and scale microservices reduces infrastructure costs by 70% or more. Improving workload prediction performance means equipping scaler services with better tools for dealing with unexpected traffic spikes, which is translated in both a smoother experience for users and lower costs for maintainers. Autoscaling is the most practical solution since it assures the automatic scaling of microservice instances in order to meet the SLA(Service Level Agreement) [5], without a human agent analyzing and constantly taking scaling decisions. It can be *reactive* or *predictive*, the latter considering multiple inputs like historical information and current trends, in order to predict future traffic patterns.

The main goal of the presented investigation was to find a performant model for microservices workload prediction, which can be later used by a proactive microservice scaler. As a consequence of our goal, the research question that led our investigation was:

"*Do deep learning algorithms lead to better results than classical time series approaches for workload predicting of a microservice autoscaler? If yes, which one is the most appropriate?*"

Previous research in this field mainly uses classical time series approaches (such as ARIMA – autoregressive integrated moving average, Brown's quadratic exponential smoothing or WMA-weighted moving average) [6–8], or simple machine learning [9, 10]. Our investigation uses different deep learning architectures: MLP (Multilayer Perceptron), CNN (Convolutional Neural Network), hybrid CNN-LSTM (CNN Long Short-Term Memory Networks); deep learning was shown to outperform classical methods on some time series prediction tasks [11], and we selected some models of varied complexity.

The contribution of this research is twofold:

- A microservice-oriented prediction methodology adapted to the particularities of this setting, is proposed. The methodology includes steps and decisions that were taken to match practical microservice demands, such as choosing to predict the number of requests, which is a metric that is not influenced by the scaling prediction, and making the prediction in time intervals of an order of minutes and predict a step into the future to allow time for services to be deployed to match the expected traffic. The prediction window size was chosen for accuracy while also allowing time for most application servers or containers to initialize the application. This methodology is also covering data preparation and processing, that is designed for prediction accuracy.
- A comparative analysis of the performance of different prediction models inside the proposed methodology is conducted; the comparison is done between the results obtained using the chosen deep-learning algorithms, and classical time series approaches, but also with some hybrid machine learning models used in industry [9]. The comparison shows important improvements over the previous results, and emphasizes MLP as the best choice for a predictive microservice scaler. MLP seems to be the most appropriate to capture the complexities of the dataset while also having the advantage of faster training time.

The paper is structured as follows: After we present the related work in the next section, we succinctly describe microservice characteristics and the practical aspects which influenced the lines of this research in Section 3.1. Section 3 introduces the proposed

methodology and in Section 4 we refine it in substeps and specify the settings for our experiments. Section 5 presents our practical implementation of the methodology on a specific dataset: the baseline models' results, the tuning process of the deep learning models for selecting the hyperparameters, an evaluation of the best performing ones, and a comparison with the baselines and other research work. Section 5.4 summarises the obtained results, and in addition, in order to emphasise their utility, a proof of concept implementation for an auto-scaling tool using this model is presented. Conclusions and future work are presented in Section 6.

The following abbreviations are used in the paper: ANN (Artificial Neural Networks), ARIMA (AutoRegressive Integrated Moving Average), CNN (Convolutional Neural Network), CNN-LSTM (CNN Long Short-Term Memory Networks); FFT (Fast Fourier Transform); MAE (Mean Absolute Error); MAPE (Mean Absolute Percentage Error); MRE (Mean Relative Error); MLP (Multilayer Perceptron); MSE (Mean Squared Error); RMSE (Root Mean Square Error); RSLR (Robust Stepwise Linear Regression); SVM (Support Vector Machine); VM (Virtual Machine).

2. Related work

Different classical time series models have been applied for web-services workload prediction. Calheiros et al. [6] apply the ARIMA model to cloud workload prediction. The model was evaluated using a trace of English Wikipedia resource requests spanning a duration of four weeks. The data of the first three weeks are used for training and the fourth for prediction using a time window of 1 hour. The obtained MAPE varies from 9% to 22% depending on the confidence interval, which was chosen from 80 to 95 in order to limit the occurrence of underestimations.

Other classical time series models have also been applied, like Brown Exponential Smoothing by Mi et al. [7] obtaining a MRE of 0.064 on the France World Cup 1998 web server trace. Another classical model is Weighted Moving Average, in which recent observations receive more weight than older ones, was applied by Aslanpour et al. [8], and was tested on a NASA server 24h trace, achieving a 5% improvement in response time on a cloud scaling simulator.

It is difficult to identify the best of these classical approaches for our task since the research outlined above used different datasets and evaluations measures. However, we can look for comparisons between different classical models on other time series problems (not necessarily related to workload prediction). Udom and Phumchusri [12] show that ARIMA performs better than other models (Moving Average, Holt's and Winter's exponential methods) in terms of MAPE on four different datasets. ARIMA was also shown to perform better on a short-term forecasting dataset than an exponential smoothing approach [13]. Zhu et al. [14] show that ARIMA outperforms Holt's exponential smoothing model in terms of MSE on air quality time series analysis.

Khan et al. [15] have used Hidden Markov Models to predict workloads for a cluster of VMs. The used dataset comes from an in-production private cloud environment, and the selected metric is the CPU utilization of the VMs. Their model identifies VMs which have similar loads, trained on a trace of 17 days and generates predictions for intervals of 15 min for the next 4 days. Still, their approach only works for a static configuration, because the training dataset is a matrix of the all VMs in the system on the all selected time intervals, and the selected metric is the CPU utilization. This means that if the configuration of the

system changes then the accuracy will not be preserved, and utilization data for the new VMs must be built, and then a retraining session is necessary. We try to propose a model which can dynamically adapt to scaling decisions without penalty in prediction accuracy.

Another example of a static system predictor is the one proposed by Syer et al. [16], which detects variation in workloads between test and production environments for multiple large-scale software systems from the telecommunications domain. As opposed to this approach which discovers various types of workloads and their deviation from the training environment, but can not adapt automatically to system re-configuration, our solution assumes requests homogeneity (discussed in Section 3) and can adapt to automatic scaling events.

Kumar and Singh [10] applied ANN for workload prediction on a seven month log of traffic from a Saskatchewan University web server and a two month one from the NASA Kennedy Space Center web server. They use a classical ANN architecture: one input layer (size 10), one hidden and one output layer, and the model is trained through the SaDE technique, which means learning the weights through evolutionary algorithms. The results of this model were compared to an ANN trained through backpropagation. The model trained with SaDE got 0.013 and 0.001 RMSE on the selected data sets, while for the one with backpropagation a RMSE of 0.265 and 0.119 was obtained.

CloudInsight [9] is one of the most complex models for workload prediction. It uses a technique called "council of experts" – an ensemble of different models, which in this case are: classical time series (autoregressive, moving average, exponential smoothing), linear regression, and machine learning – SVM. Each model has a different prediction weight, which is also real-time learned through a SVM, based on their accuracy on the dataset. The evaluation was done on a subset of the Wikipedia trace [17], on Google cloud data, and on some generated workloads. They indicated that ARIMA and SVM are the two best static predictors they have experimented with. Considering as a performance indicator the normalized RMSE, on average, the ensemble system was 13%–27% better than the baselines (ARIMA, FFT, SVM, RSLR).

A review of how deep learning methods can be applied to time series problems was presented by Gamboa in [11]. The paper distinguishes between three types of problems: classification, forecasting and anomaly detection, presents methods for modeling them, and guidance for selecting appropriate models. It also shows that using these, an improvement in performance could be achieved, on case studies for different applications in which deep learning performed better. Brownlee [18] published a comprehensive guide on applying MLPs, CNNs and LSTMs on various real datasets, and discussed their advantages over classical methods, which were used as baselines for the experiments. The study highlighted the ability of deep learning models to find non linear relationships in data, as opposed to linear methods, like ARIMA; this was the reason to focus on this kind of methods in our investigation.

Lin et al. [19] proposed a hybrid CNN-LSTM architecture for learning trends in time series. It relies on CNN to extract important features from raw time series data, and passes them to the LSTM layers to find long range dependencies in historical data. The model was shown to outperform both CNN and LSTM with around 30% lower RMSE on three real world datasets. These results look promising, and for this reason this is one of the models taken into consideration for our experiments.

There are some approaches for workload prediction of large scale systems that use LSTM models such as Tang et al. [20], Zhu et al. [21] which show it to be a suitable

approach. In our experiments, we have tested the hybrid CNN-LSTM model with the expectation that it would perform better than its individual components.

Zhang et al. [22] used deep learning based on canonical polyadic decomposition to predict workloads for cloud applications (in this case using a trace of 10 days for the PlanetLab platform, which is a global research network that supported the creation of new network services [23]). Their results indicate better performance of the deep learning model than of the state-of-the-art machine learning based approaches. However, while the model is robust in terms of request workload variety, it aims to predict CPU utilization, which, as outlined above, is not a good fit for our investigation.

A significant description of the necessity and of an implementation of a predictive autoscaler for microservices was done by Netflix in [24]. Before implementing *Scryer*, the name of the aforementioned service, they relied on Amazon Auto Scaling service of the Amazon Cloud, which was based on a reactive approach. *Scryer* uses classical time series methods such as Fast Fourier Transformation, which models a sinusoidal over the input data, and linear regression on clusters of points from the predicted time window in previous days. This model addressed three problems encountered with Amazon's scaler: dealing with rapid spikes in demand by preparing ahead of time, restoring compute capacity after outages, and factoring known usage traffic patterns. Netflix are one of the pioneers of microservice technologies, having broken down their monolith application into multiple services covering everything from video streaming, account registration, content recommendations, in the early 2010s, and later becoming an authority in this domain by developing a strong presence in the open source community based on publishing their tools [25].

A predictive scaling policy was later added in Amazon Web Services [26], based on machine learning algorithms. However, this feature is not yet available in other cloud providers such as Microsoft Azure [27] or Google Cloud [28].

Building on top of the related work presented in this section, we aim to apply and compare some deep learning methods, which were shown to be suited for time series in [18] and [19], for the specific task of workload prediction. The success of this task is highlighted by comparing error metrics with those reported by CloudInsight [9] – the specified ensemble of classical and machine learning approaches, on the same dataset, which is a subset of Wikipedia traces.

3. Scaler prediction methodology

This section presents some of the most important characteristics of Microservice architectures in the first part. Based on these characteristics, in the second part we present our proposed scaler prediction methodology which can be applied to any particular implementation of this architecture.

3.1. Microservice characteristics

Web services are generally associated with Service Oriented Architectures (SOA) [1]. The main idea of this type of architecture is to break down monolithic applications into independent parts that are loosely coupled, autonomous, offer a standard contract and act mostly as black boxes to their consumers. This means that services can be developed, updated and deployed independently offering better scalability than traditional architectures.

Microservices [29] are the modern approach of Service Oriented Architectures, and they have several important characteristics [2]:

1. **Fine Granularity** – each service is implemented to serve a specific business case.
2. **Maintainability** – changes of a feature will have limited impact on the overall code-base.
3. **Reusability** – you can select which features to import into a different system.
4. **Agility** – bug fixes and new features can be deployed without retesting or taking down other parts of the system.
5. **Autonomy** – they are separate entities, with their own tech stack, and can be deployed independently.
6. **Loose Coupling** – they communicate using lightweight network protocols such as REST and HTTP.
7. **High Scalability** – due to their autonomy and loose coupling they can scale-out horizontally without incurring heavy communication overhead.

3.1.1. Deployment

All these characteristics make microservices a good fit for cloud deployments. Cloud providers generally offer on-demand resources, which is more convenient for hosting the applications, and allowing less expensive dynamic workloads [4]. If application workloads are fluctuating, then it is advisable to scale the services accordingly, in order to provide smooth experience for the users, and in the same time to use the resources efficiently.

The problem of having unused resources during the periods of low traffic is solved automatically by cloud deployments, by allocating them to some other users who need them. Similarly, it may be necessary to request more resources when a traffic spike is foreseen. Microservice architectures are ideal for these operations because they offer high level of scalability. Due to their fine granularity it is possible to scale only the services that are in high demand, which would not be possible on monolithic applications. Also, since they are designed to be autonomous, it is simple to setup necessary dependencies such as databases without conflicts among instances.

Also, service discovery is one of the key tenets of a microservice-based architecture. Trying to hand-configure each client or to define some form of convention can be very difficult and also unsafe. In order to overcome these kinds of problems service discovery applications are offered. For example, Eureka is the Netflix Service Discovery Server and Client [30]; this server can be configured and deployed to be highly available.

3.1.2. Scaling

The microservices could be scaled manually, which is inefficient, or automatically through a dedicated service. Autoscaling is the process of automatically scaling out instances in order to meet the SLA(Service Level Agreement) [5], which is formed of a list of commitments between clients and service providers, related to different aspects of the service, such as: quality, availability, responsibilities. For example, it could be stated that the application should have 99% uptime, or it should respond to most requests within a given time range.

Autoscaling can be *reactive*, by setting up thresholds such as resource utilization, and instantiating new services when they are reached, or *predictive* by creating new instances ahead of the foreseen traffic spikes. Predictive autoscaling considers multiple inputs like historical information and current trends in order to predict future traffic patterns.

Even though predictive auto-scaling can be done efficiently without having a cloud deployment, for example scaling some service up and other down alternatively on a fixed resource environment, cloud environments are the best fit and research work is done to address this need [31].

3.2. Proposed scaler prediction methodology

We propose a methodology for finding a prediction model to be used by a proactive scaler for microservice architectures, which takes into account the specific characteristics outlined in the previous section. The main steps are the following:

- **Choose one type of services for which to define a proactive scaler**
  - Each microservice type should have its own predictive autoscaler; a microservice is specialized for a single specific task and it will operate very specific requests.
  - It is expected to obtain better prediction accuracy and resource utilization when working with a single type of requests. The request type would increase the dimensionality of the input data, therefore increasing computational resource utilization, and finally impact on the prediction accuracy, since the model would have to learn multiple features (the error will increase proportionally to the number of the request types). Additionally, to put it into practice, a new model would be required in order to estimate how many resources need to be allocated based on multiple request counters, but also on interactions between them.
- **Choose the number of requests per resource to be the selected metric for prediction**
  - The reason for choosing this metric is this metric independence of the scaler's action. Metrics such as CPU utilization or response time, predicted in [32], are affected by the outcome of the predictor, making them an unreliable target. Also, this is in line with the research done by Jindal et al. [33], who proposed a metric for measuring microservice performance based on the number of satisfied requests, called MSC (Microservice Capacity). Thus, a proactive scaler can determine the number of required instances by dividing the predicted incoming traffic to the MSC.
  - Microservices have fine granularity, therefore we can assume request homogeneity – the requests for one specific microservice are uniform (i.e., they could be solved in a similar period of time). This means that for this problem we can use this simple metric without compromising the usefulness of our predictions.
- **Model real service trace data analysis as a time series supervised learning problem**
  - A common dataset which can be extracted from any application's log is a list of timestamps when requests were handled (one such dataset could be extracted from each microservice type, as they are highly autonomous and we can demarcate exactly the requests they received and when they were completed).
  - It is possible to extract more useful information from this data if we model it as a time series problem [34]. Since specific timestamps are not required, but just general access patterns, a feasibly approach would be to group requests into a series of *buckets* (abstraction used for representing time series). A bucket has a fixed width (some time range) and variable height (the number of requests handled by the program in that range). A visualization of such a time series model is presented in Figure 1.

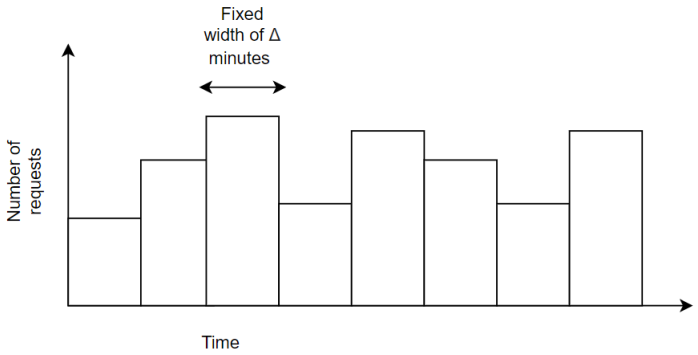


Figure 1. Abstract representation of a time series modeling traffic of one microservice type

- Microservices are highly scalable, so we need to achieve a granularity in predictions as fine as possible, meaning that scaling decisions can be taken as soon as the information is available. We can control this granularity in terms of selecting the width of the time series buckets. Therefore the lower bound (which we are aiming for) of the bucket width is dependent on technological constraints for scaling up/down microservices. This estimation also fits an observation from [15] which has used a 15 min bucket width, from analyzing autocorrelations on their dataset, which consists of a 21 days trace from an in-production distributed application.
- After converting the dataset to time series it must be prepared for being fed to a supervised learning algorithm (the supervised learning is considered because we already know the desired prediction target and we can label our data [35]) which means transforming the series into a list of vectors of the form $(input, output)$. A possible choice for this transformation is based on *sliding window* technique (which was shown to have adequate performance and allow for a wide range of algorithms to be applied to the resulting dataset [36]); more details about using this process of data preparation is detailed in Section 4.2.
- Also, we are not interested to predict the height of the first next bucket in the future, because the scaling decisions might be useless if they can not be executed in practice, meaning that a time is required between the moment in which the scaling decision is taken and the moment when the new microservice application instance is online and can actually process requests. This period of time was outlined previously as the ideal width of a bucket. In order to accommodate this requirement we need a classical approach for multi-step time series prediction (e.g., the Direct strategy from [37]), in which the prediction target is the second window in the future.
- The prediction window is limited to one, in the near future, in order to improve accuracy. This requires periodic predictions, however, once a deep learning model is trained, the actual computational overhead is small (less than a second in our experiments).

- **Apply an appropriate prediction model**
  - Choosing the most appropriate model is a complex problem, and our empirical investigation aimed to provide such a model.
- **Evaluate the results**
  - estimate the prediction error using different metrics;
  - compare the results with similar results obtained using with different prediction models;

– verify using practical usage.

The summary of the proposed methodology is depicted using a diagram in Figure 2.

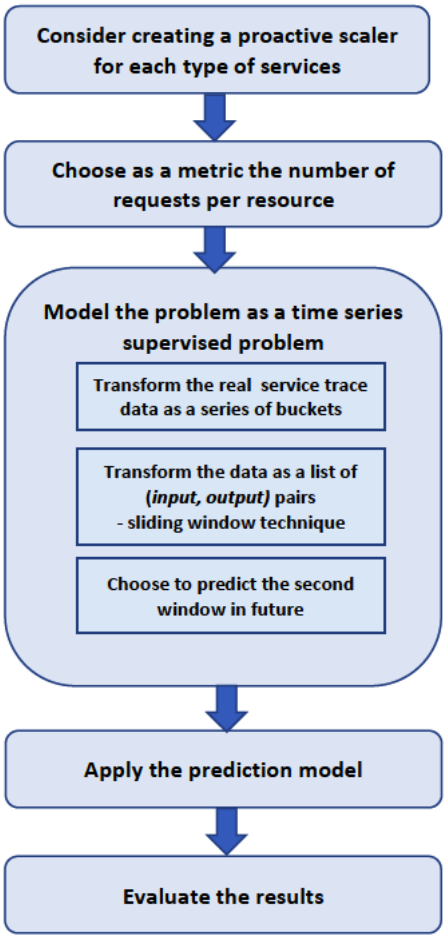


Figure 2. Diagram associated to the methodology for finding a prediction model for a proactive microservices scaler

In order to estimate the most appropriate prediction model for microservice workload prediction (to be applied in the fourth step), we have conducted an investigation based on the specified methodology. This investigation started by choosing and preparing data on which we can do the experiments, extracting a collection of prediction models that are potential candidates, followed by the preparation of their initial settings. After that we did the experiments and the evaluation of the results.

4. Methodology refinement and investigation settings

In this section we present details regarding the data used in the experiments, their preparation as corresponding supervised datasets, and the collection of prediction models that we have chosen as potential candidates.

4.1. Data sets

Very important aspects in the selection of the datasets were to follow real world user traffic patterns, to have a consistent size, and to have some variation which would showcase how the model can handle unpredictable spikes. Based on these we have chosen several Wikipedia traces. Although we do not know the specific implementation of the Wikipedia server, this dataset can be used for testing the model for two reasons:

– the requests are all of the same type (fetch the content of a wiki page) which is in line with the assumption of request homogeneity for microservices, and
– the traffic patterns come from a production server and capture realistic user traffic (random spikes, day/night variation, weekend variations, etc.).

In addition, the appropriateness of this choice is also confirmed by the fact that similar datasets were used in the analysis conducted by Kim et al. [9] that describes the algorithms for the CloudInsight service, which is a commercial cloud scaling and monitoring platform.

The raw data used for the experiments is a Wikipedia trace for 12 days in September 2007 [17], available online at http://www.wikibench.eu/. From this, two subsets of requests were extracted as separate datasets: all requests to Japanese and German Wikipedia, respectively, to facilitate the results comparison with those obtained by Kim et al. [9] which were based on the same data.

The Japanese wiki dataset is presented in Figure 3. The y-axis represents the number of requests and the x-axis the time, measured in 10 minute intervals over the whole period. It shows an interesting variation in the form of a large spike during the 5th day of measured data which could be a challenge for some prediction models.

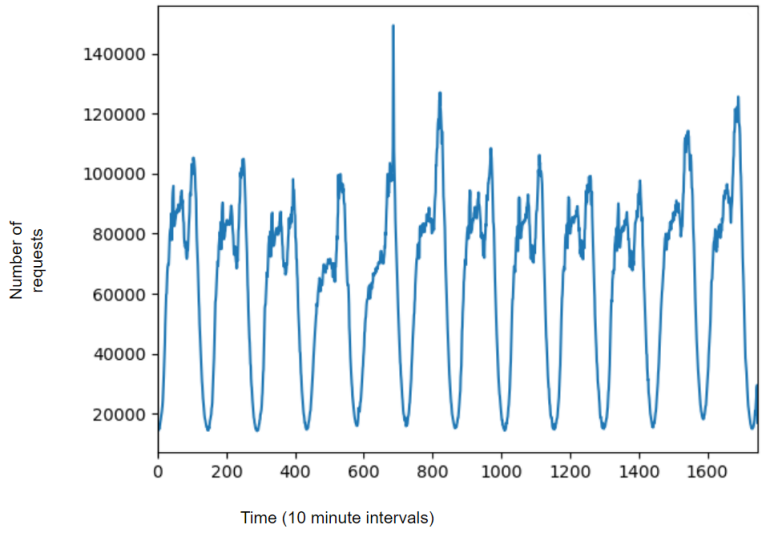


Figure 3. Japanese Wikipedia data visualization: number of requests per 10 minute intervals

The first dataset contains 111 million requests (ja.wikipedia) and the second 101 millions requests (de.wikipedia). The amount of data after preparation is the same as if we would extract from the global trace during the same time range, but with a much faster processing, because the number of buckets depends on the considered time interval, and not on the number of requests.

Since the target bucket width is given by the time in which a microservice application can be reasonably started up we can do some estimations about general technical constraints of this operation. For example, if we considered the Netflix open source stack that is among the most popular approaches for implementing microservices, we have to consider the time for initializing Spring Boot, service discovery (e.g., Eureka [30] – the Netflix default client – needs a refresh time of 30 s, which is recommended on production environments, too), and in some cases performing business logic like initializing in-memory caches from database information. The typical initialization time for microservice frameworks will also add a few seconds [38]. In addition, a typical deployment may also need time for starting up the container (e.g., Docker) or virtual machine. Considering that there are many factors which can influence this interval in our experiments we considered a permissive estimation of a few minutes. Therefore we have chosen two cases for target bucket width: 10 min and 15 min.

4.2. Data preparation

In order to turn a web request log file into a supervised dataset the following steps were taken:

- extract timestamps of all requests for a country (e.g., all lines matching ja.wikipedia);
- create buckets that contain the number of requests in a time interval;
- iterate over the buckets using the sliding window technique, and group them into (input, output) tuples.

Applying sliding window. The starting point for the sliding window time series technique [39] is a time series $(t_1, t_2, \ldots, t_{size})$, where t_i is the number of requests in the i-th bucket. Training instances are then generated with input $(t_i, t_{i+1}, \ldots, t_{i+n-1})$ and output (t_{i+n+1}), where n is the size of the sliding window. This process starts at $i = 1$ and is incremented by 1 until $i = size - n + 1$. The predicted value is t_{i+n+1} instead of t_{i+n} because a scaler using this model would need to have a buffer window during which to deploy the services. These are emphasized in Figure 4.

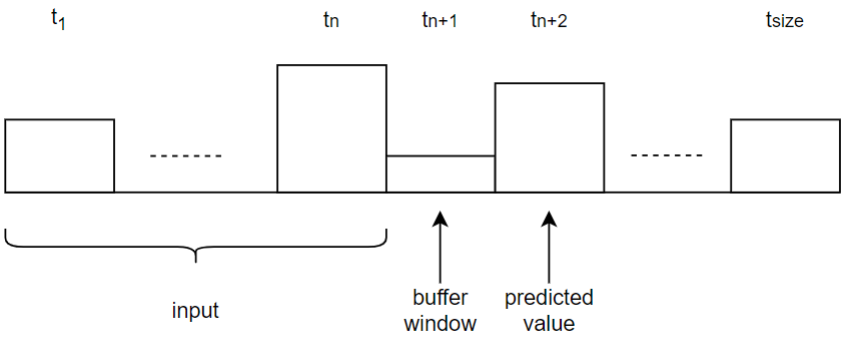


Figure 4. Sliding window technique

Input data were scaled using the min-max scaling technique: $x = \dfrac{(x - \min)}{(\max - \min)}$, which brings the dataset into the $[0, 1]$ range. The same method was applied by Kumar and Singh in [10] in order to speed-up learning. In a practical implementation, this scaling step is

more difficult to apply because it should rely on some hypothetical bounds that have to be determined for future traffic. Still, these bounds could be estimated based on the historical data.

The sizes of the datasets, after applying transformations were 1166 for the 15 min window and 1747 for the 10 min window. The sizes are determined by the sampling window of 12 days and the bucket windows of 10 and 15 minutes.

Performance metrics

The error metrics selected in this investigation are:

$$\text{Mean squared error: } MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2,$$
$$\text{Mean absolute error: } MAE = \frac{1}{n}\sum_{i=1}^{n}|Y_i - \hat{Y}_i|, \text{and}$$
$$\text{Mean absolute percentage error: } MAPE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)/Y_i,$$

where Y_i and $\hat{Y}_i$ are the observed, respectively, the predicted values [40]. MSE was used as the loss function for training because it tends to penalize big deviations in prediction, which is desirable for our problem as we want to accurately predict traffic spikes. MAE is similar, but conceptually simpler, given that each prediction error contributes in proportion to its absolute value. MAPE is independent of the problem scale and can be interpreted intuitively, therefore can be used to give a general evaluation of how well a model performs across different datasets. According to Lewis [41] a highly accurate forecast would have MAPE lower than 10%, and a good forecast between 10% and 20%.

4.3. Baseline models

Baseline models were considered in order to verify in which measure machine learning is useful for this problem, and if using it, features not considered by simpler methods could be learned. Two baseline models were applied: a naive approach, and a classical time series model – ARIMA.

The naive approach just assumes that the predicted workload is the same as the last observed workload. No proactive scaler could use this model as the predicted change in traffic is always null, but it is used in order to check if the proposed models perform better than doing no prediction at all.

ARIMA [42] is a classical approach for modeling time series. It has been selected because it has been applied with good results to workload prediction before [6], and was shown to perform better than other classical models [12–14]. Also, it is a common baseline model for machine learning solutions in time series predictions [9, 43–45]. Furthermore, it combines multiple simpler models (AR and MA) into a performant one. Autoregressive models (AR) make predictions based on previous observations while Moving average(MA) models use recent forecast errors. The integrated part indicates whether the series needs to be differenced, and how many times. Therefore, the parameters of the ARIMA model are:

- p: the number of lag observations included in the model;
- d: the number of times that the raw observations are differenced;
- q: the size of the moving average window.

4.4. Deep learning models

We have chosen in this investigation the following deep learning architectures: MLP, CNN, CNN-LSTM hybrid, since all have been shown to perform well on time series tasks [11, 18, 19]. Also, deep learning has constantly outperformed classical methods in prediction tasks [46].

All the selected models have advantages and drawbacks among each other regarding training speed, the amount of data required to produce good answers, and the tuning of the size of the sliding window to capture relevant recent information.

4.4.1. MLP – Multilayer perceptron

MLPs are quintessential deep learning models that although efficient in their own right also serve as baselines for more sophisticated architectures [47]. It is made up of an input layer, a number of hidden layers and an output layer, linked by weights which are learned through the backpropagation algorithm. While a MLP with one hidden layer is theoretically sufficient to represent any function, that layer may be too large and training could be affected by overfitting, therefore deeper models can help reducing the generalization error [48].

This model has been selected to check whether looking at a smaller sliding window, without taking into account further historical dependencies, achieves satisfactory results.

4.4.2. CNN – Convolutional neural network

CNNs are specialized in dealing with data that has a grid-like topology such as images (2d) or time series (1d) [47]. They have the ability to learn filters which assign importance to some aspects of the input data, which is done by the convolutional layers. Another type of layers that they usually contain are the pooling layers, which reduce the spatial size of the convolved features and make the representation invariant to small translations in input.

CNN has been tested in this experiment because it looks like a natural fit for a time series problem given its assumed spatial dependencies. CNNs can extract only the important features of the input, therefore they can efficiently work with a larger sliding window, and take into account more recent measurements when making a prediction.

4.4.3. CNN-LSTM – CNN long short-term memory networks

The CNN-LSTM architecture involves using Convolutional Neural Network (CNN) layers for feature extraction on input data combined with LSTMs to support sequence prediction. This hybrid model was applied on a range of time series tasks by Lin et al. [19] and was shown to outperform both CNN and LSTM models.

Recurrent Neural Networks (RNN) are Neural Networks that take into account the outcome of previous predictions, while making the current one [47]. LSTM – Long Short-Term Memory, networks are an improvement over RNNs in the sense that they are better at capturing long-term dependencies [49].

This model has been chosen because it combines the ability of CNNs to extract salient features from raw time series data with the capability of LSTMs to find long range dependencies and historical trends.

5. Experiments and evaluation

In this section we describe the experiments that we conducted, their results, and a comparative analysis of these results. The research follows a set of best practices such as: setting baselines, starting with parameters that have been shown to perform well on other problems, exploring possible solutions manually and automatic exhaustive search for fine tuning parameters. The experiments were performed using the data and models described in Section 4.

First a tuning phase has been carried out for each chosen architecture in order to choose the best parameters for each model. The details of this phase are presented in the next subsection.

For the validation we have used k-fold validation [50], which estimates how well a model will perform on previously unseen data and offers a less biased skill estimation than the classical train/test validation method. The k-fold Cross-Validation with $k = 3$ was chosen, which means splitting the training dataset into $k = 3$ equal parts; for cross-validation $k - 1$ parts are used to perform the training (the weights are reinitialised for training on each subset), and the evaluation is done on the part left out, and this process is repeated until an evaluation was done on each of the parts. Finally, the averaged accuracy of all tests was considered. The instances themselves were not shuffled inside the partitions, as their ordering is significant for LSTM models.

Each dataset was split into training (the first 90% of data points) and testing (the remaining 10%) data. After tuning (on the training set of a selected dataset), the resulting models were next trained again on all training datasets, and evaluated on the testing data, which were unseen during tuning and training.

Implementation. All the selected models were implemented in Python programming language. For machine learning models the Keras library [51] was used with some variations (described below) on the following types of layers: Dense for MLP, Conv1D, MaxPooling1D and Dense for CNN, and the previous ones with the addition of LSTM for CNN-LSTM hybrid.

The k-fold validation process was carried out using the scikit-learn library [52]. Statsmodels library [53] was used for ARIMA implementation.

Aside from the configurations described in the article, the default settings of the library were used. The algorithm used in initializing the connection weights of our neural networks models was Glorot Uniform provided by Keras, also called the Xavier initializer [54].

5.1. Hyperparameter optimisation

We have chosen for the tuning phase the Japanese wiki dataset (on the first 90% data points) described in section 4.1 because, besides the fact that includes significant patterns, it also has some interesting irregularities, like a huge spike which is not repeated. As we have previously mentioned, this dataset was also used by Kim et al. [9] and we intend to compare the results.

The selected time window for tuning was set to 10 min, because this is a reasonable prediction time to allow a scaler to spin out new instances, as shown in some previous experiments [24].

5.1.1. Naive baseline

The naive baseline leads to the following results: 2.02×10^7 MSE, 3577.8 MAE and 7.1% MAPE. This illustrates the fact that although the MAPE score would classify it as a very good predictor, it does not do anything useful and the proposed models should achieve better results.

5.1.2. ARIMA

Settings. In order to apply the ARIMA model we had to find appropriate values for its parameters: p, d, q. The value of d represents the number of times the series needs to be differenced in order to make it stationary. The series stationarity was checked using the augmented Dickey–Fuller test [55] which found the p-value to be $1.09e{-}08$. This is lower than 0.05, the commonly used threshold, meaning that we can set the d parameter to 0.

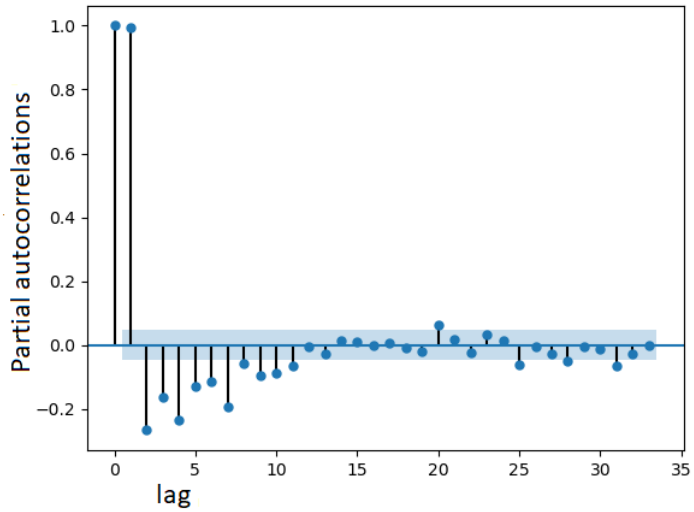


Figure 5. Partial autocorrelation plot for ARIMA

The partial autocorrelation plot (Figure 5) was analyzed to set the autoregression parameter (p). The significance region is confidently passed at 1, with a steep decline afterwards. The moving average parameter (q) is approximated from the autocorrelation plot (Figure 6) which suggests a value of around 20 would be a good start. After fitting ARIMA(1, 0, 20) the final 2 layers had p-value of 0.547 and 0.758, which meant that they were not significant enough, therefore we used 18 as the upper limit for q in our tuning.

Results. The results obtained for several values for $q : 5, 10, 15, 18$, are illustrated in Table 1, the best one being for ARIMA(1, 0, 15) with 1.42×10^7 MSE, 3056.7 MAE and 6.3% MAPE.

5.1.3. MLP

Settings. After some manual experiments we started with a MLP with 2 hidden layers $(150, 100)$ neurons, and a sliding window size of $n = 24$ (this is the window used to

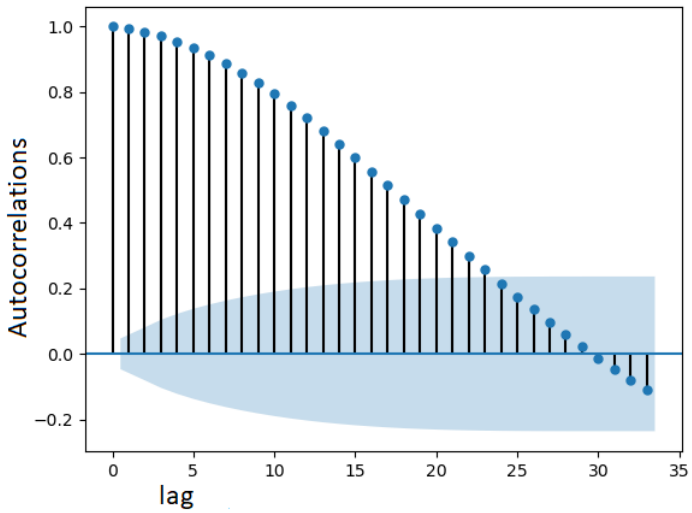


Figure 6. Autocorrelation plot for ARIMA

Table 1. ARIMA tuning – based on different values for q parameter

p	d	q	MSE / 10^6	prediction time
1	0	5	15.4	0.4 s
1	0	10	14.3	9.6 s
1	0	15	14.2	31 s
1	0	18	14.3	71 s

transform the time series data into a supervised dataset, meaning how many buckets are taken into account for each prediction, not the bucket width which was set at 10 min). To find an optimal combination of batch size and epoch numbers a 2d grid search was performed, and the results are presented in Figure 7. Batch size should ideally be a power of 2 for extra performance on GPU architectures, as some experiments were ran on Google Colab's cloud GPU[1]. Using lower batch size is more accurate but the training is slower [56]. As expected the lowest MSE is obtained for the lowest batch size (4), however it does not drop significantly at 8, regardless of epochs numbers. The selection of the epoch numbers is again a trade-off between the speed and the accuracy. We noticed that using a smaller number of epochs (50) the performance is not very good, while the difference between 100 and 250 is not very important, meaning that we can get a good approximation using a model with a epoch size of 100.

Additional experiments were done by adding Dropout layers on different values (0.2, 0.1, 0.05), however it did not improve performance. These are generally used to prevent over-fitting, when the network is too big, the data is scarce, or the training is done for too long [57], which was not the case for this experiment.

Various optimizers and activation functions were tested, and from these Adadelta optimizer and ReL (Rectified Linear) activation function were selected. ReL activation function is also the default recommendation [47] for modern neural networks, because it is non-linear while preserving many advantages of linear functions that make them generalize well. Although the ADAM optimizer is widely used in research, there is no consensus on which is the

[1]https://colab.research.google.com

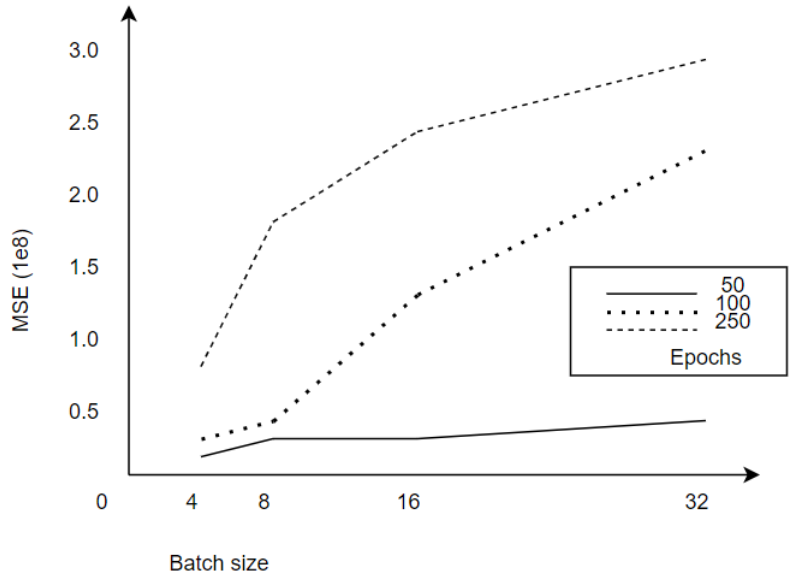


Figure 7. Grid search depending on the epoch number and the batch size

optimal one [47], therefore we choose Adadelta, which in our experiments performed better – 1.92×10^7 vs. 2.58×10^7 values for MSE.

A comprehensive grid search was performed for sliding window size and number and content of hidden layers, of around 90 combinations. Some of the best results are presented in Table 2.

Table 2. MLP tuning – based on different combinations of sliding window size and number and content of hidden layers

Sliding window	Hidden layers	MSE / 10^6
4	(100, 50, 25, 20, 10)	18.8
4	(100, 50, 50, 20, 10)	18.8
8	(100, 50, 25, 20, 10)	17.0
8	(150, 50, 50, 50, 50, 10)	17.2
8	(50, 50, 50, 50)	18.3
16	(10, 20, 30, 40, 50)	18.1
16	(100, 20, 20, 20, 10)	18.4

Results. The final parameters chosen for the proposed MLP model were: Adadelta optimizer with ReL activation function, a sliding window of size 8 with 5 hidden layers of size: $100, 50, 25, 20, 10$.

5.1.4. CNN

Settings. Firstly, a baseline model was selected through manual experimentation. This had the following structure: input of size 20, a 1d convolutional layer, a max pooling layer, a flatten layer, a dense layer of size 150 and the output layer. The same batch size, epoch number grid search was performed and it yielded similar results to those reported in Figure 7 for MLP. This was followed by iterating the same optimizers and activation function which resulted in our selection of Adadelta and softplus.

A grid search was again performed in order to find out the optimal sizes for sliding window, hidden layers and their neurons (see Table 3). This proves our assumption that CNNs can extract better features from larger sliding window as best results were obtained with a window of 128 as opposed to 8 for MLPs.

Table 3. CNN tuning – based on different combinations of sliding window size and number and content of hidden layers

Sliding window	Hidden layers	MSE / 10^6
8	(25, 10, 5)	35.3
64	(100, 20, 10, 5)	35.0
128	(100, 20, 10, 5)	21.0
128	(300, 50)	22.2
128	(10, 10, 10)	23.4
256	(100, 20, 10, 5)	23.7

Results. The parameters selected for the CNN model were: Adadelta optimizer, softplus activation function, a window of size 128 and 4 hidden layers of size: $100, 20, 10, 5$.

5.1.5. CNN-LSTM hybrid

Settings. The starting values for some parameters were influenced by the research done by Lin et al. [19]: a convolutional layer with 32 CNN filters, a max pooling layer, a LSTM layer with a couple of hundred units. In order to feed the output of the convolutional layers into the LSTM layer the input was broken into multiple sequences. This provided the time dimension which LSTM input shape specifies, as the sequences are arranged in a temporal order.

A similar search as for the previous model was performed and as a result we selected Adadelta optimizer and ReL activation function.

While searching for the size of the LSTM layer, we observed a trend where error value would become very large after a couple of epochs, of approximately $1.7 \cdot 10^{27}$. This might be linked with a gradient explosion [47], which causes a network to become unstable because of an increase in the number or values of the gradients with which the inputs are multiplied. Therefore, we applied a common solution, to rescale elements in a gradient vector if their norm exceeds 1, which has solved the issue.

A search was then performed for different combinations of sliding window size (which is transformed into a 2D structure, the input shape of the algorithm), CNN sequences and LSTM units, and the most important results are shown in the Table 4.

Table 4. CNN-LSTM tuning – based on different combinations of input shape and size of LSTM layer

Input shape	LSTM units	MSE / 10^6
(20, 15)	500	246.0
(16, 16)	150	98.3
(16, 16)	500	90.2
(12, 12)	750	93.6
(12, 12)	500	97.3
(8, 8)	500	370.1

It can be observed that the MSE values are quite larger than those reported in the validation of previous models. The reason for this is that the amount of data used for training becomes smaller as we increase the sliding window size. There was also a lot of variance for different runs of the same configuration.

Results. From the previously described experiments we selected the following parameters for this model: Adadelta optimizer, ReL function, $(16, 16)$ input shape (sliding window size equal to 256) with 500 LSTM units.

5.2. Evaluation

The evaluation was done on both Japanese and German Wikipedia traces with two time windows on each, 10min and 15min, thus obtaining 4 data sets. The sliding window size was slightly scaled when evaluating models on the 15min window with a 0.66 ratio to account for the different time ranges in the data set.
This evaluation process of retraining and testing the models has been repeated 10 times to account for the random weight initialization. The results obtained using deep learning models were averaged and then compared to baselines and to each other, and the results could be seen in Table 5.

Table 5. The MSE based comparison of the final results (the values are divided by10^6)

DataSet	Naive	ARIMA	MLP	CNN	CNN-LSTM
Jp10	20.2	15.2	8.5	11.7	7.2
Jp15	87.8	56.6	31.0	35.0	50.2
De10	16.7	10.3	5.1	10.5	21.3
De15	77.4	43.3	17.2	35.4	65.7

If we compare the results across across all datasets then we may conclude that MLP performed very well, consistently passing both baselines. On average, the MLP model was 49% more accurate than the classical ARIMA method which also indicates a better performance than CloudInsight [9] which obtained a 12% improvement over ARIMA on the same dataset.

CNN performed a bit worse, but still managed to beat the baselines in 3 out of 4 cases, while being very close on the other one.

CNN-LSTM has been very inconsistent. On Jp10 dataset it obtained the best result, beating MLP but this performance has not been repeated. In the De10 and De15 experiments it did not even beat ARIMA performance. This has not been improved even after multiple measurements or epochs, as seen on the loss plot from Figure 8, which indicates that the loss improves very little after 100 epochs.

Computational overhead. An aspect which should be noted is the computational overhead of the proposed models. The prediction time for the ARIMA model varies from 0.4 s to 31 s for the most accurate one (see Table 1). The time in which the deep learning models make a prediction (once trained) is much shorter: 0.16 s for MLP, 0.21 s for CNN and 0.24 s for CNN-LSTM.

Best performer: MLP. The comparison revealed this model to be the best performer, beating both ARIMA baseline and the CloudInsight hybrid model. A more detailed comparison with the classical method can be seen in Table 6 taking into account all error metrics. A plot of the predicted traffic on the Jp10 dataset can be seen in Figure 9.

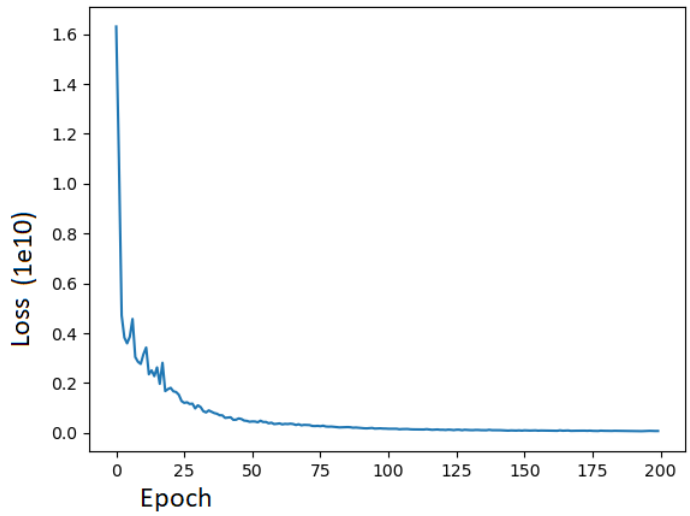


Figure 8. Training loss for CNN-LSTM on De15 with different numbers of epochs

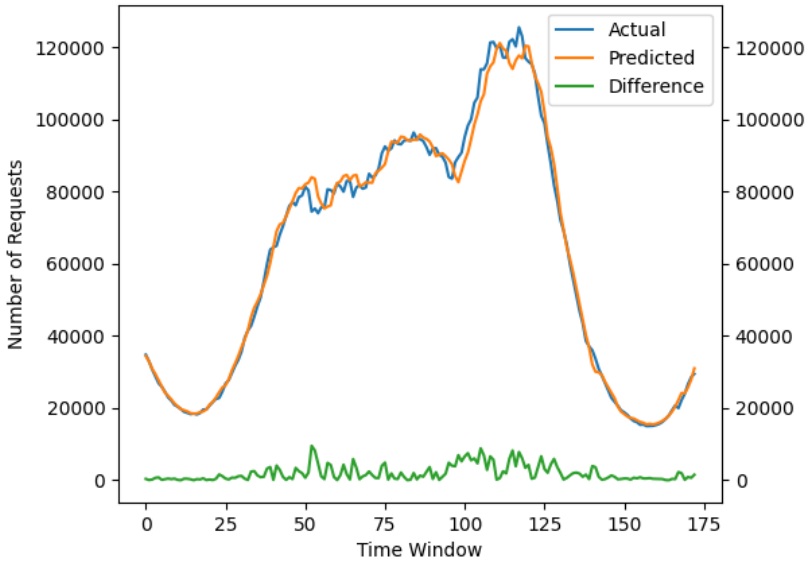


Figure 9. Actual vs. Predicted traffic for about a day on the Jp10 dataset using MLP

Table 6. MLP vs ARIMA, compared based on MSE/MAE/MAPE

	ARIMA			MLP		
DataSet	MSE	MAE	MAPE	MSE	MAE	MAPE
Jp10	15.2E6	3056	6.3%	8.5E6	1960	2.9%
Jp15	56.6E6	6124	8.8%	31.0E6	3540	3.4%
De10	10.3E6	2517	7.6%	5.1E6	1583	3.4%
De15	43.3E6	5606	13.4%	17.2E6	2787	3.9%

5.3. Threats to validity

As with any experiment based analysis, the reported results and conclusions could be subject to certain threats to validity [58]. The followings are the major threats to the validity of our work and the ways we tried to mitigate them.

- Construct validity – For our purposes we assume that the number of requests in an interval is a satisfactory prediction metric. This tends to be enough because a microservice should fit a specific business case therefore having homogeneous requests. The results might not be as accurate when a single microservice type handles widely different requests.
- External validity – We propose a prediction model building methodology which can be customized to the specific microservice deployment it is used on, because the Δt window should be chosen after performance benchmarking the selected application (see Section 3).
- Internal validity – The metric we chose (number of requests) is independent of the prediction result. The other options (e.g., CPU utilization, average response time) would change depending of the scaling actions performed and then influence further decisions, causing a small error to propagate in time.

  We used scaling on input data to improve accuracy and training time. In a real world scenario scaling could still be done using historical bounds which would be updated periodically. Values which are out of these bounds could impact accuracy [59]. In the case of a sudden burst of requests with no historic precedent the measurement of the metric may be impacted. For example if there is a bottleneck of requests waiting to be processed they may not even be counted. However, in a practical deployment the system would eventually scale to handle the requests but it would take multiple scale out commands instead of just one (which is the norm in non exceptional scenarios).
- Reliability – The selected dataset is publicly available [17] and has been used by other researchers for the same goal [9].

  The selected models have been implemented and evaluated using Keras, scikit-learn, and statsmodels libraries. The measurements analysis was in general based on their default settings and on repeating the processes, but an extended analysis of the possible measurements errors could reveal the need of some additional adaptations. From our observations, the MSE value ranges do not wildly fluctuate on multiple measurements, which is also indicated by calculating confidence intervals. For example, on a random re-evaluation with a larger number of repetitions (30) of MLP on De10 dataset with a 95% confidence level the resulting confidence interval was $6.6 \pm 0.67(1e6)$, which still convincingly outperforms the baselines.

  During experiments we have chosen k-fold cross validation with $k = 3$, but we are aware of the fact that a higher value for k could estimate a more accurate confidence interval [60–62]. Our choice was justified by the impact on the computational time of a higher value for k. We tried various settings and layer distributions for some of the more complex models (CNN and CNN-LSTM) and choosing for example $k = 10$ would have led to a much higher asymptotical computational complexity. Still, we acknowledge that would be worth investigating the results that could be obtained using a much higher value of k for cross validation.

5.4. Results analysis

In order to find an efficient model for a proactive auto-scaler for microservices, we have started by analysing the most suitable steps, and we arrived to a methodology adapted to the microservices specifity.

Inside the proposed methodology we compared a naive and a classical time series method – ARIMA, with three deep learning models, MLP, CNN, CNN-LSTM, over two traces of Wikipedia traffic data and two time windows (of 10 and 15 minutes).

This analysis emphasized that MLP(Multilayer Perceptron) shows considerable improvements in performance over the classical method, of around 49% in MSE, which is also better than some state of the art models currently used for this task, like the council of experts employed by CloudInsight [9].

It also showed that the sophisticated hybrid CNN-LSTM can obtain great results (having the best performance on Jp10), however it requires considerably more tuning and training time. Given a larger data trace and tuning effort, it might become the most accurate model.

MLPs are much faster to train than the other deep learning models which facilitates the periodic workload data update a practical application might need.

All these recommend MLP as the best choice for application in a practical proactive auto-scaler. This model was selected as the most appropriate from our implementation (based on the selected collection of models) of the methodology described in Section 3. Still our investigation maybe also seen as a starting point for other applications of this methodology.

5.5. Practical usage

In order to emphasise a possible practical usage of the model presented in this research, we developed a proof of concept implementation of a predictive scaling tool, which is available at https://github.com/StefanSebastian/MicroserviceMonitoring/tree/master/monitor_scaler_app.

The tool, modelled in Figure 10, was designed to simplify the process of monitoring and automatic scaling as much as possible. The main components are:

– a server application which consumes the data stream from all microservice instances (through Kafka message queue) generates various statistics and stores aggregates into the local database for training the prediction model,
– a dashboard monitor application which displays all active microservices and their performance, and

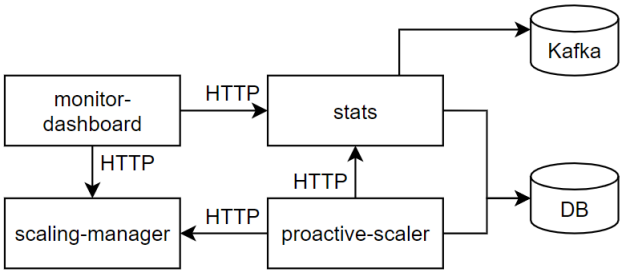


Figure 10. The schema of the proof of concept scaling tool

– a scaling manager which is responsible with starting or stopping instances of the service (in the demo implementation it executes Docker commands to start or stop containerized microservices)
– a proactive scaler which uses historic data to predict traffic patterns (containing a Python implementation of the proposed MLP model) and generates scaling decisions, which are then forwarded to the scaling manager.

An additional Java client is also provided which captures traffic from Spring Cloud microservice implementations and puts onto a kafka queue which feeds data into the scaling system.

This scaling tool was tested on a simplistic microservice system (https://github.com/StefanSebastian/MicroserviceMonitoring/tree/master/demoapp) built on the Spring Cloud stack: a Zuul load balancer, an Eureka name server, and a microservice which simulates workloads, and was shown to work on a manually prepared scenario. The scenario consisted of one spike of traffic repeated over and over again, for which we compared the system performance of reactive and predictive scaling approaches. The conclusion was that in the proactive approach the average processing time of the system was 14% better.

6. Conclusions

The paper proposes a methodology for microservice oriented workload prediction and analyzes whether deep learning models are appropriate to be used as a prediction model for this kind of data. The methodology is adapted to practical microservice demands, such as the metric selection of the number of requests, which are not influenced by the scaling prediction, and the prediction in time intervals of an order of minutes, with a buffer window in which the services can be deployed.

An empirical investigation was conducting in order to find the most appropriate deep learning model to be used for a microservice proactive auto-scaler. The tests and the comparison analysis led to the conclusion that considering the accuracy, but also the computational overhead and the time duration for prediction, MLP (MultiLayer Perceptron) model qualifies as a reliable foundation for the development of proactive micro-service scaler applications.

Future plans include investigation of other models, but also development of a more complex proof of concept project that considers realistic scenarios, with varied traffic patterns over a longer period of time to showcase the accuracy of the proposed tool.

References

[1] T. Erl, *Service-Oriented Architecture: Analysis and Design for Services and Microservices*, 2nd ed. Springer International Publishing, 2016.
[2] S. Newman, *Building Microservices: Designing Fine-Grained Systems*, 2nd ed. O'Reilly Media, 2021.
[3] *2020 Digital Innovation Benchmark*, Kong Inc, 2019. [Online]. https://konghq.com/resources/digital-innovation-benchmark-2020/ Released on konghq website.
[4] M. Villamizar, O. Garcés, L. Ochoa, H. Castro, L. Salamanca et al., "Infrastructure cost comparison of running web applications in the cloud using aws lambda and monolithic and microservice architectures," in *Proceedings of 16th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid)*, 2016, pp. 179–182.
[5] R.V. Rajesh, *Spring Microservices*. Packt Publishing, 2016.

[6] R.N. Calheiros, E. Masoumi, R. Ranjan, and R. Buyya, "Workload prediction using ARIMA model and its impact on cloud applications' QoS," *IEEE Transactions on Cloud Computing*, Vol. 3, No. 4, 2015, pp. 449–458.

[7] H. Mi, H. Wang, G. Yin, Y. Zhou, D. Shi et al., "Online self-reconfiguration with performance guarantee for energy-efficient large-scale cloud computing data centers," in *Proceedings of 2010 IEEE International Conference on Services Computing*, 2010, pp. 514–521.

[8] M.S. Aslanpour, M. Ghobaei-Arani, and A. Toosi, "Auto-scaling web applications in clouds: A cost-aware approach," *Journal of Network and Computer Applications*, Vol. 95, 07 2017, pp. 26–41.

[9] I.K. Kim, W. Wang, Y. Qi, and M. Humphrey, "Cloudinsight: Utilizing a council of experts to predict future cloud application workloads," in *Proceedings of the 11th International Conference on Cloud Computing (CLOUD)*, 2018, pp. 41–48.

[10] J. Kumar and A.K. Singh, "Workload prediction in cloud using artificial neural network and adaptive differential evolution," *Future Generation Computer Systems*, Vol. 81, 2018, pp. 41–52.

[11] J.C.B. Gamboa, "Deep learning for time-series analysis," *CoRR*, Vol. abs/1701.01887, 2017. [Online]. http://arxiv.org/abs/1701.01887

[12] P. Udom and N. Phumchusri, "A comparison study between time series model and ARIMA model for sales forecasting of distributor in plastic industry," *IOSR Journal of Engineering (IOSRJEN)*, Vol. 4, No. 2, 2014, pp. 32–38.

[13] K.I. Stergiou, "Short-term fisheries forecasting: comparison of smoothing, ARIMA and regression techniques," *Journal of Applied Ichthyology*, Vol. 7, No. 4, 1991, pp. 193–204. [Online]. https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1439-0426.1991.tb00597.x

[14] J. Zhu, R. Zhang, B. Fu, and R. Jin, "Comparison of ARIMA model and exponential smoothing model on 2014 air quality index in Yanqing County, Beijing, China," *Applied and Computational Mathematics*, Vol. 4, No. 6, 2015, pp. 456–461.

[15] A. Khan, X. Yan, S. Tao, and N. Anerousis, "Workload characterization and prediction in the cloud: A multiple time series approach," in *Proceedings of 2012 IEEE Network Operations and Management Symposium*, 2012, pp. 1287–1294.

[16] M.D. Syer, W. Shang1, Z.M. Jiang, and A.E. Hassan, "Continuous validation of performance test workloads." *Automated Software Engineering*, Vol. 24, 3 2016, pp. 189–231.

[17] G. Urdaneta, G. Pierre, and M. van Steen, "Wikipedia workload analysis for decentralized hosting," *Elsevier Computer Networks*, Vol. 53, No. 11, July 2009, pp. 1830–1845.

[18] J. Brownlee, *Deep Learning for Time Series Forecasting: Predict the Future with MLPs, CNNs and LSTMs in Python*. Machine Learning Mastery, 8 2018.

[19] T. Lin, T. Guo, and K. Aberer, "Hybrid neural networks for learning the trend in time series," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, 2017, pp. 2273–2279.

[20] X. Tang, "Large-scale computing systems workload prediction using parallel improved LSTM neural network," *IEEE Access*, Vol. 7, 2019, pp. 40 525–40 533.

[21] Y. Zhu, W. Zhang, Y. Chen, and H. Gao, "A novel approach to workload prediction using attention-based LSTM encoder-decoder network in cloud environment," *EURASIP Journal on Wireless Communications and Networking*, 2019, pp. 1–18.

[22] Q. Zhang, L.T. Yang, Z. Yan, Z. Chen, and P. Li, "An efficient deep learning model to predict cloud workload for industry informatics," *IEEE Transactions on Industrial Informatics*, Vol. 14, No. 7, 2018, pp. 3170–3178.

[23] *PlanetLab – An open platform for developing, deploying, and accessing planetary-scale services.* [Online]. https://planetlab.cs.princeton.edu Read October-2020.

[24] D. Jacobson, D. Yuan, and N. Joshi, *Scryer: Netflix's Predictive Auto Scaling Engine*, 2013. [Online]. https://netflixtechblog.com/scryer-netflixs-predictive-auto-scaling-engine-a3f8fc922270 Read 17-October-2020.

[25] *Why You Can't Talk About Microservices Without Mentioning Netflix*, SmartBear Software, (2015, December). [Online]. https://smartbear.com/blog/develop/why-you-cant-talk-about-microservices-without-ment/ Read October-2020.

[26] J. Bar, *New-Predictive Scaling for EC2, Powered by Machine Learning*, (2018, November). [Online]. https://aws.amazon.com/blogs/aws/new-predictive-scaling-for-ec2-powered-by-machine-learning/ Read October-2020.

[27] *Autoscaling guidance – Best practices for cloud applications*, Microsoft, (2017, May). [Online]. https://docs.microsoft.com/en-us/azure/architecture/best-practices/auto-scaling Read 17-October-2020.

[28] *Autoscaling groups of instances*, Google, 2014. [Online]. https://cloud.google.com/compute/docs/autoscaler Read October-2020.

[29] P. Jamshidi, C. Pahl, N.C. Mendonça, J. Lewis, and S. Tilkov, "Microservices: The journey so far and challenges ahead," *IEEE Software*, Vol. 35, No. 3, 2018, pp. 24–35.

[30] *Spring Cloud Netflix*. [Online]. https://cloud.spring.io/spring-cloud-netflix/reference/html/ Read October-2020.

[31] P. Singh, P. Gupta, K. Jyoti, and A. Nayyar, "Research on auto-scaling of web applications in cloud: Survey, trends and future directions," *Scalable Computing: Practice and Experience*, Vol. 20, 05 2019, pp. 399–432.

[32] A.A. Bankole and S.A. Ajila, "Predicting cloud resource provisioning using machine learning techniques," in *Proceedings of the 26th IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*, 2013, pp. 1–4.

[33] A. Jindal, V. Podolskiy, and M. Gerndt, "Performance modeling for cloud microservice applications," in *Proceedings of the 2019 ACM/SPEC International Conference on Performance Engineering*, ICPE '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 25–32.

[34] R. Shumway and D. Stoffer, *Time Series Analysis and Its Applications with R Examples*, 3rd ed. Springer, 2011.

[35] M. Alloghani, D. Al-Jumeily, J. Mustafina, A. Hussain, and A.J. Aljaaf, *A Systematic Review on Supervised and Unsupervised Machine Learning Algorithms for Data Science*. Cham: Springer International Publishing, 2020, pp. 3–21. [Online]. https://doi.org/10.1007/978-3-030-22475-2_1

[36] T.G. Dietterich, "Machine learning for sequential data: A review," in *Structural, Syntactic, and Statistical Pattern Recognition*, T. Caelli, A. Amin, R.P.W. Duin, D. de Ridder, and M. Kamel, Eds. Berlin, Heidelberg: Springer, 2002, pp. 15–30.

[37] G. Bontempi, S. Ben Taieb, and Y.A. Le Borgne, *Machine Learning Strategies for Time Series Forecasting*. Springer Berlin Heidelberg, 01 2013, Vol. 138, pp. 62–67.

[38] M. Smeets, *Microservice framework startup time on different JVMs*, 2019. [Online]. https://technology.amis.nl/languages/java-languages/microservice-framework-startup-time-on-different-jvms-aot-and-jit/ Read 26-June-2021.

[39] R. Frank, N. Davey, and S. Hunt, "Time series prediction and neural networks," *Journal of Intelligent and Robotic Systems*, 2001, pp. 91–103.

[40] A. Botchkarev, "Performance metrics (error measures) in machine learning regression, forecasting and prognostics: Properties and typology," *Interdisciplinary Journal of Information, Knowledge, and Management*, Vol. 14, 2019, p. 045–076.

[41] C.D. Lewis, *Industrial and business forecasting methods: A practical guide to exponential smoothing and curve fitting*. London(U.A.): Butterworth Scientific, 1982.

[42] S.L. Ho and M. Xie, "The use of ARIMA models for reliability forecasting and analysis," *Computers and Industrial Engineering*, Vol. 35, No. 1–2, Oct. 1998, p. 213–216.

[43] A.O. Adewumi and C.K. Ayo, "Comparison of ARIMA and Artificial Neural Networks models for stock price prediction," *Journal of Applied Mathematics*, Vol. 2014, 03 2014, pp. 1–7.

[44] S. Siami-Namini, N. Tavakoli, and A. Siami Namin, "A comparison of ARIMA and LSTM in forecasting time series," in *Proceedings of 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 2018, pp. 1394–1401.

[45] V.R. Prybutok, J. Yi, and D. Mitchell, "Comparison of neural network models with ARIMA and regression models for prediction of Houston's daily maximum ozone concentrations," *European Journal of Operational Research*, Vol. 122, No. 1, 2000, pp. 31–40. [Online]. https://www.sciencedirect.com/science/article/pii/S0377221799000697

[46] T.J. Sejnowski, *The Deep Learning Revolution*. The MIT Press, 2018.
[47] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, Adaptive Computation and Machine Learning series. MIT Press, 2016. [Online]. https://books.google.ro/books?id=Np9SDQAAQBAJ
[48] R.D. Reed and R.J. Marks, *Neural Smithing: Supervised Learning in Feedforward Artificial Neural Networks*. Cambridge, MA, USA: MIT Press, 1998.
[49] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, Vol. 9, No. 8, Nov. 1997, p. 1735–1780.
[50] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 3rd ed. USA: Prentice Hall Press, 2009.
[51] F. Chollet and et al., *Keras*, GitHub, 2015. [Online]. https://github.com/fchollet/keras
[52] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion et al., "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, Vol. 12, 2011, pp. 2825–2830.
[53] S. Seabold and J. Perktold, "Statsmodels: Econometric and statistical modeling with Python," in *Proceedings of the 9th Python in Science Conference*, 2010.
[54] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, Machine Learning Research, Y.W. Teh and M. Titterington, Eds., Vol. 9. Chia Laguna Resort, Sardinia, Italy: PMLR, 13–15 May 2010, pp. 249–256. [Online]. https://proceedings.mlr.press/v9/glorot10a.html
[55] Y.W. Cheung and K.S. Lai, "Lag order and critical values of the augmented Dickey–Fuller test," *Journal of Business & Economic Statistics*, Vol. 13, No. 3, 1995, pp. 277–280.
[56] N. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. Tang, "On large-batch training for deep learning: Generalization gap and sharp minima," in *Proceedings of 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26*, 2017.
[57] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, Vol. 15, No. 56, 2014, pp. 1929–1958.
[58] P. Runeson and M. Höst, "Guidelines for conducting and reporting case study research in software engineering," *Empirical Software Engineering*, Vol. 14, 2008, pp. 131–164.
[59] S. Nayak, B.B. Misra, and H.S. Behera, "Impact of data normalization on stock index forecasting," *International Journal of Computer Information Systems and Industrial Management Applications*, Vol. 6, No. 2014, 2014, pp. 257–269.
[60] J.D. Rodriguez, A. Perez, and J.A. Lozano, "Sensitivity analysis of k-fold cross validation in prediction error estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 32, No. 3, 2010, pp. 569–575.
[61] R.R. Bouckaert, "Estimating replicability of classifier learning experiments," in *Proceedings of the Twenty-First International Conference on Machine Learning*, ICML '04. New York, NY, USA: Association for Computing Machinery, 2004. [Online]. https://doi.org/10.1145/1015330.1015338
[62] M. Huk, K. Shin, T. Kuboyama, and T. Hashimoto, "Random number generators in training of contextual neural networks," in *Proceedings of 13th Asian Conference on Intelligent Information and Database Systems*, N.T. Nguyen, S. Chittayasothorn, D. Niyato, and B. Trawiński, Eds. Cham: Springer International Publishing, 2021, pp. 717–730.

e-Informatica Software Engineering Journal, Volume 16, Issue 1, 2022, pages: 220108, DOI: 10.37190/e-Inf220108

Empirical AI Transformation Research: A Systematic Mapping Study and Future Agenda

Einav Peretz-Andersson*, Richard Torkar**

*Department of Computing, School of Engineering, Jönköping University, Sweden
**Department of Computer Science and Engineering,
Chalmers and University of Gothenburg, Sweden

Einav.Peretz.Andersson@ju.se, torkarr@chalmers.se

Abstract

Background: Intelligent software is a significant societal change agent. Recent research indicates that organizations must change to reap the full benefits of AI. We refer to this change as AI transformation (AIT). The key challenge is to determine how to change and which are the consequences of increased AI use.
Aim: The aim of this study is to aggregate the body of knowledge on AIT research.
Method: We perform an systematic mapping study (SMS) and follow Kitchenham's procedure. We identify 52 studies from Scopus, IEEE, and Science Direct (2010–2020). We use the Mixed-Methods Appraisal Tool (MMAT) to critically assess empirical work.
Results: Work on AIT is mainly qualitative and originates from various disciplines. We are unable to identify any useful definition of AIT. To our knowledge, this is the first SMS that focuses on empirical AIT research. Only a few empirical studies were found in the sample we identified.
Conclusions: We define AIT and propose a research agenda. Despite the popularity and attention related to AI and its effects on organizations, our study reveals that a significant amount of publications on the topic lack proper methodology or empirical data.

Keywords: AI transformation, digital transformation, organizational change, systematic mapping study

1. Introduction

Artificial Intelligence (AI) technology can yield a competitive advantage and new business models for many types of organizations [1] provided that they have sufficient knowledge, skills, and a suitable infrastructure [2]. Technology adoption is one of the driving forces of economic growth [3]. In particular, this adoption can help in tackling global challenges such as health, education, environment, science and it has significant capability to address our regional, local, and organizational challenges [4]. However, technology adoption itself can be a challenge that leads to success or failure based on how it is tackled.

AI is intrinsically software-based and entails massive software engineering [5]. The increased use of AI is closely connected to recent hardware development (the computational resources are now sufficient) and developments in software engineering (it is now possible to

Submitted: 03 Nov. 2021; Revised: 20 May 2022; Accepted: 25 May 2022; Available online: 31 May 2022

design, implement, and test AI-based software systems) [6]. Most successful AI applications are data-driven and use machine learning as core technology [7]. Organizations today are either developers or users of data-driven products and services. An organization is deemed data-driven, or AI mature, if it possesses sufficient knowledge and skills to use AI internally (to improve the organization) and externally (to improve products or services) [8]. The main question for many organizations is how to successfully adopt AI (become data-driven and achieve AI maturity). Despite the expansion, availability, and value of AI technologies, organizations are still struggling to adopt AI [1]. Recent collaborative research made by researchers from MIT, University of Toronto, and the US Census Bureau point out that the adoption rate of AI technologies in organizations is low in general and concentrated to older and larger firms [9].

Organizations are difficult to change in the ways necessary for technology adoption. With rapid development and change of AI technologies, organizations must change continuously. Various factors that could potentially influence willingness or the ability to adopt AI include the availability of relevant resources (computational, economical, and human), legislation (governance and ethics), cost, limited computational capability and infrastructure, security, organization size and structure, traditions, and organizational culture [10]. We identify several studies that explore the phenomenon of digital transformation (DT). One existing definition of DT states that it is a "radical improvement in business performance and operations outcomes due to the use of technology" [11]. DT is thus a very broad umbrella term encompassing all transformations relating to digital technologies.

We argue that the type of transformation that organizations need to undergo to benefit from AI technology is sufficiently different from DT to deserve its own definition and exploration. Our main motivation for making this distinction is connected to the primary function of AI, which is to offload cognitive work from humans to computers. This functionality will potentially lead to more drastic organizational changes than DT in general. The key problem is that AIT is understudied as a distinct phenomenon. This means that it is typically defined indirectly through digital transformation research and explorations into which factors actually contribute to failure or successful AI adoption are scarce.

The aim of this study is to aggregate the body of knowledge concerning the relationship between AI and organizational transformation (OT), to map the field by performing a systematic mapping study (SMS) and, by doing so, identifying gaps in research that represent opportunities for future studies. Our work can help organizations to optimize AIT by finding approaches and models that have been successful in similar contexts.

This study is organized as follows: Section 2 discuss the concept of AI and organization. Section 3 present the aim and the scope, lists and motivates the research questions, and discusses the methodology and the threat and validity. Section 4 summarizes the results for each research question and describes the overall implications of the results. Section 5 includes an assessment of validity threats as well as a discussion and definition of AIT. Finally, Section 6 provides conclusions and pointers to future work.

2. Background

2.1. AI and organizations

AI changes the composition of human skills and tasks required in an organization [3]. Organizations need to develop new knowledge and competences to comprehend new technologies so that they will be aligned with the strategy, processes, and structure.

Organizational adaptation to AI can be viewed as an external catalyst for change, where organizations react on a strategic and tactical level. It also acts as an augmentation to an internal catalyst, where organizations change processes and their operation to meet the technological and societal challenges [12]. Adaptation of AI is inevitable and will affect the business models.

AI will eventually change the composition, business models, and the tasks required in an organization. New business models can be a result of strategy or a strategizing action [13]. AI vastly changes organizations' resources, operations and structure. It is argued that organizations that adjust to this change will become more efficient [14].

Rapid change requires organizations to be flexible and to quickly adapt and adopt new technologies while preparing the organization from a human, societal, and technological perspective to meet this dynamic change [15]. It is suggested that the social and technological changes that organizations are experiencing in the new millennium, will lead to changes in social values, practices, and in the structure and processes of organizations [16]. It has been pointed out that AI has immense influence on organizations, such as: reducing costs, improving human task solving efficiencies, and supporting business customer relationships [17]. However, there are also limitations of AI, and humans will still play an important role in the organization as well [17]. In addition, the importance of human skills that cannot be *learned* by intelligent technologies, will only increase [18]. A reference is made to Michael Polanyi's expression "we know more than we can tell" [19]. This is known as Polanyi's Paradox [18]; where many decisions and actions made by humans cannot be learned or described, which creates an implication for intelligent technologies to duplicate human behavior or improve upon *gut feelings* [20]. Decision making based on *gut feeling* cannot explain the reasons behind the decisions, which are often described as *feel-right decisions*. Moreover, it is hard to identify which decisions are based on this kind of intuition since many employees will find it hard to admit that a crucial decision they have made is based on gut feeling [20].

The discussion of the effect of new technologies on organizations and the changes they will lead to are not new, but rather a continuous discussion of previous industrial revolutions. Research shows two different approaches toward AI: the utopian, where machines will improve human life quality, and the dystopian, where machines will take over the human society [21]. AIT triggers scholarly interests in various disciplines. The scientific literature presents various models related to smart technology transformation, regardless of whether the future of AI will be utopian, dystopian, or something in between, research must be carried out to support the best possible use of AI.

We observe large organizations such as Apple, Amazon, Microsoft, Google, Facebook, and other corporations that have the resources (capital and human) and the market position to invest and develop their use of AI technologies to transform their organization. In addition, research shows that companies upgrade their workflows and the way they work based on AI technologies which lead to enhancement of their financial and market performance [22].

2.2. What is AI transformation (AIT)?

Artificial Intelligence (AI) is used as a genre name and it is becoming increasingly discussed, following the developments related to, for example: IBM Watson, Google DeepMind, Google AlphaGo, and IBM Deep Blue. To the best of our knowledge, this is the most well-known case dealing with AIT. However, AI transformation has also been observed in other studies,

such as the impact of AI on business performance, business value, business capabilities etc. One example is an in-depth study on the impact of AI on firm performance that presents a framework for building on the business value of AI-based transformation projects based on 500 case studies from IBM, AWS, Cloudera, Nvidia, Conversica, and Universal Robots websites [23]. It becomes a dynamic tool that people and communities make use of to refer to various technologies. AI does not have a specific, universal definition but its overarching focus is intelligent systems that can think humanly, act humanly and learn as humans [24]. AI discussions often feature topics such as the possibility of machines to perform as humans in terms of thought processes, reasoning, and behavior. From a technological point of view, AI includes a number of subareas of importance [25]: machine learning deals with the intellectual ability to learn from experience and to improve in order to increase the performance at solving some task, natural language processing deals with the interpretation and production of natural (human) language, computer vision deals with the parsing of data from vision-based sensors to capture aspects of the physical world in the computer, agent-based systems deal with simulation and optimization of micro an macro world models [6].There are a number of additional subareas of AI and it is possible to view some of these different subareas as complementary or overlapping in terms of the overall mission to design intelligent computer-based systems. In this SMS, we view AI as an umbrella term for all such subareas.

The focus on AI as an interdisciplinary research area is relatively new, and the capacity of this technology is versatile and enormous [26]. The interest associated with AI involves economical, psychological, technological, political, and ethical aspects [27]. AIT receives scholarly interest from various domains as well as the attention of various industries in recent years (see the linked data sheet for more detailed information [28]). We also observe that there is a substantial scientific discussion around digital transformation [11, 29–31], but few studies focused only on AI transformation.

Out of the 52 papers we identify in this study, 23% discuss digital transformation of AI technologies (the technologies that are discussed in these papers are AI, Big Data Analytics (BDA), and Data Analytics (DA)). We observe that other concepts such as various smart industries, i.e., smart manufacturing, smart agriculture, and Industry 4.0 also discuss the concept of AIT (see subsection 4.2). This helps us to distinguish between the two concepts and to argue that this SMS mainly discusses AIT and focuses only on AI technologies, which is partly discussed in DT.

AIT should be discussed distinctly from any other DT. The reason for this is that, unlike other forms of DT, AIT will clearly shift cognitive work from human actors to computers. The consequence for many organizations is significant.

2.3. Organizational transformation

Organizational transformation can be described from various perspectives; on the one hand it denotes to be a radical change in the form or character of something or someone that completely changes the organization. Transformational change is discussed as a complex phenomenon, where the change requires a shifting of the current organization strategy, structure, process, culture, work behavior and mindset [32]. This change occurs by a breakthrough to pursue new opportunities. Furthermore, it is argued that organizations that will not identify these types of needs for a change will be disrupted [32].

On the other hand, the change can also be considered to be incremental; an ongoing, gradual, discontinuous process which leads to change [33]. It is argued that organizational

change is a continuous process in organizations as a result of various activities that occur on a regular basis, such as hiring new employees, getting new facilities, renewing the organization strategy, implementing new technologies, and restructuring [34]. Continuous and confluent organizational change can be described as a slow and evolutionary change which is not episodic or a result of a crisis [35]. The organizational ambidexterity theory states that organizations as part of their growth, in a simultaneous way, need to pursue both an evolutionary change – a discontinuous incremental change where the organization is expanding the existing business – and a revolutionary innovative change where the organization is incubating novel opportunities [36].

3. Research methodology

The following section refers to present the aim and the scope, lists and motivates for the research questions, and discusses the methodology and the threat and validity techniques used to obtain and analyze data. This part outlines the approach used in order to fulfill the purpose of this paper.

3.1. Aim and scope

The focus of this article is on change that is led by a particular purpose; AI, we are interested in both incremental and radical change that will lead to a transformation in the organization. We will follow AIT as a change agent; an incremental or radical change that can happen in the organization. By using AI capabilities, the traditional organization transforms its structure, processes, organizational learning, work routine, knowledge management, products, and services [37]. We do not focus on the process of the change or in a particular model or theory that explains the change, but rather on the concept of AIT.

To explore AIT, it is important to understand the concept of AI and its implications, while understanding its relationship to the organizational structure, leadership, culture, vision, and mission and the human attributes within the organization. Organizations are frequently integrating various technologies, but technology transformation related to AI is considered to have a strong impact on organizations [12]. AIT is related to the integration and adaptation of AI into an organization's business, although it can also be considered as a disruptive process that creates new forms of organizations [38].

The scope of this study is AIT. The aim is to follow the SMS process to aggregate the body of knowledge on AIT research and to map the field and identify the research gaps that represent opportunities for future studies [39].

3.2. Research questions

The research questions and the motivation for each question are formulated based on the aim for the SMS. In this work, we seek to answer the following research questions:

RQ1. How is AI transformation conceptualized in the literature?

Motivation. To find existing definitions of AIT in the literature, to analyze these definitions to identify contradictions, similarities, or issues. This analysis can be used to establish a common and useful definition for AIT.

RQ2. What are the research methods used in AI transformation research?

Motivation. An understanding of which research methods are applied, and how, allows us to assess the maturity of the research, and to characterize the existing body of knowledge generated in the field.

RQ2.1. What are the main theories and frameworks adopted in AI transformation research?

Motivation. AIT is inherently interdisciplinary. Due to this, theories and frameworks come from multiple disciplines, which makes it difficult for a specific discipline to make sense of results and conclusions. An understanding of the underlying theories and frameworks of AIT enables the establishment of a unified framework, in which results, and conclusions could be reinterpreted by any discipline, and by stakeholders from the private and public sector.

RQ2.2. What real-world scenarios and contexts are studied in AI transformation research?

Motivation. To identify the maturity of AIT in different domains, and to explore unique characteristics related to AI transformation in these domains.

RQ3. What are the emerging questions for future research and the important research gaps in the area?

Motivation. It is important to identify the major trends of AIT research and to identify research gaps, as they seed new research opportunities. In addition, an ever-increasing number of organizations are looking into how to transform due to AI. The identified research gaps may allow new research that helps these organizations reap the benefits and mitigate the risks involved in AIT.

By addressing these research questions, we aim to provide an insight concerning AIT definitions existing in literature. Secondly, we propose categories, based on the theories used in the literature, which may increase the clarity about existing research relating to AIT. Thirdly, we strive to offer a foundation for future research by finding research gaps in this research field.

3.3. Literature review procedure

Systematic mapping study (SMS) can be described as identifying, evaluating, and interpreting the available knowledge within a particular phenomenon of interest [39]. We follow the Kitchenham procedure [39] for performing the SMS. SMS is a form of literature review where one can gain a transparent and rigorous assessment of the literature. Furthermore, they aim to provide a foundation and empirical answer for one or more research question [39, 40] and to discover research trends [41].

Three databases (Science Direct, Scopus, and IEEE Xplore) are used as main literature sources. The motivation for this selection is that the first two databases are common in management and organization studies, while the third is linked to the profile of this study, which is interdisciplinary and can offer a technological perspective. Hence, our aim was to find a good sample, rather than finding all articles [41]. The papers are first selected based on title, keywords, and abstracts. The second screening is performed by two external reviewers. The third screening is performed based on the Mixed Methods Appraisal Tool (MMAT), and the two external reviewers independently involved in the appraisal process review 15% of the articles (the selection is performed based on a random sampling). We have selected two reviewers from two different fields (Computer Science and Business Administration). The reason was to make sure that we are catching both approaches, and not missing relevant articles. Having two different reviewers in the review process is useful, so one researcher extract the data and the two others reviewing the extraction [39]. In this

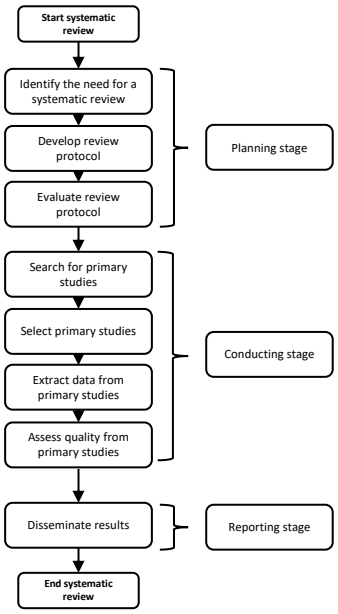


Figure 1. SMS process [39]

way we also reduce the bias, though, given that this step involves human judgment, the threat cannot be eliminated [39]. Based on the screening, a full text reading is conducted to ensure that the right articles are selected. The search strategy and the selection criteria are thoroughly described in Sections 3.3.3 and 3.3.4.

An SMS provides information about the effect of some phenomenon "across a wide range of settings and empirical methods" [39] and gives a robust and thorough view of the current status of research in a particular discipline, by collecting and summarizing the empirical work that exist [42]. Worth mentioning in this context is that the purpose of an SMS is not necessarily to be complete or exhaustive (something we can never assure) but rather to be systematic and transparent. Concerning the former it will allow other researchers to reproduce the results (now or in the future), and concerning the latter an SMS provides a clear view of different sources of evidence and how said evidence is weighted.

An SMS is conducted in three phases: planning the review, conducting the review, and reporting the review. Each phase is divided into a step-by-step process, where an evolution from phase-to-phase must occur. Once the last step is achieved one can progress to the next phase (see Figure 1).

3.3.1. Define and evaluate review protocol

We develop a review protocol to specify the methods we use, and to reduce the possibility for bias. The components of the review protocol are the research questions, the search strategy for collecting primary studies, the exclusion and inclusion criteria, assessment of quality, and data extraction strategy. The external reviewers evaluate and validate the review protocol and, as per their suggestions, changes are incorporated to refine the protocol.

3.3.2. Source selection

We select as sources the following main literature databases:
1. Science Direct;
2. Scopus;
3. IEEE Xplore.

We have also carried out an additional pilot search in the proceedings of top-tier software engineering conferences (ICSE, ESEC/FSE and ASE) and ACM digital library to ensure the validity of the search results. We have followed the same search patterns employed for the three already included databases, and we found 104 conference articles and 378 articles at ACM digital library. We reviewed the title and the abstract but did not encounter any additional papers that discussed AIT (the article's focus was more on the technology than the organization). Our mapping will provide a basis for more in-depth follow-up studies on specific subtopics for which additional databases would be more appropriate. This work can serve as a foundation for future research investigating AIT.

3.3.3. Search strategy

We divide the search into two stages: pilot search and primary search. For each search, we perform the following:
1. Keywords: Keywords are identified based on the research questions,
2. Variants: Synonyms and alternate spellings of search keywords are identified,
3. Search keyword connectors: Combinations of **OR** and **NOT** are used to define sub searches.

Following the SMS methodology and the research questions, in order to identify the most relevant keywords, we perform a pilot search where we evaluate various combinations of relevant keywords. Additionally, we check which word combinations provide the greatest number of hits.

Based on this pilot, we identify the following keyword search terms:
transformation* **OR** organizational change* **OR** learning organization* **OR** change management* **OR** organization restructuring* **OR** organization redesign* **OR** organization design* **OR** technology adoption* **AND** Artificial Intelligence* **OR** AI* **OR** Machine learning* **OR** ML* **OR** Data mining* **OR** Data analytics* **OR** Decision support system* **OR** Expert system* **OR** Knowledge-based system* **OR** Intelligence system* **OR**/**AND** Human machine*

In addition, we consult with key stakeholders within the field of business administration, economics, and AI, to review the keywords to make sure that we remain within the scope of AIT.

3.3.4. Inclusion/exclusion criteria

To select the most relevant studies and exclude irrelevant studies, we establish inclusion and exclusion criteria (see Table 1). We limit the study to existing management and organization studies (MOS) during the period January 2010 to September 2020, since there is a significant growth of publications on these issues within this time frame. We include only studies that relate to AIT. For publications that are within the frame of our inclusion criteria, the following filters are applied as exclusion criteria:
– **Filter 1:** remove publication types other than journal articles;

– **Filter 2:** remove non-English language studies;
– **Filter 3:** remove duplicate studies.

For quality purposes, we limit the selection criteria to journal articles that are published in the English language [43]. The reasoning behind this filtering is that, in most mature areas journals are identified as a more influential and reliable source than other publication channels.

Table 1. Exclusion/Inclusion Criteria

Inclusion criteria
Studies involving AIT
Studies published between 2010–2020
AI and organizational change
AI and organizational restructuring
Business, Management and Accounting
Decision Sciences, Psychology
Exclusion Criteria
Publication types other than journal articles
Duplicate studies
Non-English language studies

After the search, six stages of selection are used to reduce the initial 571 papers (Scopus), 252 papers (IEEE, only two duplicates), and 143 papers (Science Direct, 24 duplicates). The search and selection processes are described below and summarized in Figure 2.

1. Screening the articles based on the title and abstract, and articles that we can identify from the title and the abstracts that are relevant to the SMS, are categorized as *include*, while articles that are irrelevant are categorized as *exclude* (see Table 2 – initial include).
2. A second screening of the included articles is conducted by two external reviewers who evaluate and validate the screening incorporated changes to refine *include* articles (see Table 2 – final include).
3. Krippendorff's α (K_α) (inter-rater reliability statistic) is used to estimate the reliability of the evaluation [45]. The Krippendorff's α results for each database are presented in Table 2. A $K_\alpha > 0.8$ implies a strong inter-rater reliability, i.e., the reviewers were in strong agreement.
4. A third screening is conducted based on the MMAT, which is a tool used to appraise the quality of empirical studies and designed to support systematic reviews that have various methods, i.e., qualitative, quantitative, and mixed methods [46]. The screening questions are used as an indication of the level of quality of the empirical investigation. The screening questions are: (1) Are there clear research questions? (2) Do the collected data allow to address the research questions? Responding 'No' or 'Can't tell' = 0 (the paper is not an empirical study), 'Yes' = 1 [46] (see Tables 3–5). Two external reviewers are independently involved in the appraisal process. We randomly select: Scopus: 14 articles (15% of 95 articles), IEEE: 2 articles (14% of 14 articles), Science direct: 3 articles (18% of 17 articles).
5. The remaining 47 papers are used as the basis for the full-text review. The basic structure of the search and selection process can be seen in Figure 1.
6. The last assessment is based on a full-text reading and leads to the further exclusion of 7 studies.

Table 2. Overview of Inclusion/Exclusion

Source	Number of articles	Initial include	Final include	K_α^1
Scopus	571	129	95	0.91
IEEE	252	16	14	0.85
Science Direct	143	19	17	0.93

[1]Krippendorff's Alpha (K_α) test score

Table 3. MMAT (Scopus)

Scopus	MMAT	Screening question	
		0	1
Qualitative	73	47	26
Quantitative	18	6	12
Mixed methods	4	2	2
Total	95	55	40

Table 4. MMAT (IEEE)

IEEE	MMAT	Screening question	
		0	1
Qualitative	11	9	2
Quantitative	3	1	2
Total	14	10	4

Table 5. MMAT (Science Direct)

Science Direct	MMAT	Screening question	
		0	1
Qualitative	15	11	4
Quantitative	2	1	1
Total	17	12	5[1]

[1]Two articles have been excluded since it was out of the frame of management and organization

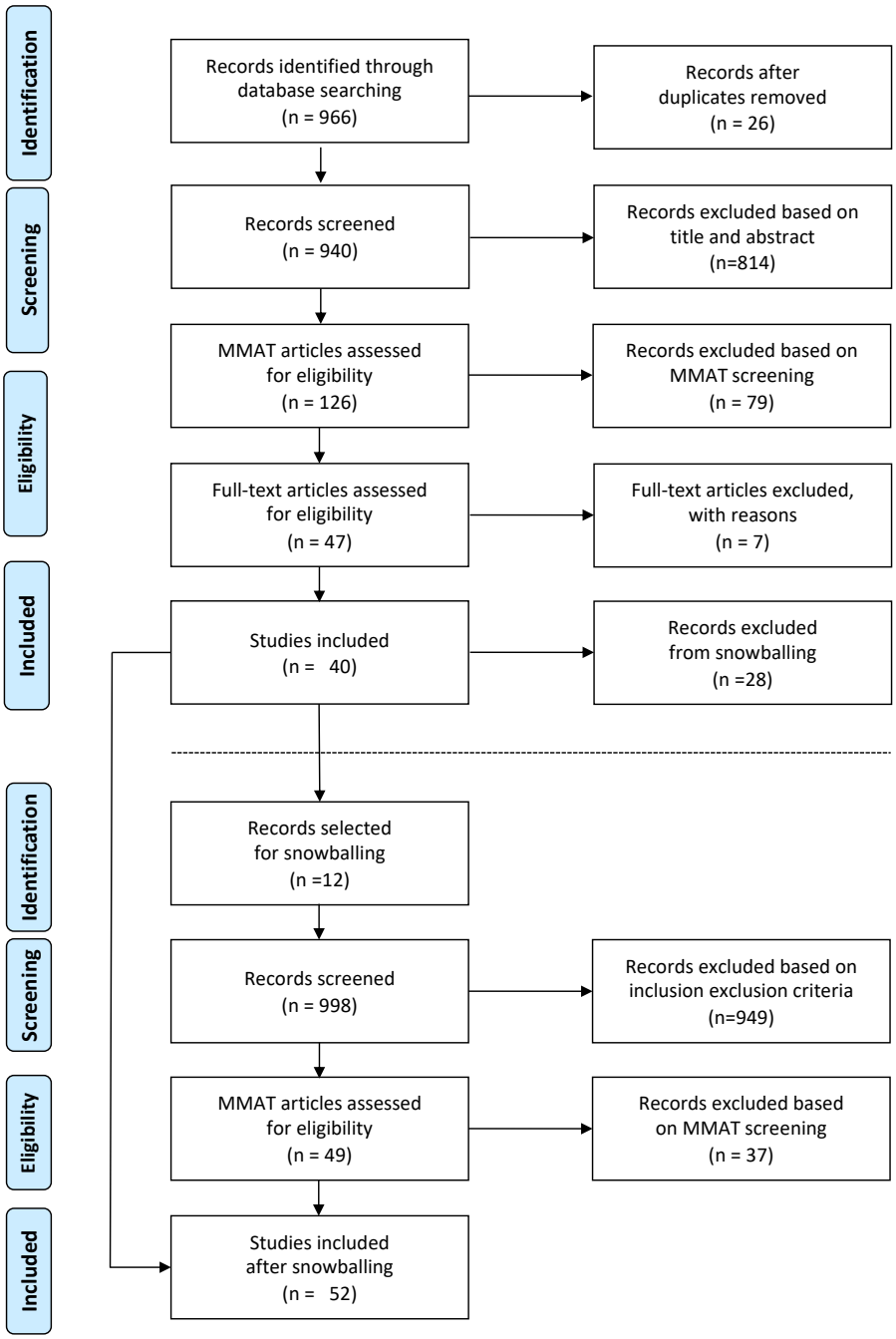


Figure 2. The phases of the SMS through PRISMA [44]

The remaining 40 papers are classified as primary studies and incorporated in the analysis for this study. The basic structure of the search and selection process can be seen in Figure 2.

As a final step, to control for bias, we conduct snowball sampling on the primary studies according to suggested guidelines for secondary search procedures [47]. We identify 12 studies, from the selection of primary studies, to increase the numbers of articles which discuss topics related to AIT. The motivation for selecting the 12 studies as the starting

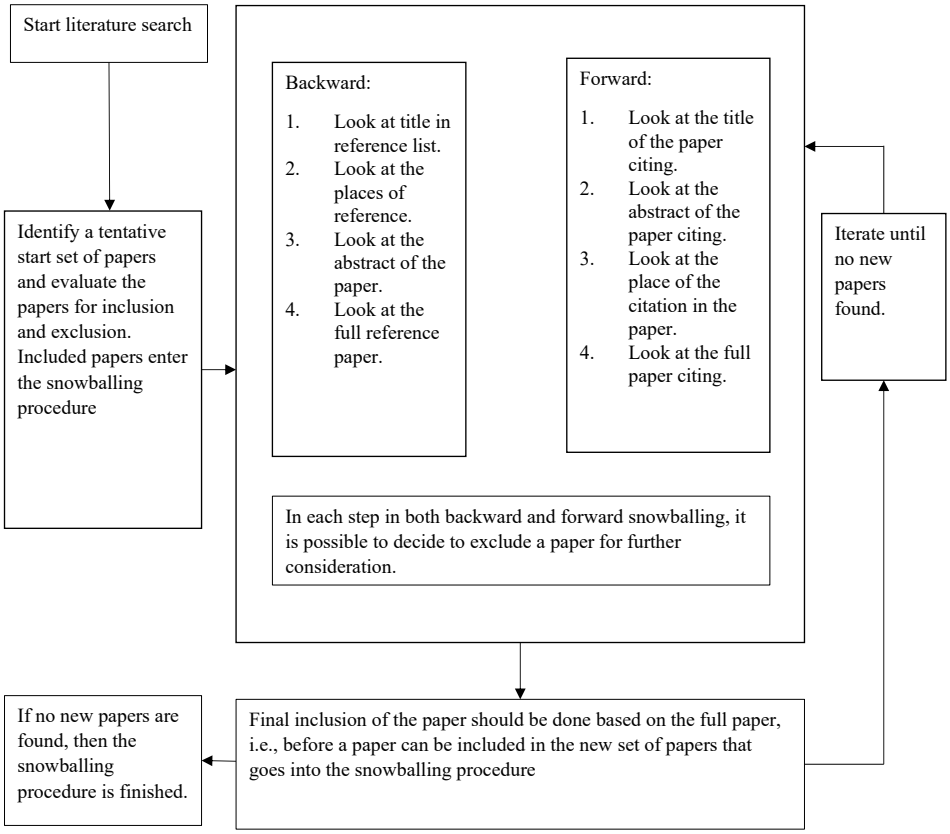


Figure 3. Snowballing procedure [47]

set is based on: the variety in disciplines and publishers, the number of Google Scholar[1] citations in relation to all 40 articles (we have decided to include), and the rank of the journal. We also include articles that thoroughly discuss topics closely related to AIT.

Snowballing is a complementary tool that increases the likelihood of finding all relevant papers on a subject [47]. We perform one-step backward snowballing, which means that we review the reference list of each selected article and follow the same inclusion and exclusion criteria as mentioned in Section 3.3.4.

The studies we review are published between January 2010 to September 2020 and for this reason we cannot perform forward snowballing. However, to complement the snowball sampling, we contact the authors of the primary studies to potentially identify some additional papers. We evaluate the papers retrieved as a consequence of this contact and determine, using our inclusion and exclusion criteria, that none of the papers are to be included as primary studies. Figure 3 describes the snowball procedure we follow, and Figure 2 shows the phases of the snowballing through PRISMA.

We screen the reference list of the 12 studies (in total 998 references) and based on our exclusion and inclusion criteria, we exclude 949 articles. The remaining 49 articles are scrutinized according to MMAT, and 37 articles are excluded. In total 12 papers are included in the full-text review. Hence, after the snowballing procedure, a total of 52 papers are selected as primary studies.

[1]Google Scholar, http://scholar.google.com

3.4. Validity threats

There can be different threats to the validity of study results. There exist additional databases which we are not including in this study. In addition, there are likely other keywords, or combinations of keywords, that would result in different sets of found, included, and excluded papers. We use a particular research design, but there are other ways to perform SMS. One validity threat is human judgment in data extraction and analysis. Additionally, the focus of this SMS is on articles published within the interval 2010–2020. Since AIT is gaining traction in the research community, later SMS will most likely include a significantly higher number of well-performed empirical studies.

To overcome the SMS limitations and to validate its results, several actions were taken. By following the suggested SMS guidelines [39] and by performing our analyses in the prescribed way, we reduce the risk of biased assumptions and conclusions. The analysis of our SMS threats to validity, considering construct validity, reliability, internal and external validity.

Construct validity refers to establishing the correct operational measures for the concepts under study. It describes how closely the phenomenon under study represents what the researchers had in mind and what is investigated according to the research questions [48]. The main constructs in our study are the concepts of "AI Transformation" and "systematic mapping study". Regarding the first, we identified some field roots and discussed related work. We could have used keywords for specific AI-related technologies (NLP, ML, machine vision, neural networks, deep learning, etc.), but our focus was on the broader concept of AI and the related transformation of the organization to support or adopt AI technologies. It is important to perform follow-up studies that focus exclusively on specific areas of AI (such as Deep Learning, Neural networks, and Natural Language Processing) but in the present study, we have chosen to focus on empirical work that considers the general toolbox of AI, without specifying particular areas.

As for the second construct, we followed the guidelines [39] to design our research questions, search criteria, and review protocol. We also did a pilot study and documented all steps to address possible threats to construct validity. We used additional databases (such as ACM) and different keywords (such as deep learning) in our pilot study, and based on the results, we decided which databases and keywords to use. We used keywords that we argue are sufficiently stable to be used as search strings. A broad search of general publication databases, which index the majority of well-regarded publications, was conducted so that all papers on the selected topic could be found. Moreover, we have also carried out an additional search in the proceedings of top-tier software engineering conferences (ICSE, ESEC/FSE and ASE) to ensure the validity of the search results. Hence, our work could be complemented with a systematic literature review that covers a larger number of databases and keywords in order to give a broader overview of this topic.

Reliability focuses on whether the data are collected, and the analysis is conducted in a way that it can be repeated by other researchers with the same results. All steps and processes have been well documented, so replications of our study should yield similar results. The selection of databases was based on providing coverage for management and organizational studies (the first two databases), while the last one was linked to the profile of this study, which is interdisciplinary and can offer a technological perspective. Hence, we have relatively good coverage of the topic of AIT. We established a rigorous search strategy (see Section 3.3.3) and addressed relevant questions related to AIT. The search strategy was tested and reviewed by two external reviewers, and Krippendorff's α statistic

was calculated to ensure that a high inter-rater agreement had been reached. The MMAT tool was used to evaluate the quality of empirical studies and was designed to support the systematic review. The design of this SMS followed a rigorous structure to ensure reproducibility and control for bias.

Internal validity concerns the analysis of the data [48]. Selecting primary studies and assessing them individually pose the greatest threats. Our major source of data was a journal on AIT. In order to increase the reliability of our conclusions, we extended our literature review to several rounds in order to integrate the most complete primary studies possible. We recognize that a much broader search string could have been beneficial. Furthermore, we could include data from a wide range of sources, include keywords strongly associated with AI technology, and include all types of articles. Based on our pilot study, we defined the scope of our study, which was not to obtain an exhaustive sample but rather a representative sample. Since the topic we are interested in is multi-disciplinary, we opted for breadth (disparate databases in terms of venues covered) instead of depth (e.g., exclusive focus on classical computer science or software engineering venues). The second threat stems from the bias of individual researchers in assessing the primary studies they have been assigned. In the analysis, we use various methods to increase the trustworthiness of the results. By following this structure (i.e., by following a predetermined protocol and determining the differences collaboratively), we decrease the risk of assumptions biases.

External validity refers to the domain in which a study's findings are generalizable [48]. The scope of our systematic mapping study was on AIT within the interval 2010–2020. There may be limitations in generalizing our findings to broader time periods, or broader choices of primary research, for example, books and white papers. The results of our current study were drawn from qualitative analysis. To enable analytical and statistical generalizations, quantitative analysis can be considered to complement our findings.

4. Results

This section presents the results for each of the research questions as stated in Section 3.2. The grounds for the results are the papers found in the SMS. The number of papers that have been kept in each step described in Figure 2. It can be seen, that in the end 52 papers have been kept to fulfil the aim and the scope of this SMS and to answer the research questions. A complete list of papers included in the SMS can be found in Appendix A and on the online link [28].

4.1. Evaluation of methodological quality

The primary use of the MMAT tool in this SMS is to support the identification of empirical studies based on the screening questions. The 52 papers included in this review have been re-evaluated based on the MMAT quality criteria for these empirical studies. MMAT categorizes papers into: qualitative studies, quantitative randomized controlled trails, quantitative non-randomized studies, quantitative descriptive studies, and mixed-methods studies). We perform this categorization of the 52 included papers and assess their quality based on the MMAT methodological quality criteria.

We rate the papers into two groups: low methodological quality studies and high methodological quality studies. Studies which score 0 in one or more of the MMAT methodological quality criteria questions, are categorized as studies with a low methodological quality.

Studies which score 1 in all of the MMAT methodological quality criteria, are categorized as studies with a high methodological quality.

The MMAT-based quality assessment reveals that, for a subgroup of the qualitative studies (11 out of 33), the methodological quality is considered low. These studies are lacking adequate explanation of how the findings are derived from the data and an evaluation of whether the results are sufficiently substantiated by data. This implies that the credibility of the reported findings can be put into question. However, 22 out of the 33 qualitative studies are considered as studies with high methodological quality. In the quantitative group of papers, 7 studies out of 18 are considered to be of low methodological quality. These studies are lacking discussion about the risk of non-response bias, which can indicate that there are potential validity and reliability issues. The mixed-methods study is considered to be of high methodological quality. We conclude that a majority of the studies (63%) are of high methodological quality.

4.2. AI transformation conceptualization (RQ1)

Research question 1 (RQ1) concerns in which ways AIT is conceptualized in the literature. The motivation behind RQ1 is to find existing definitions of AIT in the literature, and to analyze these definitions to identify contradictions, similarities, or issues. This analysis can potentially be used to establish a common and useful definition for AIT.

Method description and motivation

This research question is explored using content analysis, which helps to reduce and organize large data to concrete concepts that describe a particular phenomenon [49]. It can be employed using both quantitative and qualitative approaches. It can be used inductively or deductively. Quantitative content analysis relies on the measurement instrument and its reliability, while qualitative content analysis relies on the knowledge and experience of the scholar [50].

Quantitative content analysis is defined as "the systematic, objective, quantitative analysis of message characteristics" [51], in this view content analysis is a quantitative method that includes human coding and computer text analysis. In addition, the quantitative content analysis approach does not rely on the researcher. Moreover, the empirical results can be reproduced if sufficient care has been taken during the design, execution, and reporting of the research. On the other hand, qualitative content analysis follows a similar coding process of a phenomenon, but mainly relies on the researcher's comprehension of the text/context.

In this study, we apply both methods: first we perform an inductive content analysis (ICA) to improve our understanding of the existing definitions. Inductive content analysis is used when there is insufficient or fragmented knowledge about a particular phenomenon [52]. It is used as a tool to identify repetition or commonality of use of a word, phrase, or text which appears in a document. The concept of content analysis is to identify commonalities in the text, gather it into groups, and evolve understanding of it [53].

The process of ICA comprises three steps: preparation, organization, and reporting of results. In the preparation step, the focus is on collecting the data. In this study, the collection of data for the analysis is performed based on the guidelines by Kitchenham [39]. In total, 52 primary studies were included in the analysis. This process and the systematic procedure of the literature review strengthens the trustworthiness of the data collection [49].

We argue that the methods for selecting the data for the SMS ensure an acceptable level of trustworthiness for answering the research questions of our interest. In the organization step, we review the conceptualization of AIT in the literature. This is a crucial step in understanding the work that has been done within the field [54], and will help us to find common understanding, definitions, and keywords used.

Results and analysis

When reviewing the articles in the final selection, we find that only 21% ($n = 11$) include a clear definition of transformation related to technology. The remaining 79% ($n = 41$) articles discuss AI transformation without providing a definition.

We follow the abstraction process [52] and identify five general themes. The purpose of these themes is that they help us gain a better understanding of the different perspectives discussed related to AIT, which is the main topic of our investigation.

The first theme is focused on *transformation*, where emphasis is put on the process of change, and transition from the current state to a new state. This type of transition seems to usually happen in the form of evolution or revolution. The second theme, *fourth industrial revolution*, includes common phrases related to digital technologies that provide intelligent and innovative solutions, such as smart city, smart manufacturing, and smart agriculture. The third theme, *the organization and its environment*, consists of the forces that influence the organization's current status, such as adoption, adaptation, and integration of smart technologies. The fourth theme, *enterprise architecture* is focused on the way the organization strategizes and organizes, as well as its capabilities and structure. The last theme, *idea transformation*, concerns how organizations transform through ideation as a form of innovation. It can be radical, incremental, or a consequence of the ambidexterity of the organization.

The overview of AIT literature by means of categories indicates that prior studies lack an integrated approach to AIT and the associated challenges due to this transformation. The literature uses digital transformation as a common denominator for any kind of technological transformation. In all reviewed articles that discuss ideas related to AIT, the authors use digital transformation as a concept. However, digital transformation *per se* does not always involve AI. Hence, AI, in our view, is focused on smart technologies, intelligent machines which can work, act and have human-like abilities [55]. We follow the overarching definition of Russell and Norvig [24], which discusses the possibility of machine to perform as humans in terms of thought processes, reasoning, and behavior, i.e., intelligent systems that can think humanly, act humanly, and learn as humans.

We are unable to find any definition for AIT in the literature. One likely reason for this is the lack of a universal definition of AI. For example, depending on the context, AI is sometimes described as including areas such as machine learning, big data analytics, and even Internet of things. In other contexts, machine learning, natural language processing, and computer vision are described as sub areas of AI. Some definitions of AI assume the narrow, data-driven applied AI that is pervasive in many sectors today. Other definitions assume the general, human-like AI. There are definitions of AI that benchmark the level of intelligence by comparing with human performance. Other definitions assume objective measures of intelligence. These multiple views of intelligence and of AI are captured well in what could arguably be considered as the standard textbook on AI [24].

It is important to define and clarify the meaning of AI before defining AIT. Once a suitable definition of AI is adopted, it can serve as a starting point to define and describe AIT. We propose a definition of AIT in Section 5.

Evaluation of validity

We performed an additional quantitative content analysis of the abstracts and titles of the articles included in the study. We counted the frequencies of words (excluding punctuation and stop words) to explored the patterns and clusters of terms used. This quantitative content analysis is fully reproducible in that a researcher can perform the same analysis on the same abstract and title corpus and achieve identical results[2].

The top-20 most frequent words in the abstracts and titles of the articles included in this study are listed in Table 6 [28]. It is clear that the most frequent words correspond well together with the five manually identified themes. In Table 7, we report on an analysis of bigrams (consecutive written words) in the abstracts of the papers included in this study. When reviewing the list of most frequent bigrams, we identify a clear mapping to the five identified themes and, in addition, some key phrases related to academic research.

Table 6. The top-20 most frequent words in the abstracts and titles of the articles included in this study. For a full list of words, including common English language construct words (refer to the linked data sheet for more detailed information [28])

Word	Frequency	Word	Frequency
data	84	analytics	30
study	77	transformation	29
business	61	value	28
research	60	adoption	27
digital	52	paper	26
big	43	case	25
smart	40	technologies	24
technology	40	process	23
organizational	33	model	22
new	32	impact	21

Table 7. The top-20 most frequent bigrams (consecutive written words) in the abstracts of the articles included in this study

No.	Bigram	Frequency	No.	Bigram	Frequency
1	big data	45	11	originality value	11
2	artificial intelligence	20	12	publishing limited	11
3	data analytics	19	13	change management	10
4	digital transformation	19	14	data driven	10
5	dynamic capabilities	13	15	digital technologies	10
6	case study	12	16	firm performance	10
7	decision making	11	17	case studies	9
8	design methodology	11	18	industry 4.0	9
9	emerald publishing	11	19	smart manufacturing	8
10	methodology approach	11	20	business value	7

[2]The R scripts used to perform the content analysis are provided in the linked data sheet [28].

AI receives significant attention and the discussions on AI and its consequences are becoming more and more frequent. The question is what is actually known about such consequences. We argue that there is a need for a useful definition of AIT. The reason for this is that, unlike other forms of digital transformation, AI shifts cognitive work from human actors to computers. The consequences for many organizations are therefore likely be more significant. We also suggest more focused research related to specific AI technologies and their respective impact on organizations.

4.3. The main research methods used in AI transformation research (RQ2)

Research question 2 (RQ2) concerns which research methods are used in research related to AIT. The motivation behind RQ2 is that we want to acquire an understanding of *which* research methods are commonly used, as well as gaining more knowledge concerning *how* the methods are used and reported in published work. This allows us to assess the maturity of the research, and to characterize the existing body of knowledge generated in the field.

We review the 52 primary studies included in this study. The analysis reveals that there is a multitude of research designs employed in AIT research. The majority of research tends to be qualitative ($n = 33$) in nature. Also, 18 articles employ a quantitative approach, and one article uses a mixed-methods approach [28].

The quantitative studies are primarily based on surveys or questionnaires. Common analysis approaches include structural equation modeling and partial least squares, descriptive statistic, correlation analysis, and basic regression analysis.

The qualitative studies primarily use case studies and interviews as the method of data collection ($n = 21$). In some cases, secondary data are used for additional data collection. This document analysis involves, for example: white papers, archive documents, and other forms of documentation. Analysis is mainly performed through content analysis using various coding techniques. The use of data triangulation increases the credibility of the results. In these studies, the authors overcome a common bias that would occur when only one research method is used. However, it does not imply that the results can be generalized. In 33% of the qualitative studies ($n = 11$), the primary analysis method is not presented. For the purpose of scientific clarity and reproducibility, the full disclosure and motivation of the primary data analysis approach is paramount [56]. The lack of such descriptions and motivations significantly reduces the credibility of the findings, and the conclusions that have been drawn.

We identify one article which uses a mixed-methods approach to gather empirical data from a real-world setting (Stantec in Edmonton, Canada) [57]. In this article, a case study is performed. In the case study, interviews are combined with regular check-ins, document analysis with data mining, social network analysis, surveys, and a snowball sampling strategy. The use of mixed-methods to answer a specific research question provides both breadth and depth evidence [46]

In many cases, it is difficult to classify published empirical research articles in a simple, unambiguous way, according to which data collection and analysis method are used. One reason is that many published research articles do not provide clear descriptions of how the data collection and analysis are performed. Another reason is that some research articles use multiple methods for data collection and analysis. We identify which of the selected articles do not describe their analysis approaches ($n = 11$). We then study the remaining articles ($n = 41$) to extract any listed data collection or analysis method. We argue that

our categorization is sufficiently correct to allow us to summarize the nature and maturity of the selected articles.

4.4. The theoretical perspectives and frameworks in the field (RQ2.1)

Research question 2.1 (RQ2.1) concerns which theories and frameworks are adopted in AIT research. The motivation behind RQ2.1 is that we identify AIT as inherently interdisciplinary. Due to this, theories and frameworks may come from multiple disciplines, which could make it difficult for a specific discipline to make sense of results and conclusions. An understanding of the underlying theories and frameworks of AIT enables the establishment of a unified framework, in which results, and conclusions could be reinterpreted by any discipline, and by stakeholders from the private and public sector.

The linked data sheet [28] describes the main theories and frameworks adopted in AIT research (see Figure 4 for a stacked bar graph of the 52 included papers). Out of 52 articles, 14 (26%) of the studies clearly mentioned the use of a theory, model, or framework. These 14 studies are found to use 19 different theories that can be grouped into three major categories. The first category uses theories/frameworks within the domain of business and economics: socio-technical systems, the contingency theory, network theory, the theory of the growth of the firm, the resource-based view, the organizational evolutionary theory, and the dynamic capabilities view theory.

The second category uses theories/frameworks within the domain of psychology: the stimulus-organism-response, the psychological reactance theory, decision making and mental models, and the information processing theory. Additionally, one can find theory that is

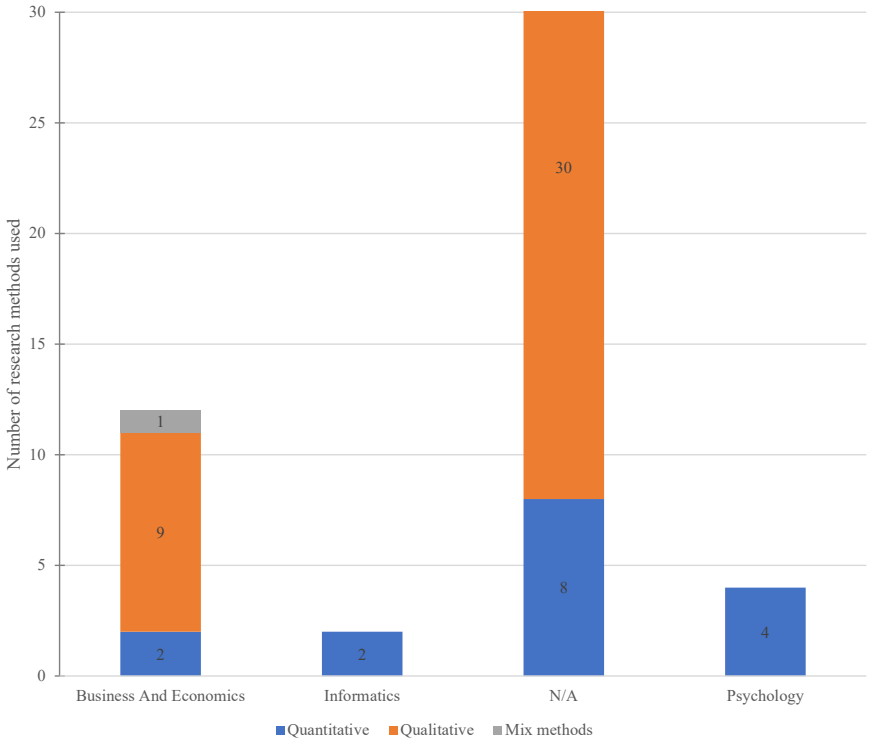


Figure 4. Stacked bar graph of the types of research methods used versus the theories used with respect to the 52 included papers

used within the domain of computer science and information technology, i.e., technology--organization-environment. In this category, we also decide to include the diffusion of innovations theory even though it can be related to various domains (e.g., business and economics, psychology, and so on).

We observe that 32 of the qualitative studies lack theoretical grounding, while only five of the quantitative and the mixed-methods studies do not discuss theoretical grounds (11 quantitative and 3 studies use theory a a foundation). Since qualitative studies tend to be more descriptive and generally not aim for statistical generalizability, the use of theory helps to clarify the logic behind the selected methods. Also, it allows the researcher to reveal existing biases about a study and support the researcher with the primary analysis and interpretation [58]. In quantitative studies, the theory is the foundation for testing and answering the research question, and the research design is built on identifying the theoretical framework that will support the research structure [59].

Based on the results of this SMS, we emphasize the need for more theory research focused on the impact of AIT on organization.

4.5. The real-world scenarios and contexts in AI transformation research (RQ2.2)

Research question 2.2 (RQ2.2) concerns which real-world scenarios and contexts are studied in AIT research. The motivation behind RQ2.2 is that we want to identify the extent or maturity of AIT in different domains, and to explore potentially unique characteristics related to AIT in these domains.

The analysis reveals that AIT research is conducted related to a number of industrial or societal domains. See Figure 5 for a horizontal bar chart of the 52 included papers. We categorize the domains into general segments and describe sectors. The categorization leads to four sectors: the industrial sector, the service sector, the knowledge sector, and the extraction sector. In the industrial sector, manufacturing is the most common industry discussed in the literature. In the service sector, the finance industry (banking, finance, accounting and auditing, and insurance) is the most frequently studied, followed by healthcare. The last two sectors are less represented. In the knowledge sector, high-tech and information technologies are the main industries discussed in the literature. In the

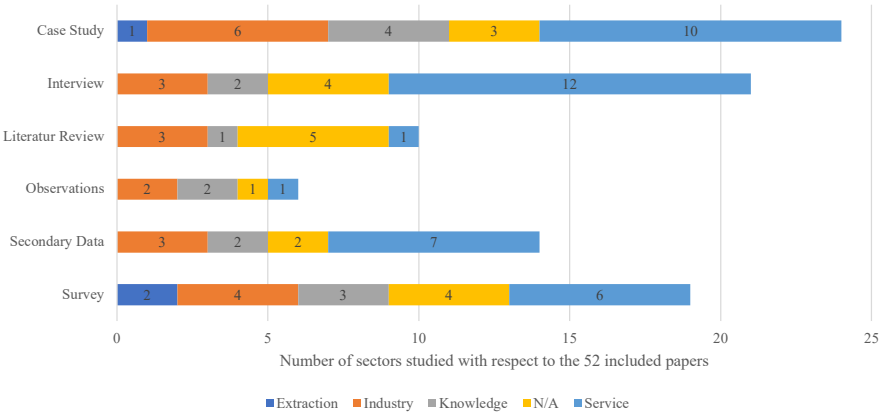


Figure 5. Horizontal bar chart of the types of data collection methods used versus the sector studied with respect to the 52 included papers

extraction sector the studied domains include agriculture, oil, and gas (refer to the linked data sheet for more detailed information [28]).

To identify the maturity of AIT research in different domains, we also review the distribution of papers in terms of publication venue. The 52 primary studies included in the review are published in 44 different journals that belong to 15 different focus areas. Of these 44 journals, 33 are ABS[3]-listed journals, and three are CORE[4]-listed journals. In addition, we present the three-citation indices based on Web of Science, which covers the articles in this study in Appendix B the linked data sheet [28]. In total, 21 articles are included in SSCI[5], 6 articles are listed SCIE[6], six articles are included in both, 7 articles are included in ESCI[7], and 4 articles are not listed in any index.

It is clear that AI is increasingly influential as technology area, and the results of this SMS shows the attention AIT has within various domains. We observe for the results that the most discussed sectors are the industrial sector (manufacturing), and the service sector, while big data analytics is the most researched AI technology when discussing AIT.

4.6. Future research (RQ3)

Research question 3 (RQ3) concerns the emerging questions for future research and the important research gaps in the area. It is important to identify the major trends of AIT research and to identify research gaps, as they seed new research opportunities. In addition, an ever-increasing number of organizations are looking into how to transform due to AI. The identified research gaps may allow new research that helps these organizations reap the benefits and mitigate the risks involved in AIT. The review of the 52 articles included in this study identifies potential opportunities for future research and outline future research directions related to AIT. This can be beneficial both to academics and professionals.We summarize the "future research" section from primary study, and we discuss the gaps appearing when mapping studies. We identify at least six avenues for future research.

Research methods

From a theoretical point of view, there is still a lot of potential for research in the field of AIT. The use of multiple measurement methods, or the use of various approaches to investigate AIT is suggested [60]. The use of mixed-methods approaches, increased sample sizes and in different industries would be of significant value [57]. In addition, future research could take alternative approaches, such as field experiments [61]. Moreover, the need to use various primary data collections to validate research findings and uncover

[3]ABS ranking list is a guide to the range and quality of journals in which business and management academics publish their research. Its purpose is to give both emerging and established scholars greater clarity as to which journals to aim for, and where the best work in their field tends to be clustered.

[4]CORE provides assessments of major journals in the computing disciplines (https://www.core.edu.au/home).

[5]SSCI, stands for Social Science Citation Index, which covers over 3,400 journals across 58 social sciences disciplines, as well as selected items from 3,500 of the world's leading scientific and technical journals (https://clarivate.com/webofsciencegroup/solutions/webofscience-ssci/).

[6]SCIE, stand for Science Citation Index Expanded which covers over 9,200 of the world's most impactful journals across 178 scientific disciplines (https://clarivate.com/webofsciencegroup/solutions/webofscience-scie/).

[7]ESCI, stands for Emerging Sources Citation Index which cover all disciplines and range from international and broad scope publications to those that provide deeper regional or specialty area coverage (https://clarivate.com/webofsciencegroup/solutions/webofscience-esci/).

the impact of AI is emphasized [62]. Furthermore, the importance of the use of various databases and sources is stressed [63]. More research is needed to strengthen the validity of smart technology transformation research [64].

Theoretical foundations

Future research should consider potential links to existing theories, which help to explain, predict, and understand AIT. The articles included in this study discuss potential opportunities for theoretical assumptions, which should be reviewed as a basis for investigation of organizational change fueled by smart technologies. Further research can be accomplished by the use of various theories related to the interaction, assessment and comparison of organizations ordinary capabilities vs. dynamic capabilities [65].

Societal aspects

Legal, ethical, societal, and economic changes which are the result of AIT are relevant for future investigation [66]. Legal and ethical considerations in relations to societal anticipation is an important aspect from an organizational perspective and it provides a broader perspective of the consequences concerning AI [11]. When studying AIT, researchers should consider the development of organizational and societal expectations, the outcomes related to opportunities, and the challenges involving AI. These factors and their implications from an organizational perspective, we argue, are highly relevant for future research.

The importance of ethical challenges related to smart technologies, new data sets, algorithms, and various AI solutions and machine learning is stressed [67]. Additional research, along those lines can be taken from different organizational perspectives (operation, strategy, structure, process, human labor, and so on). This may lead to an increase of the level of usage and understanding of the concept of AI. It is argued that an increased understanding of the factors that shape experiences on the transition age, not only of technological changes, but also of any social and economic changes, may lead to a better adaptation of smart technologies. It is further argued that there is a high value in the collaboration between academia and industry, which can help to identify business, technical, and societal challenges in the implementation of smart technologies [68].

The impact of adoption and adaption

The value of exploring the impact of investing in big data analytics to create higher-order capabilities or dynamic capabilities is discussed [62]. The impact of AI capabilities on firm performance should be studied from an organizational perspective, in a way which makes it possible to comprehend the importance AI personal expertise and AI infrastructure. In this way, organizations will be able to improve their business value and to gain a better understanding of AI. It is claimed that organizations, while adopting AI, should consider the impact on the firms [57]. A comparison of various findings and trends related to smart technologies can be beneficial to gain an understanding of the capabilities of smart technologies and its effect on the organization. Further research could explore the advantages and disadvantages of AI and its impact on organizational structures [57]. To extend the concept of technological transformation one should examine the adoption of one specific digital innovation in a particular organizational context, as well as verifying and elaborating

on this particular context, and examine how boundary relations are reconfigured in other contexts and with other digital innovations [69].

The effect on human capital

The discussion on the effect of new technologies on human capital and organizations is not new, but rather a continuous discussion of previous industrial revolutions and changes in the labor market. It is stated that "any worker who now perform his task by following specific instructions can, in principle, be replaced by a machine" [70]. The authors further claim that physical jobs that disappear from the market as a result of the industrial revolution increase the need for the mental capacity of human labor and the importance of training and retraining of the labor to better anticipate future structural changes. The importance of creative imagination, entrepreneurship, and leadership are emphasized and viewed to be irreplaceable by a machine: "without creative imagination, neither art nor science could possibly advance" [70].

Furthermore, it is emphasized that an organization's future, based on new technologies, will cause some jobs to disappear [3]. But from the nature of capitalism (or humans) it will create other jobs which we cannot easily predict [3].

The user perspective plays a vital role in the way AI transforms. Future research that focus on potential moderators to the impacts of users' psychological reactance is suggested [71]. Moreover, it is pointed out that the most important factor in organizational transformation is not the technological but rather the managerial factor, along with employee attitudes [72]. A holistic view for future research is discussed, which should emphasize the need for collaboration between researchers and practitioners to contribute for clarifying the relevance of human resources in the firms' transformation and processes [73].

A focus is suggested on the reciprocal and symbiotic relationship between intelligent technologies and human capital, which will have a complementary role in the future organization [74]. The investment made in organizations to develop new technologies, or implementing new technologies such as AI, leads to investments in human capital in a way that can complement and support the decision-making. However, this type of human capital, that is complementary to AI decision support, is not adequately researched or identified. It is emphasized that future research should emphasize and compare the behavior of employees and managers in the context of delegation of strategic decision to a human being or an algorithm [75].

Complementary contexts

Smart technologies and their effects on the organization are investigated in various contexts. To enable a thorough understanding of AIT further research can be taken in various contextual basis. Published studies could be repeated in developing countries and different industries and sectors, or to compare between organizations of similar size [22]. Similar ideas are suggested that urge to also test conceptual models and theories in various service industries [71]. The research around AIT should extend the target research areas and cover more regions such as specific European and American countries to compare findings in emerging and developed economies and to increase generalizability [76].

5. Discussion

5.1. Understanding and defining AI transformation

In the last sections, we elaborate on our impetus for conducting an SMS on AIT, as a key concept for incremental and radical change that will lead to a transformation in the organization. AI and its technologies (for example: computer vision, machine learning, natural language processing, and robotics) are reshaping organizational structure, processes, organizational learning, work routines, knowledge management, products, and services [37]. AI involves both challenges and immense opportunities, its capability to manage information and knowledge required change in organizations culture, mindset and skills and organization that will understand and act on it will probably get a competitive advantage. AI counter business, and the reciprocity relations, and influence it has on each other is discussed [77]. AI changes organizations, but organizations influence the way AI develops. Understanding this link between the two is highly relevant from a research perspective.

Researchers from various disciplines should collaborate to understand and improve the connection between the technology and the organization. AI and its effects on the organization is unavoidable [23], however, it is important to understand the concept of AI and its implications, while understanding its relationship to the organizational structure, leadership, culture, vision, and mission and the human attributes within the organization.

In this SMS, we aggregate the body of knowledge on the relationship between AI and organizational transformation, map the field, and identify the research gaps that represent opportunities for future studies. Our SMS follows Kitchenham's suggestions on conducting an SMS [39] and identifies 52 articles published in various journals. We present three main research questions and adopt both qualitative and quantitative approaches based on the analysis of the 52 articles to increase the trustworthiness of this study, and to give a thorough understanding of the phenomenon from different perspectives. In addition, the use of both methods was complementary; the strengths of one approach supplemented the weaknesses of another [78].

In general, from the review, we observe that MMAT reveals that very little empirical research is conducted on the topic of the SMS. We find that the topic is discussed in various academic disciplines and uses various methods, and theories, however only a few use established theories. We identify a number of themes as discussed in Section 4: The organization and its environment, enterprise architectures, idea transformation, and the fourth industrial revolution. Four sectors were identified: The industrial sector, the service sector, the knowledge sector, and the extraction sector (agriculture, oil, and gas). However, the most discussed sectors were the industrial sector and the service sector, while big data analytics is the most reviewed AI technology when discussing AIT.

However, we were unable to find a concise and useful definition of AIT. The available research that brings up this phenomenon is often focusing on digital transformation, and there is a substantial scientific discussion around digital transformation but few studies focused only on AI. However, we emphasize the need for a definition of AIT. The reason for this is that, unlike other forms of digital transformation, AI will clearly shift cognitive work from human actors to computers. The consequences for many organizations is significant.

We view AIT as an interdisciplinary phenomenon. In this context, we thus define AIT as:

Definition 1. *the ongoing change in organizational dimensions (strategy, structure, people, technology and processes), subject to constraints and interests of external forces (customers, suppliers, partners, competitors, regulators), and manifested in AI readiness.*

This division into organizational dimensions and external forces is suggested in an existing work on e-business transformation [79]. In this definition, **organizational dimensions** refer to strategy as the way organizations determine their goals, their actions, the implementation, and the resource allocation required for achieving these goals [80]. The *structure* is the way an organization is designed and the way it administrates, which is linked to the effectiveness, the coordination, and the communication of the organization [80]. The organizational *processes* are linked with the *strategy* and *structure*. The processes are essentially sequences of tasks, distributed in time and space. They are required to assign tasks to people and to accomplish these tasks [81]. The external forces are uncontrollable factors that can influence an organization. AI *technology* can refer to either the actual hardware and software systems which are based on AI, or to the knowledge, skills, and processes required to apply AI in the real world. These definitions of AI technology are based on typical definitions of technology (see for example [82]). Most researchers discuss the internal dimensions and the external forces as two separate agents of change. In our view, AIT occurs when one or more of the organizational dimensions or the external forces change due to the use of AI technologies. Transformation, on the one hand, can be of a revolutionary nature, where the organization changes radically and quickly along one or more of the organizational dimensions. On the other hand, transformation can also be of a gradual or incremental nature, where the organization, in a discontinuous way, respond to basic changes in its environment [83]. An organization that has a clear sense of its position along the organizational dimensions is able to align itself properly to external factors.

The AIT Playbook[8] discusses the journey of a successful organization's transformation, and the leveraging of AI capabilities to significantly advance, due to the use of AI technologies. Our definition of AIT is concretely connected to the knowledge and insights about successful AIT provided in the playbook. The AI transformation playbook describes various relevant organizational aspects (for example: resources, AI expertise, up-skilling people, adjustment of processes and strategy).

Our definition categorizes these aspects into organizational dimensions. It also adds the perspective of external forces (interests and constraints originating from outside the organization), and introduces AI readiness to quantify the level of fulfilment of the transformation. We argue that our definition provides the research community with a clear description, which can be criticised, elaborated upon, and used to frame future work. It also provides organizations with a foundation for their AI journey and a basis for evaluation of the progress.

6. Conclusions

In this study, we systematically review the field of AI and organizational transformation, and provide a thorough understanding of the field. By doing so, we identify gaps in research that represent potential opportunities for future study. Despite the popularity and attention related to AI and its effects on organizations, this Systematic mapping study (SMS) shows

[8]AI Transformation Playbook, https://landing.ai/ai-transformation-playbook/

that the number of studies discussing this topic are opinion papers rather than scientific research papers.

The results reveal that there is no existing useful definition of AIT and that in the sample we identify there are few empirical research papers. Existing work on AIT originates from various academic disciplines and domains. This shows that AI is interdisciplinary in its nature and that it has impacts on various domains and industries. AIT researchers are mainly using qualitative methods. We provide a new definition for AIT and attempt to consolidate and relate existing work from the various disciplines and domains. We also observe a clear need for research using mixed methods approaches.

This Systematic mapping study enriches the current state-of-the-art knowledge regarding AIT research. We propose several directions for future research, including: a Systematic mapping study to determine, for each specific AI technology, how it transforms organizations. Another proposed direction for future work is to explore how one particular dimension of the organization (i.e., strategy, structure, people, technology, processes) transform based on the implementation of AI technology. It could be interesting to look into AIT in various contexts, such as: private sector vs. public sector, different industries, different size of organization and the context of various countries (developing countries vs. industrialized countries and so on). The use of mixed-methods research approaches to investigate AIT will give a more broad view about this phenomenon.

This SMS reveal that there is a substantial scientific discussion around digital transformation, but only few works discuss the concept of AIT. In this SMS we develop a definition for AIT. This definition can be used as a foundation for future work involving the impact of AI on organizations.

The selected 52 papers in this SMS should be interesting for industry, academia and public sector since it may contain relevant information for practitioners. We believe that the results of this SMS can be a foundation for improvements of the collaboration between these three actors. The university responsibility should be knowledge production, the industry is responsible for market and economic production and exchange, and the government stands for policy making.

The results introduced in these papers can provide valuable insight for organizations which are adopting AI.

References

[1] J. Holmstrom, "From AI to digital transformation: The AI readiness framework," *Business Horizons*, 2021.

[2] U. Lichtenthaler, "Beyond artificial intelligence: Why companies need to go the extra step," *Journal of Business Strategy*, 2018.

[3] E. Brynjolfsson, *The Second Machine Age: Work, Progress, and Prosperity in a Time of Brilliant Technologies*. Norton and Company, 2014.

[4] J. Maclure and S. Russell, *AI for Humanity: The Global Challenges*. Springer International Publishing, 2021, pp. 116–126.

[5] R.C. Schank, "Where's the AI?" *AI Magazine*, Vol. 12, No. 4, 1991.

[6] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, Vol. 521, No. 7553, 2015, pp. 436–444.

[7] E. Ntoutsi, "Bias in data-driven artificial intelligence systems – An introductory survey," *WIREs Data Mining and Knowledge Discovery*, 2020.

[8] C. Anderson, *Creating a data-driven organization: Practical advice from the trenches*. O'Reilly Media, Inc., 2015.

[9] N. Zolas, Z. Kroff, E. Brynjolfsson, K. McElheran, D. Beede et al., "Advanced technologies adoption and use by u.s. firms: Evidence from the annual business survey," National Bureau of Economic Research, Working Paper 28290, 2020.
[10] M. Cubric, "Drivers, barriers and social considerations for AI adoption in business and management: A tertiary study," *Technology in Society*, Vol. 62, 2020, pp. 101–257.
[11] S. Akter, K. Michael, M. Uddin, G. McCarthy, and M. Rahman, "Transforming business using digital innovations: The application of AI, blockchain, cloud, and data analytics," *Annals of Operations Research*, 2020, pp. 1–33.
[12] F. Khanboubi and A. Boulmakoul, "Digital transformation in the banking sector: Surveys exploration and analytics," *International Journal of Information Systems and Change Management*, Vol. 11, No. 2, 2019, pp. 93–127.
[13] L. Achtenhagen, L. Melin, and L. Naldi, "Dynamics of business models – Strategizing, critical capabilities and activities for sustained value creation," *Long Range Planning*, Vol. 46, No. 6, 2013, pp. 427–442.
[14] S. Makridakis, "The forthcoming artificial intelligence (AI) revolution: Its impact on society and firms," *Futures*, Vol. 90, 2017, pp. 46–60.
[15] A. Schumacher, S. Erol, and W. Sihn, "A maturity model for assessing industry 4.0 readiness and maturity of manufacturing enterprises," *Procedia Cirp*, Vol. 52, 2016, pp. 161–166.
[16] H.A. Simon, *Administrative Behavior: A Study of Decision-making Processes in Administrative Organization*, 3rd ed. Free Press, 1976.
[17] A. Ebbage, "Banking on artificial intelligence," *Engineering and Technology*, Vol. 13, No. 10, 2018, pp. 66–69.
[18] H. David, "Why are there still so many jobs? The history and future of workplace automation," *Journal of Economic Perspectives*, Vol. 29, No. 3, 2015, pp. 3–30.
[19] M. Polanyi, *The Tacit Dimension*. University of Chicago Press, 2009.
[20] E. Sadler-Smith and E. Shefy, "The intuitive executive: Understanding and applying 'gut feel' in decision-making," *Academy of Management Perspectives*, Vol. 18, No. 4, 2004, pp. 76–91.
[21] M. Tegmark, *Life 3.0: Being Human in the Age of Artificial Intelligence*, 1st ed. Alfred A. Knopf, 2017.
[22] P. Maroufkhani, W.K.W. Ismail, and M. Ghobakhloo, "Big data analytics adoption model for small and medium enterprises," *Journal of Science and Technology Policy Management*, Vol. 11, No. 4, 2020, pp. 483–513.
[23] S.L. Wamba-Taguimdje, S.F. Wamba, J.R.K. Kamdjoug, and C.T. Wanko, "Influence of artificial intelligence (AI) on firm performance: The business value of AI-based transformation projects," *Business Process Management Journal*, Vol. 26, No. 7, 2020.
[24] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*. Pearson Education, 2016.
[25] L. Deng, "Artificial intelligence in the rising wave of deep learning: The historical path and future outlook [perspectives]," *IEEE Signal Processing Magazine*, Vol. 35, No. 1, 2018, pp. 180–177.
[26] J. Lee, H. Davari, J. Singh, and V. Pandhare, "Industrial artificial intelligence for industry 4.0-based manufacturing systems," *Manufacturing Letters*, Vol. 18, 2018, pp. 20–23.
[27] E.P.R. Service, "The ethics of artificial intelligence: Issues and initiatives," European Parliament, Tech. Rep., 2020.
[28] E. Peretz-Andersson, "AI transformation: A systematic literature review (linked data sheet)," https://osf.io/3afw6/?view_only=fd36e2c55f044f1abe55e6e9d1d0f852, 2021, [Online; accessed 2021-05-21].
[29] F. Li, "The digital transformation of business models in the creative industries: A holistic framework and emerging trends," *Technovation*, Vol. 92, 2020, p. 102012.
[30] L. Heilig, E. Lalla-Ruiz, and V. Stefan, "Digital transformation in maritime ports: Analysis and a game theoretic framework," *Netnomics: Economic research and electronic networking*, Vol. 18, No. 2, 2017, pp. 227–254.
[31] A. Nadeem, B. Abedin, N. Cerpa, and E. Chew, "Digital transformation and digital business strategy in electronic commerce-the role of organizational capabilities," 2018, pp. 1–8.

[32] D. Anderson and L.A. Anderson, *Beyond Change Management: Advanced Strategies for Today's Transformational Leaders*. John Wiley and Sons, 2002.
[33] H. Tsoukas and R. Chia, "On organizational becoming: Rethinking organizational change," *Organization Science*, Vol. 13, No. 5, 2002, pp. 567–582.
[34] H. Mintzberg and F. Westley, "Cycles of organizational change," *Strategic Management Journal*, Vol. 13, No. S2, 1992, pp. 39–59.
[35] H. Arazmjoo and H. Rahmanseresht, "A multi-dimensional meta-heuristic model for managing organizational change," *Management Decision*, Vol. 58, No. 3, 2019, pp. 526–543.
[36] M.L. Tushman and C.A. O'Reilly III, "Ambidextrous organizations: Managing evolutionary and revolutionary change," *California Management Review*, Vol. 38, No. 4, 1996, pp. 8–29.
[37] P. Weill and S.L. Woerner, "Is your company ready for a digital future?" *MIT Sloan Management Review*, Vol. 59, No. 2, 2018, pp. 21–25.
[38] R. Ramilo and M.R.B. Embi, "Critical analysis of key determinants and barriers to digital innovation adoption among architectural organizations," *Frontiers of Architectural Research*, Vol. 3, No. 4, 2014, pp. 431–451.
[39] B. Kitchenham, "Procedures for performing systematic reviews," Keele University, UK, Tech. Rep., 2004.
[40] R. Mallett, J. Hagen-Zanker, R. Slater, and M. Duvendack, "The benefits and challenges of using systematic reviews in international development research," *Journal of Development Effectiveness*, Vol. 4, No. 3, 2012, pp. 445–455.
[41] K. Petersen, S. Vakkalanka, and L. Kuzniarz, "Guidelines for conducting systematic mapping studies in software engineering: An update," *Information and Software Technology*, Vol. 64, 2015, pp. 1–18.
[42] A. Bajaj and O.P. Sangwan, "A systematic literature review of test case prioritization using genetic algorithms," *IEEE Access*, Vol. 7, No. 126355–126375, 2019.
[43] I.M. Côté, P.S. Curtis, H.R. Rothstein, and G.B. Stewart, "Gathering data: searching literature and selection criteria," *Handbook of meta-analysis in ecology and evolution*, 2013, pp. 37–51.
[44] D. Moher, A. Liberati, J. Tetzlaff, D.G. Altman, and P. Group, "Preferred reporting items for systematic reviews and meta-analyses: The PRISMA statement," *PLoS Medicine*, Vol. 6, No. 7, 2009.
[45] A.F. Hayes and K. Krippendorff, "Answering the call for a standard reliability measure for coding data," *Communication Methods and Measures*, Vol. 1, No. 1, 2007, pp. 77–89.
[46] Q.N. Hong, P. Pluye, S. Fàbregues, G. Bartlett, F. Boardman et al., "Mixed methods appraisal tool (MMAT)," McGill University, Tech. Rep., 2018.
[47] C. Wohlin, "Guidelines for snowballing in systematic literature studies and a replication in software engineering," in *18th International Conference on Evaluation and Assessment in Software Engineering*. ACM Press, 2014, pp. 1–10.
[48] T.D.Cook, D. Campbell, and W.R.Shadish, *Experimental and quasi-experimental designs for generalized causal inference*. Houghton Mifflin Boston, MA, 2002.
[49] S. Elo, M. Kääriäinen, O. Kanste, T. Pölkki, K. Utriainen et al., "Qualitative content analysis: A focus on trustworthiness," *SAGE Open*, 2014, pp. 1–10.
[50] U. Flick, *The SAGE Handbook of Qualitative Data Analysis*. Sage, 2013.
[51] K.A. Neuendorf, *The Content Analysis Guidebook*, 2nd ed. SAGE, 2017.
[52] S. Elo and H. Kyngäs, "The qualitative content analysis process," *Journal of Advanced Nursing*, Vol. 62, No. 1, 2008, pp. 107–115.
[53] M. Bengtsson, "How to plan and perform a qualitative study using content analysis," *NursingPlus Open*, Vol. 2, 2016, pp. 8–14.
[54] J. vom Brocke, A. Simons, B. Niehaves, B. Niehaves, K. Riemer et al., "Reconstructing the giant: On the importance of rigour in documenting the literature search process," in *17th European Conference on Information Systems*, 2009, pp. 2206–2217.
[55] A.C. Serban and M.D. Lytras, "Artificial intelligence for smart renewable energy sector in Europe – Smart energy infrastructures for next generation smart cities," *IEEE Access*, Vol. 8, 2020, pp. 77 364–77 377.

[56] M. Allen, *The SAGE encyclopedia of Communication Research Methods*. Sage Publications, 2017.

[57] M.M. Bonanomi, D.M. Hall, S. Staub-French, A. Tucker, and C.M.L. Talamo, "The impact of digital transformation on formal and informal organizational structures of large architecture and engineering firms," *Engineering, Construction, and Architectural Management*, Vol. 27, No. 4, 2019, pp. 872–892.

[58] C.S. Collins and C.M. Stockton, "The central role of theory in qualitative research," *International Journal of Qualitative Methods*, Vol. 17, 2018, pp. 1–10.

[59] J.W. Creswell and J.D. Creswell, *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches*. Sage, 2018.

[60] M. Jocevski, N. Arvidsson, G. Miragliotta, A. Ghezzi, and R. Mangiaracina, "Transitions towards omni-channel retailing strategies: A business model perspective," *International Journal of Retail and Distribution Management*, Vol. 47, No. 2, 2019, pp. 78–93.

[61] X. Fan, N. Ning, and N. Deng, "The impact of the quality of intelligent experience on smart retail engagement," *Marketing Intelligence and Planning*, Vol. 38, No. 7, 2020, pp. 877–891.

[62] S.F. Wamba and S. Akter, "Understanding supply chain analytics capabilities and agility for data-rich environments," *International Journal of Operations and Production Management*, Vol. 39, No. 6–8, 2019, pp. 887–912.

[63] G. Elia, G. Polimeno, G. Solazzo, and G. Passiante, "A multi-dimension framework for value creation through big data," *Industrial Marketing Management*, Vol. 90, 2020, pp. 508–522.

[64] K. Tiwari and M.S. Khan, "Sustainability accounting and reporting in the industry 4.0," *Journal of Cleaner Production*, Vol. 258, 2020.

[65] Y. Gong and M. Janssen, "Roles and capabilities of enterprise architecture in big data analytics technology adoption and implementation," *Journal of Theoretical and Applied Electronic Commerce Research*, Vol. 16, No. 1, 2021, pp. 37–51.

[66] T. Nam, "Technology usage, expected job sustainability, and perceived job insecurity," *Technological Forecasting and Social Change*, Vol. 138, 2019, pp. 155–165.

[67] P. Dahlbom, N. Siikanen, P. Sajasalo, and M. Jarvenpää, "Big data and HR analytics in the digital era," *Baltic Journal of Management*, Vol. 15, No. 1, 2020.

[68] M. Gotthardt, D. Koivulaakso, O. Paksoy, C. Saramo, M. Martikainen et al., "Current state and challenges in the implementation of smart robotic process automation in accounting and auditing," *ACRN Journal of Finance and Risk Perspectives*, Vol. 9, 2020, pp. 90–102.

[69] M. Barrett, E. Oborn, W.J. Orlikowski, and J. Yates, "Reconfiguring boundary relations: Robotic innovations in pharmacy work," *Organization Science*, Vol. 23, No. 5, 2011, pp. 1448–1466.

[70] W. Leontief, *The Long-Term Impact of Technology on Employment and Unemployment*. The National Academies Press, 1983.

[71] W. Feng, R. Tu, and Z. Zhou, "Understanding forced adoption of self-service technology: The impacts of users' psychological reactance," *Behaviour and Information Technology*, Vol. 38, No. 8, 2019, pp. 820–832.

[72] D.J. Bowersox, D.J. Closs, and R. Drayer, "The digital transformation: Technology and beyond," *Supply Chain Management Review*, Vol. 9, No. 1, 2005, pp. 22–29.

[73] F. Caputo, V. Cillo, E. Candelo, and Y. Liu, "Innovating through digital revolution: The role of soft skills and big data in increasing firm performance," *Management Decision*, Vol. 57, No. 8, 2019, pp. 2032–2051.

[74] K. Conboy, P. Mikalef, D. Dennehy, and J. Krogstie, "Using business analytics to enhance dynamic capabilities in operations research: A case analysis and research agenda," *European Journal of Operational Research*, Vol. 281, No. 3, 2020, pp. 656–672.

[75] S. Schneider and M. Leyer, "Me or information technology? Adoption of artificial intelligence in the delegation of personal strategic decisions," *Managerial and Decision Economics*, Vol. 40, No. 3, 2019, pp. 223–231.

[76] W.E. Hilali, A.E. Manouar, and M.A. Idrissi, "Reaching sustainability during a digital transformation: A PLS approach," *International Journal of Innovation Science*, Vol. 12, No. 1, 2020, pp. 52–79.

[77] C.D. Francescomarino and F.M. Maggi, "Preface to the special issue on business process innovations with artificial intelligence," *Journal on Data Semantics*, Vol. 8, 2019, pp. 77–77.
[78] A. Regnault, T.Willgoss, and S. Barbic, "Towards the use of mixed methods inquiry as best practice in health outcomes research," *Journal of Patient-Reported Outcomes*, Vol. 2, No. 19, 2018.
[79] A. Farhoomand and R. Wigand, "Editorial: Special section on managing e-business transformation," *European Journal of Information Systems*, Vol. 12, 2003, pp. 249–250.
[80] A.D. Chandler, *Strategy and Structure: Chapters in the History of the Industrial Enterprise*, 3rd ed. MIT Press, 2013.
[81] B.T. Pentland, C.S. Osborn, G. Wyner, and F. Luconi, *Useful Descriptions of Organizational Processes: Collecting Data for the Process Handbook*. Center for Coordination Science, Massachusetts Institute of Technology, USA, 1999.
[82] R. Bain, "Technology and state government," *American Sociological Review*, Vol. 2, No. 6, 1937, pp. 860–874.
[83] E.Romanelli and M. Tushman, "Organizational transformation as punctuated equilibrium: An empirical test," *Academy of Management Journal*, Vol. 37, No. 5, 1994, pp. 1141–1166.
[84] Y. Chen and Z. Lin, "Business intelligence capabilities and firm performance: A study in China," *International Journal of Information Management*, Vol. 57, 2021, p. 102232.
[85] M. Aboelmaged and S. Mouakket, "Influencing models and determinants in big data analytics research: A bibliometric analysis," *Information Processing and Management*, Vol. 57, No. 4, 2020, p. 102234.
[86] R. Balakrishnan and S. Das, "How do firms reorganize to implement digital transformation?" *Strategic Change*, Vol. 29, No. 5, 2020, pp. 531–541.
[87] F. Brunetti, D.T. Matt, A. Bonfanti, A.D. Longhi, G. Pedrini et al., "Digital transformation challenges: Strategies emerging from a multi-stakeholder approach," *The TQM Journal*, Vol. 32, No. 4, 2020, pp. 697–724.
[88] C. Dremel, M.M. Herterich, J. Wulf, and J. vom Brocke, "Actualizing big data analytics affordances: A revelatory case study," *Information and Management*, Vol. 57, No. 1, 2020, p. 103121.
[89] J. Lee and D. Kim, "Development of innovative business of telecommunication operator: Case of KT-MEG," *International Journal of Asian Business and Information Management (IJABIM)*, Vol. 11, No. 3, 2020, pp. 1–14.
[90] P. Mikalef, J. Krogstie, I. Pappas, and P. Pavlou, "Exploring the relationship between big data analytics capability and competitive performance: The mediating roles of dynamic and operational capabilities," *Information and Management*, Vol. 57, No. 2, 2020, p. 103169.
[91] K. Moore, "Smart connected sensors, cyber-physical networks, and big data analytics systems in internet of things-based real-time production logistics," *Economics, Management, and Financial Markets*, Vol. 15, No. 2, 2020, pp. 16–22.
[92] N. Nguyen, R. Gosine, and P. Warrian, "A systematic review of big data analytics for oil and gas industry 4.0," *IEEE Access*, Vol. 8, 2020, pp. 61 183–61 201.
[93] P. Osterrieder, L. Budde, and T. Friedli, "The smart factory as a key construct of industry 4.0: A systematic literature review," *International Journal of Production Economics*, Vol. 221, 2020, p. 107476.
[94] R. Silva, C. Bernardo, C. Watanabe, R. Silva, and J. Neto, "Contributions of the internet of things in education as support tool in the educational management decision-making process," *International Journal of Innovation and Learning*, Vol. 27, No. 2, 2020, pp. 175–196.
[95] M. Sott, L. Furstenau, L. Kipper, F. Giraldo, J. López-Robles et al., "Precision techniques and agriculture 4.0 technologies to promote sustainability in the coffee sector: State of the art, challenges and future trends," *IEEE Access*, Vol. 8, 2020, pp. 149 854–149 867.
[96] A. Tuomi, I. Tussyadiah, E. Ling, G. Miller, and L. Geunhee, "x=(tourism_work) y=(sdg8) while y= true: automate (x)," *Annals of Tourism Research*, Vol. 84, 2020, p. 102978.
[97] Z. Zhang and T. Luo, "Knowledge structure, network structure, exploitative and exploratory innovations," *Technology Analysis and Strategic Management*, Vol. 32, No. 6, 2020, pp. 666–682.

[98] J. Brock and F.V. Wangenheim, "Demystifying AI: What digital transformation leaders can teach you about realistic artificial intelligence," *California Management Review*, Vol. 61, No. 4, 2019, pp. 110–134.

[99] D. Kalaivani and P. Sumathi, "Factor based prediction model for customer behavior analysis," *International Journal of System Assurance Engineering and Management*, Vol. 10, No. 4, 2019, pp. 519–524.

[100] R. Leung, "Smart hospitality: Taiwan hotel stakeholder perspectives," *Tourism Review*, 2019.

[101] S. Magistretti, C. Dell'Era, and A. Messeni Petruzzelli, "How intelligent is Watson? Enabling digital transformation through artificial intelligence," *Business Horizons*, Vol. 62, No. 6, 2019, pp. 819–829.

[102] A. Mitra, S. Gaur, and E. Giacosa, "Combining organizational change management and organizational ambidexterity using data transformation," *Management Decision*, 2019.

[103] L. Pee, S. Pan, and L. Cui, "Artificial intelligence in healthcare robots: A social informatics study of knowledge embodiment," *Journal of the Association for Information Science and Technology*, Vol. 70, No. 4, 2019, pp. 351–369.

[104] A. Thomas, "Convergence and digital fusion lead to competitive differentiation," *Business Process Management Journal*, 2019.

[105] K. Warner and M. Wäger, "Building dynamic capabilities for digital transformation: An ongoing process of strategic renewal," *Long Range Planning*, Vol. 52, No. 3, 2019, pp. 326–349.

[106] C. Lehrer, A. Wieneke, J.V. Brocke, R. Jung, and S. Seidel, "How big data analytics enables service innovation," *Journal of Strategic Information Systems*, Vol. 35, No. 2, 2018.

[107] R. Torres, A. Sidorova, and M. Jones, "Enabling firm performance through business intelligence and analytics: A dynamic capabilities perspective," *Information and Management*, Vol. 55, No. 7, 2018, pp. 822–839.

[108] H. Chen, R. Schütz, R. Kazman, and F. Matthes, "How Lufthansa capitalized on big data for business model renovation." *MIS Quarterly Executive*, Vol. 16, No. 1, 2017.

[109] A. Gunasekaran, T. Papadopoulos, R. Dubey, S. Wamba, S. Childe et al., "Big data and predictive analytics for supply chain and organizational performance," *Journal of Business Research*, Vol. 70, 2017, pp. 308–317.

[110] R. Basole, "Accelerating digital transformation: Visual insights from the API ecosystem," *IT Professional*, Vol. 18, No. 6, 2016, pp. 20–25.

[111] M. Hengstler, E. Enkel, and S. Duelli, "Applied artificial intelligence and trust – The case of autonomous vehicles and medical assistance devices," *Technological Forecasting and Social Change*, Vol. 105, 2016, pp. 105–120.

[112] M. Chalal, X. Boucher, and G. Marquès, "Decision support system for servitization of industrial SMEs: A modelling and simulation approach," *Journal of Decision Systems*, Vol. 24, No. 4, 2015, pp. 355–382.

[113] P. O'Donovan, K. Leahy, K. Bruton, and D. O'Sullivan, "An industrial big data pipeline for data-driven analytics maintenance applications in large-scale smart manufacturing facilities," *Journal of Big Data*, Vol. 2, No. 1, 2015, pp. 1–26.

[114] P. O'Donovan, K. Leahy, K. K. Bruton, and D. O'Sullivan, "Big data in manufacturing: A systematic mapping study," *Journal of Big Data*, Vol. 2, No. 1, 2015, pp. 1–22.

[115] S. LaValle, E. Lesser, R. Shockley, M. Hopkins, and N. Kruschwitz, "Big data, analytics and the path from insights to value," *MIT Sloan Management Review*, Vol. 52, No. 2, 2011, pp. 21–32.

Appendix A. List of selected articles

Author	Title	Year
Chen and Lin [84]	Business intelligence capabilities and firm performance: A study in China	2021
Gong and Janssen [65]	Roles and capabilities of enterprise architecture in big data analytics technology adoption and implementation	2021
Aboelmaged and Mouakket [85]	Influencing models and determinants in big data analytics research: A bibliometric analysis	2020
Akter et al. [11]	Transforming business using digital innovations: The application of AI, blockchain, cloud and data analytics	2020
Balakrishnan and Das [86]	How do firms reorganize to implement digital transformation?	2020
Brunetti et al. [87]	Digital transformation challenges: Strategies emerging from a multi-stakeholder approach	2020
Conboy et al. [74]	Using business analytics to enhance dynamic capabilities in operations research: A case analysis and research agenda	2020
Dremel et al. [88]	Actualizing big data analytics affordances: A revelatory case study	2020
Elia et al. [63]	A multi-dimension framework for value creation through big data	2020
Fan et al. [61]	The impact of the quality of intelligent experience on smart retail engagement	2020
Gotthardt et al. [68]	Current state and challenges in the implementation of smart robotic process automation in accounting and auditing	2020
Hilali et al. [76]	Reaching sustainability during a digital transformation: A PLS approach	2020
Lee and Kim [89]	Development of innovative business of telecommunication operator: Case of KT-MEG	2020
Maroufkhani et al. [22]	Big data analytics adoption model for small and medium enterprises	2020
Mikalef et al. [90]	Exploring the relationship between big data analytics capability and competitive performance: The mediating roles of dynamic and operational capabilities	2020
Moore [91]	Smart connected sensors, cyber-physical networks, and big data analytics systems in internet of things-based real-time production logistics	2020
Nguyen et al. [92]	A systematic review of big data analytics for oil and gas industry 4.0	2020
Osterrieder et al. [93]	The smart factory as a key construct of industry 4.0: A systematic literature review	2020
Serban and Lytras [55]	Artificial intelligence for smart renewable energy sector in Europe – Smart energy infrastructures for next generation smart cities	2020
Silva et al. [94]	Contributions of the internet of things in education as support tool in the educational management decision-making process	2020

Author	Title	Year
Sott et al. [95]	Precision techniques and agriculture 4.0 technologies to promote sustainability in the coffee sector: State of the art, challenges and future trends	2020
Tiwari and Khan [64]	Sustainability accounting and reporting in the industry 4.0	2020
Tuomi et al. [96]	x=(tourism_work) y=(sdg8) while y= true: automate (x)	2020
Wamba-Taguimdje et al. [23]	Influence of artificial intelligence (AI) on firm performance: The business value of AI-based transformation projects	2020
Zhang and Luo [97]	Knowledge structure, network structure, exploitative and exploratory innovations	2020
Bonanomi et al. [57]	The impact of digital transformation on formal and informal organizational structures of large architecture and engineering firms	2019
Brock and von Wangenheim [98]	Demystifying AI: What digital transformation leaders can teach you about realistic artificial intelligence	2019
Caputo et al. [73]	Innovating through digital revolution: The role of soft skills and big data in increasing firm performance	2019
Dahlbom et al. [67]	Big data and HR analytics in the digital era	2019
Feng et al. [71]	Understanding forced adoption of self-service technology: The impacts of users' psychological reactance	2019
Jocevski et al. [60]	Transitions towards omni-channel retailing strategies: A business model perspective	2019
Kalaivani and Sumathi [99]	Factor based prediction model for customer behavior analysis	2019
Leung [100]	Smart hospitality: Taiwan hotel stakeholder perspectives	2019
Magistretti et al. [101]	How intelligent is Watson? Enabling digital transformation through artificial intelligence	2019
Mitra et al. [102]	Combining organizational change management and organizational ambidexterity using data transformation	2019
Nam [66]	Technology usage, expected job sustainability, and perceived job insecurity	2019
Pee et al. [103]	Artificial intelligence in healthcare robots: A social informatics study of knowledge embodiment	2019
Schneider and Leyer [75]	Me or information technology? Adoption of artificial intelligence in the delegation of personal strategic decisions	2019
Thomas [104]	Convergence and digital fusion lead to competitive differentiation	2019
Wamba and Akter [62]	Understanding supply chain analytics capabilities and agility for data-rich environments	2019
Warner and Wäger [105]	Building dynamic capabilities for digital transformation: An ongoing process of strategic renewal	2019
Lehrer et al. [106]	How big data analytics enables service innovation	2018
Torres et al. [107]	Enabling firm performance through business intelligence and analytics: A dynamic capabilities perspective	2018
Chen et al. [108]	How Lufthansa capitalized on big data for business model renovation	2017

Author	Title	Year
Gunasekaran et al. [109]	Big data and predictive analytics for supply chain and organizational performance	2017
Basole [110]	Accelerating digital transformation: Visual insights from the API ecosystem	2016
Hengstler et al. [111]	Applied artificial intelligence and trust – The case of autonomous vehicles and medical assistance devices	2016
Chalal et al. [112]	Decision support system for servitization of industrial SMEs: A modelling and simulation approach	2015
O'Donovan et al. [113]	An industrial big data pipeline for data-driven analytics maintenance applications in large-scale smart manufacturing facilities	2015
O'Donovan et al. [114]	Big data in manufacturing: A systematic mapping study	2015
Barrett et al. [69]	Reconfiguring boundary relations: Robotic innovations in pharmacy work	2012
LaValle et al. [115]	Big data, analytics and the path from insights to value	

Appendix B. Journal publication patterns

Source title	No. of studies	ABS rating	CORE rating	Citation index
Information Processing and Management	1	2	NA	SCIE, SSCI
International Journal of Asian Business and Information Management	1	NA	NA	ESCI
Journal of Cleaner Production	1	2	NA	SCIE
Technology Analysis and Strategic Management	1	2	NA	SSCI
Journal of Science and Technology Policy Management	1	1	NA	ESCI
International Journal of Production Economics	1	3	NA	SCIE
Annals of Operations Research	1	3	NA	SCIE
Journal of Theoretical and Applied Electronic Commerce Research	1	1	NA	SSCI
Information and Management	3	3	NA	SCIE, SSCI
International Journal of Innovation Science	1	NA	NA	ESCI
Business Process Management Journal	2	2	NA	SSCI
International Journal of Innovation and Learning	1	NA	NA	ESCI
Marketing Intelligence and Planning	1	1	NA	SSCI
TQM Journal	1	1	NA	NA
Baltic Journal of Management	1	1	NA	SSCI
Engineering, Construction and Architectural Management	1	1	NA	SCIE, SSCI
Business Horizons	1	2	NA	SSCI
Management Decision	2	2	NA	SSCI

Source title	No. of studies	ABS rating	CORE rating	Citation index
Behaviour and Information Technology	1	NA	B	SCIE, SSCI
California Management Review	1	3	NA	SSCI
International Journal of Systems Assurance Engineering and Management	1	NA	NA	ESCI
Long Range Planning	1	3	NA	SSCI
International Journal of Retail and Distribution Management	1	2	NA	SSCI
Journal of the Association for Information Science and Technology	1	NA	A*	SCIE, SSCI
Managerial and Decision Economics	1	2	NA	SSCI
Tourism Review	1	1	NA	SSCI
Technological Forecasting and Social Change	2	3	NA	SSCI
Journal of Decision Systems	1	1	NA	ESCI
Strategic Change	1	2	NA	NA
Economics, Management, and Financial Markets	1	NA	NA	NA
ACRN Journal of Finance and Risk Perspectives	1	NA	NA	NA
IEEE Access	3	NA	NA	SCIE
International Journal of Information Management	1	2	NA	SSCI
Industrial Marketing Management	1	3	NA	SSCI
Annals of Tourism Research	1	4	NA	SSCI
Journal of Big Data	2	NA	NA	ESCI
MIT Sloan Manag. Rev	1	3	NA	SSCI
Journal of Business Research	1	3	NA	SSCI
International Journal of Operations Production Management	1	4	NA	SSCI
IT Professional	1	NA	C	SCIE
MIS Quarterly Executive	1	2	NA	SSCI
Journal of Management Information Systems	1	4	NA	SCIE, SSCI
Organization Science	1	4*	NA	SSCI
European Journal of Operational Research	1	4	NA	SCIE

e-Informatica Software Engineering Journal, Volume 16, Issue 1, 2022, pages: 220109, DOI: 10.37190/e-Inf220109

Reporting Consent, Anonymity and Confidentiality Procedures Adopted in Empirical Studies Using Human Participants

Deepika Badampudi*, Farnaz Fotrousi**, Bruno Cartaxo***, Muhammad Usman****
*Blekinge Institute of Technology, Sweden
**University of Hamburg, Germany
***Federal Institute of Pernambuco, Brasil
****Blekinge Institute of Technology, Sweden
deepika.badampudi@bth.se, farnaz.fotrousi@uni-hamburg.de, email@brunocartaxo.com, muhammad.usman@bth.se

Abstract

Background: Empirical studies involving human participants need to follow procedures to avoid causing harm to the subjects. However, it is not always clear how researchers should report these procedures.
Aim: This study investigates how researchers report ethical issues in the software engineering journal publications, particularly informed consent, confidentiality, and anonymity.
Method: We conducted a literature review to understand the reporting of ethical issues in software engineering journals. In addition, in a workshop, we discussed the importance of reporting the different ethical issues.
Results: The results indicate that 49 out of 95 studies reported some ethical issues. Only six studies discussed all three ethical issues. The subjects were mainly informed about the study purpose and procedure. There are limited discussions on how the subjects were informed about the risks involved in the study. Studies reported on how authors ensured confidentiality have also discussed anonymity in most cases. The results of the workshop discussion indicate that reporting ethical issues is important to improve the reliability of the research results. We propose a checklist based on the literature review, which we validated through a workshop.
Conclusion: The checklist proposed in this paper is a step towards enhancing ethical reporting in software engineering research.

Keywords: research ethics, informed consent, confidentiality, anonymity

1. Introduction

Human subjects are often involved in studies in software engineering research, mainly students and practitioners who are considered vulnerable participants [1, 2]. The research results may cause significant psychological, social and economic damage to subjects who are employees [2]. Similarly, there is a possibility that students who are subordinates could be coerced into participating in research studies [2], which may affect the validity of the results. Therefore, it is important to evaluate potential risks and vulnerabilities to participants before employing them in a research study. The researchers should take the necessary steps

Submitted: 15 Feb. 2022; Revised: 10 Jun. 2022; Accepted: 10 Jun. 2022; Available online: 22 Jun. 2022

to minimize or prevent risks [2], as well as to adequately inform the subjects about the study and its risks. Additionally, researchers should obtain informed consent explaining the purpose and procedure of the research, the potential conflict of interest, risks and benefits. The subjects are more likely to provide a reliable and honest response when they are ensured confidentiality and anonymity [3].

Given the importance of ethical issues, some journals provide guidelines on crediting authors (authorship), handling conflict of interest and reproducibility of the data and analysis software. In addition, journals provide specific guidelines on how to involve human subjects in the research and require that researchers report how they obtained informed consent. For example, Springer instructs the authors to report the ethical issues as follows – "*For all research involving human subjects, freely-given, informed consent to participate in the study must be obtained from participants and a statement to this effect should appear in the manuscript… if any of the sections are not relevant to your manuscript, please include the heading and write 'Not applicable' for that section*"[1]. We believe that it is not only important to state that authors obtained consent; however, the authors should also report the procedure of obtaining consent to improve accountability and trust.

Badampudi [4] reviewed how authors report ethical issues in the latest issues of the empirical software engineering journal. It concluded that there is limited reporting of ethical issues [4]. However, the review study only considered five issues from one journal. Our study has considered multiple volumes and issues of four different journals (more details in Section 2.2).

The contributions of our study are as follows:

- We reviewed how researchers reported consent, anonymity and confidentiality in 95 journal papers.
- In addition, we aggregated the different details reported in the primary studies and proposed a checklist that will help the authors to:
  1. Identify the consent, anonymity and confidentiality issues that are important for their study.
  2. Plan for addressing the consent, anonymity and confidentiality issues.
  3. Report the procedure to obtain consent, anonymity and confidentiality to increase accountability and trust.
- The checklist contributes to a better understanding of consent, anonymity and confidentiality by clarifying the difference between them and elaborating on what is meant by each ethical issue.
- We also conducted a workshop to discuss the checklist for consent, anonymity and confidentiality and get initial feedback.

It is important to keep in mind that both the review and the checklist we present in this paper cover only consent, anonymity and confidentiality issues related to software engineering empirical studies that use human participants directly. Ethical concerns not associated with that sphere are – the use of data from social networks, code repositories, or organization-related data, are out of scope. The review is also limited to a sample of publications in four journals.

The remainder of this paper is structured as follows: Section 2 presents background on ethical issues applicable to software engineering and the relevant related work to this study. Section 3 describes the design of our research, which is followed by Section 4, where we describe the literature review results. We present our and workshop results and checklist

[1] https://www.springer.com/gp/editorial-policies/informed-consent

for reporting ethical issues in Section 5.2. Section 6 presents the discussions and finally Section 7 concludes our study.

2. Background and related work

In this section we provide information on the ethical issues considered in our study, and elaborate the related work.

2.1. Ethical issues

Singer and Norman [1] identify four ethical issues that are relevant to software engineering empirical studies: informed consent, confidentiality, beneficence, and scientific value. Singer and Norman have discussed anonymity as part of confidentiality [1]. Whereas Coffelt [5] discussed the difference between the concepts. Anonymity is the state when the researchers can not identify the identity of individual subjects. While confidentiality refers to the state that the researchers know the subjects but take actions to protect their identity and data from being revealed [5]. We describe the ethical issues below.

- **Informed consent** can be obtained by disclosing the following information: the purpose of the study, research approach, who will access the raw data and for what purpose, risks to the subjects, anticipated benefits for the subjects, the importance of voluntariness and statement offering to answer subject's questions.
- **Anonymity** involves not collecting data that can identify or trace an individual or an organization.
- **Confidentiality** refers to protecting of the raw data and only publishing the aggregated results that cannot be traced to an individual or an organization.
- **Scientific value** relates to the study validity and, research topic importance [1]. If researchers do not ethically conduct research, it could lead to incorrect interpretation of the data and have implications on human participation such as waste of time and effort [6]. Examples of ethical issues in scientific value are: assigning participants to a disadvantaged control situation, incorrect results due to publication bias (not publishing statistically non-significant results)[7], researcher bias (flexible analyses that lead initially statistically non-significant results to become significant) [7] and experimenter expectancy bias (unintentional experimenter behavior that increases the likelihood of the hypothesis to be confirmed) [8].
- **Beneficence** has two components: human beneficence, which is maximizing benefits and minimizing harm (risk-benefit ratio), and organization beneficence which is minimizing the harm to an organization when uncovering issues and challenges in a company.

Our study, focuses on: informed consent (including the description of benefits and risks), confidentiality and anonymity. Since our goal is only on those ethical issues directly related to human participation, we did not focus on ethical issues related to scientific value in our study. Moreover, each of the ethical issues related to scientific value requires a deeper investigation. For example, a crossover study design is considered good to ensure that all participants are assigned to each control situation in the experiment. However, it is argued that crossover design may make the study more unethical in oncology clinical trials due to confounding by crossover [9]. Furthermore, if crossover design as not designed or analysed properly, it may results in invalid results [10], which affects the scientific value.

For investigating the reporting of beneficence in publications, the benefits and risks should be sufficiently discussed to investigate the beneficence. However, in our pilot study [4] we identified few studies that discussed risks explicitly; therefore, it would be difficult to investigate the beneficence unless explicitly discussed in the publication. We did not focus on beneficence reporting. However, we extracted information on risks and benefits, which will allow us to investigate beneficence.

2.2. Journals guidance to authors regarding reporting ethical issues

In this study, we analyze ethical issues as reported in a sample of papers from four leading software engineering journals: Springer – Empirical Software Engineering (EMSE), Elsevier – Information and Software Technology (IST), IEEE – Transactions on Software Engineering (TSE), and ACM – Transactions on Software Engineering and Methodology (TOSEM). These journals provide different guidance to authors regarding how to report ethical issues.

- **EMSE** recommends that an informed consent statement should appear in the paper manuscript for all research involving human subjects. The journal also touches on anonymity and confidentiality issues, although it does not recommend whether to report them or not.
- **IST** endorses that authors should include a statement in the manuscript mentioning that they obtained informed consent whenever the research involves human subjects. On the other side, the journal does not provide any instruction regarding anonymity and slightly touches the confidentiality issue.
- **TSE** does not make any recommendation regarding any of the three ethical issues.
- **TOSEM** tangentially covers the three ethical issues. However, there is no recommendation concerning whether and how to report such information in the paper.

A review of the author guidelines of these four software engineering journals indicates that they do not impose strong and detailed guidance on how the authors should report ethical issues like informed consent, anonymity, and confidentiality. Moreover, sometimes it is hard to find the instructions to the authors. For example, EMSE at least touches on all three ethical issues. However, parts of the recommendations are spread throughout the journal submission guidelines[2], while the other parts appear on the Springer editorial policies[3]. Still, the latter has a general nature since it refers to all journals published by Springer. Another example is TOSEM, which does not mention any of the three ethical issues of its submission guidelines[4]. To find such kind of information, one has to access the ACM Code of Ethics and Professional Conduct[5], and it also has a general nature. One must infer how to report ethical issues on papers by analysing a code of conduct to guide computing professionals behavior.

2.3. Related work

There are many relevant discussions in the computer science academic community about ethical issues of our profession. For instance, the various considerations on the ethics of advanced machine learning algorithms [11–13], or how software developers should be

[2]https://bit.ly/3joV9YQ
[3]https://www.springer.com/gp/editorial-policies
[4]https://dl.acm.org/journal/tosem/author-guidelines
[5]https://www.acm.org/code-of-ethics

conscious about the impacts of systems they create and the way they behave as professionals [14–17].

Still, we do not see that level of urgency when considering ethical issues of empirical studies involving human subjects in software engineering. The results of our paper substantiate this claim, as well as other few related studies.

In 2002, Singer and Vinson [1] called attention to ethical issues that had been neglected in software engineering empirical studies. Based on a review of ethical codes of many research fields, the authors identified ethical issues related to software engineering empirical studies: informed consent, scientific value, beneficence, and confidentiality. They also illustrated those four issues with real empirical studies. In 2008, Singer and Vinson [18] expanded the first discussion, this time focusing on the role of Ethics Review Boards (ERB) and how to comply with them. They provide detailed information about how to plan and which documents are needed during an ERB review.

A recent literature review investigated ethical authorship issues on diverse research disciplines [19]. The author did not find any paper discussing ethical authorship issues in software engineering. In contrast, the author found 16 articles in research areas like Medical, Science and Engineering, Chemistry, Education, and Economics. The literature review does not cover ethical issues related to empirical studies in software engineering as our study does. However, it unveils more evidence that ethical issues have low priority in our research community. Few studies report or discuss ethical issues in the software engineering research field.

Software engineering research based on Mining Software Repositories (MSR) strategies has soared during the last decade. Although data collection and analysis in MSR studies are usually automated, Gold and Krinke [20] argue that such kind of research may involve human subjects, as repositories typically contain data about developers' interactions. In this context, they discuss the ethical implications of MSR research. From the viewpoint of the process used to ensure ethical software engineering research, Strandberg [21] proposed a checklist based on authoritative guidelines for interview studies involving industrial practitioners.

A subject even more rarely discussed is how inviting participants to software engineering surveys can pose relevant ethical issues. Baltes and Diehl [22] report their experience with different sampling strategies to conduct surveys. The authors highlight that researchers should be conscious that contacting software developers may harm them even when they do not answer the survey. Baltes and Diehl received the following comment by a developer they contacted asking to participate in one of their surveys *"I consider this problem now worse than spam since Google at least filters out spam for me.* [...] *[Y]ou send one, I get one per week – or more."*

3. Research method

We used a mixed-methods approach – consisting of a literature review and a workshop – to understand 1) which ethical issues are reported in SE journal publications and 2) which ethical issues should be reported and the importance SE researchers place on reporting different ethical issues in their publications.

3.1. Research questions

RQ1 **To what extent, consent, confidentiality and anonymity are reported in software engineering journal publications?**
Rationale: Here we will describe reporting of research ethics in a sample of papers published in the four journals mentioned in Section 3.2.1. Mainly to understand to what extent and how authors discuss consent, confidentiality and anonymity.

RQ2 **Which ethical issues related to consent, confidentiality and anonymity should be reported in software engineering publications?**
Rationale: Here we will describe the importance of reporting consent, confidentiality and anonymity, and how they should be reported.

3.2. Literature review

We conducted a literature review to understand how software engineering (SE) researchers report research ethics in SE publications. We followed a systematic study selection and data extraction process. However, we did not perform the quality assessment of the included studies. In addition, our search is also limited to a few volumes in the selected journals. Thus, we do not refer to our review as a systematic literature review. We report the details of the literature review process in the sections below.

3.2.1. Data collection

We selected four journals in software engineering, namely – The Empirical Journal in Software Engineering (EMSE), Information and Software Technology (IST), Transactions on Software Engineering (TSE), and Transactions on Software Engineering and Methodology (TOSEM). We selected these journals as they are among the top-ranked SE journals and are expected to reflect the best current reporting practices. We started our search from the volumes published in the summer of 2019 and continued screening previous volumes until we reached a sample of 100 papers (excluding editorials and letters) from each of the four journals. Table 1 provides the details of screening which includes the volumes, years and the number of papers reviewed in each journal.

Table 1. Data collection description

Publisher	Journal	Volume	Year	No.
Springer	EMSE	V.23 I.6 to V.24 I.3	2018 (all issues)–Aug 2019	100
Elsevier	IST	V.103 to V.110	Nov 2018–Jun 2019	104
IEEE	TSE	V.43 I.11 to V45 I.7	Nov 2017–Jul 2019	105
ACM	TOSEM	V.24 I.4 to V.28 I.3	Aug 2015–Jul 2019	100
Total				409

3.2.2. Study selection

Our objective was to include papers that employ humans in the study. Therefore we included papers that employ human subjects or involve collecting the information that can lead to identifying an individual or an organization. We excluded papers that do not collect

information from practitioners, such as methodological papers, systematic literature reviews and solution proposals. In addition, we excluded the studies that collect information that is publicly available (data from open source) and studies that do not involve human subjects or authors themselves are subjects. All four authors were involved in the review process. To ensure that we have the same interpretation of the inclusion criteria, we conducted a pilot study of 20 papers. All authors independently reviewed the title and abstracts of the papers to either include or exclude the papers. We conducted a kappa test to evaluate the agreement level. The average Cohen kappa for all raters for our pilot study was 0.88, which indicates a high agreement [23]. However, we still discussed the papers where at least one author had a different decision. We concluded that title and abstracts might not be sufficient to determine the inclusion of human subjects in the study design. Therefore we decided also to review the research questions and data collection methods when deciding to include or exclude the paper. Table 2 provides the total number of papers included from each journal.

Table 2. Number of papers included from each journal

Journals	Included Papers
EMSE	28
IST	21
TSE	33
TOSEM	23
Total	105
After full text reading	95

3.2.3. Data extraction

To facilitate the data extraction, we devised an extraction form. We conducted a pilot extraction study to review the relevance, completeness, and interpretation of the extraction items. All four authors extracted two papers, each resulting in data extraction from eight papers in the pilot extraction. As a result of the pilot extraction, we decided to remove some of the extraction items, such as extraction of research methods, as the data collection method was perceived to be more relevant for our study. Table 3 lists the extraction items. The first, third, and fourth authors extracted the data, and the second author reviewed the extraction.

Table 3. Data extraction form

Item	Description
Data collection Procedure	How was the data collected?
Data collector	Who collected the data?
Category of subjects	Who are the subjects – students and/or practitioners
Data description	What data is collected in the study?
Ethical issues (informed consent, confidentiality, and anonymity)	What was reported on ethical issues? (Verbatim from the paper)

3.2.4. Analysis

We conducted a mixed qualitative-quantitative analysis approach. For qualitative analysis, we performed inductive content analysis [24] to categorise the extracted information relevant to informed consent, confidentiality and anonymity. We chose this approach to look for new knowledge on the phenomena instead of relying on prior knowledge. We performed the analysis in the following steps:

1. Performing initial coding: For all the extracted data, we underlined all terms related to any of the three issues of informed consent, confidentiality, and anonymity. We doubled check whether the information provided for each category could also be relevant to another category.
2. Forming final codes: We grouped the initial codes to form the final codes iteratively based on the shared characteristics of the codes that could put them in the same group.
3. Forming categories: We overviewed the final codes and categorised them based on the patterns we found within the codes. We merged the categories into high-level when they could make sense.

For example, we extracted a text from a paper regarding confidentiality as *"…the data would remain with us, and the transcripts would not be published but only the research findings supported by the anonymous quote"*. We assigned an initial coding "publishing" to the statement, later converted to the final code "sharing." Finally, we formed the category "reporting the sharing procedure" and assigned the statement to this category. For quantitative analysis, we used descriptive statistics and mainly used bar charts to visualise data quantitatively.

3.3. Workshop

We conducted a workshop study [25] to evaluate the importance of ethical issues from the perspective of software engineering researchers. The evaluation contributes to understanding what ethical issues researchers should report in software engineering publications.

The first and second authors organised the workshop study as a session of the SEthics 2021 (2nd International Workshop on Ethics in Software Engineering Research and Practice), co-located with ICSE 2021. SEthics2021 was virtual and used the ICSE Researchr platform. We conducted a survey and group discussions in the workshop to collect data. Surveys and group discussions are considered as suitable methods when evaluating artefacts (checklist in our study) based on people's perspectives [26]. The workshop study session was organized for 40 minutes in the following four parts:

1: Introduction – At the beginning of the workshop, the first author provided the following information: an introduction to ethical issues applicable to software engineering (SE) research, journal publishers' requirements to report ethical issues and a summary of our literature review results on the current state of reporting research ethics in SE publications.

2: Survey – We designed a questionnaire in the Mentimeter (https://www.mentimeter.com/). After the introduction, we asked participants to answer a survey on the importance of reporting ethical issues. The survey included the following question: How do you rank the importance of the ethical issues?

– IC1: Report the process of how the study purpose statement is communicated.
– IC2: Report the process of how the risks, and benefits are communicated to the participants. An explanation of any foreseeable risks or discomforts.

– IC3: Report how voluntariness is ensured. An explanation of the subject's right to refuse without penalty.
– C1: Report how the analysis was conducted while protecting confidentiality.
– C2: Report how data is stored and used to ensure confidentiality.
– C3: Report how the data is shared to protect confidentiality.
– A1: Report how the data and subjects are anonymised.

The participants ranked the statements between 3 (Definitely will consider) to 1 (Would not consider) using sliders in Mentimeter. We provided a guide for the participants that defined and exemplified each statement to clarify the statements. The Ethical issues IC1–IC3 are relevant to informed consent, C1–C2 are about confidentiality and A1 is related to anonymity.

3: Group discussion within the groups – We created two breakout rooms: "reporting ethical details in manuscripts" or "skipping ethical details". We asked the participants to join the breakout rooms that best represented their survey response. For example, if the participants primarily selected ratings close to 3, they should enter the breakout room: "reporting ethical details in manuscripts". The first and second authors moderated each breakout room, responsible for facilitating, documenting, and summarising the discussions. The moderators took notes to collect the breakout room discussions. Taking notes allows unobtrusively collecting data in real-time [26]. Collecting data through field notes is prone to researcher bias [26]. To mitigate researcher bias, we shared the data collected in our notes with the participants, where they had an opportunity to confirm or suggest a reformulation.

4: Final group discussions – All participants from the breakout session joined the main session to share the discussions carried out in the breakout rooms. The first and second authors again shared the summarised statements of each group with all participants in the main session.

We informed the participants about the working group in advance in the program of the SEthics21 workshop published on the website. To ensure the confidentiality of the participants, we did not report the traceability of individual responses to participants. We ensured anonymity and confidentiality to mitigate social desirability bias. ''Social desirability refers to the respondents' tendency to admit to socially desirable traits and behaviors and to deny socially undesirable ones'' [27]. We wanted the workshop participants to be honest, particularly if they disagreed with the need to report ethical aspects which could be considered sensitive in a workshop focused on ethics. Privacy (anonymity and confidentiality) can help in producing honest responses to sensitive questions [28]. Due to the pandemic, we conducted the workshop online. The sessions were recorded and uploaded to a streaming platform. We did not audio or video record the discussions of the working group; however, we recorded the summaries of the discussions. In our results, we report the recorded summaries word to word to avoid any misinterpretation. To ensure the validity of the concluded statements, we performed two real-time validations: within each breakout rooms and again in the main session, where participants validated the discussion summary.

4. How ethical issues are reported in software engineering publications

This section answers RQ1 based on 95 included primary studies. We included only studies that employed human subjects directly. Practitioners were the most commonly employed subjects in the primary studies, followed by students. In some studies, both practitioners and

students were involved. A small portion of papers also involved end-users and researchers as subjects (see Figure 1 for details).

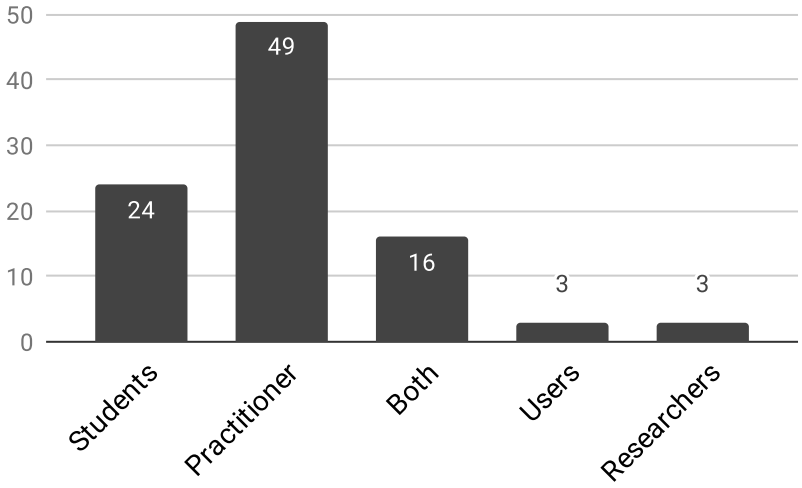


Figure 1. Number of papers in each subject category

4.1. Overview of papers that reported ethical issues (49/95)

In our sample, the data was collected from subjects mostly by conducting experiments and through surveys and interviews as see in Figure 2. Most papers used a combination of two or more methods. In addition, papers reported using additional sources to collect data such as company documents, and data from crowdsourcing platform. In few papers, a tool was used to collect data. For example, a tool was installed on developers system to observe their activities. We categorized papers in the *other* category that did not report any specific data collection method, the studies were mostly exploratory. We did not find any significant relation among the papers reporting the three ethical issues and the data collection methods.

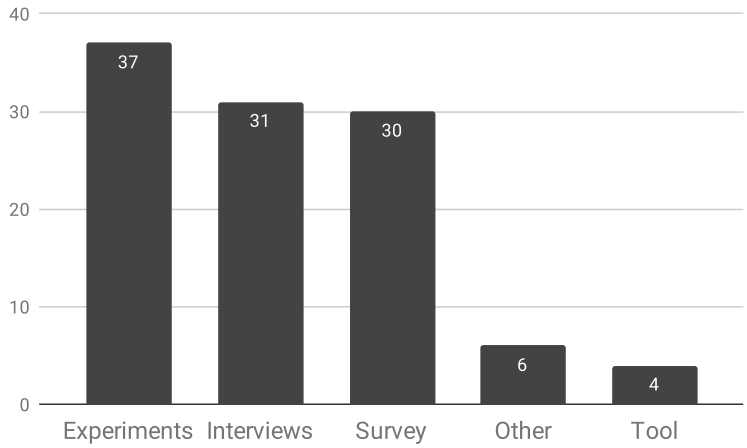


Figure 2. Methods used to collect data from subjects

From our sample of 95 papers, we found that around 50% (49 papers) considered at least one ethical issue in their study. Figure 3 shows the number of papers that report the different ethics issues. The number in round brackets (n) indicates the number of papers discussing only one ethical issue. As seen in Figure 3, not all ethical issues (informed consent, confidentiality and anonymity) are considered in all the 49 papers. Only six papers have discussed all three issues. Confidentiality and anonymity are more often mentioned together than any other combination of the three ethical issues. Although 50% of the papers in our sample report ethical issues, in most cases, however, only one issue is discussed. We provide details on what do researchers report on the ethical issues in Section 4.2.

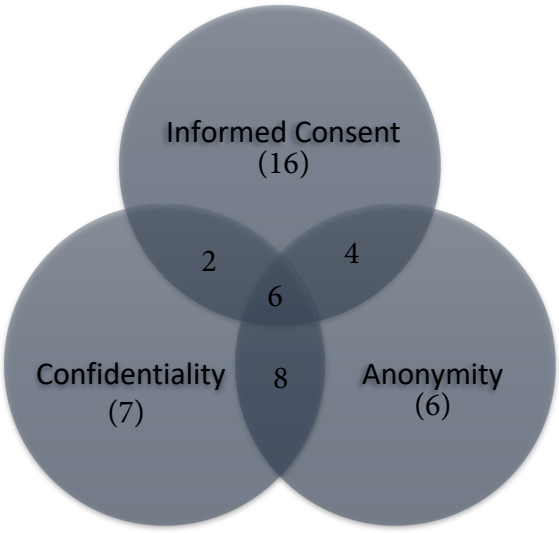


Figure 3. Number of papers discussing different ethical issues

We looked at how the researcher reported the three ethical issues. The papers that reported ethical issues reported different levels of detail. The next section provides details on what researchers report and how.

4.2. Details on ethical issues reported in software engineering publications

This section provides details on what and how much researchers report on ethical issues in the 49/95 studies. Figure 4 shows the overview of the reporting status. As seen in the figure, 28 papers discussed informed consent. However, out of 28 papers, nine papers did not discuss any details on the procedure for obtaining informed consent. The authors only mentioned that they obtained informed consent. We observed that most papers reported some details on how the authors ensured confidentiality. However, half of the papers did not report how authors achieved anonymity. Overall, 38 papers discuss details on addressing at least one ethical issue. Table 4 provides the list of the papers reporting details.

Table 4. List of papers that reported the procedure for addressing ethical issues

Paper ID	Title
ESE1	System requirements-OSS components: matching and mismatch resolution practices – an empirical study

Table 4 continued

Paper ID	Title
ESE12	Getting the most from map data structures in Android
ESE13	Older adults and hackathons: a qualitative study
ESE14	An empirical study on the impact of AspectJ on software evolvability
ESE21	Understanding the behaviour of hackers while performing attack tasks in a professional setting and in a public challenge
ESE25	An empirical study of architecting for continuous delivery and deployment
ESE26	Eye tracking analysis of computer program comprehension in programmers with dyslexia
ESE3	An industrial case study on the use of UML in software maintenance and its perceived benefits and hurdles
ESE4	Factors and actors leading to the adoption of a JavaScript framework
ESE8	Large-scale agile transformation at Ericsson: a case study
IST11	Exploratory testing: Do contextual factors influence software fault identification?
IST12	Impact of model notations on the productivity of domain modelling: An empirical study
IST14	The current state of software license renewals in the I.T. Industry
IST17	GuideGen: An approach for keeping requirements and acceptance tests aligned via automatically generated guidance
IST18	Quality requirements challenges in the context of large-scale distributed agile: An empirical study
IST5	An exploratory study of waste in software development organizations using agile or lean approaches: A multiple case study at 14 organizations
TOSEM16	Documenting Design-Pattern Instances: A Family of Experiments on Source-Code Comprehensibility
TOSEM17	Many-Objective Software Remodularization Using NSGA-III
TOSEM18	Software Change Contracts
TOSEM19	Platys: An Active Learning Framework for Place-Aware Application Development and Its Evaluation
TOSEM2	Status Quo in Requirements Engineering: A Theory and a Global Family of Surveys
TOSEM21	Mining Unit Tests for Discovery and Migration of Math APIs
TOSEM22	Code-Smell Detection as a Bilevel Problem
TOSEM23	On the Comprehension of Program Comprehension
TOSEM6	Fixing Faults in C and Java Source Code: Abbreviated vs. Full-Word Identifier Names
TOSEM9	Multi-Criteria Code Refactoring Using Search-Based Software Engineering: An Industrial Case Study
TSE1	makeSense: Simplifying the Integration of Wireless Sensor Networks into Business Processes
TSE14	Data Scientists in Software Teams: State of the Art and Challenges
TSE17	Coordination Challenges in Large-Scale Software Development: A Case Study of Planning Misalignment in Hybrid Settings
TSE18	Measuring Program Comprehension: A Large-Scale Field Study with Professionals
TSE2	Automatic Identification and Classification of Software Development Video Tutorial Fragments
TSE21	Towards Prioritizing Documentation Effort
TSE22	A Comparison of Program Comprehension Strategies by Blind and Sighted Programmers
TSE26	Understanding Diverse Usage Patterns from Large-Scale Appstore-Service Profiles
TSE3	The Good, the Bad and the Ugly: A Study of Security Decisions in a Cyber-Physical Systems Game
TSE6	Integrating Technical Debt Management and Software Quality Management Processes: A Normative Framework and Field Tests

Table 4 continued

Paper ID	Title
TSE7	Automated Refactoring of OCL Constraints with Search
TSE9	What Makes a Great Manager of Software Engineers?

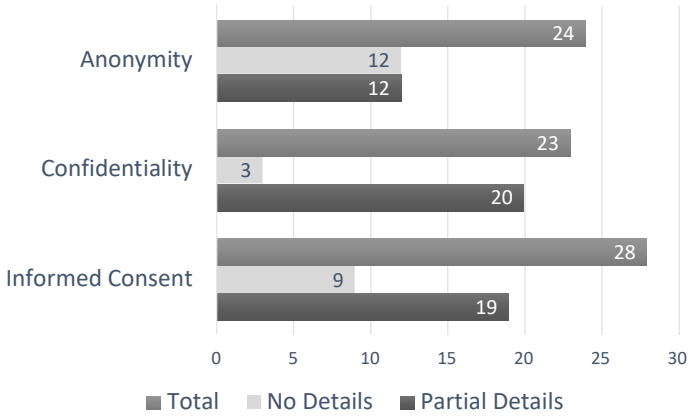


Figure 4. Overview of ethical issues reporting

The status of each paper reporting details on confidentiality, anonymity and informed consent is shown in Figure 5. As seen from Figure 3, six studies report all three ethical issues. However, out of the six, only three studies report details on how they addressed ethical issues, as seen in Figure 5. It is important to note that even though the studies (ESE4, ESE13, and TOSEM22) reported on all three ethical issues, they only report partial details. Most studies report details on at most one ethical issue. However, some of these studies have a rather detailed explanation of how they addressed the ethical issues. The level of details reported varies across the studies. We look at how researchers discuss each ethical issue in the primary studies.

4.2.1. Informed consent

Full informed consent is important to ensure that the subjects understand the implications of participating in the study. As seen in Figure 4, 19 out of 28 papers discussed the details on obtaining informed consent. Most studies provided a link to the consent form from which we extracted the details. Figure 6 provides the details on informed consent reported in the 19 studies. As seen in Figure 6, among 19 studies that provided details on informed consent, study purpose and procedure is discussed more commonly (10 studies) in the consent forms, followed by benefit explanation (eight studies) and voluntariness (seven studies). However, most of the studies (12/19 papers) discuss only one ethical issue. ESE26 reports most details on obtaining informed consent; however, it does not discuss the risks to the subjects. Only one study, i.e., TOSEM16, discusses risk. The subjects can only be fully informed about the participation if they are fully aware of the potential risks and in related to the benefits gained from participation.

The primary studies reported the following details on the procedure for obtaining informed consent:

Study purpose and procedure: The studies that mentioned that the study purpose and procedure were mainly to get honest and accurate responses from the subjects. The

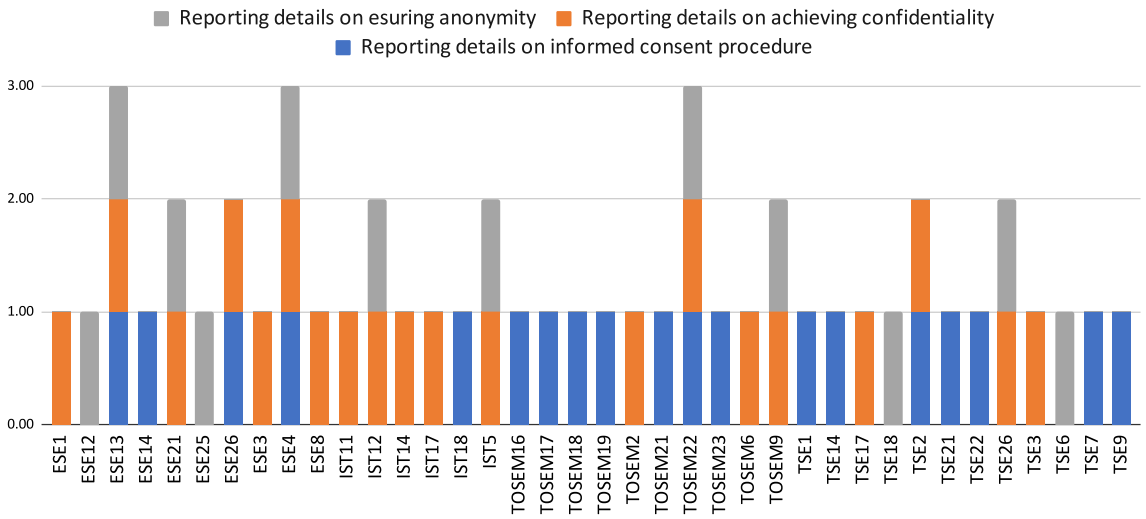


Figure 5. Papers reporting details on how ethical issues were addressed

authors communicated the study's purpose and procedure at the beginning of the interview or sent an email before the interviews/experiment.

Benefit: Some studies with students as subjects mentioned benefits in terms of extra credit. In contrast, some studies mentioned monetary benefits ranging from $100 to $200 either to some or all subjects. One of the studies (ESE26) mentioned non-monetary benefits. They provided the importance of the topic and the benefit to the software engineering community at large.

Voluntariness: It is important to discuss voluntariness together with the benefit of participation. For example, some studies mentioned that the students were not obliged to participate in the study. However, the researchers offered the participating students extra credit. Such a benefit can compromise the voluntariness as there is a penalty (no extra credit) when not participating in the study. One study (TOSEM19) explicitly mentioned that nonparticipating students received an alternative task to earn extra credit. Therefore, it is important to report the procedure to ensure voluntariness without any penalty.

Risks: One study (TOSEM16) discusses the risk of the experiment results influencing the students' grades. They reported that the study ensured that the experiment did not influence the grades.

Only one study (TOSEM16) in our sample discussed both risks and benefits (see Figure 6). However, the risks and benefits were discussed for master student subjects and not for other participants involved in the experiment (professionals and PHD students). The risks for students was the experiment influencing their grades which was mitigated by rewarding an extra point for all participants regardless of their performance.

4.2.2. Confidentiality

In total, 23 papers reported confidentiality, of which three papers do not provide any details, just stated they assure confidentiality. Using the qualitative analysis, we understood that the remaining 20 papers report confidentiality at least in one of the following aspects:

- Storing and using data (12 studies),
- Analysing data (two studies),
- Sharing data (eight studies),

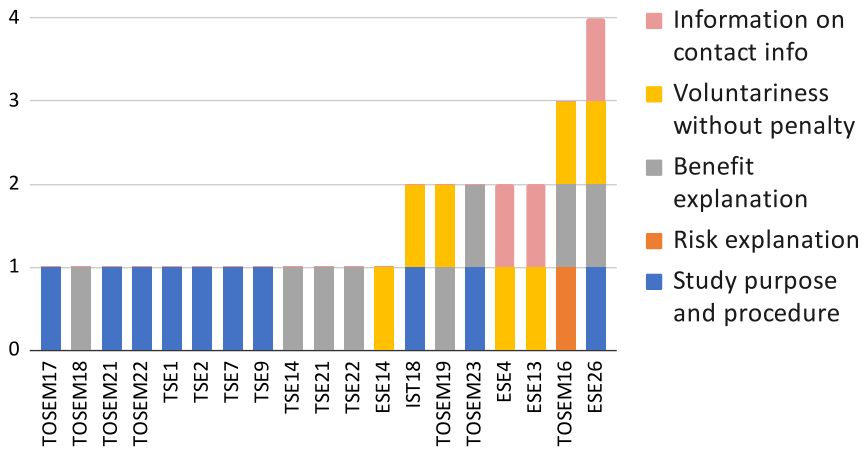


Figure 6. Studies reporting details on informed consent procedure

– Ethical approval (four studies),
– What data that is kept confidential (six studies).

As seen in Figure 7, only one paper, i.e., IST14, reported all confidentiality aspects, while most studies (14/20) address only one of the aspects. One study, i.e., ESE21, provides most details about confidentiality, however, it does not discuss the ethical issues of data analysis and approval.

The confidentiality details reported in the primary studies are as follows:

Storing and using data: Some studies reported the procedure on how authors collected and kept the private information confidential. Most of the papers that reported confidentiality have provided information on storing and using data (12/20). Researchers reported that they chose not to reveal information while storing and retrieving data. TSE26 reported using warehouse servers behind the company firewall to keep data confidential.

Analysing data: Only two studies, i.e., IST14 and TSE3, reported the procedure for

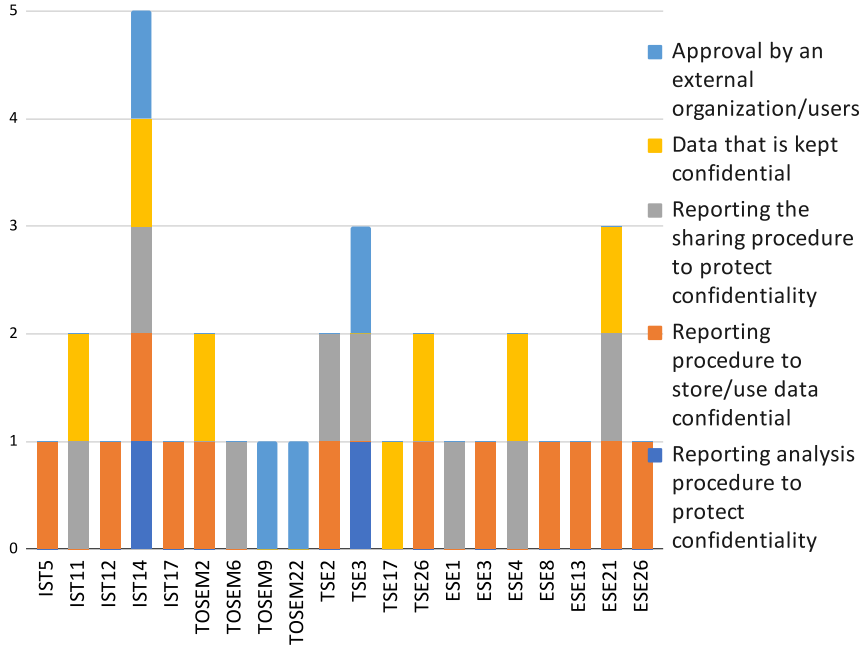


Figure 7. Studies reporting details on confidentiality

analyzing data confidential. The studies mainly cleaned the interview transcripts and recording statements to remove and destroy confidential information. One of the studies (IST14) also summarised content to ensure no confidential information is taken out of the organisation.

Sharing data: Some studies reported how the researchers protected and did not share the private data with other organisations or individuals. One of the common approaches used in the studies, for example, in ESE1, and TOSEM6, was aggregating answers/data before sharing to ensure confidentiality.

Ethical approval: Four studies, TOSEM9, TOSEM22, TSE3, IST14, reported that an Ethical committee reviewed the research project, design (e.g., questionnaire), and data to ensure their conformity with the ethical norms.

Data that is kept confidential: Some studies reported what data they did not disclose in the reports. Data such as companies' information (e.g., name, available hard-wares), faults and failures measurements are examples of the data that was kept confidential. Such information is valuable to provide transparency on how researchers ensured the confidentiality of subjects and projects.

4.2.3. Anonymity

In total, 24 papers reported anonymity, in which only 12 described the procedure on how data and subjects are anonymised. ESE25 study reported anonymity for both of its research methods, questionnaire and interview. Three studies (i.e., TOSEM9, ESE13, ESE21) used more than one data collection method. TOSEM9 used questionnaire, experiment and case study research methods but reported anonymity only for its questionnaire. ESE13 conducted the study using observation, interview and questionnaire, and ESE21 recruited experiment and interview research methods. ESE13 and ESE21 reported anonymity for the whole research, when the participants registered for the study.

The studies anonymised the name of the subjects, firms, projects, and subjects' quotes. The studies followed different approaches to anonymise subjects and data. One study, i.e., ESE21, reported that participants chose a self-selected username to anonymise the subjects. ESE13 reported anonymising subjects using the group names (of younger adults) and the registration numbers (of older adults), but did not provide further details. The studies provided several reasoning for anonymising subjects and data. Social desirability bias (ESE25) and feeling of exposure (IST5) are two reasons the studies mentioned as threats that influence the participants' answers. Social desirability bias occurs when the participants adapt their responses to make the researchers happy. The studies anonymised the subject and data to mitigate these threats. Furthermore, the studies mentioned adhering to a non-disclosure agreement (TSE6) and security policies (TSE18) as two other reasons for anonymising subjects and data.

5. Checklist for reporting consent, confidentiality and anonymity in software engineering publications

This section answers RQ2 based on the literature review and workshop results. We discussed the literature review results and the checklist derived from the literature review in a workshop to understand the importance of reporting consent, confidentiality and anonymity.

The checklist we present here covers only ethical issues related to software engineering empirical studies that use human participants directly; in particular, it focuses on informed consent, anonymity and confidentiality. It is the first step towards developing recommendations for reporting ethical issues in publications. Researchers should also consider the overall ethical aspects when designing their studies and publishing their findings. Researchers could consider publishing a pre-study protocol that: 1) justifies the planned sample sizes and the choice of study design (e.g., subject allocation to control situations in experiments), 2) specifies the main hypotheses and analysis procedures to mitigate experimenter and/or researcher bias and 3) explains any blinding methods adopted to conceal design elements from participants, data collectors or analysts. Mechanisms to publish pre-study protocol already exist. For example, there is a Registered Reports Track in Empirical Software Engineering and Measurement (ESEM) in conjunction with EMSE journal, and preregistered papers in Transactions on Software Engineering and Methodology (TOSEM).

5.1. Importance of reporting consent, confidentiality and anonymity in software engineering publications

We conducted a workshop to investigate the importance of reporting consent, confidentiality and anonymity in software engineering publications. In total, 12 researchers participated in the workshop. The participants mainly were assistant professors, one full professor, one PhD student, and senior researchers from research institutes. All workshop participants answered the survey and participated in the group discussions. The active participants in the group discussions had co-authored at least one publication on ethics.

After the introduction to the workshop, individual participants rated the importance of the ethical issues through a survey as presented in Figure 8.

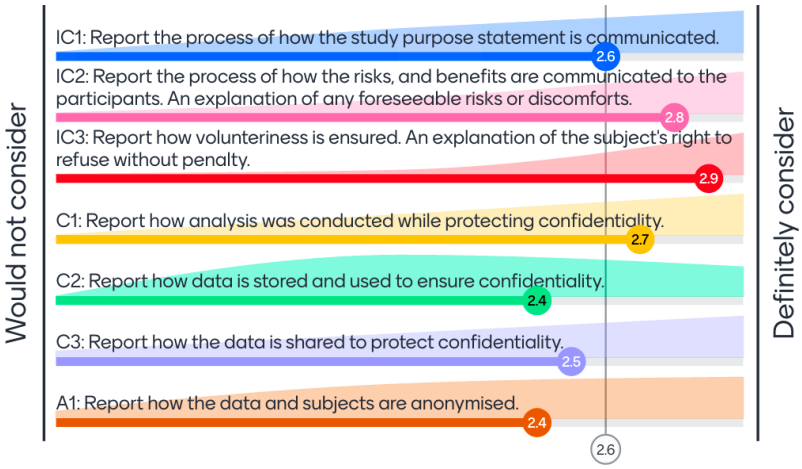


Figure 8. Importance of ethical issues

The ratings were in the range of 2.6±0.3, meaning that the participants considered the ethical issues important. Overall, the participants perceived informed consent reporting slightly more important than other ethical issues. Within informed consent, the voluntariness aspect was rated highest, which has implications for scientific values. Voluntary participation could increase the accountability and trust of the results.

After the survey, we conducted group discussions. The participants were put in two separate Zoom breakout rooms based on their agreement level. As mentioned in Section 3.3,

we named the two breakout rooms as: "reporting ethical details in manuscripts" or "skipping ethical details".

There were only two participants in the "skipping ethical details" zoom room. We summarised the discussion as follows:

- There is no concrete answer to whether researchers should include or skip ethical details in a research publication.
- Depending on the research topics, the researchers should be able to decide the ethical details to report.
- Research ethics value might be too limiting when the researchers discuss a lot of details on the procedure to address ethical issues. It is not important to report everything for the sake of reporting without adding any value to the report.

So, in conclusion, they mentioned that *"there is no clear no to reporting ethical details; the researcher should choose important details for reporting."*

The second group, with 10 participants. We summarised the discussion in "reporting ethical details in manuscripts" breakout room as follows:

- Reporting ethical details are important, e.g., for reviewers of articles to get insight into how the researchers deal with the ethical issues, or for healthcare articles to improve the research reliability.
- Providing a supplementary ethical document attached to a paper, depending on the program committee's permission and confirmation.
- Software engineering committee should research and prepare guidelines for reporting ethical details.
- Not only the ethical rules and regulations should be a matter, but also the rationale behind considering them.

The session was concluded with the message that *"we all think that reporting ethical issues is important, but we are questioning ourselves how much of it should be reported in practice."* We believe our checklist will help in deciding what and how much details on ethical issues should be reported.

5.2. Checklist for authors and reviewers on reporting consent, confidentiality and anonymity in software engineering publications

We aggregated details on reported consent, confidentiality and anonymity and created a consolidated list as shown in Table 5. Table 5 contains attributes of informed consent, confidentiality and anonymity that researchers can report to strengthen the validity of the results. When applicable, the researchers can refer to the documentation they may have used to get ethical boards' approval or to communicate to the subjects, such as consent forms. The checklist includes a description and the importance of reporting consent, confidentiality and anonymity. In addition, we provide examples from the studies that have reported the ethical issues. The checklist is not prescriptive; instead, the researchers should identify the potential risks to the subjects based on the study objectives and decide which attributes they should report to strengthen the results. In addition, the researchers should also report the potential risks to justify the measures taken to address ethical issues.

Table 5. Checklist for reporting the process for obtaining informed consent, achieving confidentiality and anonymity

ID	Attribute	Description of the ethical issue	Example
IC1	Report the process of how the study purpose statement is communicated. For example, how the participants were made aware of what the study involves, its purpose, procedures to be followed, and the likely duration of the subject's participation.	Participants may feel that they are not only being observed but also being evaluated. Hence, the purpose should be clear and the participation duration should be clear so that participants can assess the needed effort and avoid inconveniences such as boredom, frustration, and wasting of time.	IST18 – The interviewer started each interview by explaining the objective of the research to the participants and the importance of giving accurate and honest answers to the validity and reliability of the research.
IC2	Report the process of how the risks and benefits are communicated to the participants. An explanation of any foreseeable benefits and risks or discomforts to subjects.	The risks and benefits of participation should be clear so that true results are obtained. For example, students should be made aware of the impact participation will have on their grades, if any. Different risks include psychological, social, economic, legal, and physical risks. Students and practitioners should know the benefits of participating in the research study. In addition, it is important to discuss the balance of risks and benefits to the subjects.	TOSEM16 – The participants were not evaluated on the results achieved in the experiments. All students… were equally rewarded with one extra point in the exam grade, regardless of their actual performance.
IC3	Report how voluntariness is ensured. An explanation of the subject's right to refuse without penalty.	The participation should be voluntary and free from coercion. For example, students should be able to refuse participation without having any impact on their grades. When students are given credits for study participation, an alternative task should be provided when the students do not want to participate in the study	TOSEM19 – Participation in the study was not mandatory. Nonparticipants were offered an alternative task to earn points equivalent to what they would earn by participating in the study.
C1	Report how analysis was conducted while protecting confidentiality.	Participants should be ensured that their private information is protected and researchers do not reveal the information during data analysis.	IST14 – The responses of the participants were literally transcribed, allowing the destruction of the original material, on the same day of the interviews; in addition, all identifying remarks were perpetually removed and destroyed to protect all the participants.

Table 5 continued

ID	Attribute	Description of the ethical issue	Example
C2	Report how data is stored and used to ensure confidentiality	Participants should be ensured that their private information is kept confidential and researchers have chosen proper storage to record data and do not reveal the information while storing or retrieving.	TSE26 – All raw data collected for this study are kept within the data warehouse servers, which are placed behind the company firewall. Furthermore, The dataset includes only the aggregated statistics for the users covered by our study period. No actual users can be traced at all.
C3	Report how the data is shared to protect confidentiality	Participants should be ensured that their private information is protected and researchers do not share the private information with other organizations and individuals while reporting the research findings.	ESE21 – For confidentiality reasons on the industrial use cases, programs could not be shared among different companies and each hacker team only attacked the program owned by the corresponding company.
C4	Report what data is kept confidential.	Participants should ensure that their private information is protected by understanding which corresponding info the researcher hide and reveal during research analysis and reporting.	IST11 – Access to the Firm's fault data and employees was offered provided that liability issues were considered by not further disclosing the company name or the magnitude of the fault numbers.
A1	Report how the data and subjects are anonymised when needed	Participants should be ensured that their identity and personally identifiable information is publicly kept unknown	ESE12 – All questions were optional, and the survey was anonymous to encourage developers to participate.

6. Discussion

A survey published in 2001 by Hall and Flynn [29], with heads of 44 computer science departments in the UK, indicates that software engineering researchers have little regard for ethical issues when conducting studies with human participants. Only 36% think that monitoring ethical considerations is very important. Hall and Flynn's survey [29] was published 20 years ago. Our workshop study shows that researchers are more enthusiastic about reporting ethical issues. However, our literature review results show that the software engineering community still pays little attention to ethical issues when reporting empirical studies.

It is important to justify ensuring confidentiality and anonymity, as one of the workshop participants mentioned. Researchers should also consider the interactions of ethical aspects with other scientific issues, such as open, transparent research practices (traceability and reproducibility). To make the research reproducible, it is important that all relevant information – such as methodological details on what and how data was collected and analyzed – should be reported. However, the study subjects' anonymity concerns should

also be considered by withholding the information sensitive to the subjects, such as their identities. There seems to be a tradeoff between reporting all versus withholding or anonymizing some parts of the information considered sensitive by the study subjects. However, to replicate a study, the secondary data users may not need all information as long as all necessary methodological details are transparently reported. Thus, we think it is possible to address both concerns by balancing the need to report all necessary methodological details and withholding/anonymizing sensitive data. In addition, the specific context of the study may also result in a different set of tradeoffs depending on, for example, the type of research being reported. Even when researchers do not report details that can identify individuals, they can choose to keep the links between the data and subjects internally if needed for follow-up studies. Confidentiality mechanisms need to be proportionate to possible risks. Researchers need to be aware that they cannot adopt blanket solutions for all studies involving human subjects.

6.1. Comparing the results of the literature review and workshop

Informed Consent: The literature review results show that in the few papers (19/95) that provided some details about informed consent, the most commonly mentioned informed consent aspect was study purpose and procedure. The other two aspects – voluntariness and benefits explanation – were discussed in even less number of studies. The survey respondents, on the other hand, rated all three aspects worth considering to report in their papers. IC3 (Voluntariness) received the participants' highest ranking (2.9/3.0). Voluntariness, however, is covered only in a handful of papers in our sample. Likewise, the risk and benefits explanation also received a high rating from participants – however, that too is not discussed in many papers in our sample. Overall, the survey participants rated all informed consent aspects worth considering for reporting. However, the data from the literature review indicates that these aspects are practically not reported by a majority of the studies in our sample.
Confidentiality: In our sample of papers included in the literature review, only a limited number of papers (20/95) reported some details about confidentiality. The most commonly reported aspect (12/20) is about data storage and usage, followed by how data is shared (8/20). The survey respondents considered all aspects worth considering for reporting. However, the survey ratings for confidentiality related aspects are lower than informed consent ratings.
Anonymity: Anonymity related details was reported in only 12 studies in our sample of studies. In the case of the survey, overall, the participants did rate anonymity as something that should be considered for reporting. However, it received a relatively lower rating as compared to other issues. Half (12/24) of the papers in our sample that reported anonymity did not share any details on how was it done. The authors thought it enough to just report that anonymity is addressed without providing any further details.

6.2. Threats to validity

We used a combination of methods – literature review and a workshop – to investigate reporting of ethical issues in SE research. Our study may still have a few limitations. We use Petersen and Genzel's [30] classification to discuss the threats to the validity of the data collection and analysis phases of our study.

Descriptive validity is concerned with those threats that may happen due to problems in the data collection phase of a study, which may eventually distort the accurate description of the truth. With regards to the literature review, all authors first piloted the data extraction form on a sample of papers. The results of the piloting process were discussed in a joint meeting to ensure that all authors have a shared understanding of the data extraction form and process. The questionnaire of the survey was designed jointly by the first two authors, who were both involved in collecting the data at the workshop.

Interpretive validity is concerned with those threats, such as researchers' bias, that may lead to inaccurate conclusions. To avoid any issues in drawing conclusions, the first two authors jointly performed the analysis, including the coding. Furthermore, the results were presented and discussed in a joint meeting involving all four authors.

Generalizability is concerned about the extent to which the results are applicable to those that are not part of the study. In case of the literature review, our sample is quite small and therefore is not representative of the entire SE literature. Moreover, the results obtained on the reports of ethical considerations in the literature review (and therefore the checklist) are limited to the types of studies found (that is, primarily experiments, interviews and surveys, as see in Figure 2). In addition, we review journal publications only as they provide specific guidelines (see Section 2.2). The results may differ if we consider conference publications as well. Our aim was not to achieve generalizability, but rather to observe how ethical issues are reported in a sample of recent journal articles published at top SE journals. The sample gives an idea about the state of research practice on reporting ethical issues in SE research – that even most of the recent articles in this sample at top quality journals do not report necessary ethical issues appropriately. The workshop participants are also limited in number. In addition, as the participants were attending a workshop on ethics in SE, they were likely more positive towards reporting ethical issues. We think further evaluation of the checklist, involving more SE researchers, is needed in future.

7. Conclusions and future work

Our literature review results based on 95 primary studies indicate limited reporting of consent, anonymity and confidentiality issues in SE publications. The studies included in our sample mostly discussed the process of obtaining informed consent. However, this was limited to informing the subjects on the study purpose and procedure in most cases. We identified different aspects of confidentiality reporting. Most studies discuss the details of storing and using data to maintain confidentiality. Half of the studies that mentioned anonymity did not provide information on how they anonymised.

In the workshop, the participants rated the procedure to: obtain voluntariness, communicate risks and benefits, and analyse to preserve confidentiality as the top three ethical issues to report. However, in our literature review, we observe that the risks of participation and the analysis process to preserve confidentiality were the least discussed aspects.

Finally, we propose a checklist that SE researchers can use to identify the ethical issues related to informed consent, confidentiality and anonymity applicable to their study and consider when reporting their findings.

The proposed checklist is only based on selected empirical software engineering literature. In the future, we plan to compare it with related works from other disciplines (e.g., [31, 32]) as well and see how can we further improve it.

Acknowledgement

We would like to acknowledge that this work was supported by the Knowledge Foundation through the OSIR project (reference number 20190081) at Blekinge Institute of Technology, Sweden.

References

[1] J. Singer and N.G. Vinson, "Ethical issues in empirical studies of software engineering," *IEEE Transactions on Software Engineering*, Vol. 28, No. 12, 2002, pp. 1171–1180.

[2] J.E. Sieber, "Protecting research subjects, employees and researchers: Implications for software engineering," *Empirical Software Engineering*, Vol. 6, No. 4, 2001, pp. 329–341.

[3] M. Jefford and R. Moore, "Improvement of informed consent and the quality of consent documents," *The Lancet Oncology*, Vol. 9, No. 5, 2008, pp. 485–493.

[4] D. Badampudi, "Reporting ethics considerations in software engineering publications," in *2017 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*. IEEE, 2017, pp. 205–210.

[5] T. Coffelt, "Confidentiality and anonymity of participants," in *The SAGE Encyclopedia of Communication Research Methods*, M. Allen, Ed. SAGE Publications, Inc, 2017.

[6] B. Kitchenham, L. Madeyski, and P. Brereton, "Problems with statistical practice in human-centric software engineering experiments," in *Proceedings of the evaluation and assessment on software engineering*, 2019, pp. 134–143.

[7] M. Jørgensen, T. Dybå, K. Liestøl, and D.I. Sjøberg, "Incorrect results in software engineering experiments: How to improve research practices," *Journal of Systems and Software*, Vol. 116, 2016, pp. 133–145.

[8] R. Rosnow and R. Rosenthal, *People studying people: Artifacts and ethics in behavioral research*. WH Freeman, 1997.

[9] V. Prasad and C. Grady, "The misguided ethics of crossover trials," *Contemporary Clinical Trials*, Vol. 37, No. 2, 2014, pp. 167–169.

[10] S. Vegas, C. Apa, and N. Juristo, "Crossover designs in software engineering experiments: Benefits and perils," *IEEE Transactions on Software Engineering*, Vol. 42, No. 2, 2015, pp. 120–135.

[11] K. Xivuri and H. Twinomurinzi, "A systematic review of fairness in artificial intelligence algorithms," in *Responsible AI and Analytics for an Ethical and Inclusive Digitized Society*, D. Dennehy, A. Griva, N. Pouloudi, Y.K. Dwivedi, I. Pappas et al., Eds. Springer International Publishing, 2021, pp. 271–284.

[12] K. Boyd, "Ethical sensitivity in machine learning development," in *Conference Companion Publication of the 2020 Conference on Computer Supported Cooperative Work and Social Computing*, CSCW '20 Companion. Association for Computing Machinery, 2020, pp. 87–92.

[13] B. Zhang, M. Anderljung, L. Kahn, N. Dreksler, M.C. Horowitz et al., "Ethics and governance of artificial intelligence: Evidence from a survey of machine learning researchers," *Journal of Artificial Intelligence Research*, Vol. 71, 2021, pp. 591–666.

[14] D. Spinellis, "The social responsibility of software development," *IEEE Software*, Vol. 34, No. 2, 2017, pp. 4–6.

[15] F.F.S. Flores and S.R.L. de Meira, "Houston, we may have a problem: Results of an exploratory inquiry on software developers' knowledge about codes of ethics," in *International Systems Conference (SysCon)*, 2019, pp. 1–6.

[16] A. McNamara, J. Smith, and E. Murphy-Hill, "Does ACM's code of ethics change ethical decision making in software development?" in *Proceedings of the 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE 2018. Association for Computing Machinery, 2018, pp. 729–733.

[17] T. Ahmed and A. Srivastava, "Understanding and evaluating the behavior of technical users. A study of developer interaction at StackOverflow," *Human-centric Computing and Information Sciences*, Vol. 7, No. 1, 2017, pp. 1–18.

[18] N.G. Vinson and J. Singer, "A practical guide to ethical research involving humans," in *Guide to Advanced Empirical Software Engineering*. Springer, 2008, pp. 229–256.

[19] N.M. Minhas, "Authorship ethics: An overview of research on the state of practice," in *IEEE/ACM 2nd International Workshop on Ethics in Software Engineering Research and Practice (SEthics)*, 2021, pp. 31–38.

[20] N.E. Gold and J. Krinke, "Ethical mining: A case study on msr mining challenges," in *Proceedings of the 17th International Conference on Mining Software Repositories*, MSR '20. Association for Computing Machinery, 2020, pp. 265–276.

[21] P. Strandberg, "Ethical interviews in software engineering," in *ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*. IEEE Computer Society, 2019, pp. 1–11.

[22] S. Baltes and S. Diehl, "Worse than spam: Issues in sampling software developers," in *Proceedings of the 10th ACM/IEEE international symposium on empirical software engineering and measurement*, 2016, pp. 1–6.

[23] A.J. Viera, J.M. Garrett et al., "Understanding interobserver agreement: The kappa statistic," *Fam med*, Vol. 37, No. 5, 2005, pp. 360–363.

[24] S. Elo and H. Kyngäs, "The qualitative content analysis process," *Journal of Advanced Nursing*, Vol. 62, No. 1, 2008, pp. 107–115.

[25] R. Ørngreen and K. Levinsen, "Workshops as a research methodology," *Electronic Journal of E-learning*, Vol. 15, No. 1, 2017, pp. 70–81.

[26] K. Thoring, R. Mueller, and P. Badke-Schaub, "Workshops as a research method: Guidelines for designing and evaluating artifacts through workshops," in *Proceedings of the 53rd Hawaii International Conference on System Sciences*, 2020.

[27] I. Krumpal, "Determinants of social desirability bias in sensitive surveys: A literature review," *Quality and Quantity*, Vol. 47, No. 4, 2013, pp. 2025–2047.

[28] A.D. Ong and D.J. Weiss, "The impact of anonymity on responses to sensitive questions," *Journal of Applied Social Psychology*, Vol. 30, No. 8, 2000, pp. 1691–1708.

[29] T. Hall and V. Flynn, "Ethical issues in software engineering research: A survey of current practice," *Empirical Software Engineering*, Vol. 6, No. 4, 2001, pp. 305–317.

[30] K. Petersen and C. Gencel, "Worldviews, research methods, and their relationship to validity in empirical software engineering research," in *Joint Conference of the 23rd International Workshop on Software Measurement and the 8th International Conference on Software Process and Product Measurement*. IEEE, 2013, pp. 81–89.

[31] F.G. Miller and D.L. Rosenstein, "Reporting of ethical issues in publications of medical research," *The Lancet*, Vol. 360, No. 9342, 2002, pp. 1326–1328.

[32] W.M. Association et al., "World medical association declaration of Helsinki: Ethical principles for medical research involving human subjects," *Jama*, Vol. 310, No. 20, 2013, pp. 2191–2194.

e-Informatica Software Engineering Journal, Volume 16, Issue 1, 2022, pages: 220110, DOI: 10.37190/e-Inf220110

Reuse in Contemporary Software Engineering Practices – An Exploratory Case Study in A Medium-sized Company

Xingru Chen*, Deepika Badampudi*, Muhammad Usman*
*Department of Software Engineering, Blekinge Institute of Technology, SE-37179, Karlskrona, Sweden
xingru.chen@bth.se, deepika.badampudi@bth.se, muhammad.usman@bth.se

Abstract

Background: Software practice is evolving with changing technologies and practices such as InnerSource, DevOps, and microservices. It is important to investigate the impact of contemporary software engineering (SE) practices on software reuse.
Aim: This study aims to characterize software reuse in contemporary SE practices and investigate its implications in terms of costs, benefits, challenges, and potential improvements in a medium-sized company.
Method: We performed an exploratory case study by conducting interviews, group discussions, and reviewing company documentation to investigate software reuse in the context of contemporary SE practices in the case company.
Results: The results indicate that the development for reuse in contemporary SE practices incurs additional coordination, among other costs. Development with reuse led to relatively fewer additional costs and resulted in several benefits such as better product quality and less development and delivery time. Ownership of reusable assets is challenging in contemporary SE practice. InnerSource practices may help mitigate the top perceived challenges: discoverability and ownership of the reusable assets, knowledge sharing and reuse measurement.
Conclusion: Reuse in contemporary SE practices is not without additional costs and challenges. However, the practitioners perceive costs as investments that benefit the company in the long run.

Keywords: software reuse, contemporary SE practices, software reuse costs and benefits, software reuse challenges and improvements, InnerSource

1. Introduction

Software reuse is commonly practiced in organizations and is described as "*the systematic use of existing software assets to construct new or modified ones or products*" [1]. The benefits of software reuse such as improved product quality, faster time-to-market and reduced development costs [1–3] are well-acknowledged. Although software reuse has been studied for more than five decades, with the constant changes in architecture patterns and styles (e.g., microservices), and processes (e.g., InnerSource [4]), the research in software reuse still remains relevant. In 2019, Barros-Justo et al. [5] conducted a tertiary study to investigate the trends in software reuse research. They identified many software reuse

Submitted: 08 Apr. 2022; Revised: 02 Jul. 2022; Accepted: 28 Jul. 2022; Available online: 02 Sep. 2022

research proposals related to 1) requirements engineering, testing, and design activities, 2) evolution/maintenance and variability management in project and process management, and 3) other general reuse topics, such as decision-making based on systematic software reuse and metrics to evaluate the reuse performance. Capilla et al. [6] also identified new software reuse research opportunities in the context of new application domains, new software reuse techniques and methods.

The growing popularity of open source use has also impacted software reuse. Mikkonen and Taivalsaari [7] identified the growing popularity of opportunistic design, which is "*developing new software systems by routinely reusing and combining components (open source components and modules online) that were not designed to be used together*". Xu et al. [8] also identified a trend in increased library reuse of Maven libraries in Maven Central[1]. Although opportunistic reuse is the opposite of systematic reuse as Barros-Justo et al. [2] and Capilla et al. [9] stated, in the long run, there is a need to systematize the maintenance of the external reusable assets from the open source. In addition to the open source software, the open source "way of working" is adopted by various software organizations, e.g., GlobalSoft/SoftCom [10], and IBM (CMS) [11, 12]. Inspired by the open source way of working, Tim O'Reilly coined the term InnerSource [4] as "*the use of open source development techniques within the corporation*" in 2000. Vitharana et al. [12] further conceptualized InnerSource, particularly consumer contributions to reusable assets, as participatory reuse. The authors described participatory reuse as "the scenario in which potential reusers participate in the entire development process (e.g., analysis, design, development, testing) to ensure that the project assets meet their reuse needs."

In addition to the open source and IS practices, the changing technology also impacts software reuse, i.e., the unit of reuse changed from components to microservices. Organizations are increasingly adopting microservices, together with DevOps practices (e.g., continuous integration and deployment) and container-based solutions (e.g., Docker) to improve the delivery time and scalability of their products and systems (cf. [6, 13–15]).

Studies have investigated the impact of opportunistic reuse [9], microservices [16], and InnerSource [17] on software reuse. Capilla et al. [9] found negative impacts of opportunistic reuse on software reuse. Their results indicate that the integration of reusable assets found opportunistically increases the number of smells and issues in most cases. Gouigoux and Tamzalit [16] identified increased reuse as one of the main benefits of migrating from monolith to microservices based architecture solutions. InnerSource (IS) practices facilitate software reuse [17]. When consumers of reusable assets also participate in developing and maintaining the reused assets, it further promotes reuse. The above mentioned software engineering (SE) practices can be referred to as contemporary SE practices.

While studies investigate the impact of individual contemporary SE practices on software reuse, there is no empirical investigation of software reuse in a combination of contemporary SE practices. We refer to *software reuse in contemporary SE practices* as organizations practicing both opportunistic reuse – leveraging open-source assets and libraries wherever possible, and participatory reuse with the help of IS practices for collaboratively developing reusable assets, together with the adoption of new technical solutions such as microservices-based architectures and DevOps practices.

It is important to investigate if the previously well-known challenges of software reuse are still applicable in the context of contemporary SE practices and discover the new implications of software reuse. For example, when developing reusable assets in participatory

[1]Statistics for the Maven Repository, https://search.maven.org/stats

reuse, additional coordination may be required when accepting contributions from other teams. Furthermore, other teams may need additional documentation to understand how they can contribute. The ownership of the reusable assets can be complicated when different teams are involved in development. Opportunistic reuse involves additional integration effort due to differences in the architectural style and technology in the target project and the reusable asset from open source [9].

Barros-Justo et al. [18] pointed out the empirical evidence of software reuse in practice, particularly in medium-sized companies, is limited. Also, existing systematic literature studies on software reuse highlight the need for more empirical studies [1–3]. Therefore, we contribute by investigating the state-of-the-practice of software reuse in the context of contemporary SE practices in S-Group Solutions AB[2], a medium-sized Swedish IT company.

In our study we characterize software reuse in contemporary SE practices. We conducted an exploratory case study to investigate how practitioners practice software reuse with contemporary SE practices and how they perceive its costs, benefits, challenges, and improvements. We collected the data using in-depth interviews, group discussions and document analysis. Reduced time and improved product quality are the main identified benefits. The study participants perceived additional coordination with stakeholders as a cost in both, development and use of a reusable asset. The study participants identified discoverability and ownership of the reusable assets, knowledge sharing and reuse measurement as the top focused challenges and improvement areas. The participants were in consensus on adopting IS patterns[3] in order to address the top listed challenges and improvement areas.

The remainder of the paper is structured as follows: Section 2 presents the related work; Section 3 describes the study design; Section 4 provides the study results; Section 5 discusses the results in comparison to the related works and provides discussion on threats to validity; Section 6 concludes the paper and proposes the future work.

2. Related work

Companies adopt software reuse practices to achieve certain benefits (e.g., better productivity), which leads to additional costs. Likewise, adopting software reuse practices also results in some challenges, which researchers try to solve by proposing improvement suggestions. This section will present an overview of the related works on software reuse costs, benefits, challenges, and improvements.

Many studies identified increased development productivity and better product quality as software reuse benefits, which includes both internal [19–21] and opportunistic reuse [19, 22, 23]. Furthermore, less maintenance effort [18, 19, 23], standardized architecture [21] and higher documentation quality [18] have also been identified as benefits of software reuse.

Relatively fewer studies investigated software reuse costs than benefits. Kruger and Berger [20] discovered that the majority of the additional reuse costs relate to the development for reuse phase. They noted that developing assets for reuse is generally more costly than developing for single use. However, Mohagheghi et al. [21] investigated the relation

[2]https://sgroup-solutions.se/
[3]https://patterns.innersourcecommons.org/

between software reuse and increased rework and did not find a cause-effect relationship between them.

The implementation of the software reuse initiative is not without challenges. Barros-Justo et al. [18] replicated Bauer et al.'s study [19], investigating the challenges and problems related to software reuse practices. Both studies [18, 19] identified the same software reuse related challenges including licensing issues, "not invented here" syndrome, inadequate granularity of reusable assets, accessibility of reusable assets, decrease of code understandability and difficulties in modifying the code due to software reuse. Mäkitalo et al. [23] and Barros-Justo et al. [18] also found that fixing compatibility and dependency related issues is particularly challenging in case of reusable assets. In addition to these technical challenges, coordination among the teams working on the development of the reusable assets is also a challenge [20].

Some improvement suggestions have also been proposed in the literature to address the challenges associated with the development of reusable assets. For example, using a written reuse guidebook to improve the understandability of the reusable assets [24], tools to help improve the discoverability of the reusable assets [24, 25] and allocating developers a separate time budget to develop or maintain the reusable assets [24].

Barros-Justo et al. [18] pointed out that few empirical studies on software reuse exist in small to medium-sized companies, therefore, they conducted a survey study in a medium-sized company to fill the gap. Our study further contributes to the medium-sized company context. We conducted an exploratory case study to cover the topic in more depth, using interviews and group discussions. In our study, we collected data about software reuse costs and benefits as well as about reuse related challenges and improvements in the context of contemporary SE practices. Moreover, we also discussed the feasibility of adopting selected IS patterns to address the identified challenges and improvement areas.

3. Study design

This section presents the details of the study design.

3.1. Research method

The study is part of a research project aimed at improving the internal reuse practices of the partner companies. In a joint discussion involving both company and research team members, it was decided to start with an initial study to understand the current state-of-the-reuse practice at the case company. We chose an exploratory case study [26] as our research method to investigate the current reuse practice in the company. We used three data collection methods (see Section 3.4 and Table 1): interviews, group discussions and company documentation.

3.2. Research questions

We formulated the following three research questions to guide our study:
RQ1: How software reuse is conducted in the context of contemporary SE practices in a medium-sized company? Motivation: To understand the software reuse strategies in contemporary SE practices, we aim to characterize the reuse process. We investigated the company's reuse related activities, roles and workflows.

RQ2: What are the costs and benefits of practicing software reuse in the context of contemporary SE practices in a medium-sized company? Motivation: To understand practitioners' perceptions of software reuse costs and benefits in the context of contemporary SE practices. Cost is the extra/additional effort required to develop, maintain or use the reusable assets.
RQ3: What are the challenges in practicing software reuse in the context of contemporary SE practices in a medium-sized company, and how can they be improved? Motivation: To understand what challenges, issues or problems the practitioners in a medium-sized company encountered in software reuse in the context of contemporary SE practices. To collect the improvements from the practitioners view and discuss other possible interventions with practitioners that can facilitate software reuse.

3.3. Case company and unit of analysis

The case company, S-Group Solutions AB, is a private Swedish IT company that focuses on developing spatial information and geographical information systems (GIS). The company offers its solutions to the public sector and the target customers are mainly local governments and authorities. S-Group Solutions AB has 65 employees and can therefore be classified as a medium-sized company [27]. The software development organization of the company consists of 29 people and it is divided into three teams corresponding to four solution areas. Each development team consists of an average of five developers each, with one senior developer acting as the tech lead. Each solution area has a corresponding project manager, a product owner and a tester. The development organization has one software architect who oversees and guides all teams and is responsible for maintaining the integrity of the overall software design and architecture. In addition, the company also has a support team, a UX engineer and a technical writer. The development teams follow agile practices (e.g., daily standup and sprint planning) to manage their work. S-Group Solutions AB uses Azure DevOps and has continuous integration and delivery (CI/CD) pipeline, which updates every midnight. Currently, S-Group Solutions AB is migrating some codes from a monolithic architecture to a microservices-based architecture. The unit of analysis is the software reuse practice at the case company. Currently, two of the three teams are more involved in the development and use of the reusable assets, while the other team is relatively new in this reuse journey.

3.4. Data collection

The data is collected through semi-structured interviews, multiple group discussions and company documentation. The aim of each data collection method and its corresponding research questions (RQs) are presented in Table 1. We used group discussions and the company documentation to validate and triangulate the interview data (see aims in Table 1). The software architect was our contact person at the case company, who has a long working experience (12 years) at the company and has a leading role in introducing the software reuse related practices. We used semi-structured interviews to collect data since it allows improvisation and exploration of the studied objects [26] and captures unexpected information on the studied topic [28]. The group discussions are used since it collects in-depth perspectives through interactive conversations with multiple participants, not only with the moderator/interviewer as interviews do. Allowing multiple opinions provides more descriptive and elaborated data.

Table 1. Aims and corresponding research questions of the data collection methods

Data collection methods	Aims	RQ
Interviews	Understand how software reuse is practiced in the company, and collect the practitioners' perceptions on software reuse costs, benefits, challenges and improvements	RQ1, RQ2, RQ3
Group discussions	1) Validate the interview results, and get additional inputs	RQ1, RQ2, RQ3
	2) Collect the challenge prioritization results from the company's perspective 3) Discuss the feasibility and application of the interventions which were proposed by the authors and collect the feedback from the company	RQ3
Company documentation	Understand the company structure in a written form and triangulate it with the interview results	RQ1

Selection of the interview participants. To select the right person as participant candidates, we shared with the software architect a list of candidate roles related to software reuse at the company, including producers and consumers of the reusable assets and their managers and team leads. The software architect helped us identify four participants initially – the software architect himself, two tech leads (representing two different teams) involved in the development and consumption of reusable assets and one product owner. During the interviews with these four participants, we also identified the need to cover the role of a tester and a project manager. With the help of the software architect, we managed to interview one tester and one project manager. Table 2 shows the summary of the participants. In total, we have interviewed 20 percent of the population (6 out of 29), which covers all teams, and both technical and non-technical roles.

Table 2. Overview of the interview participants

PID	Team	Current role	Exp[a]	Interview duration
P1	Team 2	Product owner (PO)	27	1h10mins
P2	Team 1	System developer and tech lead (TL1)	3	1h10mins
P3	Team 3	System developer and tech lead (TL2)	7	1h30mins
P4	All teams	Software architect (SA)	12	1h20mins
P5	Team 1	Tester (T)	4	55mins
P6	Team 1	Project manager (PM)	6	1h

[a] Experience in number of years the practitioner is working with the current company.

Interview design. As mentioned in Section 3.1, we chose to conduct semi-structured interviews. The second author developed the interview guide (see Table 3) and it was reviewed independently by the other two authors and a senior researcher from the research project, which resulted in minor reformulations. We also performed a pilot interview with a practitioner from another company to test the interview guide. The interview guide contains seven aspects: introduction, participants' background, reuse practices, costs, benefits, challenges and improvements. The mapping between the interview questions and RQs are presented in Table 3. We used the interview questions as a guide and followed semi-structured interview format which allowed for flexibility in the interview.

Table 3. Mapping between the interview* questions and RQs

Interview questions	Corresponding RQs
1. Overview of the current reuse practices and your role a) Details on role and experience: i: What is your role? Please provide a short overview of the tasks that you perform in your role. ii: Which other roles do you interact with and why? iii: What is your overall experience and experience with development for and with reuse?	Demographic information
b) Details on the product, team/s and shared assets: i: What is the size of your team? ii: Which software artefacts (requirements, test cases, code, etc.) you work with? Format of requirements? iii: Do you prefer development of assets from scratch or reuse? existing/available assets? What is the motivation behind your preference? iv: Which software assets do you/your team share (across site?)? What solution do they offer? Can you give an example? v: Do you produce and/or consume the shared assets? Give examples of the shared assets your are involved in?	RQ1
2. Is there a company/project/unit wide strategy/policy/goal to develop with reuse?	RQ1
3. Is there a company/project/unit wide strategy/policy/goal to develop for reuse (i.e., developing assets, e.g., code, with the aim to make them reusable)?	RQ1
4. How is the funding of shared assets done?	RQ1
5. What is your experience regarding developing for and with reuse? What reuse[4] related activities/tasks/initiatives are you involved in?	RQ1
6. What activities, if any, are performed in your company to: a) Identify the reusable assets b) Develop/adapt reusable assets. i: What are the unique activities in development of reusable assets? c) Use reusable assets or replace existing assets with reusable assets. d) Maintain reusable assets. e) Share reusable assets or make reusable assets available. i:[–] **For all activities ask the following questions:** ii: Is there anyone response for this activity? If yes, who? If not, should there be any one responsible? iii: Who or what triggers this activity and how often? iv: What information/input is needed for the activity? v: Who provides the information needed for the activity or how is it obtained?	RQ1
7. Benefits of development for reuse and with reuse (what are the reuse benefits and how are they measured?) How reuse benefits – i: the organization, ii: your role (**incentive**), iii: the product, iv: business/customers, v: the team?	RQ2
8. Costs of development for reuse and with reuse (what are the reuse costs and how are they measured?) How reuse costs affect – i: the organization, ii: your role, iii: the product, iv: business/customers, v: the team?	RQ2
9. What are the challenges and improvement areas with respect to development for and with reuse?	RQ3

* In this interview, we want to know your view on software reuse. In particular, we want to know your experiences of developing and/or using reusable assets. Assets include components, microservices, APIs etc. developed either in-house or acquired from open source projects that could be reused within the company.

Due to the Covid-19 pandemic, we conducted the interviews online using Microsoft Teams and the interview duration was set to approximately one and half hours. To provide

[4]Reuse could also mean using the same open source component that someone else at the company has adopted.

some context and background information, we shared high-level interview questions with the participants before the interviews. The authors distributed their tasks mainly in three parts during the interview: lead the interview, ask follow-up questions and take notes. All three authors were involved in all the parts by switching their tasks in different interviews. We requested participants' permissions to audio record the interviews.We guaranteed that the data will only be stored in a local drive and will only be used in an aggregate form during the analysis and presentation of results. All interview participants gave their consent for audio recording of the interviews.

Selection of the group discussions participants. As described previously, the study is part of a research project. Keeping in view the relevance of the topics covered in the study and the long-term goals of the project, we formed a discussion group to take the study and the project forward. The discussion group consists of five members – including two participants from the case company – the software architect and the project manager – and the authors representing the project's research team. The software architect has a leading role in software reuse practice in the company and interacts with all development teams about the technical issues. Therefore, his involvement was necessary for the study and project. The project manager's role is also important as he is responsible for planning and managing the more active teams in developing and maintaining the reusable assets. Including the project manager ensures coverage of project management and planning-related perspectives in the discussions.

Group discussions design and company documentation review. Similar to the interviews, group discussions were also held online through Microsoft Teams due to the Covid-19 pandemic. We conducted discussions to validate interview results, prioritize the challenges, and discuss the potential improvements to address the prioritized challenges. In the group discussion on validating interview results, we presented the interview results to the company contact person. The results of the interview data validation are provided in Section 3.5. Prior to the next group discussion, we asked the company contact person to conduct an internal discussion with the team members to prioritize the challenges identified in the interviews. In addition, we investigated the possible solutions for the identified challenges from the existing literature. Then in a group discussion, the company contact mentioned the prioritized challenges (reported in Section 4.3.3). In the same group discussion meeting, we provided potential solutions to mitigate the prioritized challenges. We then discussed the feasibility of the proposed solutions with the contact person and the project manager (see Section 4.3.3). The project manager provided company documentation that included information about the people involved in reuse practices, reuse context and the reuse activities, which we used to triangulate the interview results. On average, the discussions lasted for an hour. The research team took extensive notes during the discussions. The discussions concluded with one of the authors sharing the summary and confirming the next steps (e.g., who is expected to do what before the next group discussion).

3.5. Interview data analysis approach

To enable the data analysis, the first author transcribed word to word of approximately seven hours of audio recordings from the interviews. The other two authors did a preliminary analysis by extracting and analyzing the relevant data from the transcripts. The authors held a joint meeting to discuss the interview credibility, the transcription quality and the preliminary analysis findings. At the end of the discussion, we reached a consensus on the findings and agreed that all six interviews are eligible for the study. We presented the

results from the preliminary analysis to the software architect. Apart from a few minor corrections, the software architect agreed and was able to relate to the results.

Taking inspiration from the recommended thematic synthesis steps [29] and four-steps data analysis process [30], we used the following integrated approach (both inductive and deductive approaches) to code and analyze our data. The entire data analysis process is described as follows:

Generating start list and clustering. We created a general start list for clustering according to the research questions and context information. The start list acted as a preliminary theme to group the raw data according to the general domain instead of content-specific, enabling inductive coding. The start list contained seven aspects: personal background, reuse context, reuse activities, reuse costs, reuse benefits, reuse challenges and reuse improvements. The first author extracted the relevant text segments from the transcripts and grouped them according to the generated start list.

Inductive coding within clusters. We followed the descriptive coding method according to Saldana's coding manual book [31]. The first and second authors agreed on a code naming style and then they independently coded the text segments from the reuse benefits cluster to pilot the code naming style. During the coding process, more text segments from the transcripts were extracted when needed. The piloting results showed that we were consistent in the code meanings and the corresponding text segments. However, we needed to agree on a common name for each code. For example, we named the same text segments "save time" and "reduced development time". We eventually used "reduced development time" and agreed that the code names should be more explicit. After the piloting, the two authors independently coded clusters related to costs, challenges and improvements, and arranged several meetings to address the disagreements. Apart from refining the code names, the coding results showed that the first and second authors had similar opinions on the codes and text segments. We merged two codes into one for the benefits cluster related to maintenance, and merged another two codes into one for the costs cluster related to team coordination. To enhance comprehension, we also cleaned the text segments jointly. All the changes are logged to ensure code traceability. A codebook was generated when the two authors reached a consensus on the codes and code descriptions. Using the codebook as a reference, the first author went through all six transcripts again to ensure we did not miss any relevant text segments. The first author extracted 15 new text segments, which resulted in two code modifications. When there was no disagreement between the first and second authors, they asked the third author for review. The review contains four parts: the suitableness of the unit (broad or brief) of the extracted text segments, the relevance between the codes and the text segments, the coverage of the codes and the clarity of the code descriptions. We discussed the review results and addressed the disagreements in a joint meeting among all authors. In the review process, the third author suggested the removal of one code for benefits and three codes for challenges due to their low relevance to software reuse. The relevance of these codes was discussed jointly and all three authors agreed to remove them. In addition, we further refined the code names and extracted additional text segments according to the review suggestions.

Translate codes into themes. We used pattern coding [31] to generate themes according to the codes relations and thematic map to visualize and organize the codes and themes. The first author independently came up with the themes for four clusters. Then the second and third authors individually reviewed the appropriateness of the themes. The disagreements were addressed iteratively through multiple discussions.

Validation of the interview findings. To reduce the risk of researchers' bias in data interpretation, we shared the study report with the company contact person for review. The contact person commented that one of the challenges was about technical incompatibility than a challenge caused by software reuse. Therefore, we removed this challenge from our results. Overall, the contact person confirmed that the results reflected the reality. In addition, the results were also presented, discussed and agreed upon in one of the group discussions.

4. Results

This section presents the results per research question.

4.1. RQ 1. Reuse practice and contemporary SE practices in a medium-sized company

The software reuse process in S-Group Solutions AB consists of both participatory reuse and opportunistic reuse, as presented in Figure 1. The potential reusable assets at the case company are code (packages and microservices), requirements and automated test cases. To better explain how software reuse is practiced in the contemporary software engineering practices environment, we characterize different reuse activities in participatory reuse and opportunistic reuse.

In participatory reuse (see the solid-lined box on top in Figure 1), the development teams that use the reusable assets also participate in the development process. The software architect, product owners, project managers and tech leads conduct a solution vision meeting to propose reusable candidates before the project initiates and share the knowledge among the teams. To facilitate the communication across different solution areas, people in the same role sit together, i.e., testers at one place and product owners at one place, which also facilitates exploring potential reuse opportunities across different teams. After identifying the internal reusable candidates in solution vision, the software architect and tech leads perform technical analysis to discuss overall design, such as the API design. Once the technical analysis is completed, the relevant team develops the reusable code assets themselves. When the consumers or other developers want to contribute to the internal reusable code assets, they need to coordinate with the owner of the reusable assets to align the needs from both sides (consumers and producers) and understand the code commit requirements. There are two types of contributions: One is to add new functionalities to the reusable assets and the other is to fix the bugs in the reusable assets. The reusable assets include npm packages[5], NuGet packages[6] and microservices, which are stored in the internal DevOps repository (Azure DevOps server). The internal DevOps repository has the capability of keyword searching, which helps to look up the shared reusable assets. For the most reused packages, a read-me file provides a short description of the package. All developers are potential producers and consumers of the shared reusable assets, i.e., all developers within the company can reuse the shared packages, and if the consumers identify the need to fix or add something, they need to do that on their own. After the update, a new version number is assigned to the revised package. The case company has reached nearly 40% of the reuse rate, as a ratio of reusable assets from previous projects and the

[5]https://www.npmjs.com/
[6]https://www.nuget.org/

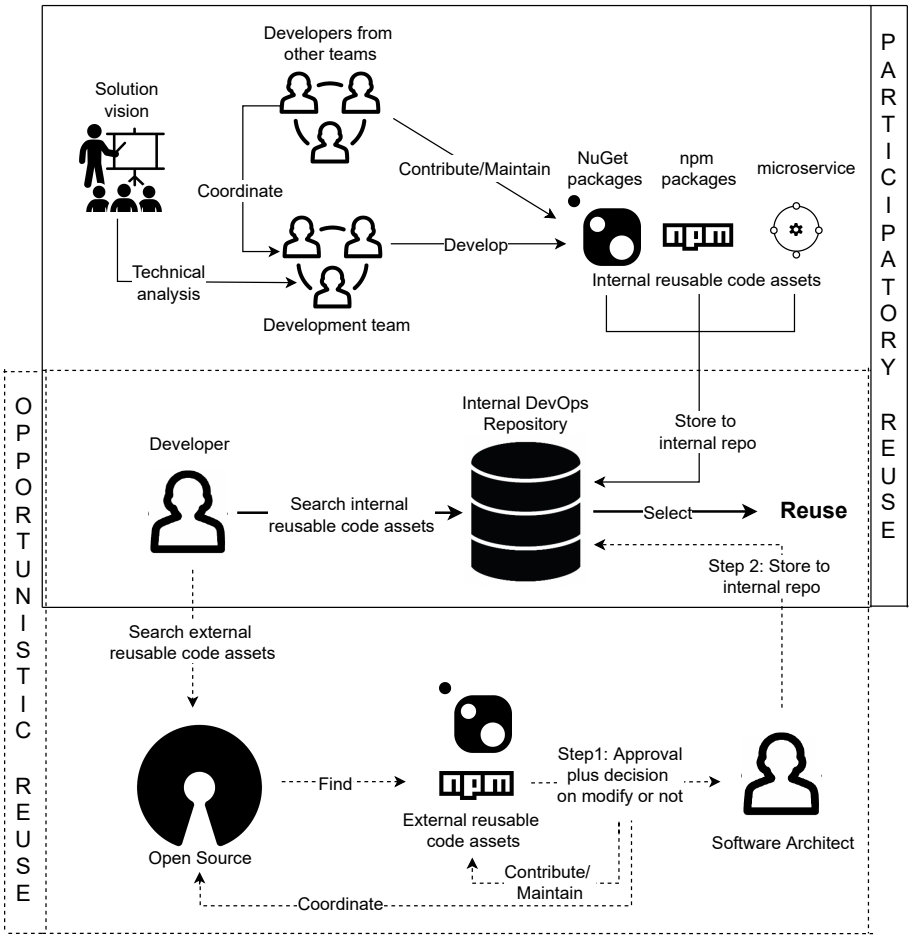


Figure 1. Software reuse and contemporary SE practices in S-Group Solutions AB

newly developed project. The company aims to increase the participatory reuse since it supports the development and maintenance of the reusable assets and also facilitates the company-wide adoption of the reuse practices.

In opportunistic reuse (see the dotted-lined box at bottom in Figure 1), the company reuses from the open-source community. However, developers need to get approval from the software architect before importing the external reusable assets into the internal DevOps repository and reusing them. The approval includes risk analysis, such as the fit to purpose check and the associated community activeness check. The developers need to coordinate with the open source community for bug fixes and new feature requests. They also make upstream contributions. If the open source maintainers do not respond in time, the software architect and developers need to decide whether to modify the external reusable code assets themselves and store the modified ones into the internal DevOps repository (provided that the OSS license permits). The opportunity reuse in the company is limited in package reuse only.

According to the software reuse process description in S-Group Solutions AB, InnerSource is practiced by accepting other teams' developers participation in the development and maintenance of the reusable assets. All code is stored and shared organization-wide in the internal DevOps repository, expect for some sensitive code which is kept private within the team that developed it.

RQ1 – Summary. We characterize the reuse in contemporary SE practices in S-group Solutions AB, which follows both participatory reuse and opportunistic reuse. All reusable assets are managed in the internal DevOps repository. Developers from different teams collaborate in developing and maintaining the internal reusable assets. Developers also retrieve external reusable assets from Open Source community and make upstream contributions for bug fixes or maintain the assets locally.

4.2. RQ2. Software reuse costs and benefits

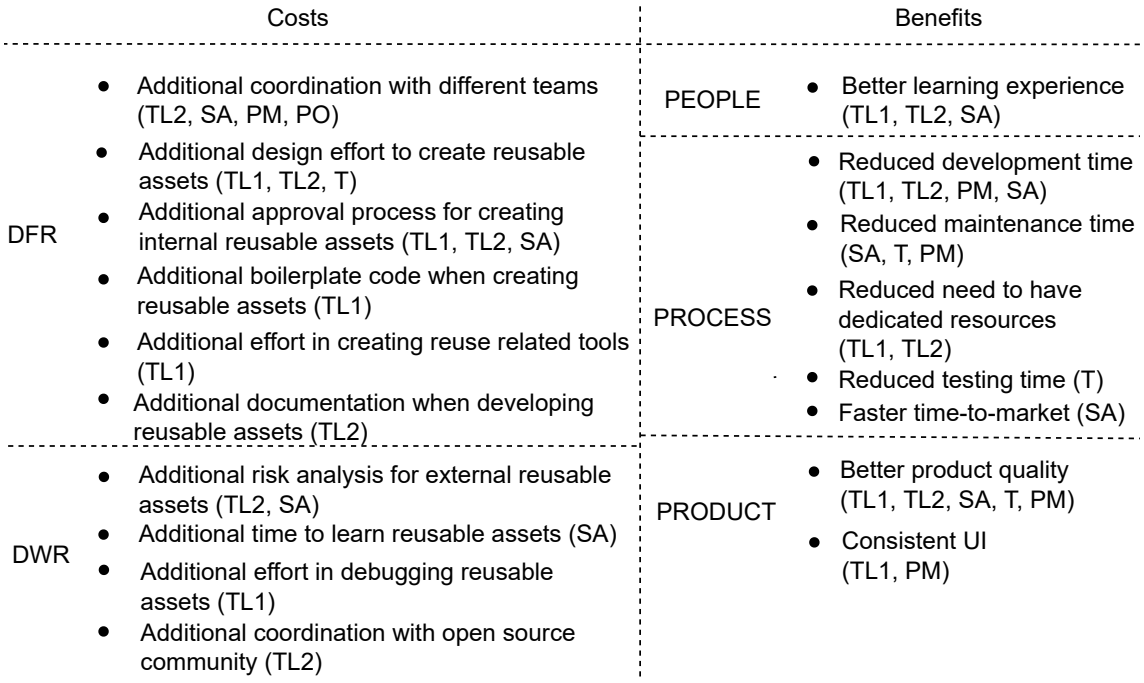


Figure 2. Costs and benefits of the software reuse in the context of contemporary SE practices

Software reuse includes an upfront cost in creating reusable assets, which pays off when reusable assets are integrated in new solutions. Figure 2 provides the classifications of the practitioner perceived costs and benefits, mapping with the participants by the role abbreviations (see Table 2). The listed codes follow the order of their coverage among the participants – from more to less common, reflecting which costs and benefits are considered relevant by different study participants. Sections 4.2.1 and 4.2.2 discuss the reuse costs and benefits perceived by the participants, respectively.

4.2.1. Reuse costs perceived by the participants

This section describes the identified costs related to development for reuse (DFR) and development with reuse (DWR). DFR contains all activities for "creating, acquiring or re-engineering reusable assets", while DWR contains all activities for "using reusable assets in the creation of new software products" [32]. We identified six costs in DFR and four costs in DWR.

In theme DFR, the software reuse costs are as follows:

1. **Additional coordination with different teams** is perceived as a cost by four participants (one of the tech leads, the software architect, the project manager and the product owner). The producers and consumers need additional synchronization and communication to develop or maintain the internal reusable assets, especially in participatory reuse. The software architect described it as "*developing something that fits a few other people, you have to take their needs into consideration and integrate that into the specific product*" and it usually "*ends up in a prioritize discussion*" for the purpose of matching release time as described by the product owner.
2. When creating reusable code assets, **additional design effort to create reusable assets** is needed to make reusable assets easy to use, less error-prone and avoid breaking changes. Such cost is reflected in technical analysis process in the participatory reuse followed in the company (see Figure 1), which was shared by two tech leads and the tester. The additional design effort is not limited to reusable code assets but also reusable automated test cases. One tech lead described it as "*usually you care more about the design of the (reusable) component. But as soon as it is common, you need to design it better so that it is easier for other teams to use as well.*". And the tester shared that creating auto test using page object "*adds small overhead in the short term but will probably be time-saving in the long term.*"
3. **T**hree participants mentioned that additional approval process for creating internal reusable assets is needed before the implementation. The product owner approves the reusable asset functionality, and the project leader approves from the workload and time perspective. One tech lead said reusable microservices need to "*go through, from the product owner, project leader, all of that, before you create the (reusable) microservices*". The additional approval process is part of the solution vision activity in the participatory reuse (see Figure 1).
4. When reusing, some technical problems might constraint the developers from efficient reuse and they need to take additional actions to achieve reuse. One tech lead discussed the cost that developers have to write **additional boilerplate code when creating reusable assets**. He described this cost as "*every time we need to create a new package to address a lot of boilerplate code that we need to implement*". Boilerplate code is code that is copy-and-pasted without modification, e.g., the definitions of getting and setting instance variables method in object-oriented programs.
5. Tools can help in promoting reuse. However, if the developers need to **create the reuse related tools** themselves then it involves **additional effort**. One of the developers added – "*create stuff (reuse tool) that is easy for reuse requires additional effort. However, it can only take a very little time in the long run. It will take some time in the beginning to set everything up and to get it working.*"
6. **Additional documentation when developing reusable assets** is brought up by one tech lead and he described it as "*if you develop some reusable components, you try to add more documentation, describing what component is, so the developers who reuse it will understand.*"

In theme DWR, the software reuse costs are as follows:

1. One tech lead and the software architect pointed out **additional risk analysis for external reusable assets** in opportunistic reuse. To acquire an external reusable asset, the tech lead said that "*when you have some package candidates, you need to check (if they fit) requirements, you need to do like prototyping and testing*". The software architect added that "*every time we choose to do something like picking a new open-source framework or open-source tool, it has to go through that process*

(risk analysis) where it goes through a few lines within the company to assure that, for example, licenses, agreements actually meets the terms for including this in the product and so on."

2. The software architect highlighted the cost of **additional time to learn reusable assets**, and he said that "*understanding is going to take a bit longer if you are not familiar with that specific project or that specific component.*"
3. One tech lead pointed out the cost of **additional effort in debugging reusable assets**. The debugging process jumps over the reused code and developers have to copy the source code into the project to enable the debugging process. He described this cost as "*it is kind of a little bit of a hassle to debug that, because you need to remove the package and use the actual source code as a reference instead*".
4. **Additional coordination with open source community** is a cost in opportunistic reuse. One of the tech lead explained that "*If it is a new bug (in the external reusable components), then sometimes you need to request for the fix and sometimes it is a problem because you need to wait for such fix or if possible you need to do some workarounds.*"

4.2.2. Reuse benefits perceived by the participants

Although the participants pointed out costs in both DFR and DWR, most of them stated that the benefits outweigh the costs. We identified eight benefits of software reuse in the case company (see Figure 2), which were classified into three themes according to the context facets [33]: people, process and product.

The software reuse benefits the engineers that are involved in the development, integration and maintenance of the reusable assets. In the people theme, we identified **better learning experience** as a reuse benefit. The software architect shared better learning experience as a benefits for the developers that are involved in software reuse. Such benefit is gained from the additional time that developers spend in learning reusable assets. During the learning, the developers will understand what the reusable assets are about and how they were built. A well-designed reusable asset will help developers grasp knowledge faster than development from scratch. The software architect perceives the value in "*getting a much broader understanding of things*" and "*learning much faster than doing it all by yourself*". Such knowledge gaining will also help the company develop better-skilled teams.

In addition to the above benefit for people, the participants also shared the following process related benefits of practicing software reuse:

1. As a result of the additional costs in the DFR, reusing the assets **reduces time in development, maintenance, testing and delivery** (see first four codes for theme PROCESS in Figure 2). The software architect highlighted that software reuse helps "*faster time-to-market*" since developers do not need to develop everything. The project manager, the software architect and two tech leads perceive the main benefit of reusing software is that "*it will save a lot of time instead of we have it (the code) from scratch*", namely, reduced development time. Meanwhile, as a result of reuse, changes or fixes can be propagated easily, which helps reduce the maintenance time. One tech lead said software reuse "*gain (benefits) from a maintainability point of view where you can fix things in one place and reflect all over the entire product*". On the other hand, the tester added the reusable code requires less testing effort and described it as "*when developers reuse stuff, because they reuse something that we know how good quality is, we do not have to spend the same amount of time on testing that specific code once*

again." Overall, the company can benefit from the time-saving perspective and be more competitive in the market.

2. Two tech leads highlighted the benefit of **reduced need for dedicated resources** for consumers because they can rely on the producers of the reusable assets, who have competence in a particular area. One of them described that it is "*a good thing that if it (reusable asset) is more specific area, all the developers do not need to learn such area. So other teams (consumers), they just reuse with such kind of component.*" Teams, even the company could benefit from the reduced resources and further reduce the costs.

Lastly, we also identified the following product related benefits of practicing software reuse:

1. Due to the careful design in producing reusable assets and evolution after several reuses, **better product quality** is discussed by five participants. They said that the software with more reused content has better quality (reduced defects, bugs, deficiencies) as "*such kind of components (reusable components), they are more or less tested. And they contain less bugs than in the components we just developed*". Good product quality could gain reputation for the company and increase the competitiveness.
2. Two participants mentioned benefits in **consistent UI**, however, from different perspectives. One tech lead emphasized the company brand value because of the **consistent UI** and said "*we will share the same, maybe header, sidebars, dashboard, so the users or the customers will recognize our product by whichever application they are using.*" The project manager also mentioned a direct benefit to the customers, i.e., **a consistent UI** leading to a consistent user experience for the customers across different products and modules from the case company, and he described this benefit as "*if you reuse a component that has UI artifacts, it will also look and feel the same and work the same way. You can help create consistency in our UIs.*".

RQ2 – Summary. Costs in DFR result mostly from designing, developing, coordinating for creating reusable assets and their documentation. However, it pays off when developers start to reuse more. Costs in DWR result from learning, analysing, and coordinating for using the reusable assets. The main benefits of software reuse are related to time-saving, better product quality and improved learning experience.

4.3. RQ3. Software reuse challenges and improvements

From interview, the participants also shared the challenges they face in practicing software reuse in the context of contemporary SE practices and the improvements they would like to implement. In total, we identified 14 challenges, which are divided into the following two groups.

– The challenges with improvement suggestions: In this group there are five challenges for which the participants also shared some improvement suggestions (see Section 4.3.1 for details).
– The challenges without improvement suggestions: In this group there are nine challenges, without any specific improvement suggestions by the study participants (see Section 4.3.2 for details).

In addition to the challenges above, we also identified three improvement suggestions (generic improvements) that could not be mapped to any of the challenges, which are described at the end of Section 4.3.1.

In the group discussions, the software architect and project manager prioritized the challenges based on the company's needs in the group discussions. We proposed IS related improvements to address the top concerning challenges, namely discoverability and ownership of reusable assets, knowledge sharing and reuse measurement. The prioritized challenges and IS related improvements that the authors suggested are described in Section 4.3.3.

Figure 3 provides the classifications of the practitioner perceived challenges and improvements, mapping with the participants by the role abbreviations (see Table 2).

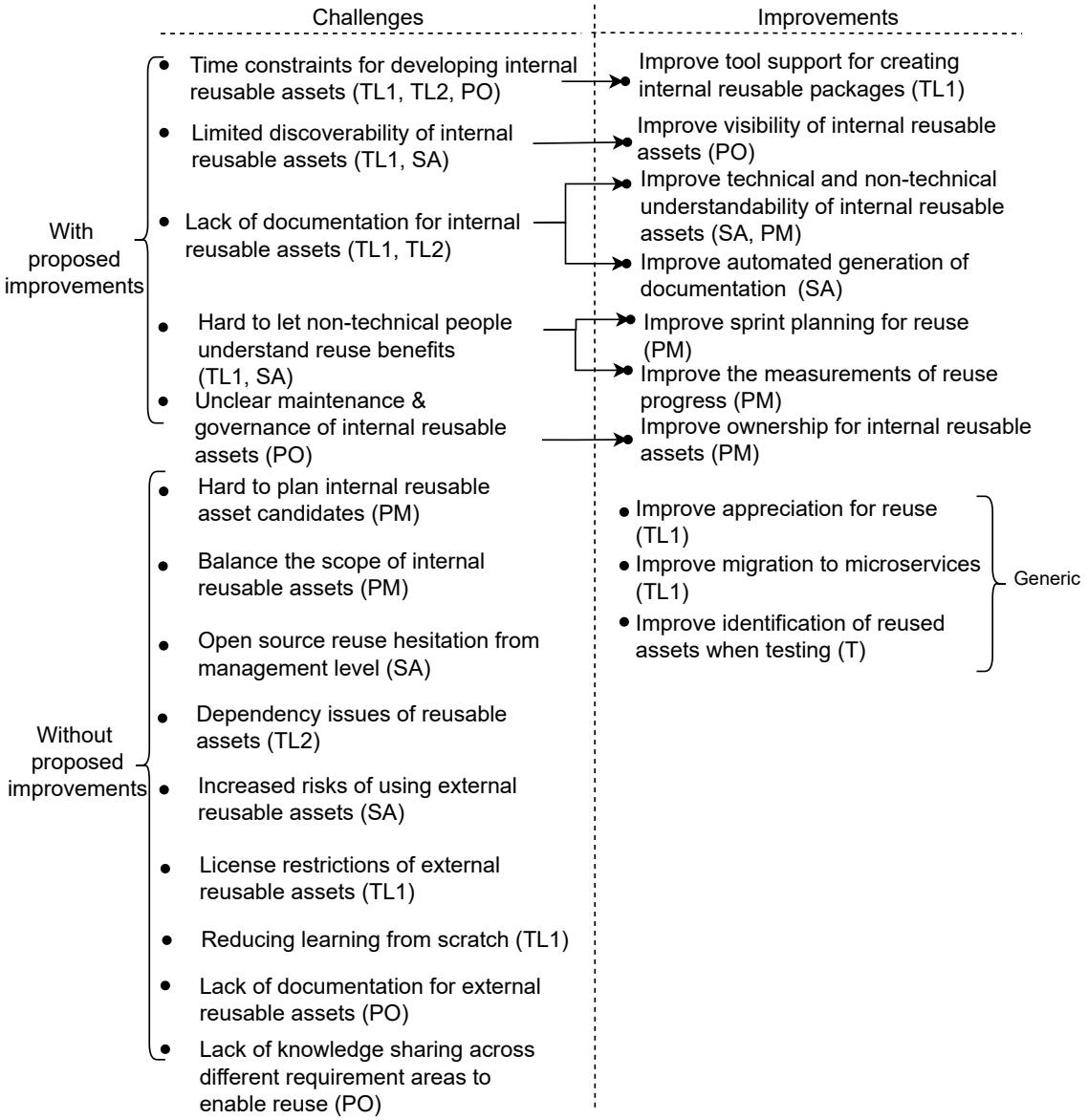


Figure 3. Challenges and improvements of the software reuse in the context of contemporary SE practices

4.3.1. Challenges along with improvement suggestions

We elicited five out of fourteen challenges with participants' proposed improvements. In addition, the participants mentioned three generic improvements which are also described in this section. The challenges with proposed improvements are as follows:

1. **Time constraints for developing internal reusable asset** is a big concern raised by two tech leads and the product owner. Tight release schedule is always a constraint for software product delivery in the industry. Adding DFR into an existing development life cycle is even more demanding. One of the tech leads gave an example that "*we have a notification system, which were very easily could be made to a microservice. But for now, since we are close to release, we will keep it in our application for now, but it will probably become a requirement and as new application or service of itself in the future*".

   Improvement suggestion – Improve managing time constraints

   Improving *tool support* for creating internal reusable packages is necessary to manage the time constraints for developing reusable assets. One of the tech lead suggested supporting tools for creating reusable packages – "*to have a lot of tools that do things for us or to create packages more easily*". To enable development of reusable assets while ensuring timely delivery, the project manager wanted to **improve *sprint planning* for reuse** and suggested the following: "*a good approach from my perspective is to have the developers give me two estimates, one where we do not develop it as a reusable component, and one where we do. So I can discuss reuse priorities with the product owner. If the product owner does not agree to prioritize reuse tasks, then my suggestion is to implement it for that specific area, but then we are allocated time afterwards for converting it to a reusable component.*"

2. Two participants perceived **limited discoverability of internal reusable assets** as a challenge. Reuse will not happen if the developers cannot find the available reusable assets. One of the tech lead and the software architect identified the challenges that people in the company are not able to discover all the existing reused assets within the company. The software architect emphasized "*it (the challenge) is the discoverability of the things for developers to know what actually exists internally*".

   Improvement suggestion – Improve visibility of internal reusable assets

   To improve the discoverability challenge, the product owner wanted to **improve *visibility* of internal reusable assets** by managing and grouping the similar user stories representing a workflow. The right categorization helps to improve visibility and hence facilitate software reuse. He added "*we probably have thousands of user stories, but to be able to get that in a manageable way, you probably need to step up a little bit, maybe on workflow level.*"

3. **Lack of documentation for internal reusable assets** was mentioned by two tech leads. One of the tech lead said "*usually they (internal packages) do not have good*

documentation". And low quality of documentation or missing documentation, may hinder understandability of internal reusable assets.

Improvement suggestion – Improve documentation for internal reusable assets

The participants identified concrete improvement suggestions to improve documentation. The software architect wanted to **improve *automated generation* of documentation** since they considered manually writing the documentation for reusable assets is as an overhead and suggested automating this process: "*since we want to spare the developers from writing too much documentation, we are looking into how to automate it entirely.*"
The software architect and project manager both suggested on how to **improve *technical and non-technical* understandability of internal reusable assets**. The software architect suggested that they need to improve documentation of reusable assets in a way that helps developers understand the capability of the reusable assets and "*how to use the package (reusable assets)*". The project manager suggested that they need to improve documentation so that "*other people within the company know what is available for reuse than only the developers.*"

4. One of the tech leads and the software architect found it was **hard to let non-technical people understand reuse benefits** which led to a lack of management support. The software architect brought up that "*sometimes it is a challenge to get them (non-technical people) to understand what is the actual benefit for it (software reuse)*".

Improvement suggestion – Improve management's perception on reuse benefit

The project manager wanted to **improve the *measurement* of reuse progress** to demonstrate the reuse rate to the management. He said that "*when we develop a new web application, I want to be able to see that in this new web application, how much did we reuse. So basically how much of the code base that's in this web application is from reuse. And that could be one version of kind of measuring how much implementation time is saved.*"

5. The product owner pointed out the challenge of **unclear maintenance/governance of internal reusable assets**. He raised the following question – "*when we have written it (the reusable assets), who should maintain it (the reusable assets)*".

Improvement suggestion – Improve maintenance/governance of internal reusable assets

The project manager identified the need to **improve *ownership* for internal reusable assets** and suggested that they need "*a clear strategy of who is responsible and who owns this (reusable) component*".

Improvement suggestion – Generic improvements

Improvements in reward are mentioned by one of the tech leads to scale reuse instead of providing suggestions to mitigate the listed challenges. It is important to **improve *appreciation* for reuse**, which motivates the developers to not only produce but also consume reusable assets. One tech lead mentioned "*not incentive or maybe like said appreciation that we make time to create something that will save time later on.*"

The tester wanted to **improve *identification of reused assets* when testing**. He suggested enhancing the traceability of reused code in the system under test – "*I think from my (tester) point of view, one area for improvement is to clarify when we reuse stuff, because it is not always very clear*".

One of the tech lead identified a need in having a clear vision to **improve *migration* to microservices**, he described it as "*from the architecture point of view, what components, to plan, extract (for migration)*".

4.3.2. Challenges without proposed improvements

In this section we discuss nine out of fourteen challenges that participants brought up without any associated improvement suggestions.

1. The project manager found it is **hard to plan internal reusable assets candidates**. He added that: "*identify is this functionality that should be implemented as a reusable component, and taking that decision, that is hard to get in black or white. And usually you have to move down some grey area to kind of take that decision.*"
2. After identifying the reusable asset candidates, practitioners are faced with a dilemma of the scope of the reusable assets. The project manager found it challenging to balance how general and specific the reusable assets should be, namely **balance the scope of internal reusable assets**. He got this input from developers that "*they (developers) find very hard when it comes to reusable components, to find the right level of how generic the component should be*".
3. The software architect identified **open source reuse hesitation from management level**. He said "*people that started as a developer and he now have another role in higher-up management*" are hesitant towards open-source reuse. He added that "*mentioning open-source to a person who worked with proprietary systems and closed was not really easy. And open-source is misunderstood in many ways, I would say.*" The management can hinder opportunistic reuse if they are not willing to take in the open source software.
4. When performing reuse, one tech lead identified **dependency issues of reusable assets**. Many dependencies need to be taken care of when reusing the package and this dependency overhead creates lots of work for developers and is not good for users as well. He added: "*some (reusable) components are good for us, but it has a lot of dependencies.*"
5. The software architect highlighted the **increased risks of using external reusable assets**. The company relies on the quality of the reused external assets. And he added that "*it is a bit more risky to include things from the outside*".

6. **License restrictions of external reusable assets** may prevent the reuse of external assets. One of the tech leads pointed that sometimes they "*cannot reuse because it depends on licensing*".
7. With increased software reuse, one of the tech lead highlighted an issue in **reducing learning from scratch**. He described that "*if we keep reusing stuff and not code anything ourselves, that might be an issue. Get experience that way, to do things from scratch as well.*" However, the software architect viewed learning benefits in terms of understanding and knowledge gained from reusing as described in Section 4.2.2.
8. The product owner and one tech lead raised a concern in **lack of documentation for external reusable assets**. The product owner stated that developers also need to know what exists externally, and what can be brought into the company. Incomplete documentation in external reusable assets hinders the opportunistic reuse practice: "*to find and also to see can it (reusable asset) reports in that way that we want to utilize it and maybe incorporate it in our product as the way it is or something else with the license agreement. That is hard to find it on that level.*" And the tech lead also said sometimes "*the real read-me files have none*" in external packages.
9. **Lack of knowledge sharing across different requirement areas to enable reuse** indicates limited transparency. The product owner explained that this knowledge sharing problem occurred because different teams work on their specific requirement areas and they lacked central communication for sharing. However, he emphasized that "*when every part is developed, it is very important that we need to share knowledge, so several people know about this functionality.*"

4.3.3. Prioritized challenges along with improvement suggestions -

Although practitioners perceived some challenges with software reuse, they considered reuse to be important and wanted to invest in further improving the software reuse process in the company. We presented the overall interview findings and asked the software architect to prioritize the challenges and improvements they would like to implement. Based on the company's requirements and internal discussions with the relevant stakeholders, the software architect prioritized discoverability and ownership of reusable assets, knowledge sharing and reuse measurement as the focus areas for further investigation. We identified some IS patterns from the InnerSource Commons[7] that could address the top challenges and improvement areas. The IS patterns that we discussed with the product manager and the software architect are discussed below:

1. **Discoverability of the reusable assets.** *InnerSource Portal* pattern[8] aims to create an intranet portal that allows the project owners to advertise their projects to the entire organization. Though the case company does not have shared IS projects, they can use the portal to find all the reusable assets in an efficient way.
2. **Ownership of the reusable assets.** The participants emphasized there is a need to have a clear strategy about the ownership. To complement the participant's suggestion, we proposed two InnerSource patterns – *30 Days Warranty* pattern[9] and *Trusted Committer* pattern[10], to address the maintenance/governance challenge. *Thirty Days Warranty* pattern assigns the contributors the responsibility to pass the knowledge and

[7]https://innersourcecommons.org/
[8]https://patterns.innersourcecommons.org/p/innersource-portal
[9]https://patterns.innersourcecommons.org/p/30-day-warranty
[10]https://patterns.innersourcecommons.org/p/trusted-committer

solve the problems about their contributions within a certain period. It creates a buffer time for the one responsible for the reusable assets to understand the contributed code and gain the ability to maintain them. *Trusted Committer* pattern aims to assign a trusted committer role to the most active contributors and allocate bandwidth to facilitate the maintenance/governance of the reusable assets.

3. **Knowledge sharing of the reuse related information**. To facilitate knowledge sharing, we suggested to improve work and decision transparency in the group discussions, namely, 1) ask all teams to publish their roadmaps and backlog planning, 2) publish decisions and allow for discussions from other teams. Such suggestions were generated from IS pattern – *Transparent Cross-team Decision Making Using RFCs*[11], which helps increase the chance of other teams' participation by publishing internal requests for comments (RFCs) documents.
4. **Reuse measurement**. We suggested *Cross-team Project Valuation* pattern[12] to further address the reuse measurement improvement. Such a pattern aims to create a model to calculate the value of cross-team projects (in our case, the shared reusable assets) and demonstrate the increased productivity when people from other teams are also involved in development or maintenance.

We discussed the feasibility of the above proposed IS patterns with the product manager and the software architect. In the discussions, we concluded that many patterns from InnerSource Common[13] could help address the top concerning challenges. Moreover, the project manager agreed to conduct a follow-up investigation to check the company's readiness for adopting more InnerSource practice to improve the development and maintenance of the reusable assets.

RQ3 – Summary. In the group discussions, S-Group Solutions AB rated discoverability and ownership of the reusable assets, knowledge sharing of the reuse related information and reuse measurement as the major improvement areas of the software reuse practice. Apart from the improvements proposed by the participants, IS patterns help address a lot of software reuse challenges.

5. Discussion

This section further discusses and compares our results in software reuse costs, benefits, challenges, and improvements with the related works.

5.1. Software reuse costs in the context of contemporary SE practices

In our study, we identified that practicing software reuse results in additional costs – more in case of development for reuse as compared to development with reuse. Our findings are in line with the results reported by Kruger and Berger [20] and Agresti [24]. We found additional coordination in participatory reuse as we anticipated. Additional coordination is needed in the case of opportunistic reuse as well. Kruger and Berger [20] also found that practicing software reuse results in additional synchronization and coordination among different teams. They found additional coordination when handing over reusable assets

[11] https://patterns.innersourcecommons.org/p/transparent-cross-team-decision-making-using-rfcs
[12] https://patterns.innersourcecommons.org/p/crossteam-project-valuation
[13] https://innersourcecommons.org/

to different functional teams, such as development teams and quality assurance teams. In comparison, our identified additional coordination occurred when other consumer teams wanted to participate and contribute to developing the reusable assets. It could also be argued that additional coordination helps increase the transparency between different teams and enhance the internal collaboration.

Our study and Agresti [24] found extra costs in understanding the reusable code. However, Agresti [24] identified extra cost when the reused assets need extensive modification. Our participants did not bring up such a cost. Comparing Agresti's study [24] with our study, we think the reason could be that our case company follows a relatively more systematic process when performing software reuse, such as using the solution vision process and the technical analysis before developing the reusable assets (see Figure 1). Moreover, our case company did not mention additional integration effort in opportunistic reuse as we assumed.

5.2. Software reuse benefits in the context of contemporary SE practices

We identified better product quality and time-saving in development, maintenance, testing and delivery as the main software reuse benefits in the case company. Multiple secondary studies (cf. [1–3]) and primary studies [18, 20, 21, 24, 25, 34] on software reuse also identified that the software reuse practices contribute to better product quality and time saving in one or more phases of the software development cycle.

Bauer et al. [25] and our study found that software reuse helps in improving the consistency of the product. However, in our study, software reuse is found to contribute to consistent user interface experience across different modules, while in Bauer's et al. [25] study, software reuse contributes to feature consistency over the range of products. Literature related to internal reuse [21, 34] and our study found the learning benefit in software reuse, however, from different perspectives. We identified that internal reuse practice also offers some learning opportunities to the developers – they could learn more from understanding and reusing well-designed reusable assets. Goldin et al. [34] also found that requirement management and reuse help the new employees complete the onboarding process easier and quicker from the learning perspective.

Barros-Justo et al. [18] identified higher documentation quality as a benefit of software reuse. We did not find higher documentation quality as a benefit of software reuse in our study. However, the participants pointed out that reusable assets require additional documentation (for details, see Section 4.2) as we anticipated. This upfront cost may contribute to higher document quality later. There maybe two reasons for not having higher documentation quality due to the reuse practices in the case company. First, the case company is still at the beginning (about two years) of their software reuse journey. They need more time to adapt to the reuse approaches. Second, in a medium-sized company, it is difficult to invest extra resources to create additional documentation.

The participants also brought up that due to the availability of the reusable assets, the consumer teams do not need to dedicate resources in those domain areas that are already covered by the reusable assets. However, with this benefit placed, the consumers may take things for granted and start to ask for more features in the reusable assets. Riehle et al. [35] reported a similar scenario wherein the producer teams were over-burdened due to the large number of change requests from the consumers of the reusable common assets.

5.3. Software reuse challenges in the context of contemporary SE practices

In our study, the participants were positive about having internal reusable assets. However, they also pointed out some challenges related to the management of the internal reusable assets, including discoverability, knowledge sharing and the ownership of the internal reusable assets. Barros-Justo et al. [18] and Bauer and Vetro [19] also reported that finding the relevant reusable asset is a common problem. Due to the boundaries between projects, reusable assets become unavailable for the developers across projects [19] – such a way of working potentially constraints software reuse and it represents the same challenge that we also identified in our case company – namely, lack of knowledge sharing across different teams to enable reuse. According to our study, Bauer and Vetro [19], and Barros-Justo et al. [18], practitioners rely on the repository search and communication with their colleagues as the main methods for finding the relevant internal reusable assets.

The question of who will own and maintain the reusable assets in participatory reuse is important. In our study, the participants brought up this question as an important challenge to deal with as we anticipated. Kruger and Berger [20] also identified the challenges in coordinating in and between teams, especially when the responsibilities are not clear.

Some participants in our study also shared that it is hard to explain the benefits of practicing software reuse to the non-technical persons (e.g., senior management), which may hinder the organization-wide adoption of software reuse. Morisio et al. [36] and Kolb et al. [37] also found that the senior management support is essential for promoting the software reuse process to the entire organization.

In our study, we found it is difficult to define the scope of the reusable assets at the initial stage. Kolb et al. [37] also shared a similar finding, however in their case, the challenge was about adding new features to an existing reusable component.

As discussed previously, software reuse practices offer learning opportunities to the developers. However, interestingly some participants cautioned that too much reliance on reuse may have a negative impact on the capability of the developers to write own code. Bauer et al. [25] also discussed the challenge of trying to strike a balance between acceptable level of reuse and excessive reuse.

Our study identified that the reuse of packages and components may lead to additional dependencies that need to be taken care of. Bauer et al. [25] identified dependency explosion was the major issue for Google in software reuse, especially the ripple effects caused by changes in reused code assets. Barros-Justo et al. [18] also noted dependency issues when reusable assets are integrated into the new solutions. We suggest practitioners could adopt and follow the practices proposed by Gustavsson [38] for managing the open source dependencies, e.g., establishing a forum for conscious decisions on open source dependencies, maintaining a dependency list and scanning for security issues. For opportunistic reuse, dealing with license restrictions was also shared as a challenge by the participants in our study, which is in line with some related works [18, 19].

5.4. Software reuse related improvements in the context of contemporary SE practices

First, we discuss those improvements that the case company has already implemented as a result of this study. The improvements are aimed at improving the development, integration and documentation of the reusable assets:

1. Additional boilerplate code: To remove the need to write additional boilerplate code while developing a reusable package, the company has developed a mechanism to create a template that includes all startup code required for initiating the development of a reusable package.
2. Additional effort in debugging: In cases when the bugs are related to the reused shared packages, the participants shared that they need to spend some additional time on debugging as they need to copy the code of the reusable package to a new project to perform the debugging. The company has now developed the support to address this issue.
3. Lack of documentation: Lack of documentation for reusable packages was identified as one of the challenges. Some documentation for reusable packages is now automated, thus saving the time and effort spent on manually creating the reusable package documentation.

In addition to the three implemented improvements discussed above, we also agreed to investigate the feasibility of adopting more IS practices and patterns to improve the development, maintenance and governance of the reusable assets in software reuse.

In the case company, the reusable assets are maintained in a repository with some keyword searching options. We suggested the case company to adopt the *InnerSource Portal* pattern to enhance the discoverability of reusable assets. For the same discoverability purpose, Agresti et al. [24] suggested cleaning up the reusable code library, setting criteria to qualify the reusable code, finding a manager to look after the library, and having an online keyword-search capability across different sources. Moreover, Bauer et al. [25] suggested that the reusable assets should be listed in the marketplace and the reusable assets from different libraries should be merged to avoid the duplicates. The similarity of the discoverability improvements among our case company, related InnerSource pattern and the discussed related works [24, 25] is that we all focused on the management and the search facility of the reusable assets.

The ownership of reusable assets affects the developers and the project managers. However, we did not find ownership related improvements in the selected related works. The patterns – *30 Day Warranty* and *Trusted Committer* pattern that we introduced to the company, could help in solving the ownership issues, reducing the effort in locating people, and synchronizing the meeting schedules and release plans [4]. As for the reuse measurements, the case company started using the reuse rate to track the percentage of the reused code assets. We also suggested *Cross-team Project Valuation* pattern to the company. Mohagheghi and Conradi [1] conducted a literature review, investigating the metrics about software reuse quality, productivity and economic benefits. They aggregated and categorized different metrics from 11 studies from 1994 to 2005. We argue that there is a need to extend such a literature review since the reusable assets (e.g., microservices) and the reuse type (e.g., participatory reuse) have evolved since 2005.

Compared to our study, Agresti et al. [24] also provided suggestions for improving the understandability of reusable assets, such as better comments in the code, better structured software modules and a written reuse guidebook. However, they did not mention the need for non-technical people, e.g., managers to understand the value of reuse.

The development and maintenance of reusable assets also have budgetary implications. Like our study, Agresti et al. [24] also discussed the need for improvements in resource planning to facilitate developers that are working on the reusable assets in addition to other tasks. They [24] suggested allocating additional budget for the developers to facilitate them for contributing to the reusable assets.

5.5. Threats to validity

We discuss threats to validity in two phases using the validity threats categorization proposed by Peterson and Gencel [39]: (1) study design and data collection, and (2) data analysis.

5.5.1. Study design and data collection

Theoretical validity. The theoretical validity is concerned with construct definition, evaluation comprehension and the selection of subjects. We decided the study objective based on the company's needs through a joint discussion with the company contact person. The interview guide is developed and reviewed iteratively among authors, and a pilot semi-structured interview is performed before the actual interviews to evaluate the interview questions' comprehension. As for the recruitment strategy, we provided the reuse related role descriptions to help the contact person identify the relevant people for the interview. The sample size is small, but we managed to cover at least 20% of the population, all teams, and related roles. The sample size of the group discussions is small and the participants are from the interviews. However, the selected two participants are the most relevant and experienced people in software reuse practice in the company. In addition, we asked the two participants to gather opinions from their colleagues and prepare documentation before they came to the discussions.
Descriptive validity. The descriptive validity is concerned with factual accuracy. We transcribed the interviews word to word and tried to use the actual text segments to describe the results as much as possible. Moreover, we presented the preliminary study results and shared the study report to the software architect. And he confirmed that the results captured the reality.

5.5.2. Data analysis

Interpretative validity. The interpretive validity is concerned with capturing the relevant information and researchers' bias in interpretation. We transcribed the interviews word to word to avoid misinterpretations. We followed the Cruzes's and Dybå's [29] recommended steps of thematic analysis to analyze the transcripts. The first and second authors independently analyzed and generated the code to confirm the results. The third author validated the data credibility as mentioned in Section 3.5, which resulted in some minor changes regarding code names and code descriptions. We also presented the results to the company to eliminate misinterpretation.
Generalizability. The generalizability is concerned with the context information which influences the study transferability. Our focus is medium-sized companies and we introduced the company context information in detail (see Section 3.3 and Section 4.1), so that other relevant companies could relate our case to their context and get some useful insights. Furthermore, the detailed context information helps the researchers to include the details when reporting findings on software reuse in contemporary SE practices.

6. Conclusion and future work

In this paper, we reported the results of an exploratory case study on software reuse practice in the context of contemporary SE practices conducted in a medium-sized company. The reported study covers the software reuse process, costs, benefits, challenges and improvements. We obtained the data from six semi-structured interviews, four group discussions and relevant documentation, followed by a rigorous process to analyze the collected data.

The study elaborates how the case company is practicing software reuse, including participatory reuse and opportunistic reuse. Participatory reuse is an organizational-wide reuse collaboration between the producers and consumers of the reusable assets, while opportunistic reuse relates to the reuse of external assets from open source communities or other third parties. The results show that the software reuse costs mainly relates to the development of the reusable assets, their documenting and the time spent in additional coordination between the teams working on the common reusable assets. In our study, the participants perceived that the benefits of software reuse outweigh the associated costs, thus were in favor of further improving the software reuse practices. Software reuse benefits many stakeholders in terms of people, process and products. The main perceived benefits are related to time-saving and product quality, which are highly aligned with the investigated related works. The study participants were aware of the software reuse challenges and suggested some concrete improvements. According to the interviews and group discussions results, discoverability and ownership of the reusable assets, knowledge sharing and reuse measurement are the top concerning challenges and improvements for the case company, which have a great potential to be addressed by certain InnerSource patterns and practices.

The case company is interested in adopting InnerSource patterns and practices to systematize the software reuse process. We are planning a follow up investigation at the case company to ascertain the company's readiness for adopting InnerSource practices for improving the development and maintenance of the reusable assets. With the help of the relevant stakeholders, the idea is to assess the application of the proposed improvements in terms of costs and importance and select specific InnerSource practices for implementation in the case company. In the long term, we are interested in evaluating the effectiveness of the adopted practices for improving the state of software reuse in the case company.

Acknowledgements

We would like to acknowledge that this work was supported by the Knowledge Foundation through the OSIR project (reference number 20190081) at Blekinge Institute of Technology, Sweden. We would also like to thank the practitioners from the case company for collaborating with us. Lastly, we would like to thank Prof. Claes Wohlin for participating in the initial discussion with the case company and providing valuable feedback on the study design and the initial version of the study report.

References

[1] P. Mohagheghi and R. Conradi, "Quality, productivity and economic benefits of software reuse: A review of industrial studies," *Empirical Software Engineering*, Vol. 12, No. 5, 2007, pp. 471–516.

[2] J.L. Barros-Justo, F. Pinciroli, S. Matalonga, and N. Martínez-Araujo, "What software reuse benefits have been transferred to the industry? A systematic mapping study," *Information and Software Technology*, Vol. 103, 2018, pp. 1–21.

[3] D. Bombonatti, M. Goulão, and A. Moreira, "Synergies and tradeoffs in software reuse – A systematic mapping study," *Software: Practice and Experience*, Vol. 47, No. 7, 2017, pp. 943–957.

[4] D. Cooper and K.J. Stol, *Adopting InnerSource*. O'Reilly Media, Incorporated, 2018.

[5] J.L. Barros-Justo, F.B. Benitti, and S. Matalonga, "Trends in software reuse research: A tertiary study," *Computer Standards and Interfaces*, Vol. 66, 2019, p. 103352.

[6] R. Capilla, B. Gallina, C. Cetina, and J. Favaro, "Opportunities for software reuse in an uncertain world: From past to emerging trends," *Journal of Software: Evolution and Process*, Vol. 31, No. 8, 2019, p. e2217.

[7] T. Mikkonen and A. Taivalsaari, "Software reuse in the era of opportunistic design," *IEEE Software*, Vol. 36, No. 3, 2019, pp. 105–111.

[8] B. Xu, L. An, F. Thung, F. Khomh, and D. Lo, "Why reinventing the wheels? An empirical study on library reuse and re-implementation," *Empirical Software Engineering*, Vol. 25, No. 1, 2020, pp. 755–789.

[9] R. Capilla, T. Mikkonen, C. Carrillo, F.A. Fontana, I. Pigazzini et al., "Impact of opportunistic reuse practices to technical debt," in *IEEE/ACM International Conference on Technical Debt (TechDebt)*. IEEE, 2021, pp. 16–25.

[10] M. Höst, K.J. Stol, and A. Oručević-Alagić, "Inner source project management," in *Software Project Management in a Changing World*. Springer, 2014, pp. 343–369.

[11] S. Fox, "IBM internal open source bazaar." *Presentation at the IBM Linux Technology Center in November 2007*.

[12] P. Vitharana, J. King, and H.S. Chapman, "Impact of internal open source development on reuse: Participatory reuse in action," *Journal of Management Information Systems*, Vol. 27, No. 2, 2010, pp. 277–304.

[13] A. Balalaie, A. Heydarnoori, and P. Jamshidi, "Microservices architecture enables DevOps: Migration to a cloud-native architecture," *IEEE Software*, Vol. 33, No. 3, 2016, pp. 42–52.

[14] P. Jamshidi, C. Pahl, N.C. Mendonça, J. Lewis, and S. Tilkov, "Microservices: The journey so far and challenges ahead," *IEEE Software*, Vol. 35, No. 3, 2018, pp. 24–35.

[15] J. Soldani, D.A. Tamburri, and W.J. Van Den Heuvel, "The pains and gains of microservices: A systematic grey literature review," *Journal of Systems and Software*, Vol. 146, 2018, pp. 215–232.

[16] J.P. Gouigoux and D. Tamzalit, "From monolith to microservices: Lessons learned on an industrial migration to a web oriented architecture," in *IEEE International Conference on Software Architecture Workshops (ICSAW)*. IEEE, 2017, pp. 62–65.

[17] M. Capraro and D. Riehle, "Inner source definition, benefits, and challenges," *ACM Computing Surveys (CSUR)*, Vol. 49, No. 4, 2016, pp. 1–36.

[18] J.L. Barros-Justo, D.N. Olivieri, and F. Pinciroli, "An exploratory study of the standard reuse practice in a medium sized software development firm," *Computer Standards and Interfaces*, Vol. 61, 2019, pp. 137–146.

[19] V. Bauer and A. Vetro, "Comparing reuse practices in two large software-producing companies," *Journal of Systems and Software*, Vol. 117, 2016, pp. 545–582.

[20] J. Krüger and T. Berger, "An empirical analysis of the costs of clone-and platform-oriented software reuse," in *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2020, pp. 432–444.

[21] O.P.N. Slyngstad, A. Gupta, R. Conradi, P. Mohagheghi, H. Rønneberg et al., "An empirical study of developers views on software reuse in statoil asa," in *Proceedings of the ACM/IEEE international symposium on empirical software engineering*, 2006, pp. 242–251.
[22] S.A. Ajila and D. Wu, "Empirical study of the effects of open source adoption on software development economics," *Journal of Systems and Software*, Vol. 80, No. 9, 2007, pp. 1517–1529.
[23] N. Mäkitalo, A. Taivalsaari, A. Kiviluoto, T. Mikkonen, and R. Capilla, "On opportunistic software reuse," *Computing*, Vol. 102, No. 11, 2020, pp. 2385–2408.
[24] W.W. Agresti, "Software reuse: developers' experiences and perceptions," *Journal of Software Engineering and Applications*, Vol. 4, No. 01, 2011, p. 48.
[25] V. Bauer, J. Eckhardt, B. Hauptmann, and M. Klimek, "An exploratory study on reuse at Google," in *Proceedings of the 1st international workshop on software engineering research and industrial practices*, 2014, pp. 14–23.
[26] P. Runeson and M. Höst, "Guidelines for conducting and reporting case study research in software engineering," *Empirical Software Engineering*, Vol. 14, No. 2, 2009, pp. 131–164.
[27] E.C. (2003), *Commission Recommendation of 6 May 2003 concerning the definition of micro, small and medium-sized enterprises, C (2003) 1422*. [Online]. http://data.europa.eu/eli/reco/2003/361/oj.[Accessed:Apr.16,2021]
[28] C.B. Seaman, "Qualitative methods in empirical studies of software engineering," *IEEE Transactions on Software Engineering*, Vol. 25, No. 4, 1999, pp. 557–572.
[29] D.S. Cruzes and T. Dyba, "Recommended steps for thematic synthesis in software engineering," in *International Symposium on Empirical Software Engineering and Measurement*, 2011, pp. 275–284.
[30] K. Petersen and C. Wohlin, "A comparison of issues and advantages in agile and incremental development between state of the art and an industrial case," *Journal of Systems and Software*, Vol. 82, No. 9, 2009, pp. 1479–1490.
[31] J. Saldaña, *The coding manual for qualitative researchers*. SAGE Publications Limited, 2021.
[32] *IEEE Standard for Information Technology – System and Software Life Cycle Processes – Reuse Processes*, IEEE Std. 1517–2010, Aug. 2010.
[33] K. Petersen and C. Wohlin, "Context in industrial software engineering research," in *3rd International Symposium on Empirical Software Engineering and Measurement*. IEEE, 2009, pp. 401–404.
[34] L. Goldin and D.M. Berry, "Reuse of requirements reduced time to market at one industrial shop: A case study," *Requirements Engineering*, Vol. 20, No. 1, 2015, pp. 23–44.
[35] D. Riehle, M. Capraro, D. Kips, and L. Horn, "Inner source in platform-based product engineering," *IEEE Transactions on Software Engineering*, Vol. 42, No. 12, 2016, pp. 1162–1177.
[36] M. Morisio, M. Ezran, and C. Tully, "Success and failure factors in software reuse," *IEEE Transactions on Software Engineering*, Vol. 28, No. 4, 2002, pp. 340–357.
[37] R. Kolb, I. John, J. Knodel, D. Muthig, U. Haury et al., "Experiences with product line development of embedded systems at testo ag," in *10th International Software Product Line Conference (SPLC'06)*. IEEE, 2006, pp. 10–pp.
[38] T. Gustavsson, "Managing the open source dependency," *Computer*, Vol. 53, No. 2, 2020, pp. 83–87.
[39] K. Petersen and C. Gencel, "Worldviews, research methods, and their relationship to validity in empirical software engineering research," in *2013 joint conference of the 23rd international workshop on software measurement and the 8th international conference on software process and product measurement*. IEEE, 2013, pp. 81–89.

e-Informatica Software Engineering Journal (EISEJ) is an international, fully open access (CC-BY 4.0 without any fees for both authors and readers), blind peer-reviewed computer science journal using a fast, continuous publishing model (papers are edited, assigned to volume, receive DOI & page numbers, and are published immediately after acceptance without waiting months in a queue to be assigned for a specific volume/issue) without paper length limit that concerns theoretical and practical issues pertaining development of software systems. Our aim is to focus on empirical software engineering, as well as data science in software engineering.

The journal is published by *Wrocław University of Science and Technology* under the auspices of the *Software Engineering Section of the Committee on Informatics of the Polish Academy of Sciences*.

Aims and Scope

The purpose of **e-Informatica Software Engineering Journal** is to publish original and significant results in all areas of software engineering research.

The scope of **e-Informatica Software Engineering Journal** includes methodologies, practices, architectures, technologies and tools used in processes along the software development lifecycle, but particular stress is laid on empirical evaluation using well-chosen statistical and data science methods.

e-Informatica Software Engineering Journal is published online and in hard copy form. The on-line version is from the beginning published as a gratis, no authorship fees, open-access journal, which means it is available at no charge to the public. The printed version of the journal is the primary (reference) one.

Topics of interest

- Software requirements engineering and modeling
- Software architectures and design
- Software components and reuse
- Software testing, analysis and verification
- Agile software development methodologies and practices
- Model driven development
- Software quality
- Software measurement and metrics
- Reverse engineering and software maintenance
- Empirical and experimental studies in software engineering (incl. replications)
- Evidence-based software engineering
- Systematic reviews and mapping studies (see SEGRESS guidelines)
- Statistical analyses and meta-analyses of experiments
- Robust statistical methods
- Reproducible research in software engineering
- Object-oriented software development
- Aspect-oriented software development
- Software tools, containers, frameworks and development environments
- Formal methods in software engineering.
- Internet software systems development
- Dependability of software systems
- Human-computer interaction
- AI and knowledge based software engineering
- Data science in software engineering
- Prediction models in software engineering
- Mining software repositories
- Search-based software engineering
- Multiobjective evolutionary algorithms
- Tools for software researchers or practitioners
- Project management
- Software products and process improvement and measurement programs
- Process maturity models

Funding acknowledgements: Authors are requested to identify who provided financial support for the conduct of the research and/or preparation of the article and to briefly describe the role of the sponsor(s), if any, in study design; in the collection, analysis and interpretation of data; in the writing of the paper. If the funding source(s) had no such involvement then this should be stated as well.

The submissions will be accepted for publication on the base of positive reviews done by international Editorial Board and external reviewers.

English is the only accepted publication language. To submit an article please enter our online paper submission site.

Subsequent issues of the journal will appear continuously according to the reviewed and accepted submissions.

The journal is included in the IC Journal Master List (ICV=7.59 was obtained in 2013) and indexed by Scopus, DBLP, DOAJ, BazTech etc.

Paper copies of selected issues of the journal are available from our Publisher (please contact oficwyd@pwr.wroc.pl for details).

http://www.e-informatyka.pl/

e-Informatica

ISSN 1897-7979