e-Informatica Software Engineering Journal, Volume 13, Issue 1, 2019, pages: 141–202, DOI 10.5277/e-Inf190105

Empirical Studies on Software Product Maintainability Prediction: A Systematic Mapping and Review

Sara Elmidaoui*, Laila Cheikhi*, Ali Idri*, Alain Abran**

*Software Project Management Team (SPM), ENSIAS, Mohamed V University, Rabat, Morocco

**Department of Software Engineering and Information Technology, École de Technologie Supérieure, Montréal, Canada

sarah.elmidaoui@gmail.com, laila.cheikhi@gmail.com, ali.idri@um5.ac.ma, alain.abran@etsmtl.ca

Abstract

Background: Software product maintainability prediction (SPMP) is an important task to control software maintenance activity, and many SPMP techniques for improving software maintainability have been proposed. In this study, we performed a systematic mapping and review on SPMP studies to analyze and summarize the empirical evidence on the prediction accuracy of SPMP techniques in current research.

Objective: The objective of this study is twofold: (1) to classify SPMP studies reported in the literature using the following criteria: publication year, publication source, research type, empirical approach, software application type, datasets, independent variables used as predictors, dependent variables (e.g. how maintainability is expressed in terms of the variable to be predicted), tools used to gather the predictors, the successful predictors and SPMP techniques, (2) to analyze these studies from three perspectives: prediction accuracy, techniques reported to be superior in comparative studies and accuracy comparison of these techniques.

Methodology: We performed a systematic mapping and review of the SPMP empirical studies published from 2000 up to 2018 based on an automated search of nine electronic databases.

Results: We identified 82 primary studies and classified them according to the above criteria. The mapping study revealed that most studies were solution proposals using a history-based empirical evaluation approach, the datasets most used were historical using object-oriented software applications, maintainability in terms of the independent variable to be predicted was most frequently expressed in terms of the number of changes made to the source code, maintainability predictors most used were those provided by Chidamber and Kemerer (C&K), Li and Henry (L&H) and source code size measures, while the most used techniques were ML techniques, in particular artificial neural networks. Detailed analysis revealed that fuzzy & neuro fuzzy (FNF), artificial neural network (ANN) showed good prediction for the change topic, while multilayer perceptron (MLP), support vector machine (SVM), and group method of data handling (GMDH) techniques presented greater accuracy prediction in comparative studies. Based on our findings SPMP is still limited. Developing more accurate techniques may facilitate their use in industry and well-formed, generalizable results be obtained. We also provide guidelines for improving the maintainability of software.

Keywords: systematic mapping study, systematic literature review, software product maintainability, empirical studies

Submitted: 27 Jun 2018; Revised: 5 Apr 2019; Accepted: 6 Apr 2019; Available online: 17 Jul 2019

1. Introduction

Maintainability of a software product is defined in SWEBOK [1] as a quality characteristic that "must be specified, reviewed, and controlled during the software development activities in order to reduce maintenance costs". Many techniques for software product maintainability prediction (SPMP) have been proposed as a means to better manage maintenance resources through a defensive design [2]. However, predicting software maintainability remains an open research area since the maintenance behaviors of software systems are complex and difficult to predict [3]. Moreover, industry continues to search for appropriate ways to help organizations achieve reliable prediction of software product maintainability.

A number of studies have been conducted in this context [4–9]. For instance, Riaz et al. [4] conducted a systematic literature review (SLR) on a set of 15 primary studies dating from 1985 to 2008 to investigate techniques and methods used to predict software maintainability. They found that the number of studies varied from one to two per year illustrating that this research topic was still in emergence in 2008 and had not yet reached a certain level of maturity. Moreover, they showed that the choice among prediction models for maintainability was not obvious (12 out of 15 studies had proposed models). Size, complexity and coupling were commonly used independent variables for maintainability, while maintainability expressed in terms of an ordinal scale based on expert judgment was the most commonly used dependent variable. A subsequent SLR (from 1985 to 2010) by Riaz [5] identified seven primary studies that focused on relational database-driven applications (RDBAs). The results showed little evidence for maintainability prediction for relational database-driven applications. He found that: expert judgment was the most common prediction technique, coupling related measures were the most common predictors, and subjective assessment was the most common dependent variable.

Orenyi et al. [6] conducted a survey on object-oriented (OO) software maintainability using a set of 36 studies published between 2003 and 2012. The authors investigated the use of a quality model, sub-characteristics or measures and techniques, and noted that regression analysis techniques were the most used (31% of the 36 studies). Dubey et al. [7] provided an overview of a set of 21 studies on maintainability techniques for OO systems published between 1993 and 2011. In these latter two studies (not SLRs) the authors did not provide a detailed analysis. Fernandez-Saez et al. [8] conducted a systematic mapping study (SMS) on a set of 38 primary studies (collected from 1997 to 2010) in order to discover empirical evidence related to the use of UML diagrams in source-code maintenance and the maintenance of UML diagrams themselves. They found that "the use of UML is beneficial for source code maintenance, since the quality of the modifications is greater when UML diagrams are available, and most research concerns the maintainability and comprehensibility of the UML diagrams themselves". To explore the use of UML documentation in software maintenance, the authors have published results from a survey of software industry maintenance projects [9] by 178 professionals from 12 different countries. The findings were summarized as follows: "59% indicated the use of a graphical notation and 43% UML, most effective UML diagrams for software maintenance were class, use case, sequence and activity diagrams, the benefits of using UML diagrams result in less time needed for a better understanding and, thus an improved defect detection, and larger teams seem to use UML more frequently in software maintenance".

A summarized context of this related work is presented in Table 1 in terms of: purpose of the study, research or mapping questions addressed, type of study (SLR, SMS or another form of literature review, such as survey, review, etc.), period of collection, and the number of primary studies for each study.

As can be seen from Table 1, while all studies shared an interest in the maintainability of the software, they focused on different aspects or topics within the field. The period of collection and number of primary studies varied among the reviews. Only three studies conducted a rigorous review with SLR and SMS addressing

Table 1. Summarized context of related work

Study ID	Purpose	Research or mapping questions addressed	Type	Period of collection	#of studies
[4]	Understand the state of the art of the software maintainability prediction techniques and metrics.	1) techniques, 2) accuracy measures, 3) independent variables, 4) dependent variables.	SLR	1985–2008	15
[5]	Understand the state of the art of the software maintainability prediction techniques and metrics in RDBAs.	1) techniques, 2) accuracy measures, 3) independent variables, 4) dependent variables.	SLR	1985–2010	7
[6]	Review existing studies in the area of OO software maintainability measurement.	Not provided	Survey	2003–2012	36
[7]	Review of studies on software maintainability model with OO system.	Not provided	Survey	1993–2011	23
[8]	Review of studies on maintenance of UML diagrams and their use in the maintenance of code.	1) UML Diagrams, 2) dependent variable, 3) state of the art, 4) factors	SMS	1997–2010	38
[9]	Survey on the use of UML in software maintenance in order to gather information and opinions from a large population.	Not provided	Survey	February to April of 2013	—

some research or mapping questions. The SMS [8] focused on empirical studies concerning the maintenance of UML diagrams and their use in the maintenance of code. However, the scope of this study was broader and focused not only on UML diagrams but also provided a state-of-the art review of software product maintainability prediction in general. The SLR [4] addressed four research questions (see Table 1), while our study addressed additional questions related to publication trends, publication sources, research types, empirical approaches, software application types, datasets, and tools used to gather these independents variables. Moreover, in our study, to provide answers to the mapping questions, we classified the selected studies according to a set of proposed criteria, whereas study [4] only extracted data for some research questions, presenting them in tables as reported in the primary studies without providing any analysis. Furthermore, none of the previous studies dealt with the accuracy of SPMP techniques whereas our study analyzes and summarizes the evidence regarding prediction accuracy of SPMP techniques as well as identifies the most accurate in comparative studies.

Since the publication of SLRs [4, 5] and SMS [8] studies a number of new empirical studies have been published, some proposing new techniques, such as machine learning techniques, others evaluated existing ones, while still others provided comparative studies to identify the most accurate. Furthermore, since the first SLR on software maintainability was published in 2008, it was important to investigate what further research had occurred since. Moreover, the number of primary studies investigated was very small (from 7 to 15) and the results obtained cannot be conclusive. To establish the state-of-the-art on this topic and reach a certain level of external

validity [4], research published during the last 10 years of studies providing empirical validation of their finding needs to be investigated. This study differs from previous reviews in several ways: it provides an up-to-date state-of-the-art review of SPMP (from 2000 to 2018), the search was conducted on nine digital libraries, a set of 82 primary studies were selected, and classification criteria were proposed for purposes of detailed and precise analysis of the results. A set of eight mapping questions (MQs) were addressed related to: (1) publication year, (2) publication source, (3) research type, (4) empirical approach, (5) software application type, (6) datasets, (7) independent variables (e.g., factors used as predictors) and dependent variables (e.g., how maintainability is expressed in terms of the variable to be predicted), and (8) techniques used, as well as a set of three research questions (RQs) related to: (1) prediction accuracy, (2) techniques reported superior in comparative studies and (3) accuracy comparison of these techniques (see Table 2). Therefore, the objective of this study was twofold:

– to classify SPMP studies according to the proposed criteria (see Table 3), and,
– to analyze and summarize the empirical evidence of SPMP technique accuracy prediction in current research.

The rest of the paper is organized as follows. Section 2 presents the methodology used to conduct the study including the mapping and research questions to be addressed, the research strategy and selection of the primary studies. Section 3 summarizes the results by providing answers to the mapping questions. Section 4 provides the results of the research questions. Section 5 presents the threats to validity of the work. Section 6 offers conclusions and possible future directions.

2. Research methodology

In this study, we used the guidelines of Petersen et al. [10] for conducting systematic reviews, which include planning, conducting and reporting. According to Kitchenham, "Systematic Mapping Studies (SMS) use the same basic methodology as SLRs but aim to identify and classify all research related to a broad software engineering topic rather than answering questions about the relative merits of competing technologies that conventional SLRs address" [11]. In the planning step, the review protocol was developed which describes the procedure for conducting the review. The steps of this protocol are summarized as follows: (1) establishment of a set of mapping and research questions to address the issues related to the review, (2) identification of the search strategy including identification of search terms, selection of sources to be searched, and the search process, (3) selection of the set of primary studies using inclusion and exclusion criteria, (4) mapping of publications by extracting data from each selected study, and (5) data synthesis by grouping the overall results in order to facilitate analysis and provide answers to the mapping and research questions. The protocol was established by holding frequent meetings between authors. A detailed description of each of these steps is provided in the following subsections.

2.1. Mapping and research questions

In addition to our primary motivation to provide and summarize evidence from published empirical studies on SPMP, according to our set of criteria, we identified eight mapping questions (MQs) and three research questions (RQs) – see Table 2.

The MQs are related to the structuring of the SPMP research area with respect to the properties and categories described in Table 3. These categories are defined and explained in the Tables A1 and A2 in the Appendix.

2.2. Search strategy

The search strategy used to identify the primary studies included the following steps: identify the search terms, apply these search terms to electronic databases to retrieve candidate studies, use the search process to ensure that all relevant studies are identified.

Table 2. Mapping and research questions

ID	Mapping questions	Motivation
MQ1	How has the frequency of SPMP studies changed over time?	To identify the publication trend of SPMP studies over time.
MQ2	What are the main publication sources?	To identify what and how many publication sources for SPMP studies.
MQ3	What research types were used?	To identify the different research types used in SPMP studies.
MQ4	What empirical approaches were used?	To identify the empirical approaches that have been used to validate SPMP techniques.
MQ5	What types of software applications were used?	To identify the software application types on which the SPMP studies focused.
MQ6	What datasets were used?	To identify the datasets used for SPMP empirical studies, including the number of projects in the empirical studies.
MQ7	What dependent and independent variables were used?	To identify: A) How maintainability was expressed in terms of the variable to be predicted (e.g., dependent variable). B) What measures or factors were used as predictors (i.e., independent variables) for SPMP. C) Successful predictors for maintainability as reported by the selected studies. D) Tools used to gather predictors.
MQ8	What techniques were used in SPMP?	To identify and classify the techniques used in SPMP studies.
ID	Research questions	Motivation
RQ1	What is the overall prediction accuracy of SPMP techniques?	To identify to what extend the SPMP techniques provide accurate prediction.
RQ2	Which SPMP techniques were reported to be superior in comparative studies?	To identify SPMP techniques reported to be superior in comparative studies.
RQ3	Which of the SPMP techniques reported to be superior in comparative studies also provided greater accuracy?	To compare SPMP techniques that have been reported to be superior in the comparative studies using the same prediction context in terms of accuracy prediction.

2.2.1. Search terms

The search terms were identified based on the MQs and RQs by identifying keywords, synonyms and alternative spellings. The main search terms were: "maintainability", "empirical", "software", "prediction", and "technique". Table 4 provides the main search terms and their alternatives spellings. As can be seen from Table 4, for alternative terms related to maintainability we considered all the maintainability sub-characteristics proposed in the standard ISO 9126 and used in previous SLRs [4, 5].

The search terms were derived using the following series of steps [12]:

- Define the main search terms matching the mapping questions listed above.
- Identify synonyms and alternative spellings for the main terms.
- Use the Boolean OR to concatenate synonymous and alternative terms in order to retrieve any record containing either (or all) of the terms.
- Use the Boolean AND to connect the main terms in order to retrieve any record containing all the terms

The following set of search terms were used to extract the primary studies: "(maintainability **OR** analyzability **OR** modifiability **OR** testability **OR** compliance **OR** stability) **AND** (empirical* **OR** evaluation* **OR** validation* **OR** experiment* **OR** control* experiment **OR** case study **OR** survey) **AND** (software product **OR** software **OR** application **OR** system **OR** soft-

Table 3. Classification criteria

Property	Categories
Research types	Solution proposal (SP), evaluation research (ER)
Empirical approaches	History-based evaluation (HbE), case study (CS), experiment or family of experiments (Ex)
Software application types	Object-oriented applications (OOA), procedure-oriented applications (POA), web-based applications (WbA), service-oriented applications (SOA), component-based applications (CbA), not identified (NI)
Datasets	Software engineering researchers (SER), open source software systems/projects (OSS), private software projects/systems (PSP) dependent variable change, expert opinion, maintainability index, maintainability level, maintainability time, others
Independent variables	Chidamber and Kemerer (C&K), Li and Henry (L&H), class diagram, source code size, McCabe complexity (McCabe), software quality attributes, Martin's measures, Halstead measures, Brito e Abreu and Carapuça (BA&C), factors, coding rule measures, quality model for object-oriented design (QMOOD) measures, maintainability index (MI), web-based application (WbA) measures, Jensen measures, effort measures, sequence diagram, Lorenz and Kidd (L&K) measures, fault measures, database measures
Techniques	Machine learning (ML), artificial neural network (ANN), fuzzy & neuro fuzzy (FNF), regression & decision trees (DT), case-based reasoning (CBR), Bayesian networks (BN), evolutionary algorithm (EA), support vector machine & regression (SVM/R), inductive rule based (IRB), ensemble methods (EM), clustering methods (CM); statistical: regression analysis (RA), probability density function (PD), Gaussian mixture model (GMM), discriminant analysis (DA), weighted functions (WF), stochastic model (SM)

Table 4. Search terms

Main terms	Alternative terms
maintainability	analyzability, modifiability, testability, stability, compliance
empirical	evaluation, validation, experiment, control experiment, case study, survey
software	software product, software, application, system, software engineering
prediction	prediction, evaluation, assessement, estimation, measurement
technique	method, technique, model, tool, approach

ware engineering) **AND** (predict* **OR** evaluat* **OR** assess* **OR** estimat* **OR** measur*) **AND** (method* **OR** technique* **OR** model* **OR** tool* **OR** approach*)"

2.2.2. Literature resources

To search for primary studies, nine relevant and important digital libraries in software engineering used in previous SLRs and SMSs [4, 5, 12] were chosen, which included journals, books, and conference proceedings from: IEEE Xplore, Science Direct, Springer Link, Ebsco, ACM Digital Library, Google Scholar, Scopus, Jstore, and DBLP. The preconstructed search terms established in the first step were applied to this set of nine digital libraries. The search focused on title, abstract and keywords, and ranged from 2000 to 2018.

2.2.3. Search process

To ensure selection of the maximum number of studies related to SPMP, a first round search (automated) was performed using the search terms on each digital library to gather the overall set of candidate studies. A second search round (manual) was performed, which consisted of examining the reference lists of the set of candidate studies in order to identify new candidates based on the title. If the full study was not available, the

authors were contacted to obtain a copy of the published work.

2.3. Study selection

After applying the search process, the full text of the candidate studies retrieved were assessed by two authors according to the following inclusion and exclusion criteria.

- **Inclusion criteria (IC)**: (1) empirical studies addressing prediction or assessment of software product maintainability and/or its sub-characteristics, (2) empirical studies using SPMP techniques.
- **Exclusion criteria (EC)**: (1) studies that discuss the process of software maintenance, (2) studies that concentrate on software maintainability generally and do not present a technique to predict the software maintainability, (3) studies published before 2000, (4) short studies (2–3 pages), (5) secondary studies, and (6) studies by the same author; if results were the same in both studies, the most recent was used, otherwise both studies were used.

The study was retained if it satisfied both inclusion criteria, and rejected if it did not satisfy at least one of the exclusion criteria. Once applied, the decision to retain or reject the study depended on the evaluation of the two authors. In case of doubt or disagreement, a discussion based on review of the full text ensued until an agreement was reached. Duplicate titles and titles out of scope of the review were rejected.

2.4. Study quality assessment

Quality assessment (QA) criteria were used to assess the relevance of the candidate studies. QA is necessary in order to limit bias in conducting mapping and review studies, to gain insight into potential comparisons and to guide the interpretation of findings [12]. The quality of the relevant studies was evaluated based on seven questions as follows:

- **QA1**: Are the objectives of the study clearly described and appropriate?
- **QA2**: Are the factors or measures used as predictors of maintainability defined?
- **QA3**: Are the datasets adequately described?
- **QA4**: Are the SPMP techniques well-presented and defined?
- **QA5**: Is the accuracy criteria well-presented and discussed?
- **QA6**: Is the most accurate technique clearly stated?
- **QA7**: Are the findings of the study clearly stated and presented?

These questions have three possible answers: "Yes", "partially", and "No". These answers are scored as follows: (+1), (+0.5), and (0) respectively. The quality score for each study was computed by summing up the scores of the answers to the QA questions. The maximum score for all questions is 7 and the minimum 0. Studies that scored greater than 50% of the perfect score were considered for the review as in [4, 12]. The QA was performed independently by two of the authors. In the case of disagreement, the two authors discussed the issue until a final consensus was reached. After applying the QA criteria, 82 primary studies with an acceptable quality score (i.e., more than 3.5) were selected. The detailed quality scores for each study are presented in Table A3 in the Appendix.

2.5. Data extraction and data synthesis

A data extraction form was completed with information for each selected primary study to determine which apply to one of more of the mapping or research questions. Two independent researchers performed the extraction. In the case of disagreement, a discussion was held to reach consensus after a thorough review of the study. To facilitate synthesis and analysis of the data, the information collected was tabulated and grouped into a file (see Table 5). Various visualization techniques (such as charts and frequency tables, etc.) were used to synthesize the data, accumulate and combine facts from the selected primary studies in order to formulate answers to the mapping and research questions. A narrative summary reports the principal findings of the study, including collection of a number of studies that state similar and comparable viewpoints.

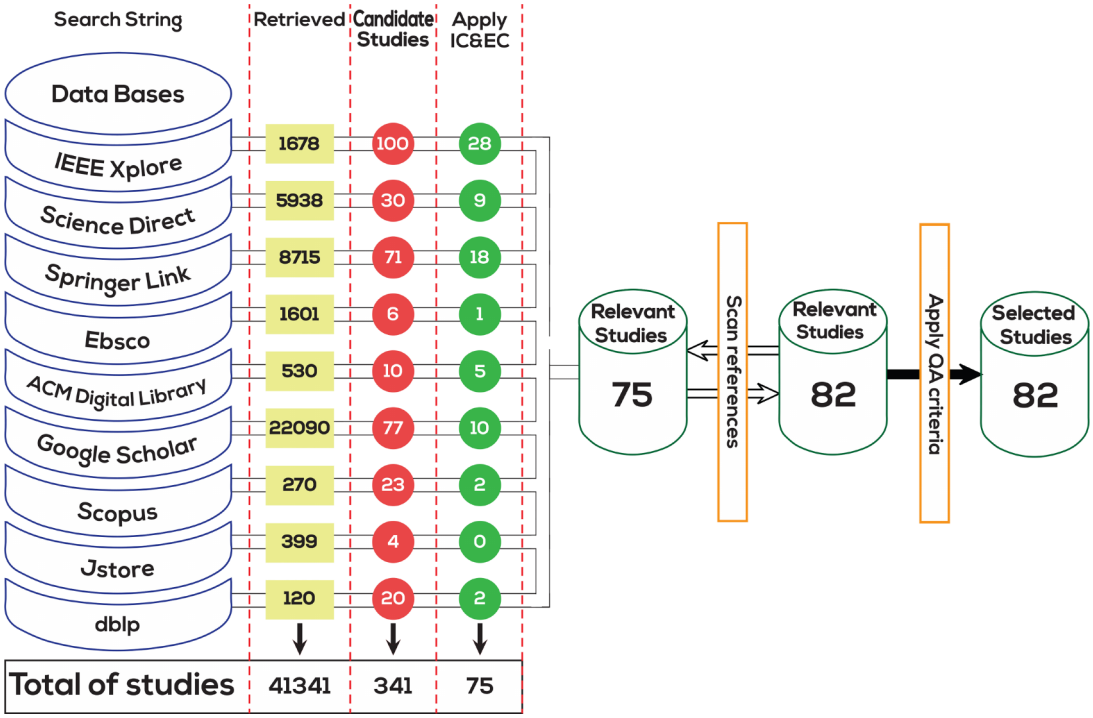

Figure 1. Search process steps and results

3. Mapping Results

To conduct the study the process defined during the planning phase was implemented. Data retrieval, study selection, data extraction, and data synthesis were executed according to the review protocol developed by the authors. To begin with, the protocol was carried out by the first author in order to search studies related to the SPMP area. The first and second author then discussed the candidate studies after removing duplicates. Finally, the selected studies were checked by reading the full text of each study in order to confirm whether the paper was to be included or excluded from the list of primary studies. In cases of disagreement, the authors discussed the studies until an agreement was reached.

Figure 1 presents the search steps together with their corresponding results: (1) Applying the search terms on the nine online databases resulted in 41341 studies, (2) Removing duplicate studies and those not related to the SPMP topic resulted in 341 candidate studies, (3) Applying the inclusion and exclusion criteria resulted in 75 relevant studies, (4) Scanning the list of references and citations resulted in seven more studies for a total of 82 relevant studies (see Table A4 in the Appendix for the summary of the search results). All 82 studies were retained since they had an acceptable quality score (see Table A5 in the Appendix).

This section presents and discusses the results obtained from review of the 82 primary studies by providing answers to the mapping questions (MQ1-8) in the following subsections. The classification of each of the selected studies was based on the established classification criteria (see Table 3, and Tables A1 and A2 in the Appendix) and can be found in Table A6 in the Appendix.

3.1. Publication years (MQ1)

Figure 2 presents the distribution of SPMP studies per year, beginning in 2000. Interest in SPMP increased slowly over the decade from 2003 to 2010, reached a peak in 2012 and in 2017 (10 and 11 studies, respectively) and decreased thereafter while remaining relatively high between 2012 and 2017. Only three studies are shown for 2018 since most of the published studies were

Table 5. Data extraction form

Data extractor
Data checker
Study identifier
Name of database
Publication year
Author name(s)
URL
Article title
MQ2: Publication source
MQ3: Research type (see Table 3 and Table A1 in the Appendix)
MQ4: Empirical approach type (see Table 3 and Table A2 in the Appendix)
MQ5: Software application type (see Table 3)
MQ6: Datasets (see Table 3) – Categories of datasets – Historical datasets: name and number of projects
MQ7: Dependent and independent variables (see Table 3) – Common types of factors or measures used as independent variables (predictors). – Common types of factors or measures used as dependent variables. – Successful predictors of maintainability as reported in the selected primary studies. – Tools (tool name, description).
MQ8: Techniques (see Table 3) – Categories of techniques: statistical and machine learning.
RQ1: Prediction accuracy – Most used accuracy criteria. – Accuracy prediction of SPMP techniques per most used dependent variable topics (identified in MQ7).
RQ2: SPMP techniques reported to be superior in comparative studies – Techniques reported to be superior in comparative studies. – Strengths and weakness of these techniques. – Techniques having been reported to be superior and not.
RQ3: Accuracy comparison of the SPMP techniques identified in RQ2 – Selection of studies under the same prediction context (e.g., dataset, accuracy criteria, etc.). – Accuracy comparison of SPMP techniques under this context. – Selection of the most accurate SPMP techniques.

not yet online at the time the SMS was conducted.

3.2. Publication sources (MQ2)

Table 6 presents the distribution of the selected primary studies over publication sources. Only six journals (IJCA, JC, IST, IJSAEM, ESE, and JSS), six conferences (SIGSOFT, QR2MSE, ICSM, ICRITO, ICACCI, and CSMR) and one symposium (HASE) had more than one selected study. The other publication sources had only one study and have been grouped into others.

Figure 3 shows graphically the distribution of primary studies by source. Of the 82 selected studies, 41 (50%) were published in journals, 34 (42%) at conferences, four (5%) at a symposium, two (2%) in a workshop, and one (1%) a chapter.

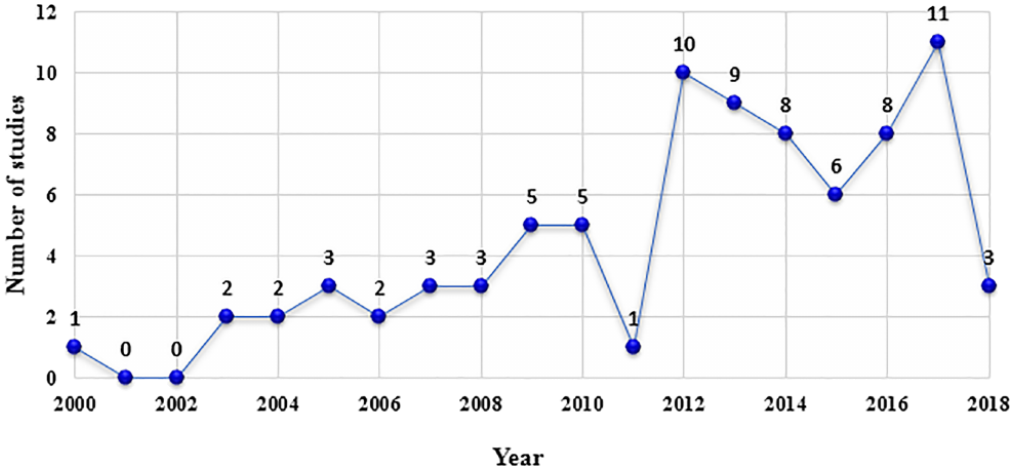

Figure 2. Distribution of selected SPMP studies per year

Table 6. Publication sources

Source	Type	#of studies	Percentage
Information and Software Technology (IST)	Journal	4	5%
Journal of Systems & Software (JSS)	Journal	4	5%
International Journal Computer Applications (IJCA)	Journal	3	4%
Empirical Software Engineering (ESE)	Journal	3	4%
Journal of Computing (JC)	Journal	2	2%
International Journal of System Assurance Engineering and Management (IJSAEM)	Journal	2	2%
SIGSOFT Software Engineering Notes (SIGSOFT)	Conference	2	2%
International Conference on Quality, Reliability, Risk, Maintenance, and Safety Engineering (QR2MSE)	Conference	2	2%
IEEE International Conference on Software Maintenance (ICSM)	Conference	2	2%
European Conference on Software Maintenance and Reengineering (CSMR)	Conference	2	2%
International Conference on Reliability, Infocom Technologies and Optimization (ICRITO)	Conference	2	2%
International Conference on Advances in Computing, Communications and Informatics (ICACCI)	Conference	2	2%
International Symposium on High Assurance Systems Engineering (HASE)	Symposium	2	2%
Others (conference, symposium, journal, chapter, workshop)		1 each source	63%

3.3. Research types (MQ3)

Two main research types were identified from the selected studies: solution proposal (SP) and evaluation research (ER). Figure 4 shows that SP was the most frequently used (48 studies or 59%) followed by ER (34 studies or 41%), indicating that the goal of researchers was to propose new techniques or adapt old ones (SP), and then evaluate and/or compare existing techniques (ER) to improve SPMP. Furthermore, of the 82 selected studies, 41 (50%) conducted comparative studies to identify the most relevant techniques for predicting software product maintainability of which 14 (34%) were SP studies and 27 (66%) were ER studies.

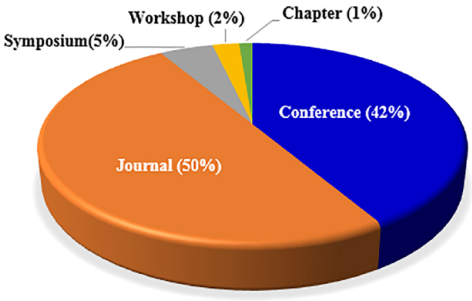

Figure 3. Distribution of primary SPMP studies by publication source

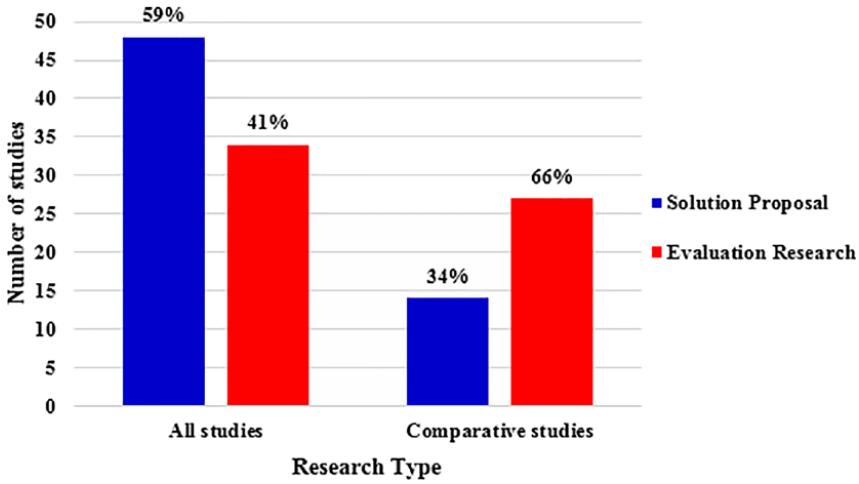

Figure 4. Research types of SPMP studies

3.4. Empirical approaches for validating SPMP techniques (MQ4)

Figure 5 shows the three main empirical approaches used to validate SPMP techniques, which are history-based evaluation (HbE), experiment (Ex), and case study (CS). From Figure 5, HbE and Ex were the most frequently employed approaches: 48 studies (58%) were empirically validated on previously completed software projects (HbE) and 26 studies (32%) were validated under controlled conditions (Ex).

As shown in Table 7, the number of studies using these two approaches has increased over time. Note that only eight out of 82 (10%) of selected studies investigated an SPMP technique in a real-life context through a case study (CS).

3.5. Software application types (MQ5)

To validate SPMP techniques, the selected studies used data from different types of software applications. A set of four main types were identified: object-oriented applications (OOA), procedure-oriented applications (POA), web-based applications (WbA), service-oriented applications (SOA), and component-based applications (CbA).

Figure 6 shows that OOA were the most frequently used with 65 studies (79%), followed by POA and SOA with four studies, each (5%), WbA with two studies (2%), and CbA with one study (1%). The remaining studies, denoted by NI (Not Identified), did not specify the type of software applications studied. The high percentage for OOA to empirically validate SPMP techniques is due to the use of historical datasets (MQ6), most of which involved object-oriented projects. Moreover, based on the distribution of primary studies using empirical approaches (MQ4) by software application type (MQ5), it can be seen in Figure 6 that OOA were frequently used in three empirical approaches: history-based evaluation (HbE) was the most frequently used, followed by experiment (Ex),

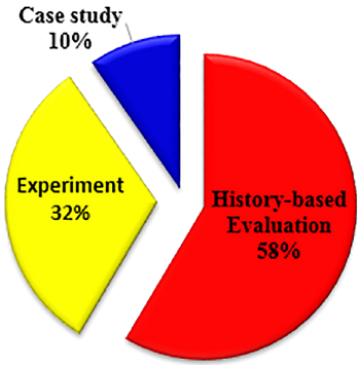

Figure 5. Empirical approaches for validating SPMP techniques

Table 7. Distribution of SPMP empirical approaches per time period

Empirical approach	2000–2005	2006–2011	2012–2018	Total
History-based evaluation (HbE)	2	10	36	48
Experiment (Ex)	5	4	17	26
Case Study (CS)	1	5	2	8

and then case study (CS). Three other software application types were used less frequently: POA was only used in HbE and Ex approaches, WbA was used equally in CS and Ex approaches, SOA was only used in HbE, while CbA was only used in CS.

Figure 7 shows the frequency of research types (MQ3), empirical approaches (MQ4) and software applications types (MQ5). It can be remarked that:

- OOA were the most frequently studied in both research types (31 for evaluation research and 34 for solution proposal),
- POA, WbA, SOA, and CbA software application types were less used (eight studies for solution proposal research), and
- the remaining six studies did not clearly identify the software applications types considered.

Moreover, almost all evaluation research studies (31 of 34 studies) used the HbE evaluation approach while the majority of solution proposal studies used either Experiment (23 studies) or HbE evaluation (17 studies) approaches. The case study approach was less used and only in solution proposal (eight studies).

3.6. Datasets (MQ6)

A variety of datasets from various sources were used in the selected the primary studies. Three main categories of datasets based on their origin were identified:

- Software engineering researchers (SER): Public datasets used by researchers from the software engineering community: UIMS (user interface management system), QUES (quality evaluation system), VSSPLUGIN (visual source safe PLUGIN), PeerSim (peer-to-peer simulator), etc.
- Open source software systems/projects (OSS): Freely available datasets, such as JHotdraw, Jtreeview (Java TreeView), JEdit, Lucene, etc.
- Private software projects/systems (PSP): Private data from large industrial projects, such as: MIS (medical imaging system), FLM (file letter monitoring system), EASY (EASY classes online services collection), SMS (student management System), IMS (inventory management system), APB (angel bill printing), and from academic software projects developed by students, such as bank information system (BIS) and Aggarwal datasets.

Table 8 presents the number and percentage of studies per dataset sources. The PSP datasets were the most frequently used with 32 studies (39%) each, followed by OSS datasets with 27 studies (33%) and SER datasets with 25 studies (30%). Note that some studies may have used more than one dataset. For example, S34 used

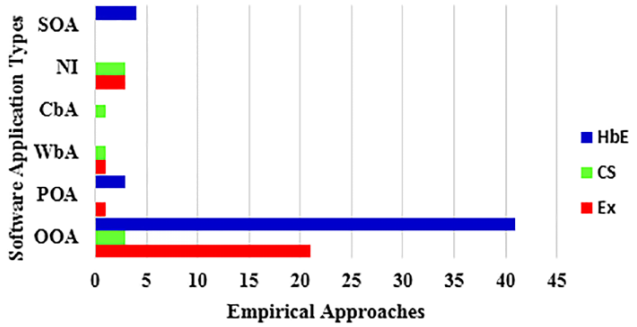

Figure 6. Frequency of software application type per empirical approach

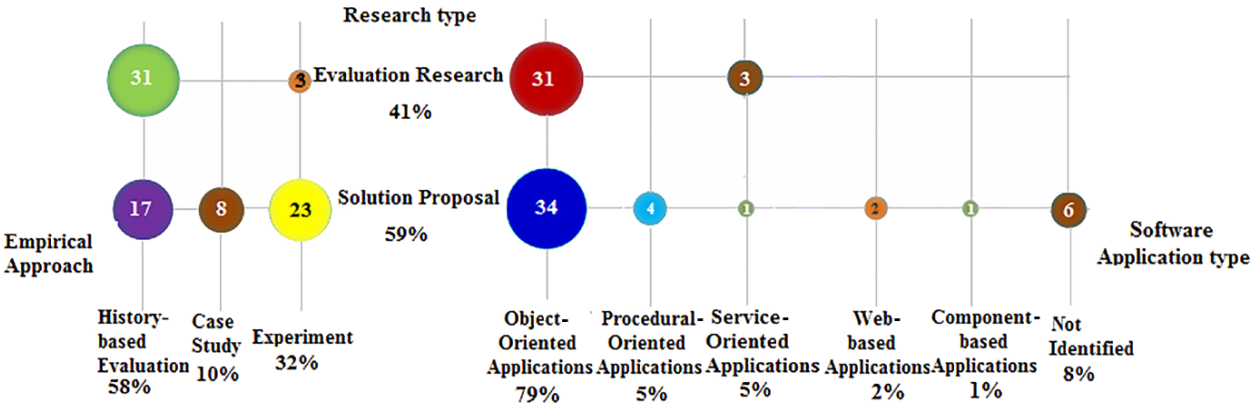

Figure 7. Frequency of research types, empirical approaches and software application types

Table 8. Number of SPMP studies per dataset sources

Dataset sources	Used in	# of studies	Percent
Private software projects (PSP)	S1, S3, S4, S7, S10, S11, S12, S13, S15, S17, S18, S20, S22, S24, S25, S30, S34, S36, S40, S47, S50, S51, S52, S53, S67, S74, S75, S78, S79, S80, S81, S82	32	39%
Open sources software projects (OSS)	S5, S8, S16, S27, S28, S34, S36, S39, S41, S46, S48, S49, S59, S60, S61, S62, S63, S64, S65, S66, S68, S70, S71, S72, S73, S76, S77	27	33%
Software engineering researchers (SER)	S2, S6, S9, S14, S19, S21, S23, S26, S29, S31, S32, S33, S35, S37, S38, S42, S43, S44, S45, S54, S55, S57, S56, S58, S69	25	30%

both PSP and OSS datasets and was counted twice.

Within these dataset sources, some empirical studies used historical data to evaluate and/or compare SPMP techniques with other techniques, referred to as historical datasets. When researchers collect data on their own, they can make it available for future use or not. When the available data is used by other research workers, it is referred to as historical datasets. From the set of 82 selected studies, 48 (which are related to HbE (MQ4)) used historical datasets. Table 9 summarizes the historical datasets used, the number and percentage of the primary studies that used the dataset, the number of projects or classes and the source reference of the dataset. Note that one study may involve more than one dataset and in that case is counted only once. As can be seen from Table 9, among the 48 HbE empirical approaches, the most frequently used historical dataset was UIMS (24 studies) followed by QUES (22 studies), which amounts to 56% for only two relatively small OOA

Table 9. Distribution of HbE empirical approaches over historical datasets

Datasets	# of studies	Percent	# of project	Source
UIMS	24	29%	1 project (39 classes)	[13]
QUES	22	27%	1 project (71 classes)	[13]
JEdit	6	7%	1 project (415 classes)	[14]
eBay	4	5%	1 projet (1524 classes)	[15]
Lucene	3	4%	1 project (385 classes)	[14]
JHotdraw	3	4%	1 project (159 classes)	[14]
Art of Illusion	3	4%	1 project (739 classes)	[16]
jTDS	3	4%	1 project (64 classes)	[17]
BIS	2	2%	1 project (28 classes)	[18]
MIS	2	2%	1 project (4500 modules)	[19]
JUnit	2	2%	1 project (251 classes)	[20]
Ivy	2	2%	1 project (614 classes)	[16]
Camel	2	2%	1 project (422 classes)	[16]
Eclipse	2	2%	1 project (10 594 classes)	[16]
FLM	2	2%	1 project (55 classes)	[21]
EASY	2	2%	1 project (84 classes)	[21]

datasets of one project each. While this creates a limitation in terms of bias in the evaluation of numerous studies, it permits a basis for comparison across findings using the same dataset. Datasets that were used in two to four studies included: JEdit, eBay, JHotdraw, jtds, Lucene, Art of Illusion, Eclipse, MIS, FLM, BIS, Ivy, Junit, Camel, and EASY. The remaining datasets were used in only one study each (not included in Table 9).

Furthermore, all these datasets (4th column) developed software projects using the object-oriented paradigm (including classes, methods, attributes, polymorphism, etc.), except MIS and Aggarwal datasets which developed software projects using the procedure-oriented paradigm (POA) and eBay software applications using the service-oriented paradigm.

Figure 8 is extracted from Table 9 and includes only software engineering researchers and open source datasets from publicly available industrial or professional datasets, such as: UIMS, QUES, JEdit, Lucene, JHotdraw (no private or student datasets were included). For instance, the two popular datasets published by Li and Henry [13] (UIMS and QUES), which are frequently used in predicting maintainability, are OO commercial systems developed using the Ada programming language. The other datasets (JEdit, Lucene, JHotdraw, Art of Illusion, jTDS, JUnit, Ivy, Camel, Eclipse, and eBay) are OO systems implemented in Java. The public availability of these datasets allows researchers and practitioners to conduct verifiable, repeatable, comparatives studies [22], provided that they use the same prediction context (e.g., dependent and independent variables, datasets, accuracy criteria, and validation method).

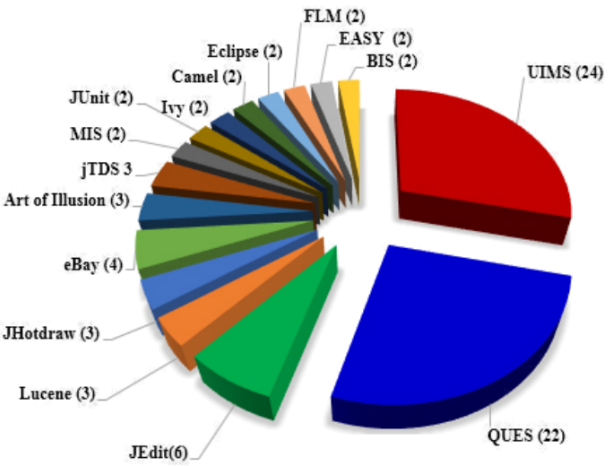

Figure 8. Historical datasets used for SPMP studies

3.7. Dependent and independent variables used in SPMP studies (MQ7)

This section identifies and discusses the dependent variables and the measures used to express maintainability (predicted output). It then presents the

Table 10. Classification of the dependent variables

Topic	Sub-topic	Supported studies	# of studies
Change	Changes in LOC	S2, S6, S9, S14, S19, S21, S23, S24, S26, S29, S31, S32, S33, S35, S37, S38, S42, S43, S44, S45, S47, S48, S49, S52, S54, S55, S56, S57, S58, S59, S60, S61, S62, S64, S65, S66, S69, S70, S71, S72, S73, S74, S77	46
	Change in module	S10, S15	
	Change in class	S30	
Expert opinion	Expert opinion based on ordinal scale	S11, S18, S20, S25, S27, S28, S36, S41, S50, S67	10
Maintainability index	Maintainability index	S8, S16, S48, S68, S75	8
	Relative maintainability index	S39, S76	
	Maintainability index satisfaction	S40	
Maintainability level	Understandability level, modifiability level, analyzability level	S4, S22, S51, S53, S78, S80, S82	7
Maintainability time	Understandability time	S3, S12, S78	8
	Modifiability time	S3, S12, S78	
	Completion time of understandability	S80, S82	
	Time to repair the design of a structure	S17	
Other	Maintainability expressed in terms of number of revised lines of code and number of revisions	S46, S63	2
	Maintainability efficiency	S79	1
	Maintainability effectiveness	S79	1
	Understandability effectiveness	S81	1
	Modifiability effectiveness	S81	1
	Understandability efficiency	S81	1
	Modifiability efficiency	S81	1
	Modifiability completeness	S3	1
	Modifiability correctness	S3	1
	Error prone modules	S1	1
	Detected fault	S13	1
	Maintainability measured using probabilistic quality model	S34	1
	WbA maintainability	S5	1
	Perceived maintainability	S7	1

factors or measures used as independent variables (predictors), the tools used to gather them and the reported successful predictors of software product maintainability from the 82 primary studies.

3.7.1. Dependent variables

The dependent variable (predicted output), software maintainability, was measured differently in the 82 selected studies. As shown in Table 10 and Figure 9, we identified five main research topics related to maintainability (or its sub-characteristics). Other less used research topics were also identified, but are not included in Figure 9. The scope of this review included the maintainability sub-characteristics as identified by ISO 9126 [23] or its successor ISO 25010 [24], such as: changeability, modifiabil-

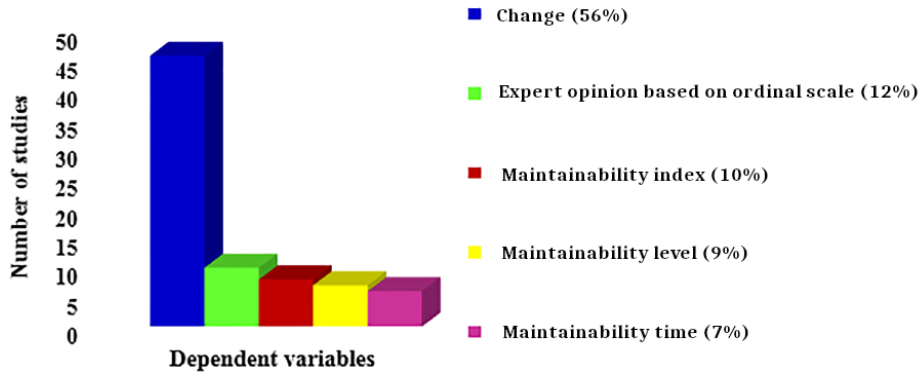

Figure 9. Distribution of selected SPMP studies per most used dependent variable

ity, stability, analysability, testability, modularity, and reusability, or as defined by a particular study (S4, for example, identified two sub-characteristics of maintainability: understandability and modifiability).

As shown in Table 10:

- The topic most frequently referred as the dependent variable is change, 46 studies (56%):
  - Changes in LOC studies used the number of lines of code changed per class by counting the number of lines in the code that were changed.
  - Changes of modules studies used the changes made to each module due to faults discovered during system testing and maintenance.
  - Changes of classes studies used the change of an attribute, a method or a class affected by decomposition of the system and its sub-systems.
- The second topic referred to studies that predict SPM based on expert opinion: 10 studies (12%) expressed maintainability using an ordinal scale based on expert opinion. The maintainability was qualified as: poor, average, very good, or very high, high, medium, low, or excellent, average, bad, etc.
- The third topic referred to studies that used a maintainability index (MI) to determine the maintainability of the software product (eight studies – at 10%). Some studies used the maintainability index calculated as a factored formula of average Halstead volume per module, average extended cyclomatic complexity, average lines of code, and average percent of lines of comments per module measures. Some studies used relative maintainability index calculated for each source code element for which metrics were calculated (e.g. methods, classes) using the goodness value. Other studies used the maintainability index satisfaction expressed in terms of maintenance time satisfaction, maintenance man-hour satisfaction, and maintenance cost satisfaction.
- The fourth topic referred to studies that predict maintainability in terms of understandability, modifiability and analyzability levels, which are evaluated based on the subject's difficulty to: understand the system, carry out modification tasks, and diagnose the system (seven studies – 9%).
- The fifth topic refers to studies that predict the maintainability in terms of understandability time, and/or modifiability time spent by subjects answering the understandability questions or understanding source code and carrying out modifications, or the time to repair the design of structure (six studies – 7%).
- The other research maintainability topic included less used dependent variables such as: modifiability completeness, modifiability correctness, number of revised lines of code and number of revisions, maintainability efficiency, maintainability effectiveness, understandability effectiveness, modifiability effectiveness, understandability efficiency, modifiability efficiency, error prone modules, detected fault, maintainability measured using a probabilistic quality model, WbA maintainability, and perceived maintainability.

3.7.2. Independent variables

In order to predict software product maintainability, the selected primary studies used various factors or measures as independent variables (or predictors) i.e., different inputs to the SPMP techniques. This subsection presents the independent variables used, those most used as predictors and the tools used to collect them. For the remainder of this paper, the terms independent variables and predictors will be used interchangeably. Table A7 in the Appendix provides the full list of the predictors used, the corresponding total number of frequencies, supported studies and percentage.

For the 82 primary studies, Chidamber and Kemerer (C&K) measures were the most used (50 studies – 61%), followed by

- Li and Henry (L&H) measures (33 studies – 40%),
- Class diagram measures (24 studies – 29%), which included measures related to method, attribute, class, or relationships (associations, aggregations, generalization and dependency),
- Source code size measures using different lines of code (LOC) measures (20 studies – 24%),
- McCabe cyclomatic complexity (17 studies – 21%), and
- Software quality attributes (such as stability, changeability and analyzability, readability of source code, document quality, understandability of software, simplicity, accessibility, etc.) (eight studies – 10%).

The least used predictors included measures such as: factors, Lorenz and Kidd (L&K) measures, coding rule measures, maintainability index (MI), web-based application (WbA), sequence diagram measures (scenarios, messages and conditions), Martin's measures, QMOOD measures, Fault, database measures, Halstead measures, and Brito e Abreu and Carapuça (BA&C), etc.

Figure 10 shows the number of studies for the most frequently used predictors. Note that one study may involve more than one type of predictor. Figure 10 is extracted from Table A7, while the least used predictors were discarded.

Furthermore, it was observed that object-oriented measures were the most used predictors. This is mainly due to the wide use of object-oriented software applications (OOAs) in SPMP empirical studies, i.e., 65 out of the 82 selected studies (see Section 3.5 Figure 6 and 7).

As shown in Figure 11, the frequently used OO measures were RFC (response for a class) and LCOM (lack of cohesion in methods), followed by WMC (weighted methods per class), DIT (depth of inheritance tree), NOC (number of children), LOC (lines of code or size1), MPC (message passing coupling), NOM (number of local methods), DAC (data abstraction coupling), Size2 (number of properties), and CBO (coupling between object). Such types of measures were collected at the design or source code levels.

Table 11 presents the list and description of the tools used to compute these measures, as well as the primary studies that used them. Note that only 46 out of the 82 studies provided information on the data collection tools used. The Classic-Ada metrics analyzer was the most commonly used (24 studies), followed by Chidamber and Kemerer Java Metric (CKJM) tool (eight studies), Intellij IDEA tool (four studies), LocMetrics tool (three studies), Krakatau Professional tool and Understand tool (two studies each), and one study each for Columbus tool, web application reverse engineering (WARE) tool, Analyst4j standalone tool, COIN tool, ObjectAid UML Explorer, JHawk tool, JDepend tool, Classycle tool, SourceMeter static code analysis tool, Customized tool, and C and C++ code counter (CCCC) tool. Four other studies used their own private tools.

Regarding successful predictors of SPM, 25 (30%) of the 82 selected studies explicitly reported useful measures for software product maintainability based on empirical evaluation – see Table 12:

- Chidamber & Kemerer and Li & Henry measures (DIT, NOC, WMC, RFC, CBO, LCOM, MPC, DAC, NOM, SIZE1, and SIZE2) reported good correlation with maintainability in 14 studies (17%).

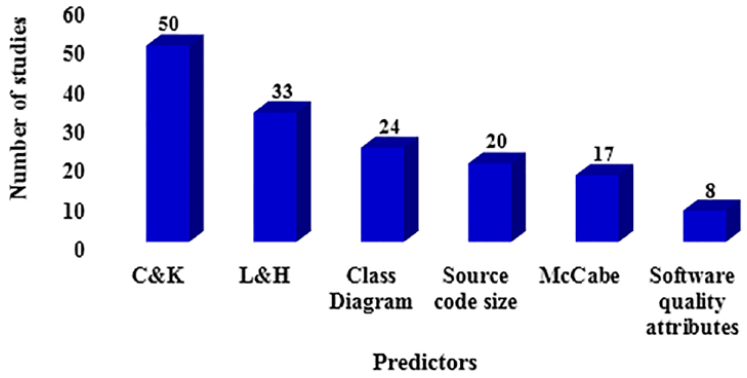

Figure 10. The number of the SPMP studies for the most frequently used predictors

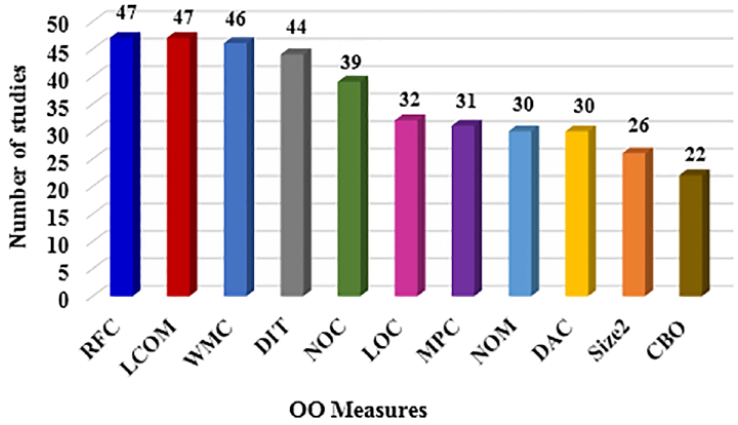

Figure 11. The number of SPMP studies for the most frequently used OO measures

Table 11. Tools used to collect measures

Name	Description	ID
Classic-Ada metrics analyzer	Classic-Ada was implemented in LEX and YACC UNIX environments and was designed on the Mach operating system running on a NeXTstation using a GNU C compiler. The system was ported to an Ultrix system running on a VAX station [13].	S2, S6, S9, S14, S19, S21, S23, S26, S29, S31, S32, S33, S35, S37, S38, S42, S43, S44, S45, S54, S57, S56, S58, S69
CKJM	Chidamber and Kemerer Java Metric extraction tool is freely available. It calculates C&K metrics by processing the bytecode of Java files [25].	S48, S49, S55, S59, S61, S66, S71, S72
Intellij IDEA	Intellij IDEA is a free and open source Java IDE developed by JetBrains and available as Apache 2 licensed and community edition [26].	S48, S49, S62, S64
LocMetrics	LocMetrics[1] counts total lines of code, blank lines of code, comment lines of code, lines with both code and comments, logical source lines of code, McCabe VG complexity, and number of comment words	S66, S71, S72
Krakatau Professional	Krakatau Professional was developed by Power Software Inc. It is a fully-featured software metrics tool designed for source code quality and software measurement specialists [27].	S8, S41

Table 11 continued

Name	Description	ID
Understand	Understand[2] is very efficient at collecting metrics about the code and providing different ways for you to view it. There is a substantial collection of standard metrics available as well as options for writing custom metrics.	S73, S77
Columbus	Columbus is a framework that supports project handling, data extraction, data representation, data storage and filtering [28].	S39
WARE	WARE is an integrated tool that automatically extracts information from the application and allows more abstract representations to be reconstructed [29].	S5
COIN	Cohesion Inheritance (COIN) is a tool for evaluating cohesion, inheritance and size metrics of class hierarchies in Java projects [30].	S68
Analyst4j standalone tool	Analyst4j is based on the Eclipse platform. It features search, metrics, analyzing quality, and report generation for Java programs [31].	S28
ObjectAid UML Explorer	ObjectAid UML Explorer[3] has been used to extract the UML diagrams from the Java source code.	S63
JHawk	JHawk[4] is a general-purpose metrics collection tool that calculates a variety of metrics from OO systems.	S63
JDepend	JDepend[5] has been used to generate design quality metrics for each package in the system and verify the relations between the packages.	S63
Classycle	Classycle's Analyser tool[6] analyzes the static class and package dependencies in Java applications.	S63
SourceMeter	SourceMeter[7] is an innovative tool built for precise static source code analysis of C/C++, Java, C#, Python, and RPG projects. This tool makes it possible to find the weak spots of a system under development from the source code only, without the need to simulate live conditions.	S76
CCCC	CCCC[8] is a free software tool by Tim Littlefair for measurement of source code related metrics.	S52
Customized tools	Customized tools have been implemented to integrate and analyze data from previous tools and to compute the new coupling, instability and abstractness metrics for every package in the system [32].	S63
Private	Tools constructed and developed for each study according to the context to automatically collect metrics.	S4, S5, S46, S52

[1]http://www.locmetrics.com
[2]http://www.scitools.com
[3]http://www.scitools.com
[4]http://www.virtualmachinery.com/jhawkprod.htm
[5]http://clarkware.com/software/JDepend.html
[6]http://classycle.sourceforge.net/
[7]http://www.sourcemeter.com/
[8]http://cccc.sourceforge.net/

– Class diagram measures (NA, NM, NC, NAgg, NGenH, NAssoc, NDep, MaxDIT, NGen, and NAggH) were found to be useful in predicting maintainability in five studies (6%).
– The other measures were reported useful in two or one study each.

The remaining 36 studies did not report useful measures, since most were interested in com-

Table 12. Successful predictors of SPM in 25 of the SPMP studies

Successful predictors	Supported by
DIT, NOC, WMC, RFC, CBO, LCOM, MPC, DAC, NOM, SIZE1, SIZE2	S6, S8, S9, S14, S21, S32, S46, S47, S48, S49, S52, S58, S60, S66
NA, NM, NC, NAgg, NAggH, NGen, NGenH, NAssoc, NDep, MaxDIT	S3, S12, S22, S51, S68
MI, CC, NODBC, SCCR	S52, S68
TWP, TLOC, WO, SS, ClS, TL, TCC, TWPR, TWPDC	S5, S68
Coding effort, RDCRatio	S7
Average fan-out, data flow, average McCabe	S1
ACLOC, AMLOC, AVPATHS, CDENS, COF, n, N, PPPC	S8
NPAVGC, OSAVG, CSA, SWMC, POF	S16
LLOC, McCabe, rule violations	S39
NOA, Coh, CAMC, LCC, LSCC, SCOM, PCCC, OL2, CBO_U, CBO_IUB, OCMEC, TCC	S46, S68
B, CLOC, Command, CONS, CSA, CSO, Cyclic, Dcy, NAA, OCmax, OSmax, SLoc, STAT, V, Query	S48
B, CLOC, Command, Inner*, Dcy*	S49
NclienP, NAggR, NAssoc, NservP, NwebP	S53
LCOM3, LOC, Ce	S60
NPM, Ca, DAM, MOA	S66
MIF, AIF, DCi, Coh, DCd	S68

paring the accuracy of their proposed or evaluated SPMP techniques rather than in identifying successful predictors. See Table A8 in the Appendix for the acronyms of the successful predictors.

3.7.3. Summary

Table 13 presents the predictors (independent variable) used by each maintainability research topic.

- Studies focusing on predicting maintainability in terms of change used mainly C&K and L&H measures, and in particular, change expressed in terms of number of LOC changed in a class. This was because the datasets used (e.g., UIMS, QUES, FLM, EASY, and Lucene, etc.) focused on OO software applications.
- Studies on maintainability based on expert opinion using an ordinal scale used quality attributes, such as readability of source code, document quality, stability, changeability, analyzability as dependent variable, or measures related to source code size, McCabe, C&K, class and coding rules.
- Studies on maintainability index or relative maintainability index used C&K, source code size, Halstead, class, Lorenz and Kidd, Brito e Abreu and Carapuça, and McCabe measures, while the maintainability index satisfaction used satisfaction attributes.
- Studies on maintainability level in terms of sub-characteristics (understandability, modifiability and analysability) used class diagram as well as sequence diagram measures and factors.
- Studies on maintainability time used class diagram measures for understandability time and modifiability time, while some used software quality attributes.
- Most of the remaining topics used class diagram, source code size, as well as factors and McCabe measures.

Furthermore, some studies, including S6 and S8, reported that C&K and L&H measures (which are related to OO design attributes such as coupling, cohesion and inheritance) were statistically significant and highly correlated to maintainability. Note also, that C&K and L&H measures as predictors were most often used to pre-

Table 13. Type of independent variable by dependent variable topic

Topic (dependent variable)	Predictor measures (independent variables)
Change	C&K, L&H, McCabe, maintainability index, database, class, Halstead, source code size, Jensen, effort
Expert opinion based on ordinal scale	Quality attributes, source code size, McCabe, coupling, C&K, class, coding rules
Maintainability index	C&K, source code size, Halstead, class, Lorenz and Kidd, Brito e Abreu and Carapuça, McCabe, quality attributes
Maintainability level	Class diagram, sequence diagram class
Maintainability time	Class diagram, quality attributes
Modifiability correctness	Class diagram, factors
Modifiability completeness	Class diagram, factors
Maintainability efficiency	factors
Maintainability effectiveness	factors
Understandability effectiveness	factors
Modifiability effectiveness	factors
Understandability efficiency	factors
Modifiability efficiency	factors
Error prone modules	McCabe, module level
Detected fault	fault

dict maintainability expressed in terms of change as predicted output.

3.8. Techniques used in SPMP studies (MQ8)

From the 82 selected primary studies we identified two major categories of techniques that have been applied to predict software product maintainability: machine learning (ML) and statistical techniques. Figure 12 shows that ML techniques were the most frequently used, being adopted by 70% (57 studies) compared to statistical techniques with 51% (42 studies). Note that we include all studies using single techniques in the review results section. Note, too, that a study may use techniques from the two categories (more details in Table A6 in the Appendix).

The statistical techniques include regression analysis (RA), probability density function (PD), gaussian mixture model (GMM), discriminant analysis (DA), weighted functions (WF) and stochastic model (SM):

- **RA** were the most frequently used statistical techniques with 35%. This category includes: Linear Regression (LR), Multiple Linear Regression (MLR), Logistic Regression (LgR), Backward Elimination (BE), Stepwise Selection (SS), Multiple Adaptive Regression Splines (MARS), Projection Pursuit Regression (PPR), polynomial regression (PR), Least Median of Squares Regression (LMSR), Pace Regression (PaceR), Isotonic Regression (IR), Regression by Discretization (RegByDisc), Additive Regression (AR), Gaussian Process Regression (GPR), and Least Absolute Shrinkage and Selection Operator (Lasso), followed by
- **PD** with 4%, **SM**, **GMM**, **DA** and **WF** with 1% each.

The ML techniques were categorized according to [33, 34] as follows: Artificial Neural Network (ANN), Fuzzy & Neuro Fuzzy (FNF), Regression & Decision Trees (DT), Ensemble Methods (EM), Case-Based Reasoning (CBR), Bayesian Networks (BN), Evolutionary Algorithm (EA),

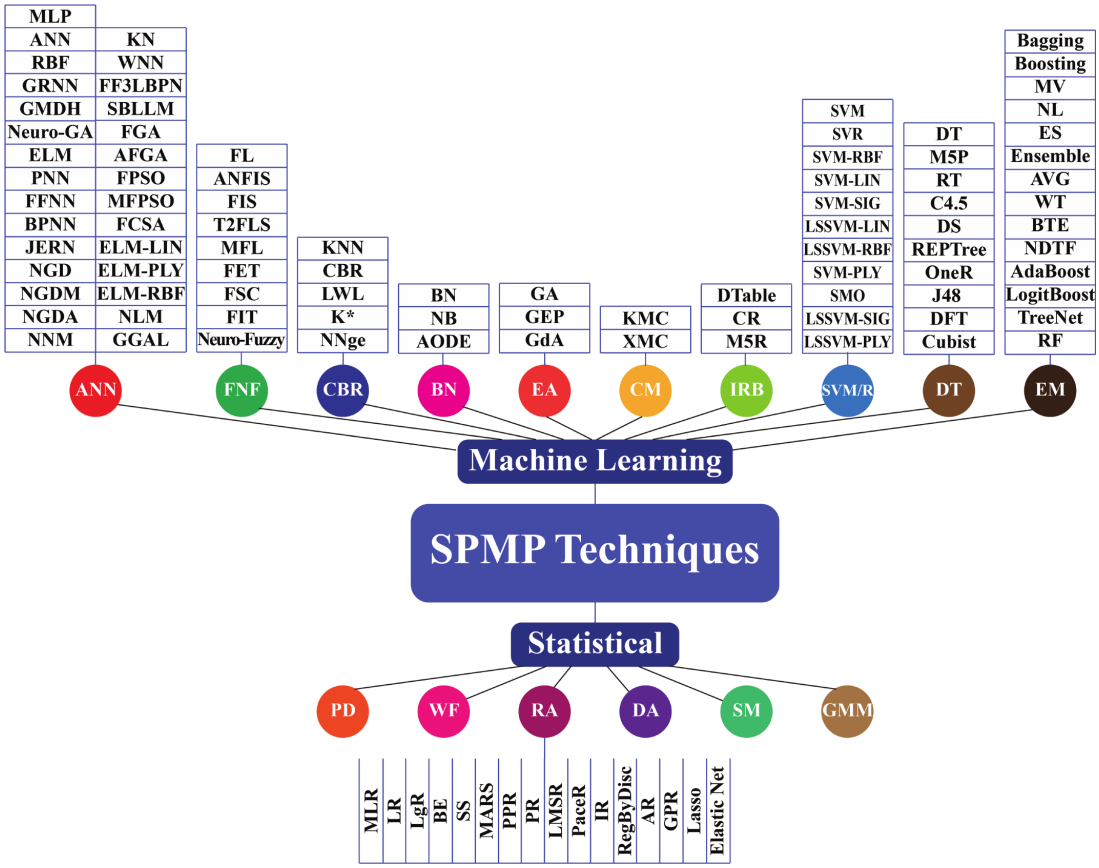

Figure 12. Techniques used in SPMP studies

Support Vector Machine & Regression (SVM/R), Inductive Rule Based (IRB), and Clustering Methods (CM).

– **ANN** were the most used techniques with 38%. It includes Multilayer Perceptron (MLP), Radial Basis Function Network (RBF), Probabilistic Neural Network (PNN), Group Method of Data Handling (GMDH), General Regression Neural Network (GRNN), Feed Forward Neural Network (FFNN), Back Propagation Neural Network (BPNN), Kohonen Network (KN), Ward Neural Network (WNN), Feed Forward 3-Layer Back Propagation Network (FF3LBPN), Extreme Learning Machines (ELM), Sensitivity Based Linear Learning Method (SBLLM), Neuro-Genetic Algorithm (Neuro-GA), Functional Link Artificial Neural Network (FLAAN) with Genetic Algorithm (FGA), Adaptive FLANN-Genetic (AFGA), FLANN-Particle Swarm Optimization (FPSO), Modified-FLANN Particle Swarm Optimization (MFPSO), FLANN-Clonal Selection Algorithm (FCSA), ELM with Linear (ELM-LIN), ELM with Polynomial (ELM-PLY), ELM with Radial Basis Function Kernels (ELM-RBF), ANN with Levenberg Marquardt Method (NLM), GRNN with Genetic Adaptive Learning (GGAL), Jordan Elman Recurrent Network (JERN), ANN with Normally Gradient Descent Method (NGD), ANN with Gradient Descent With Momentum (NGDM), ANN with Gradient Descent With Adaptive Learning Rate (NGDA) and ANN with Quasi-Newton Method (NNM).
– **SVM/R** with 24%, includes Support Vector Machine (SVM), Support Vector Regression (SVR), Sequential Minimal Optimization (SMO), SVM with Radial Basis Function Kernel (SVM-RBF), SVM with Linear Kernel (SVM-LIN), SVM with Sigmoid Kernel (SVM-SIG), Least Square Support Vector Machine (LSSVM) with Linear Kernel (LSSVM-LIN), LSSVM with Radial Basis Function Kernel (LSSVM-RBF), SVM with

Polynomial Kernel (SVM-PLY), LSSVM with Sigmoid Kernel (LSSVM-SIG) and LSSVM with Polynomial Kernel (LSSVM-PLY).
- **FNF** with 20%, includes Fuzzy Logic (FL), Adaptive Neuro-Fuzzy Inference Systems (ANFIS), Fuzzy Inference Systems (FIS), Type-2 Fuzzy Logic System (T2FLS), Mamdani-based Fuzzy Logic (MFL), Fuzzy Entropy Theory (FET), Fuzzy Subtractive Clustering (FSC), Fuzzy Integral Theory (FIT), and Neuro-Fuzzy.
- **DT** with 18%, includes Regression Tree (RT), M5 For Inducing Trees of Regression Models (M5P), Decision Stump (DS), Reduced Error Pruned Tree (REPTree), Decision Tree Forest (DFT), C4.5, OneR, J48, and Cubist.
- **EM** with 15%, includes Ensemble Selection (ES), Average-based Ensemble (AVG), Weighted-based Ensemble (WT), Best-in-Training-based Ensemble (BTE), Majority-Voting Ensemble (MV), Non-Linear Ensemble (NL), Nonlinear Ensemble Decision Tree Forest (NDTF), Adaptive Boosting (Adaboost), Bagging, Boosting, Ensemble, Random Forest (RF), TreeNet, and LogitBoost.
- **BN** with 7%, includes Naive-Bayes (NB) and Aggregating One-Dependence Estimators (AODE).
- **CBR** with 6%, includes Kstar (K*), Locally Weighted Learning (LWL), *K* Nearest Neighbor (IBK or KNN), and Nearest-Neighbor-Like algorithm that uses Non-Nested generalized exemplars (NNge).
- **EA** with 6%, includes Genetic Expression Programming (GEP), Genetic Algorithm (GA) and Greedy Algorithm (GdA).
- **IRB** with 4%, includes Decision Table (Dtable), Conjunctive Rule Learner (CR), and M5 Rules (M5R).
- **CM** with 2%, includes K-Means Clustering (KMC) and x-Means Clustering algorithm (XMC).

4. Review Results

This section presents and discusses the results of this review by providing answers to the three research questions (RQ1-3) in Table 2. Through these questions, the following subsections analyze the SPMP techniques from three perspectives: prediction accuracy, techniques reported superior in comparative studies and accuracy comparison of the techniques. Note that only studies with consistent results about accuracy have been taken into account, thereby excluding S56.

4.1. Prediction accuracy (RQ1)

From the results of MQ7, change, expert opinion, maintainability index, maintainability level, and maintainability time were the most used dependent variable topics (i.e., measures used to express maintainability, the predicted output) from a set of 74 selected SPMP studies. Table A9 in the Appendix shows the details of the SPMP techniques, the accuracy criteria used, and the mapping to the corresponding studies, grouped by the most addressed dependent variable topics. As can be seen, different accuracy criteria were used such as: mean magnitude of relative error (MMRE), percentage relative error deviation (Pred(25) and Pred(30)), coefficient of correlation R, Coefficient of determination (R-squared), root mean square error (RMSE), normalized RMSE (NRMSE), mean absolute error (MAE), mean absolute relative error (MARE), magnitude of relative error (MRE), accuracy, precision, weighted average precision (WAP), recall, F-measure, specificity, etc., where MMRE, Pred(25) and Pred(30) were the most dominant. MMRE measures the mean of the difference between the actual and the predicted value based on the actual value, while Pred measures the percentage of predicted values that have an MRE less than or equal to 0.25 or 0.30 [3].

Note that we included studies that used MMRE and/or Pred to evaluate prediction accuracy in this research question. Topics for which there was no MMRE or Pred were discarded. Note too, that low MMRE or high Pred(25) or Pred(30) values indicated good prediction accuracy [35, 36].

Change: Selected studies on the change topic (including changes of lines in the code, or changes made to each module, or changes of an attribute,

a method or a class to predict the maintainability of a software) used MMRE, Pred(25), and Pred(30) in 16 out of 44 studies as accuracy criteria. We also looked into the average performance of the different prediction techniques. As shown in Figure 13, FNF had the lower value in terms of MMRE and the highest value in terms of Pred(30), ANN had the highest value in terms of Pred(25). Moreover, FNF provided greater accuracy in terms of MMRE and Pred(30). The remaining studies (24 out of 44) used different accuracy criteria such as R-squared, R, MAE, MARE, RMSE, NRMSE, precision, recall, F-measure, specificity, accuracy, etc., while four studies did not provide the accuracy criteria used (see Table A9 in the Appendix for more details).

Maintainability index: Eight studies used the maintainability index for prediction accuracy. Most studies under this topic used various accuracy criteria such as: coefficient of correlation, R-squared, adjusted R-squared, standard error of the estimate and Spearman's coefficient of correlation (Rs), etc. Only study S68 used MMRE, and Pred(30), while study S16 used MMRE as accuracy criteria. Note that a set of 105 experiments were performed in S68 and S16.

The distribution of prediction performance of these two studies is shown in Figure 14 in terms of MMRE and Pred(30). The MMRE ranged from 1% to 100%, while the Pred(30) varied from 40% to 100%.

Maintainability time: All studies (8) under this topic predicted maintainability in terms of understandability time, and/or modifiability time while performing tasks related to maintainability. Accuracy was evaluated using various accuracy criteria such as: R-squared and qMRE, etc. One study (S3) used MMRE and Pred(30) as accuracy criteria in three experiments and the RA (MLR) technique to predict maintainability time.

Table 14 shows its prediction accuracy as well as prediction context. The average MMRE was 70% and the average Pred(30) was 38%. The result shows that the experiment using Spain data had the highest accuracy.

4.2. SPMP techniques reported to be superior in comparative studies (RQ2)

From the results of MQ3, comparative studies about SPMP techniques presenting better performance were identified. Table 15 shows the details of these studies in terms of compared techniques and the results of the comparison; that is the techniques reported to be superior. The comparative studies proposed and/or evaluated SPMP techniques, and then compared them together or with other published studies such as: S2, S6, S9, S23, S26, S32, S37, and S38 (Table 15, second column).

As can be seen from Table 15 (third column), the MLP technique was reported superior in six studies, SVM was reported superior in four studies, GMDH, BN and ELM were reported to be superior in three studies, DT, MARS, BN, Neuro-GA, GEP, GA, and LSSVM techniques were reported to be superior in two studies each, and the rest of the techniques were reported only once.

Table 15. Summary of SPMP techniques reported to be superior

ID	Compared techniques	Techniques reported superior
S2	GRNN, WNN	GRNN
S6	BN, RT, BE, SS	BN
S9	MARS, MLR, ANN, SVR, RT	MARS
S10	GMM, SVM-RBF, DT	GMM
S15	AODE, SVM-LIN, NB, BN, KNN, C4.5, OneR, RBF	AODE
S19	PPR, ANN, MARS	PPR
S23	ANFIS, FFNN, FIS, RBF, GRNN	ANFIS
S26	ELM, RT, BE, SS, BN (S6)	ELM
S29	MLP, WNN, GRNN (S2)	MLP

Table 15 continued

ID	Compared techniques	Techniques reported superior
S33	GMDH, GA, PNN, BN, RT, BE, SS (S6), MARS, MLR, ANN, RT, SVR (S9), GRNN, ANFIS (S23)	GMDH, GA, PNN
S35	MLP, WNN (S2)	MLP
S36	DT, LR, ANN	DT
S38	MLP, SVM, RBF, M5P	MLP, SVM
S41	DT, BPNN, SVM	BPNN
S42	MFL, ANFIS, SVM, PNN, RBF, BN (S6), MARS (S9)	MFL
S43	SBLLM, ELM, RT, BE, SS, BN (S6)	SBLLM, ELM
S44	K*, FSC, PR, KNN, MLR, LMSR, PPR, IR, RegByDisc, GPR, MLP, RBF, GRNN, GMDH, SVR, M5R, AR, ANFIS, DS, M5P, REPTree, LWL, CR, DTable, MARS (S9)	K*, FSC
S45	XMC, KMC	XMC
S47	GMDH, GRNN, FF3LBPN	GMDH
S48	SS, BE, Oman & Hagemeister model [37]	SS, BE
S49	NB, BN, LgR, MLP	BN, MLP
S52	KN, MLR, BPNN, FFNN, GRNN	KN
	MLP, RBF, SVM, M5P	MLP, SVM
S54	MLP, RBF, SVM, DT	DT, RBF, SVM
	MLP, SVM, LgR, KMC, GEP	GEP, SVM
S55	Neuro-GA, ANN, BN, RT, BE, SS (S6),MARS, MLR, ANN, RT, SVR (S9)	Neuro-GA
S57	Neuro-GA, BN, RT, BE, SS (S6), MARS, MLR, ANN, RT, SVR (S9)	Neuro-GA
S58	FGA, AFGA, FPSO, MFPSO, FCSA, BN, RT, BE, SS (S6),MARS, MLR, ANN, RT, SVR (S9), FIS (S32), SVM-RBF (S37), MLP, RBF, SVM, M5P (S38)	FGA, AFGA, FPSO, MFPSO, FCSA
S59	GA, Dtable, RBF, BN, SMO	GA
S60	GGAL, GMDH, LR, M5R, DT, SVM, K*, JERN, BPNN, KN, PNN, GRNN	GGAL, GMDH
S61	SVM-SIG, SVM-RBF, SVM-LIN	SVM-SIG
S62	GEP, DFT, SVM, LR, MLP, RBF	GEP
S65	ELM-PLY, LR, NB, ELM-LIN, ELM-RBF, SVM-SIG, SVM-LIN, SVM-RBF	ELM-PLY
S66	Cubist, LR, Lasso, Elastic Net	Cubist
S68	M5P, MLR, MLP, SVR	M5P
S69	Neuro Fuzzy, BN, RT, BE, SS (S6), MARS, MLR, ANN, RT, SVR (S9), FL (S32), SVM, RBF (S37), MLP, RBF, SVM (S38), ANN, Neuro-GA (S55), Neuro-GA (S57)	Neuro Fuzzy
S70	SVM-RBF, SVM-LIN, SVM-SIG	SVM-RBF
S71	MARS, MLR, SVM	MARS
S72	LSSVM-LIN, LSSVM-RBF, LSSVM-SIG	LSSVM-LIN
S73	BN, MLP, LgR, NB, J48, NNge	BN, MLP
S77	LSSVM-RBF, LR, PR, LgR, DT, SVM-LIN, SVM-PLY, SVM-RBF, ELM-LIN, ELM-PLY, ELM-RBF, LSSVM-LIN, LSSVM-PLY, NGD, NGDM, NGDA, NNM	LSSVM-RBF

Table 16 provides a description of the techniques reported to be superior in more than two studies with their strengths and weaknesses as provided by the authors.

Some of the SPMP techniques identified in comparative studies have been reported to be superior in some studies and not in others. Note that Figure 15 includes techniques that were reported superior and not, at least one time each. For example, we note that:

- MLP technique was reported to be superior in six studies (S29, S35, S38, S54, S73), while not in six (S44, S54, S58, S62, S68, S69).
- SVM technique was reported superior in four studies (S38, S54, S61, S70) and not in eight (S10, S15, S42, S58, S60, S62, S65, S69).

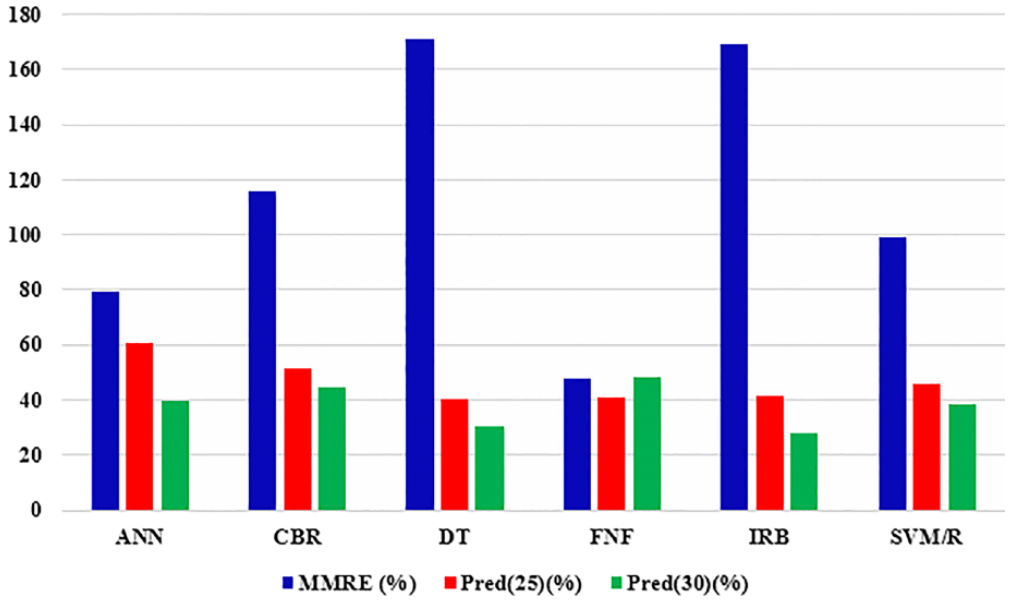

Figure 13. Average performance of different change prediction techniques (16 studies)

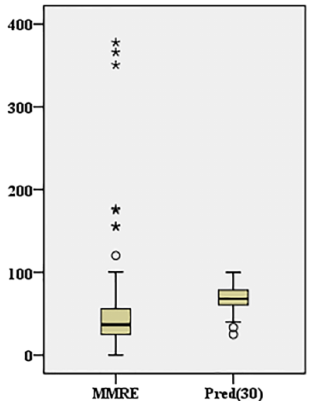

Figure 14. Performance distribution of maintainability index (S16 and S68)

- GMDH was reported superior in three studies (S33, S47, S60) and not in one (S44).
- MARS was reported superior in two studies (S9, S71) and not in eight (S19, S33, S42, S44, S55, S57, S58, S69).
- DT was reported superior in two studies (S36, S54) and not in four (S10, S41, S60, S77).
- BN was reported superior in three studies (S6, S49, S73) and not in eight (S15, S42, S43, S55, S57, S58, S59, S69).
- Neuro-GA was reported superior in two studies (S55, S57) and not in one (S69).
- RBF was reported superior in one study (S54) and not in 10 (S15, S23, S38, S42, S44, S54, S58, S59, S62, S69).
- M5P was reported superior in one study (S68) and not in four (S38, S44, S54, S58).
- GRNN was reported superior in one study (S2) and not in seven (S23, S29, S33, S44, S47, S52, S60).
- ELM was reported superior in three studies (S26, S43, S65) and not in one (S77).

From Figure 15, we note that no technique is definitively better than any other. Therefore, the choice of the best technique to predict maintainability is not obvious since every technique has advantages and drawbacks. Moreover, since the prediction context (e.g., dataset, accuracy criteria, etc.) is different among the studies, the literature results on the most accu-

Table 14. Prediction performance for maintainability time

ID	MMRE	Pred(30)	Dataset type	Prediction context: Software developement project	Prediction context: Dependent variable	Prediction technique
S3	58.30	46.00	Spain data	Object-oriented	Understandability time	RA
S3	67.60	38.50	Italy data	Object-oriented	Understandability time	RA
S3	85.00	30.00	All data	Object-oriented	Understandability time	RA

Table 16. Strengths and weaknesses of the most accurate SPMP techniques

Technique	Description	Strength	Weakness
Multilayer perceptron (MLP)	"MLP are feed forward networks that consist of an input layer, one or more hidden layers of nonlinearly activating nodes and an output layer. Each node in one layer connects with a certain weight to every other node in the following layer" [38].	– "Minimizes the prediction error of the output variables" (S29, S35). – "Uses back propagation algorithm as the standard learning algorithm for any supervised learning" (S38, S54).	
Support vector machine (SVM)	"SVM are a group of supervised learning methods that can be applied to classification or regression problems" [39].	– "Minimizes the empirical error and maximizes the geometric margin" (S38, S54).	
Group method of data handling (GMDH)	"GMDH was introduced by Ivakhnenko and Ivakhnenko & Koppa for constructing an extremely high order regression type model and is based on forward multi-layer neural network structure where learning procedure is self-organized" [40, 41].	– "Ideal for complex, unstructured systems where the investigator is only interested in obtaining a high order input-output relationship" (S33). – "Predicts the outcome even with smaller training sets" (S33). – "Computational burden is reduced with GMDH" (S33). – "Can automatically filter out input properties that provide little information about location and shape of hyper surface" (S47).	– "Heuristic in nature and not based on a solid foundation as is regression analysis" (S33).

rate techniques are not sufficient to generalize the results.

4.3. Accuracy comparison of SPMP techniques reported to be superior in comparative studies (RQ3)

For a meaningful comparison, techniques are compared based on the same prediction context. From our investigation, we found that most of the comparative studies used the following prediction context:

- UIMS and QUES datasets (see MQ6),
- L&H and C&K metrics (see MQ7),
- Change dependent variable (see MQ7),
- MMRE and/or Pred(0.25), and/or Pred(0.30) accuracy criteria (RQ1), and
- Object-oriented software development paradigm (see MQ5).

The purpose of this section is to compare the techniques reported to be superior (see Table 15,

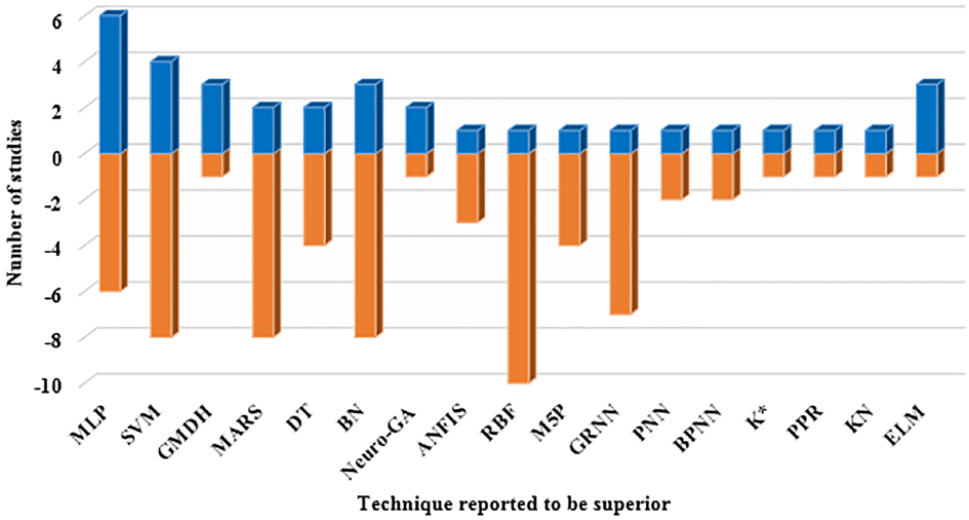

Figure 15. Techniques reported to be superior and not per study (bars above zero line indicate that techniques in horizontal axis are more accurate, whereas bars below zero line indicate that techniques in horizontal axis are not accurate)

third column), and which have this prediction context. Table 17 depicts the corresponding values of MMRE, and/or Pred(0.25), and/or Pred(0.30) for each technique per dataset. From comparative studies, 12 studies (20 experiments) were selected for UIMS and QUES datasets, and two studies (four experiments) for both datasets (i.e., the two datasets were merged). Note that one study may include more than one experiment.

Using MMRE and Pred as accuracy criteria for comparison, it is important to note that "to have a prediction model to be considered accurate, either MMRE < 0.25 and/or either Pred(0.25) > 0.75 or Pred(0.30) > 0.70, needed to be achieved" [35, 36]. That is, a low MMRE value or a high Pred(25) or Pred(30) value indicates good prediction accuracy. Table 17 shows that:

- For UIMS dataset, FGA, AFGA, and MF-PSO achieved a significantly better prediction accuracy than the other techniques. They are near in terms of MMRE (MMRE = 0.24 for FGA, MMRE = 0.25 for AFGA and MF-PSO). Besides, BN and ELM provide better accuracy than the other techniques in terms of Pred (Pred(0.25) = 0.44 and Pred(0.30) = 0.46 for BN followed by Pred(0.25) = 0.39 and Pred(0.30) = 0.45 for ELM).
- For QUES dataset, MFL and K* achieved the same MMRE value of 0.27. Moreover, they are near equal in terms of Pred: (Pred(0.25) = 0.52 and Pred(0.30) = 0.62 for MFL, while Pred(0.25) = 0.56 and Pred(0.30) = 0.66 for K*). Thus, the MFL and K* techniques provide better accuracy prediction compared to the remaining techniques.
- The GMDH, GA, and PNN techniques outperformed the MFL in both datasets (UIMS and QUES) with MMRE values of 0.21, 0.22 and 0.23, respectively, Pred(0.25) values of 0.69, 0.66 and 0.68, respectively, and Pred(0.30) values of 0.72, 0.72 and 0.75, respectively. Therefore, the GMDH was more accurate compared to the other techniques.

Here also, as stated in the previous section, no conclusion can be drawn about the most suitable technique for software product maintainability. Indeed, a technique can be more accurate in one study and less accurate in another. In addition, the accuracy of SPMP techniques is highly dependent on the prediction context (e.g., datasets used, accuracy criteria, etc.). Therefore, further studies are needed to reach a consensus on the most accurate technique for predicting maintainability of a software product.

Table 17. Prediction accuracy for UIMS, QUES, and BOTH datasets

ID	Technique	MMRE	Pred (0.25)	Pred (0.30)	Dataset	ID	Technique	MMRE	Pred (0.25)	Pred (0.30)	Dataset
S6	BN	0.97	0.44	0.46	UIMS	S6	BN	0.45	0.39	0.43	QUES
S9	MARS	1.86	0.28	0.28	UIMS	S9	MARS	0.32	0.48	0.59	QUES
S26	ELM	0.96	0.39	0.45	UIMS	S38	MLP	0.71	–	0.40	QUES
S38	MLP	1.39	–	0.23	UIMS	S38	SVM	0.44	–	0.51	QUES
S38	SVM	1.67	–	0.23	UIMS	S26	ELM	0.35	0.36	0.38	QUES
S42	MFL	0.53	0.30	0.35	UIMS	S42	MFL	0.27	0.52	0.62	QUES
S43	SBLLM	1.96	0.17	0.25	UIMS	S43	ELM	0.35	0.36	0.38	QUES
S43	ELM	0.96	0.17	0.25	UIMS	S43	SBLLM	0.34	0.50	0.56	QUES
S44	FSC	0.65	0.33	0.41	UIMS	S44	FSC	0.37	0.54	0.61	QUES
S44	K*	0.56	0.36	0.41	UIMS	S44	K*	0.27	0.56	0.66	QUES
S54	MLP	1.39	–	0.23	UIMS	S54	MLP	0.71	–	0.40	QUES
S54	SVM	1.64	–	0.23	UIMS	S54	SVM	0.44	–	0.56	QUES
S55	Neuro-GA	0.53	–	–	UIMS	S55	Neuro-GA	0.41	–	–	QUES
S57	Neuro-GA	0.31	–	–	UIMS	S57	Neuro-GA	0.37	–	–	QUES
S58	FGA	0.24	–	–	UIMS	S58	FGA	0.32	–	–	QUES
S58	AFGA	0.25	–	–	UIMS	S58	AFGA	0.32	–	–	QUES
S58	FPSO	0.27	–	–	UIMS	S58	FPSO	0.29	–	–	QUES
S58	MFPSO	0.25	–	–	UIMS	S58	MFPSO	0.32	–	–	QUES
S58	FCSA	0.27	–	–	UIMS	S58	FCSA	0.37	–	–	QUES
S69	Neuro-Fuzzy	0.28	–	–	UIMS	S69	Neuro-Fuzzy	0.33	–	–	QUES
S33	GMDH	0.21	0.69	0.72	BOTH	S33	PNN	0.23	0.68	0.75	BOTH
S33	GA	0.22	0.66	0.72	BOTH	S42	MFL	0.45	0.34	0.40	BOTH

5. Threats to validity

Three kinds of threats [10, 12] to the validity of this study are discussed as follows:

Construct validity: Construct threats to validity are related to the exhaustiveness and relevance of the primary studies. As previously noted, although maintainability and maintenance are different, they are often confounded and some studies do not make a clear distinction between them. Therefore, the search query was tailored to extract all available studies related to SPMP. Even though 82 primary studies were identified based on our search terms using keywords related to SPMP techniques, such a list may not be complete and a suitable study may have been left out. To ensure selection of the maximum number of studies, the search process was performed automatically on nine digital libraries and then manually by examining the reference section of the set of candidate studies to identify further studies. To identify additional studies, we established a set of inclusion and exclusion criteria.

Internal validity: Internal validity deals with data extraction and analysis. This threat is related to the reliability of the extracted data for the review, which can also be problematic. To accomplish this, two authors carried out the data extraction independently, keeping in mind the mapping and research questions, and their results compared. A third author reviewed the final results. When a disagreement arose, a discussion took place until an agreement was reached. If both authors extracted the same information for a specific paper, the extracted information was adopted. If the extracted information by the two authors was different for a specific paper, a meeting was held in which the full text of the paper was investigated.

External validity: External validity, which is very important for generalization of the results, is related to the context and conclusions drawn based on the data extracted. The results of this review were based only on the SPMP studies included in this paper. From each SPMP study, we extracted the dataset(s) used, and the dependent

and independent variables validated empirically using experiments, surveys or case studies. Since we refrained from deriving or adjusting any data, the comparison between SPMP studies was impartial.

6. Conclusion and future guidelines

Industry and practitioners continue to search for effective ways to increase the maintainability of software products and reduce costs. In this paper, we reported on a follow-up systematic mapping and review to provide and summarize evidence on published empirical SPMP studies. After a thorough search of nine digital libraries and analysis of the relevance and quality of candidate studies, 82 primary studies were selected from 2000 to 2018. This study classified the SPMP studies according to publication year, publication source, research type, empirical approach, software application type, datasets, independent variables, dependent variables, and techniques used. The SPMP techniques were investigated from the following perspectives: prediction accuracy, techniques reported to be superior in comparative studies, and accuracy comparison. The main findings (Sections 3 and 4), how they differ from previous studies and new findings from this systematic mapping and review are summarized as follows:

- *What are the research types used in SPMP studies*? Empirical studies were broadly categorized into two categories: evaluation research and solution proposal. The most frequent SPMP studies were solution proposals, followed by evaluation research.
- *What empirical approaches were used*? The most frequently used empirical approach was history-based evaluation, followed by experiment and case study.
- *What datasets were used*? Historical datasets freely available to the public, such as those provided by software engineering researchers (SER) and private datasets, such as those used in academic or industrial (PSP) contexts were frequently used, followed by Software engineering researcher (SER) datasets.
- *What types of software applications were used*? Many types of software applications were used in these empirical studies, those used most frequently were object-oriented software applications.
- *What dependent and independent variables were used*?
  - Maintainability in terms of the dependent variable to be predicted was most frequently expressed in terms of the number of changes made to the source code, followed by expert opinion based on an ordinal scale. This finding confirms, to some extent, the result of [4], but in reverse order, where it was reported that the most common dependent variable employed an ordinal scale based on expert judgment, followed by change measurements.
  - For the independent variables (predictors), the most frequent predictors of software maintainability were those provided by Chidamber and Kemerer (C&K), Li and Henry (L&H), class diagram, source code size measures and McCabe complexity, which were gathered at the design and source code levels. This finding confirms, in reverse order, the result of [4]. Moreover, C&K and L&H measures, as predictors, were most often used to predict the maintainability expressed in terms of changes as a predicted output.
  - The researchers used very few of the same data collection tools, thereby potentially leading to unknown error of measurement results since the measuring tools used have not been compared on similar benchmarks.
- *What techniques were used in SPMP*? The machine learning techniques were the most widely used in the literature. This finding is inconsistent with the results of [4] where the authors found that the commonly used maintainability prediction models were based on statistical techniques. This can be explained by the switch to machine learning techniques

that have gained the interest of researchers since 2008.

– *What is the overall prediction accuracy of SPMP techniques*? Several accuracy criteria were used to evaluate SPMP techniques. MMRE and Pred accuracy criteria were the most frequently used in the selected primary studies. Based on these criteria, FNF was the most accurate technique for predicting maintainability expressed in terms of changes based on MMRE and Pred(30), while ANN was the most accurate technique based on Pred(25).
– *Which SPMP techniques were reported superior in comparative studies*? We found that MLP, SVM, GMDH, and ELM were the most accurate techniques among selected comparative studies. Even if these techniques had better accuracy prediction in some studies, this was not the case in other studies. Therefore, no technique was definitively better than any other.
– *Which of the SPMP techniques reported to be superior in comparative studies also provided greater accuracy*? Accuracy comparison of techniques reported superior from comparative studies was carried out based on the same prediction context, in other words, the same datasets (UIMS or QUES or BOTH), the same metrics (L&H and C&K), the same dependent variable (Change), the same accuracy criteria (MMRE and/or Pred(0.25) and/or Pred(0.30), and the same software development paradigm (object-oriented). The results show that:
  - FGA, AFGA, and MFPSO achieved a significantly better prediction accuracy in terms of MMRE for the UIMS dataset,
  - MFL and K* were the most accurate for the QUES dataset, and
  - GMDH was the most accurate for both datasets.

  From this analysis we cannot conclude which is the most suitable technique for all cases as it is highly dependent on the prediction context (e.g., datasets used, accuracy criteria, etc.).

These findings may be useful to industry for comparing available SPMP models to improve the maintainability of software projects, and to researchers conducting further research into new SPMP techniques more performant than existing ones. Moreover, practitioners can choose the techniques used for predicting maintainability based on their prediction contexts as a solution in their practice.

In addition to the above findings, the following research gaps were identified:

– More free datasets should be made available to conduct empirical studies. In contrast to private datasets, public ones allow researchers to compare results in order to obtain generalizable results. Additional publicly available datasets can be used, such as the International Software Benchmarking Standard Group (ISBSG[1] repository of 8,261 completed software projects with more than 100 data fields, and PROMISE repository which is a collection of publicly available datasets grouped into one repository (http://promise.site.uottawa.ca/SERepository/).
– Most of the studies dealt with small datasets, such as UIMS and QUES with a single project each and related to projects developed using the Ada programming language. Large datasets based on the most frequently used programming languages in the industry are needed. This represents a serious challenge for the study of SPMP techniques. For instance, within the ISBSG, the most used programming languages include Java, COBOL, Oracle and .Net which represent 30%, 23%, 22% and 20%, respectively. It would be beneficial to SPMP research community to address this limitation.
– Moreover, dataset properties, such as type of data (categorical or numerical), missing values, outliers, etc., were not addressed by the research community.
– The majority of studies used data from OO software projects. As a result, there is a need for studies that examine maintainability for other types of applications such as web, mo-

[1]ISBSG, Development and Enhancement repository, February 2018, (http://www.isbsg.org).

bile, model-driven, and cloud computing applications.
- SPMP studies are needed that focus on maintainability before delivery of the software product in order to detect problems and quality failures early, while the source code is not available. Such studies should be based on the 'requirements' and accordingly researchers must determine what 'independent variables' or 'predictors' can be collected based on the requirements.
- Few studies address maintainability from the process level. More studies are needed to investigate how software development factors as well as software process management factors (such as project planning, requirement analysis, architectural design, development team, etc.) affect software maintainability.
- Few studies use ensemble techniques. More studies are needed using ensemble techniques since they use various single techniques to obtain a more accurate result.

Researchers interested in carrying out future research on SPMP, including empirical studies and benchmarking studies, would do well to investigate these research gaps and suggested research avenues.

References

[1] P. Bourque and R.E. Fairley, *Guide to the software engineering body of knowledge (SWEBOK (R)): Version 3.0*. IEEE Computer Society Press, 2014.

[2] P. Oman and J. Hagemeister, "Construction and testing of polynomials predicting software maintainability," *Journal of Systems and Software*, Vol. 24, No. 3, 1994, pp. 251–266.

[3] A. Kaur and K. Kaur, "Statistical comparison of modelling methods for software maintainability prediction," *International Journal of Software Engineering and Knowledge Engineering*, Vol. 23, No. 06, 2013, pp. 743–774.

[4] M. Riaz, E. Mendes, and E. Tempero, "A systematic review of software maintainability prediction and metrics," in *Proceedings of the 3rd International Symposium on Empirical Software Engineering and Measurement*. IEEE Computer Society, 2009, pp. 367–377.

[5] M. Riaz, "Maintainability prediction of relational database-driven applications: a systematic review," in *16th International Conference on Evaluation & Assessment in Software Engineering*. IET, 2012, pp. 263–272.

[6] B.A. Orenyi, S. Basri, and L.T. Jung, "Object-oriented software maintainability measurement in the past decade," in *International Conference on Advanced Computer Science Applications and Technologies (ACSAT)*. IEEE, 2012, pp. 257–262.

[7] S.K. Dubey, A. Sharma, and A. Rana, "Analysis of maintainability models for object oriented system," *International Journal on Computer Science and Engineering*, Vol. 3, No. 12, 2011, p. 3837.

[8] A.M. Fernández-Sáez, M. Genero, and M.R. Chaudron, "Empirical studies concerning the maintenance of UML diagrams and their use in the maintenance of code: A systematic mapping study," *Information and Software Technology*, Vol. 55, No. 7, 2013, pp. 1119–1142.

[9] A.M. Fernández-Sáez, D. Caivano, M. Genero, and M.R. Chaudron, "On the use of UML documentation in software maintenance: Results from a survey in industry," in *18th International Conference on Model Driven Engineering Languages and Systems (MODELS)*. IEEE, 2015, pp. 292–301.

[10] 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.

[11] B.A. Kitchenham, D. Budgen, and O.P. Brereton, "Using mapping studies as the basis for further research – A participant-observer case study," *Information and Software Technology*, Vol. 53, No. 6, 2011, pp. 638–651.

[12] A. Idri, F.A. Amazal, and A. Abran, "Analogy-based software development effort estimation: A systematic mapping and review," *Information and Software Technology*, Vol. 58, 2015, pp. 206–230.

[13] W. Li and H. Sallie, "Object oriented metrics that predict maintainability," *Journal of Systems and Software*, Vol. 23, No. 2, 1993, pp. 111–122.

[14] A. Kaur, K. Kaur, and K. Pathak, "Software maintainability prediction by data mining of software code metrics," in *International Conference on Data Mining and Intelligent Computing (ICDMIC)*. IEEE, 2014, pp. 1–6.

[15] P. Omidyar, *eBay web services*. [Online]. http://developer.ebay.com [accessed: 2018-10-26].

[16] B.R. Reddy and O. Aparajita, "Performance of maintainability index prediction models: A feature selection based study," *Evolving Systems*, 2017.

[17] A. Jain, S. Tarwani, and A. Chug, "An empirical investigation of evolutionary algorithm for software maintainability prediction," in *Students' Conference on Electrical, Electronics and Computer Science (SCEECS)*, 2016, pp. 1–6.

[18] M. Genero, J.A. Olivas, M. Piattini, and F.P. Romero, "A controlled experiment for corroborating the usefulness of class diagram metrics at the early phases of OO developments," in *ADIS*, 2001.

[19] X. Jin, Y. Liu, J. Ren, A. Xu, and R. Bie, "Locality preserving projection on source code metrics for improved software maintainability," in *Australasian Joint Conference on Artificial Intelligence*. Springer, 2006, pp. 877–886.

[20] K. Beck and E. Gamma, *JUnit*. [Online]. https://junit.org [accessed: 2018-10-26].

[21] R. Malhotra and A. Chug, "Application of group method of data handling model for software maintainability prediction using object oriented systems," *International Journal of System Assurance Engineering and Management*, Vol. 5, No. 2, 2014, pp. 165–173.

[22] S.J. Sayyad and T. Menzies, *The PROMISE Repository of Software Engineering Databases*, School of Information Technology and Engineering, University of Ottawa, Canada. [Online]. http://promise.site.uottawa.ca/SERepository Accessed: 2017-09-11.

[23] *Software Engineering – Product Quality – Part 2: External Metrics, Part 3: Internal Metrics, Part 4: Quality in Use Metrics*, ISO/IEC Std. TR 9126-2-3-4, 2003, 2004.

[24] *Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE) – System and software quality models*, Geneva, ISO Std. 25 010, 2010.

[25] D. Spinellis, *Chidamber and Kemerer Java Metrics (CKJM)*. [Online]. http://www.spinellis.gr/sw/ckjm/ [accessed: 2017-11-21].

[26] *Jetbrains Homepage*. [Online]. http://www.jetbrains.com/idea/ [accessed: 2017-11-19].

[27] *Krakatau Professional Homepage*. [Online]. http://www.powersoftware.com/kp/ [accessed: 2017-11-19].

[28] R. Ferenc, A. Beszedes, M. Tarkiainen, and T. Gyimothy, "Columbus – reverse engineering tool and schema for C++," in *Proceedings of the International Conference on Software Maintenance (ICSM'02)*, ICSM '02. Washington, DC, USA: IEEE Computer Society, 2002, pp. 172–181.

[29] G.A. Di Lucca, A.R. Fasolino, F. Pace, P. Tramontana, and U. De Carlini, "WARE: a tool for the reverse engineering of web applications," in *Proceedings of the Sixth European Conference on Software Maintenance and Reengineering*, 2002, pp. 241–250.

[30] B.R. Reddy, S. Khurana, and A. Ojha, "Software maintainability estimation made easy: A comprehensive tool coin," in *Proceedings of the Sixth International Conference on Computer and Communication Technology 2015*. New York: ACM, 2015, pp. 68–72.

[31] *Analyst4j standard tool*. [Online]. https://codeswat.com/ [accessed: 2018-01-01].

[32] S. Almugrin, W. Albattah, and A. Melton, "Using indirect coupling metrics to predict package maintainability and testability," *Journal of System and Software*, Vol. 121, No. C, 2016, pp. 298–310.

[33] R. Malhotra, "A systematic review of machine learning techniques for software fault prediction," *Applied Soft Computing*, Vol. 27, 2015, pp. 504–518.

[34] 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, 2012, pp. 41–59.

[35] S.D. Conte, H.E. Dunsmore, and Y. Shen, *Software engineering metrics and models*. Benjamin-Cummings Publishing Co., Inc., 1986.

[36] A.R. Gray and S.G. MacDonell, "A comparison of techniques for developing predictive models of software metrics," *Information and software technology*, Vol. 39, No. 6, 1997, pp. 425–437.

[37] P. Oman and J. Hagemeister, "Construction and testing of polynomials predicting software maintainability," *Journal of Systems and Software*, Vol. 24, No. 3, 1994, pp. 251–266.

[38] S. Haykin, *Neural networks: a comprehensive foundation*. Prentice Hall PTR, 1994.

[39] V. Vapnik, *The nature of statistical learning theory*. New York: Springer-Verlag, 1995.

[40] A.G. Ivakhnenko, "The group method of data of handling; A rival of the method of stochastic approximation," *Soviet Automatic Control*, Vol. 13, 1968, pp. 43–55.

[41] A.G. Ivakhnenko and Y. Koppa, "Regularization of decision functions in the group method of data handling," *Soviet Automatic Control*, Vol. 15, No. 2, 1970, pp. 28–37.

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

[43] J. Magne and M. Shepperd, "A systematic review of software development cost estimation studies," *IEEE Transactions on Software Engineering*, Vol. 33, No. 1, 2007, pp. 33–53.

[44] S. Muthanna, K. Kontogiannis, K. Ponnambalam, and B. Stacey, "A maintainability model for industrial software systems using design level metrics," in *Proceedings Seventh Working Conference on Reverse Engineering*. IEEE, 2000, pp. 248–256.

[45] M. Thwin and T. Quah, "Application of neural networks for estimating software maintainability using object-oriented metrics," in *International Conference on Software Engineering and Knowledge Engineering*, 2003, pp. 69–73.

[46] M. Genero, M. Piattini, E. Manso, and G. Cantone, "Building UML class diagram maintainability prediction models based on early metrics," in *5th International Workshop on Enterprise Networking and Computing in Healthcare Industry*. IEEE, 2003, pp. 263–275.

[47] M. Kiewkanya, N. Jindasawat, and P. Muenchaisri, "A methodology for constructing maintainability model of object-oriented design," in *Fourth International Conference on Quality Software*. IEEE, 2004, pp. 206–213.

[48] G.A. Di Lucca, A.R. Fasolino, P. Tramontana, and C.A. Visaggio, "Towards the definition of a maintainability model for web applications," in *Eighth European Conference on Software Maintenance and Reengineering*. IEEE, 2004, pp. 279–287.

[49] C. Van Koten and A. Gray, "An application of Bayesian network for predicting object-oriented software maintainability," *Information and Software Technology*, Vol. 48, No. 1, 2006, pp. 59–67.

[50] J.H. Hayes and L. Zhao, "Maintainability prediction: A regression analysis of measures of evolving systems," in *21st International Conference on Software*

Maintenance (ICSM'05). IEEE, 2005, pp. 601–604.

[51] S.C. Misra, "Modeling design/coding factors that drive maintainability of software systems," *Software Quality Journal*, Vol. 13, No. 3, 2005, pp. 297–320.

[52] Y. Zhou and H. Leung, "Predicting object-oriented software maintainability using multivariate adaptive regression splines," *Journal of Systems and Software*, Vol. 80, No. 8, 2007, pp. 1349–1361.

[53] S.S. Dahiya, J.K. Chhabra, and S. Kumar, "Use of genetic algorithm for software maintainability metrics' conditioning," in *15th International Conference on Advanced Computing and Communications*. IEEE, 2007, pp. 87–92.

[54] M. Genero, E. Manso, A. Visaggio, G. Canfora, and M. Piattini, "Building measure-based prediction models for UML class diagram maintainability," *Empirical Software Engineering*, Vol. 12, No. 5, 2007, pp. 517–549.

[55] K. Shibata, K. Rinsaka, T. Dohi, and H. Okamura, "Quantifying software maintainability based on a fault-detection/correction model," in *13th Pacific Rim International Symposium on Dependable Computing*. IEEE, 2007, pp. 35–42.

[56] K. Aggarwal, Y. Singh, A. Kaur, and R. Malhotra, "Application of artificial neural network for predicting maintainability using object-oriented metrics," *World Academy of Science, Engineering and Technology, International Journal of Computer, Electrical, Automation, Control and Information Engineering*, Vol. 2, No. 10, 2008, pp. 3552–3556.

[57] Y. Tian, C. Chen, and C. Zhang, "AODE for source code metrics for improved software maintainability," in *Fourth International Conference on Semantics, Knowledge and Grid*. IEEE, 2008, pp. 330–335.

[58] Y. Zhou and B. Xu, "Predicting the maintainability of open source software using design metrics," *Wuhan University Journal of Natural Sciences*, Vol. 13, No. 1, 2008, pp. 14–20.

[59] H. Yu, G. Peng, and W. Liu, "An application of case based reasoning to predict structure maintainability," in *International Conference on Computational Intelligence and Software Engineering*. IEEE, 2009, pp. 1–5.

[60] A. Sharma, P. Grover, and R. Kumar, "Predicting maintainability of component-based systems by using fuzzy logic," in *International Conference on Contemporary Computing*. Springer, 2009, pp. 581–591.

[61] L. Wang, X. Hu, Z. Ning, and W. Ke, "Predicting object-oriented software maintainability using projection pursuit regression," in *First International Conference on Information Science and Engineering*. IEEE, 2009, pp. 3827–3830.

[62] H. Mittal and P. Bhatia, "Software maintainability assessment based on fuzzy logic technique," *ACM SIGSOFT Software Engineering Notes*, Vol. 34, No. 3, 2009, pp. 1–5.

[63] M.O. Elish and K.O. Elish, "Application of treenet in predicting object-oriented software maintainability: A comparative study," in *13th European Conference on Software Maintenance and Reengineering*. IEEE, 2009, pp. 69–78.

[64] S. Rizvi and R.A. Khan, "Maintainability estimation model for object-oriented software in design phase (MEMOOD)," *arXiv preprint arXiv:1004.4447*, 2010.

[65] A. Kaur, K. Kaur, and R. Malhotra, "Soft computing approaches for prediction of software maintenance effort," *International Journal of Computer Applications*, Vol. 1, No. 16, 2010, pp. 69–75.

[66] C. Jin and J.A. Liu, "Applications of support vector mathine and unsupervised learning for predicting maintainability using object-oriented metrics," in *Second International Conference on Multimedia and Information Technology*, Vol. 1. IEEE, 2010, pp. 24–27.

[67] L. Cai, Z. Liu, J. Zhang, W. Tong, and G. Yang, "Evaluating software maintainability using fuzzy entropy theory," in

9th International Conference on Computer and Information Science. IEEE, 2010, pp. 737–742.

[68] S.O. Olatunji, Z. Rasheed, K. Sattar, A. Al-Mana, M. Alshayeb, and E. El-Sebakhy, "Extreme learning machine as maintainability prediction model for object-oriented software systems," *Journal of Computing*, Vol. 2, No. 8, 2010, pp. 49–56.

[69] P. Dhankhar, H. Mittal, and A. Mittal, "Maintainability prediction for object oriented software," *International Journal of Advances in Engineering Sciences*, Vol. 1, No. 1, 2011, pp. 8–11.

[70] S.K. Dubey and A. Rana, "A fuzzy approach for evaluation of maintainability of object oriented software system," *International Journal of Computer Applications*, Vol. 49, No. 21, 2012.

[71] S.K. Dubey, A. Rana, and Y. Dash, "Maintainability prediction of object-oriented software system by multilayer perceptron model," *ACM SIGSOFT Software Engineering Notes*, Vol. 37, No. 5, 2012, pp. 1–4.

[72] N. Tagoug, "Maintainability assessment in object-oriented system design," in *International Conference on Information Technology and e-Services*. IEEE, 2012, pp. 1–5.

[73] S. Sharawat, "Software maintainability prediction using neural networks," *environment*, Vol. 3, No. 5, 2012, pp. 750–755.

[74] H.A. Al-Jamimi and M. Ahmed, "Prediction of software maintainability using fuzzy logic," in *International Conference on Computer Science and Automation Engineering*. IEEE, 2012, pp. 702–705.

[75] R. Malhotra and A. Chug, "Software maintainability prediction using machine learning algorithms," *Software Engineering: An International Journal*, Vol. 2, No. 2, 2012.

[76] T. Bakota, P. Hegedűs, G. Ladányi, P. Körtvélyesi, R. Ferenc, and T. Gyimóthy, "A cost model based on software maintainability," in *28th International Conference on Software Maintenance (ICSM)*. IEEE, 2012, pp. 316–325.

[77] Y. Dash, S.K. Dubey, and A. Rana, "Maintainability prediction of object oriented software system by using artificial neural network approach," *International Journal of Soft Computing and Engineering (IJSCE)*, Vol. 2, No. 2, 2012, pp. 420–423.

[78] P. Hegedűs, G. Ladányi, I. Siket, and R. Ferenc, "Towards building method level maintainability models based on expert evaluations," in *Computer Applications for Software Engineering, Disaster Recovery, and Business Continuity*. Springer, 2012, pp. 146–154.

[79] D. Chandra, "Support vector approach by using radial kernel function for prediction of software maintenance effort on the basis of multivariate approach," *International Journal of Computer Applications*, Vol. 51, No. 4, 2012.

[80] H. Aljamaan, M.O. Elish, and I. Ahmad, "An ensemble of computational intelligence models for software maintenance effort prediction," in *International Work-Conference on Artificial Neural Networks*. Springer, 2013, pp. 592–603.

[81] P. Hegedűs, T. Bakota, G. Ladányi, C. Faragó, and R. Ferenc, "A drill-down approach for measuring maintainability at source code element level," *Electronic Communications of the EASST*, Vol. 60, 2013.

[82] X.L. Hao, X.D. Zhu, and L. Liu, "Research on software maintainability evaluation based on fuzzy integral," in *International Conference on Quality, Reliability, Risk, Maintenance, and Safety Engineering (QR2MSE)*. IEEE, 2013, pp. 1279–1282.

[83] F. Ye, X. Zhu, and Y. Wang, "A new software maintainability evaluation model based on multiple classifiers combination," in *International Conference on Quality, Reliability, Risk, Maintenance, and Safety Engineering (QR2MSE)*. IEEE, 2013, pp. 1588–1591.

[84] M.A. Ahmed and H.A. Al-Jamimi, "Machine learning approaches for predicting software maintainability: A fuzzy-based transparent model," *IET software*, Vol. 7, No. 6, 2013, pp. 317–326.

[85] S.O. Olatunji and A. Ajasin, "Sensitivity-based linear learning method and extreme learning machines compared for

software maintainability prediction of object-oriented software systems," *ICTACT Journal On Soft Computing*, Vol. 3, No. 03, 2013.

[86] A. Mehra and S.K. Dubey, "Maintainability evaluation of object-oriented software system using clustering techniques," *Internationa Journal of Computers and Technology*, Vol. 5, No. 02, 2013, pp. 136–143.

[87] J. Al Dallal, "Object-oriented class maintainability prediction using internal quality attributes," *Information and Software Technology*, Vol. 55, No. 11, 2013, pp. 2028–2048.

[88] A. Kaur, K. Kaur, and K. Pathak, "A proposed new model for maintainability index of open source software," in *Proceedings of 3rd International Conference on Reliability, Infocom Technologies and Optimization*. IEEE, 2014, pp. 1–6.

[89] A. Pratap, R. Chaudhary, and K. Yadav, "Estimation of software maintainability using fuzzy logic technique," in *International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT)*. IEEE, 2014, pp. 486–492.

[90] L. Geeta, A. Kavita, and B. Rizwan, "Maintainability measurement model for object oriented design," *International Journal of Advanced Research in Computer Science and Software Engineering*, Vol. 4, No. 11, 2014, pp. 945–956.

[91] R. Malhotra and A. Chug, "A metric suite for predicting software maintainability in data intensive applications," in *Transactions on Engineering Technologies*. Springer Netherlands, 2014, pp. 161–175.

[92] S. Misra and F. Egoeze, "Framework for maintainability measurement of web application for efficient knowledge-sharing on campus intranet," in *Computational Science and Its Applications – ICCSA 2014*. Cham: Springer International Publishing, 2014, pp. 649–662.

[93] 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.

[94] L. Kumar and S.K. Rath, "Neuro-genetic approach for predicting maintainability using Chidamber and Kemerer software metrics suite," in *Recent Advances in Information and Communication Technology 2015*. Cham: Springer International Publishing, 2015, pp. 31–40.

[95] S.O. Olatunji and A. Selamat, "Type-2 fuzzy logic based prediction model of object oriented software maintainability," in *Intelligent Software Methodologies, Tools and Techniques*. Cham: Springer International Publishing, 2015, pp. 329–342.

[96] L. Kumar, D.K. Naik, and S.K. Rath, "Validating the effectiveness of object-oriented metrics for predicting maintainability," *Procedia Computer Science*, Vol. 57, 2015, pp. 798–806.

[97] L. Kumar and S.K. Rath, "Hybrid functional link artificial neural network approach for predicting maintainability of object-oriented software," *Journal of Systems and Software*, Vol. 121, No. C, 2016, pp. 170–190.

[98] A. Chug and R. Malhotra, "Benchmarking framework for maintainability prediction of open source software using object oriented metrics," *International Journal of Innovative Computing, Information and Control*, Vol. 12, No. 2, 2016, pp. 615–634.

[99] L. Kumar, K. Mukesh, and K.R. Santanu, "Maintainability prediction of web service using support vector machine with various kernel methods," *International Journal of System Assurance Engineering and Management*, Vol. 8, No. 2, 2017, pp. 205–6222.

[100] S. Tarwani and A. Chug, "Predicting maintainability of open source software using gene expression programming and bad smells," in *5th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, 2016, pp. 452–459.

[101] S. Tarwani and A. Chug, "Sequencing of refactoring techniques by greedy algorithm for maximizing maintainability," in *International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 2016, pp. 1397–1403.

[102] L. Kumar, S.K. Rath, and A. Sureka, "Empirical analysis on effectiveness of source code metrics for predicting change-proneness," in *Proceedings of the 10th Innovations in Software Engineering Conference*, ISEC '17. New York, NY, USA: ACM, 2017, pp. 4–14.

[103] G. Kanika and C. Anuradha, "Evaluation of instance-based feature subset selection algorithm for maintainability prediction," in *International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 2017, pp. 1482–1487.

[104] K. Shivani and T. Kirti, "Maintainability assessment for software by using a hybrid fuzzy multi-criteria analysis approach," *Management Science Letters*, Vol. 7, 2017, pp. 255–274.

[105] K. Lov and K.R. Santanu, "Software maintainability prediction using hybrid neural network and fuzzy logic approach with parallel computing concept," *International Journal of System Assurance Engineering and Management*, Vol. 8, No. S2, 2017, pp. 1487–1502.

[106] L. Kumar, A. Krishna, and S.K. Rath, "The impact of feature selection on maintainability prediction of service-oriented applications," *Service Oriented Computing and Applications*, Vol. 11, No. 2, 2017, pp. 137–161.

[107] L. Kumar, S.K. Rath, and A. Sureka, "Using source code metrics and multivariate adaptive regression splines to predict maintainability of service oriented software," in *18th International Symposium on High Assurance Systems Engineering (HASE)*, 2017, pp. 88–95.

[108] L. Kumar, S.K. Rath, and A. Sureka, "Using source code metrics to predict change-prone web services: A case-study on ebay services," in *IEEE Workshop on Machine Learning Techniques for Software Quality Evaluation – MaLTeSQuE*. IEEE, 2017, pp. 1–7.

[109] R. Malhotra and R. Jangra, "Prediction and assessment of change prone classes using statistical and machine learning techniques," *Journal of Information Processing Systems*, Vol. 13, No. 4, 2017, pp. 778–804.

[110] G. Szőke, G. Antal, C. Nagy, R. Ferenc, and T. Gyimóthy, "Empirical study on refactoring large-scale industrial systems and its effects on maintainability," *Journal of Systems and Software*, Vol. 129, 2017, pp. 107–126.

[111] Y. Gokul and M. Gopal, "An authoritative method using fuzzy logic to evaluate maintainability index and utilizability of software," *Advances in Modelling and Analysis B*, Vol. 60, No. 3, 2017, pp. 566–580.

[112] P. Hegedűs, I. Kádár, R. Ferenc, and T. Gyimóthy, "Empirical evaluation of software maintainability based on a manually validated refactoring dataset," *Information and Software Technology*, Vol. 95, No. 1, 2018, pp. 313–327.

[113] L. Kumar and S. Ashish, "A comparative study of different source code metrics and machine learning algorithms for predicting change proneness of object oriented systems," *arXiv preprint arXiv:1712.07944*, 2018.

[114] G. Scanniello, C. Gravino, M. Genero, J.A. Cruz-Lemus, and G. Tortora, "On the impact of UML analysis models on source-code comprehensibility and modifiability," *ACM Trans. Softw. Eng. Methodol.*, Vol. 23, No. 2, 2014, pp. 13:1–13:26.

[115] A.M. Fernández-Sáez, M.R.V. Chaudron, M. Genero, and I. Ramos, "Are forward designed or reverse-engineered UML diagrams more helpful for code maintenance?: A controlled experiment," in *Proceedings of the 17th International Conference on Evaluation and Assessment in Software Engineering*. New York, NY, USA: ACM, 2013, pp. 60–71.

[116] G. Scanniello, C. Gravino, G. Tortora, M. Genero, M. Risi, J.A. Cruz-Lemus, and G. Dodero, "Studying the effect of UML-based models on source-code comprehensibility: Results from a long-term investigation," in *Proceedings of the 16th In-*

ternational Conference on Product-Focused Software Process Improvement, Vol. 9459, New York, 2015, pp. 311–327.

[117] A.M. Fernández-Sáez, M. Genero, D. Caivano, and M.R. Chaudron, "Does the level of detail of UML diagrams affect the maintainability of source code?: A family of experiments," *Empirical Software Engineering*, Vol. 21, No. 1, 2016, pp. 212–259.

[118] G. Scanniello, C. Gravino, M. Genero, J.A. Cruz-Lemus, G. Tortora, M. Risi, and G. Dodero, "Do software models based on the UML aid in source-code comprehensibility? Aggregating evidence from 12 controlled experiments," *Empirical Software Engineering*, Vol. 23, No. 5, 2018, pp. 2695–2733.

Appendix A. Appendix

Table A1. Research approaches

Research approaches	What is it [42]
ER	Empirical studies that evaluate and/or compare existing SPMP techniques.
SP	Empirical studies in which an SPMP technique is proposed, either as a new technique or as a significant adaptation of an existing one, or propose a solution to a defined problem.

Table A2. Empirical types

Empirical types	What is it [43]
HbE	Studies evaluating SPMP techniques of previously completed software projects.
Ex	An empirical method applied under controlled conditions to evaluate an SPMP technique.
CS	An empirical study that investigates an SPMP technique in a real-life context, e.g. in-depth study of the prediction processes of one, or a very small number, of software projects.

Table A3. QA score of selected primary studies

ID	Author	QA1	QA2	QA3	QA4	QA5	QA6	QA7	Score
S1	S. Muthanna et al.	1	1	1	1	1	0	1	6
S2	M.M.T Thwin et al.	1	1	1	1	0.5	1	1	6.6
S3	M. Genero et al.	1	1	1	1	0.5	1	1	6.6
S4	M. Kiewkayna et al.	1	1	0.5	1	0.5	0	1	5
S5	G.A.D. Lucca et al.	1	1	0.5	1	0.5	0.5	1	5.5
S6	C.V. Koten et al.	1	1	1	1	1	1	1	7
S7	J.H. Hayes et al.	1	0.5	1	1	0.5	0.5	1	5.5
S8	S.C. Misra	1	1	1	1	0.5	1	1	6.6
S9	Y. Zhou et al.	1	1	1	1	1	1	1	7
S10	X. Jin et al.	1	1	1	1	1	1	1	7
S11	S.S. Dahiya et al.	1	0.5	0.5	1	0.5	0.5	1	5
S12	M. Genero et al.	1	1	0.5	1	1	0.5	1	6

Table A3 continued

ID	Author	QA1	QA2	QA3	QA4	QA5	QA6	QA7	Score
S13	K. Shibata et al.	1	0.5	0.5	1	0.5	0.5	1	5
S14	K.K.Aggarwal et al.	0.5	1	0.5	0.5	0.5	1	0.5	4.5
S15	Y. Thian et al.	1	1	1	1	1	1	1	7
S16	Y. Zhou et al.	1	1	1	1	1	1	1	7
S17	YU, Haiquan et al.	1	1	1	1	1	0	1	6
S18	A. Sharma et al.	1	1	1	1	0.5	0.5	1	6
S19	W. Li-jin et al.	1	1	1	1	1	1	1	7
S20	H. Mittal et al.	1	1	0.5	1	0.5	1	1	6
S21	M. O. Elish et al.	1	1	1	1	1	1	1	7
S22	S. Rizvi et al.	1	1	1	1	0.5	1	1	6.6
S23	A.Kaur et al.	1	1	1	1	1	1	1	7
S24	C. Jin et al.	1	1	1	0.5	1	1	1	6.6
S25	L. CAI et al.	1	0.5	0.5	1	0.5	0.5	1	5
S26	S. O. Olatunji et al.	1	1	1	1	1	1	1	7
S27	P. Dhankhar et al.	1	1	0.5	1	0.5	0.5	1	5.5
S28	S.K. Dubey et al.	1	1	0.5	1	0.5	0.5	1	5.5
S29	S. K. Dubey et al.	1	1	1	1	1	1	1	7
S30	N. Tagoug et al.	1	0.5	1	1	1	0.5	1	5.5
S31	S. Sharawat et al.	1	1	1	1	0.5	0.5	1	6
S32	H.A. Al-Jamimi et al.	1	1	1	1	1	1	1	7
S33	R. Malhotra et al.	1	1	1	1	1	1	1	7
S34	T. Bakota et al.	1	1	1	1	1	1	1	7
S35	Y. Dash et al.	1	1	1	1	1	1	1	7
S36	P. Hegedűs et al.	1	1	1	1	1	1	1	7
S37	D. Chandra	1	1	1	1	1	1	1	7
S38	H. Aljamaan et al.	1	1	1	1	1	1	1	7
S39	P. Hegedűs et al.	1	1	0.5	1	1	1	1	6.6
S40	X.L. Hao et al.	1	1	1	1	1	0	1	6
S41	F. Ye et al.	1	1	1	1	0.5	1	1	6.6
S42	M.A. Ahmed et al.	1	1	1	1	1	1	1	7
S43	S.O. Olatunji et al.	1	1	1	1	1	1	1	7
S44	A. Kaur et al.	1	1	1	1	1	1	1	7
S45	A. Mehra et al.	1	1	1	1	0.5	1	1	6.6
S46	J. Al Dallal.	1	1	1	1	1	1	1	7
S47	R. Malhotra et al.	1	1	1	1	1	1	1	7
S48	A. Kaur et al.	1	1	1	1	1	1	1	7
S49	A. Kaur et al.	1	1	1	1	1	1	1	7
S50	A. Pratap et al.	1	1	0.5	1	1	0.5	0.5	5.5
S51	R. Kumar et al.	1	1	1	1	0.5	1	1	6.6
S52	R. Malhotra et al.	1	1	1	1	1	1	1	7
S53	S. Misra et al.	1	1	0.5	1	0.5	0.5	0.5	5
S54	M.O. Elish et al.	1	1	1	1	1	1	1	7
S55	L. Kumar et al.	1	1	1	1	1	1	1	7
S56	S.O. Olatunji et al.	1	1	1	1	1	1	1	7
S57	A.K. Soni et al.	1	1	1	1	1	1	1	7
S58	L. Kumar et al.	1	1	1	1	1	1	1	7
S59	A. Jain et al.	1	1	1	1	1	1	1	7
S60	A. Chug et al.	1	1	1	1	1	1	1	7
S61	L. Kumar et al.	1	1	1	1	1	1	1	7
S62	S. Tarwani et al.	1	1	1	1	1	1	1	7
S63	S. Almugrin et al.	1	1	1	1	1	0.5	1	6.5

Table A3 continued

ID	Author	QA1	QA2	QA3	QA4	QA5	QA6	QA7	Score
S64	S. Tarwani et al.	1	1	1	1	0.5	0.5	1	6
S65	L. Kumar et al.	1	1	1	1	1	1	1	7
S66	K. Gupta et al.	1	1	1	1	1	1	1	7
S67	S. Kundu et al.	1	1	0.5	1	1	0	1	5.5
S68	B.R. Reddy et al.	1	1	1	1	1	1	1	7
S69	L. Kumar et al.	1	1	1	1	1	1	1	7
S70	L. Kumar et al.	1	1	1	1	1	1	1	7
S71	L. Kumar et al.	1	1	1	1	1	1	1	7
S72	L. Kumar et al.	1	1	1	1	1	1	1	7
S73	R. Malhotra et al.	1	1	1	1	1	1	1	7
S74	G. Szoke et al.	1	1	1	1	0.5	0	1	5.5
S75	Y. Gokul et al.	1	1	0.5	1	0.5	0	1	5
S76	P. Hegedűs et al.	1	1	1	1	1	0	1	6
S77	L. Kumar et al.	1	1	1	1	1	1	1	7
S78	G. Scanniello et al.	1	1	1	0.5	0.5	0	1	5
S79	A.M. Fernández-Sáez et al.	1	1	1	0.5	0.5	0	1	5
S80	G. Scanniello et al.	1	1	1	0.5	0.5	0	1	5
S81	A.M. Fernández-Sáez et al.	1	1	1	0.5	0.5	0	1	5
S82	G. Scanniello et al.	1	1	1	0.5	0.5	0	1	5

Table A4. Search results for each of the nine databases

Database name	# of search results	# of duplicate studies	# of candidate studies	# of relevant studies
IEEE Explore	1678	15	100	28
Science Direct	5938	20	30	9
Springer Link	8715	45	71	18
Ebsco	1601	16	6	1
ACM Digital Library	530	14	10	5
Google Scholar	22090	30	77	10
dblp	120	80	20	2
Scopus	270	17	23	0
Jstore	399	26	4	2
Total	41341	263	341	75

Table A5. List of the 82 selected studies

ID	Author	Ref.	Title
S1	S. Muthanna et al.	[44]	S. Muthanna, K. Kontogiannis, K. Ponnambalam, and B. Stacey, "A maintainability model for industrial software systems using design level metrics," in *Proceedings Seventh Working Conference on Reverse Engineering*. IEEE, 2000, pp. 248–256
S2	M.M.T Thwin et al.	[45]	M. Thwin and T. Quah, "Application of neural networks for estimating software maintainability using object-oriented metrics," in *International Conference on Software Engineering and Knowledge Engineering*, 2003, pp. 69–73

Table A5 continued

ID	Author	Ref.	Title
S3	M. Genero et al.	[46]	M. Genero, M. Piattini, E. Manso, and G. Cantone, "Building UML class diagram maintainability prediction models based on early metrics," in *5th International Workshop on Enterprise Networking and Computing in Healthcare Industry*. IEEE, 2003, pp. 263–275
S4	M. Kiewkayna et al.	[47]	M. Kiewkanya, N. Jindasawat, and P. Muenchaisri, "A methodology for constructing maintainability model of object-oriented design," in *Fourth International Conference on Quality Software*. IEEE, 2004, pp. 206–213
S5	G.A.D. Lucca et al.	[48]	G.A. Di Lucca, A.R. Fasolino, P. Tramontana, and C.A. Visaggio, "Towards the definition of a maintainability model for web applications," in *Eighth European Conference on Software Maintenance and Reengineering*. IEEE, 2004, pp. 279–287
S6	C.V. Koten et al.	[49]	C. Van Koten and A. Gray, "An application of Bayesian network for predicting object-oriented software maintainability," *Information and Software Technology*, Vol. 48, No. 1, 2006, pp. 59–67
S7	J.H. Hayes et al.	[50]	J.H. Hayes and L. Zhao, "Maintainability prediction: A regression analysis of measures of evolving systems," in *21st International Conference on Software Maintenance (ICSM'05)*. IEEE, 2005, pp. 601–604
S8	S.C. Misra	[51]	S.C. Misra, "Modeling design/coding factors that drive maintainability of software systems," *Software Quality Journal*, Vol. 13, No. 3, 2005, pp. 297–320
S9	Y. Zhou et al.	[52]	Y. Zhou and H. Leung, "Predicting object-oriented software maintainability using multivariate adaptive regression splines," *Journal of Systems and Software*, Vol. 80, No. 8, 2007, pp. 1349–1361
S10	X. Jin et al.	[19]	X. Jin, Y. Liu, J. Ren, A. Xu, and R. Bie, "Locality preserving projection on source code metrics for improved software maintainability," in *Australasian Joint Conference on Artificial Intelligence*. Springer, 2006, pp. 877–886
S11	S.S. Dahiya et al.	[53]	S.S. Dahiya, J.K. Chhabra, and S. Kumar, "Use of genetic algorithm for software maintainability metrics' conditioning," in *15th International Conference on Advanced Computing and Communications*. IEEE, 2007, pp. 87–92
S12	M. Genero et al.	[54]	M. Genero, E. Manso, A. Visaggio, G. Canfora, and M. Piattini, "Building measure-based prediction models for UML class diagram maintainability," *Empirical Software Engineering*, Vol. 12, No. 5, 2007, pp. 517–549
S13	K. Shibata et al.	[55]	K. Shibata, K. Rinsaka, T. Dohi, and H. Okamura, "Quantifying software maintainability based on a fault-detection/correction model," in *13th Pacific Rim International Symposium on Dependable Computing*. IEEE, 2007, pp. 35–42
S14	K.K.Aggarwal et al.	[56]	K. Aggarwal, Y. Singh, A. Kaur, and R. Malhotra, "Application of artificial neural network for predicting maintainability using object-oriented metrics," *World Academy of Science, Engineering and Technology, International Journal of Computer, Electrical, Automation, Control and Information Engineering*, Vol. 2, No. 10, 2008, pp. 3552–3556
S15	Y. Thian et al.	[57]	Y. Tian, C. Chen, and C. Zhang, "AODE for source code metrics for improved software maintainability," in *Fourth International Conference on Semantics, Knowledge and Grid*. IEEE, 2008, pp. 330–335
S16	Y. Zhou et al.	[58]	Y. Zhou and B. Xu, "Predicting the maintainability of open source software using design metrics," *Wuhan University Journal of Natural Sciences*, Vol. 13, No. 1, 2008, pp. 14–20

Table A5 continued

ID	Author	Ref.	Title
S17	YU, Haiquan et al.	[59]	H. Yu, G. Peng, and W. Liu, "An application of case based reasoning to predict structure maintainability," in *International Conference on Computational Intelligence and Software Engineering*. IEEE, 2009, pp. 1–5
S18	A. Sharma et al.	[60]	A. Sharma, P. Grover, and R. Kumar, "Predicting maintainability of component-based systems by using fuzzy logic," in *International Conference on Contemporary Computing*. Springer, 2009, pp. 581–591
S19	W. Li-jin et al.	[61]	L. Wang, X. Hu, Z. Ning, and W. Ke, "Predicting object-oriented software maintainability using projection pursuit regression," in *First International Conference on Information Science and Engineering*. IEEE, 2009, pp. 3827–3830
S20	H. Mittal et al.	[62]	H. Mittal and P. Bhatia, "Software maintainability assessment based on fuzzy logic technique," *ACM SIGSOFT Software Engineering Notes*, Vol. 34, No. 3, 2009, pp. 1–5
S21	M.O. Elish et al.	[63]	M.O. Elish and K.O. Elish, "Application of treenet in predicting object-oriented software maintainability: A comparative study," in *13th European Conference on Software Maintenance and Reengineering*. IEEE, 2009, pp. 69–78
S22	S. Rizvi et al.	[64]	S. Rizvi and R.A. Khan, "Maintainability estimation model for object-oriented software in design phase (MEMOOD)," *arXiv preprint arXiv:1004.4447*, 2010
S23	A.Kaur et al.	[65]	A. Kaur, K. Kaur, and R. Malhotra, "Soft computing approaches for prediction of software maintenance effort," *International Journal of Computer Applications*, Vol. 1, No. 16, 2010, pp. 69–75
S24	C. Jin et al.	[66]	C. Jin and J.A. Liu, "Applications of support vector mathine and unsupervised learning for predicting maintainability using object-oriented metrics," in *Second International Conference on Multimedia and Information Technology*, Vol. 1. IEEE, 2010, pp. 24–27
S25	L. CAI et al.	[67]	L. Cai, Z. Liu, J. Zhang, W. Tong, and G. Yang, "Evaluating software maintainability using fuzzy entropy theory," in *9th International Conference on Computer and Information Science*. IEEE, 2010, pp. 737–742
S26	S.O. Olatunji et al.	[68]	S.O. Olatunji, Z. Rasheed, K. Sattar, A. Al-Mana, M. Alshayeb, and E. El-Sebakhy, "Extreme learning machine as maintainability prediction model for object-oriented software systems," *Journal of Computing*, Vol. 2, No. 8, 2010, pp. 49–56
S27	P. Dhankhar et al.	[69]	P. Dhankhar, H. Mittal, and A. Mittal, "Maintainability prediction for object oriented software," *International Journal of Advances in Engineering Sciences*, Vol. 1, No. 1, 2011, pp. 8–11
S28	S.K. Dubey et al.	[70]	S.K. Dubey and A. Rana, "A fuzzy approach for evaluation of maintainability of object oriented software system," *International Journal of Computer Applications*, Vol. 49, No. 21, 2012
S29	S.K. Dubey et al.	[71]	S.K. Dubey, A. Rana, and Y. Dash, "Maintainability prediction of object-oriented software system by multilayer perceptron model," *ACM SIGSOFT Software Engineering Notes*, Vol. 37, No. 5, 2012, pp. 1–4
S30	N. Tagoug et al.	[72]	N. Tagoug, "Maintainability assessment in object-oriented system design," in *International Conference on Information Technology and e-Services*. IEEE, 2012, pp. 1–5
S31	S. Sharawat et al.	[73]	S. Sharawat, "Software maintainability prediction using neural networks," *environment*, Vol. 3, No. 5, 2012, pp. 750–755

Table A5 continued

ID	Author	Ref.	Title
S32	H.A. Al-Jamimi et al.	[74]	H.A. Al-Jamimi and M. Ahmed, "Prediction of software maintainability using fuzzy logic," in *International Conference on Computer Science and Automation Engineering*. IEEE, 2012, pp. 702–705
S33	R. Malhotra et al.	[75]	R. Malhotra and A. Chug, "Software maintainability prediction using machine learning algorithms," *Software Engineering: An International Journal*, Vol. 2, No. 2, 2012
S34	T. Bakota et al.	[76]	T. Bakota, P. Hegedűs, G. Ladányi, P. Körtvélyesi, R. Ferenc, and T. Gyimóthy, "A cost model based on software maintainability," in *28th International Conference on Software Maintenance (ICSM)*. IEEE, 2012, pp. 316–325
S35	Y. Dash et al.	[77]	Y. Dash, S.K. Dubey, and A. Rana, "Maintainability prediction of object oriented software system by using artificial neural network approach," *International Journal of Soft Computing and Engineering (IJSCE)*, Vol. 2, No. 2, 2012, pp. 420–423
S36	P. Hegedűs et al.	[78]	P. Hegedűs, G. Ladányi, I. Siket, and R. Ferenc, "Towards building method level maintainability models based on expert evaluations," in *Computer Applications for Software Engineering, Disaster Recovery, and Business Continuity*. Springer, 2012, pp. 146–154
S37	D. Chandra	[79]	D. Chandra, "Support vector approach by using radial kernel function for prediction of software maintenance effort on the basis of multivariate approach," *International Journal of Computer Applications*, Vol. 51, No. 4, 2012
S38	H. Aljamaan et al.	[80]	H. Aljamaan, M.O. Elish, and I. Ahmad, "An ensemble of computational intelligence models for software maintenance effort prediction," in *International Work-Conference on Artificial Neural Networks*. Springer, 2013, pp. 592–603
S39	P. Hegedűs et al.	[81]	P. Hegedűs, T. Bakota, G. Ladányi, C. Faragó, and R. Ferenc, "A drill-down approach for measuring maintainability at source code element level," *Electronic Communications of the EASST*, Vol. 60, 2013
S40	X.L. Hao et al.	[82]	X.L. Hao, X.D. Zhu, and L. Liu, "Research on software maintainability evaluation based on fuzzy integral," in *International Conference on Quality, Reliability, Risk, Maintenance, and Safety Engineering (QR2MSE)*. IEEE, 2013, pp. 1279–1282
S41	F. Ye et al.	[83]	F. Ye, X. Zhu, and Y. Wang, "A new software maintainability evaluation model based on multiple classifiers combination," in *International Conference on Quality, Reliability, Risk, Maintenance, and Safety Engineering (QR2MSE)*. IEEE, 2013, pp. 1588–1591
S42	M.A. Ahmed et al.	[84]	M.A. Ahmed and H.A. Al-Jamimi, "Machine learning approaches for predicting software maintainability: A fuzzy-based transparent model," *IET software*, Vol. 7, No. 6, 2013, pp. 317–326
S43	S.O. Olatunji et al.	[85]	S.O. Olatunji and A. Ajasin, "Sensitivity-based linear learning method and extreme learning machines compared for software maintainability prediction of object-oriented software systems," *ICTACT Journal On Soft Computing*, Vol. 3, No. 03, 2013
S44	A. Kaur et al.	[3]	A. Kaur and K. Kaur, "Statistical comparison of modelling methods for software maintainability prediction," *International Journal of Software Engineering and Knowledge Engineering*, Vol. 23, No. 06, 2013, pp. 743–774
S45	A. Mehra et al.	[86]	A. Mehra and S.K. Dubey, "Maintainability evaluation of object-oriented software system using clustering techniques," *Internationa Journal of Computers and Technology*, Vol. 5, No. 02, 2013, pp. 136–143

Table A5 continued

ID	Author	Ref.	Title
S46	J. Al Dallal.	[87]	J. Al Dallal, "Object-oriented class maintainability prediction using internal quality attributes," *Information and Software Technology*, Vol. 55, No. 11, 2013, pp. 2028–2048
S47	R. Malhotra et al.	[21]	R. Malhotra and A. Chug, "Application of group method of data handling model for software maintainability prediction using object oriented systems," *International Journal of System Assurance Engineering and Management*, Vol. 5, No. 2, 2014, pp. 165–173
S48	A. Kaur et al.	[88]	A. Kaur, K. Kaur, and K. Pathak, "A proposed new model for maintainability index of open source software," in *Proceedings of 3rd International Conference on Reliability, Infocom Technologies and Optimization*. IEEE, 2014, pp. 1–6
S49	A. Kaur et al.	[14]	A. Kaur, K. Kaur, and K. Pathak, "Software maintainability prediction by data mining of software code metrics," in *International Conference on Data Mining and Intelligent Computing (ICDMIC)*. IEEE, 2014, pp. 1–6
S50	A. Pratap et al.	[89]	A. Pratap, R. Chaudhary, and K. Yadav, "Estimation of software maintainability using fuzzy logic technique," in *International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT)*. IEEE, 2014, pp. 486–492
S51	G. Laxmi et al.	[90]	L. Geeta, A. Kavita, and B. Rizwan, "Maintainability measurement model for object oriented design," *International Journal of Advanced Research in Computer Science and Software Engineering*, Vol. 4, No. 11, 2014, pp. 945–956
S52	R. Malhotra et al.	[91]	R. Malhotra and A. Chug, "A metric suite for predicting software maintainability in data intensive applications," in *Transactions on Engineering Technologies*. Springer Netherlands, 2014, pp. 161–175
S53	S. Misra et al.	[92]	S. Misra and F. Egoeze, "Framework for maintainability measurement of web application for efficient knowledge-sharing on campus intranet," in *Computational Science and Its Applications – ICCSA 2014*. Cham: Springer International Publishing, 2014, pp. 649–662
S54	M.O. Elish et al.	[93]	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
S55	L. Kumar et al.	[94]	L. Kumar and S.K. Rath, "Neuro-genetic approach for predicting maintainability using Chidamber and Kemerer software metrics suite," in *Recent Advances in Information and Communication Technology 2015*. Cham: Springer International Publishing, 2015, pp. 31–40
S56	S.O. Olatunji et al.	[95]	S.O. Olatunji and A. Selamat, "Type-2 fuzzy logic based prediction model of object oriented software maintainability," in *Intelligent Software Methodologies, Tools and Techniques*. Cham: Springer International Publishing, 2015, pp. 329–342
S57	L. Kumar et al.	[96]	L. Kumar, D.K. Naik, and S.K. Rath, "Validating the effectiveness of object-oriented metrics for predicting maintainability," *Procedia Computer Science*, Vol. 57, 2015, pp. 798–806
S58	L. Kumar et al.	[97]	L. Kumar and S.K. Rath, "Hybrid functional link artificial neural network approach for predicting maintainability of object-oriented software," *Journal of Systems and Software*, Vol. 121, No. C, 2016, pp. 170–190
S59	A. Jain et al.	[17]	A. Jain, S. Tarwani, and A. Chug, "An empirical investigation of evolutionary algorithm for software maintainability prediction," in *Students' Conference on Electrical, Electronics and Computer Science (SCEECS)*, 2016, pp. 1–6

Table A5 continued

ID	Author	Ref.	Title
S60	A. Chug et al.	[98]	A. Chug and R. Malhotra, "Benchmarking framework for maintainability prediction of open source software using object oriented metrics," *International Journal of Innovative Computing, Information and Control*, Vol. 12, No. 2, 2016, pp. 615–634
S61	L. Kumar et al.	[99]	L. Kumar, K. Mukesh, and K.R. Santanu, "Maintainability prediction of web service using support vector machine with various kernel methods," *International Journal of System Assurance Engineering and Management*, Vol. 8, No. 2, 2017, pp. 205–6222
S62	S. Tarwani et al.	[100]	S. Tarwani and A. Chug, "Predicting maintainability of open source software using gene expression programming and bad smells," in *5th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, 2016, pp. 452–459
S63	S. Almugrin et al.	[32]	S. Almugrin, W. Albattah, and A. Melton, "Using indirect coupling metrics to predict package maintainability and testability," *Journal of System and Software*, Vol. 121, No. C, 2016, pp. 298–310
S64	S. Tarwani et al.	[101]	S. Tarwani and A. Chug, "Sequencing of refactoring techniques by greedy algorithm for maximizing maintainability," in *International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 2016, pp. 1397–1403
S65	L. Kumar et al.	[102]	L. Kumar, S.K. Rath, and A. Sureka, "Empirical analysis on effectiveness of source code metrics for predicting change-proneness," in *Proceedings of the 10th Innovations in Software Engineering Conference*, ISEC '17. New York, NY, USA: ACM, 2017, pp. 4–14
S66	K. Gupta et al.	[103]	G. Kanika and C. Anuradha, "Evaluation of instance-based feature subset selection algorithm for maintainability prediction," in *International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 2017, pp. 1482–1487
S67	S. Kundu et al.	[104]	K. Shivani and T. Kirti, "Maintainability assessment for software by using a hybrid fuzzy multi-criteria analysis approach," *Management Science Letters*, Vol. 7, 2017, pp. 255–274
S68	B.R. Reddy et al.	[16]	B.R. Reddy and O. Aparajita, "Performance of maintainability index prediction models: A feature selection based study," *Evolving Systems*, 2017
S69	L. Kumar et al.	[105]	K. Lov and K.R. Santanu, "Software maintainability prediction using hybrid neural network and fuzzy logic approach with parallel computing concept," *International Journal of System Assurance Engineering and Management*, Vol. 8, No. S2, 2017, pp. 1487–1502
S70	L. Kumar et al.	[106]	L. Kumar, A. Krishna, and S.K. Rath, "The impact of feature selection on maintainability prediction of service-oriented applications," *Service Oriented Computing and Applications*, Vol. 11, No. 2, 2017, pp. 137–161
S71	L. Kumar et al.	[107]	L. Kumar, S.K. Rath, and A. Sureka, "Using source code metrics and multivariate adaptive regression splines to predict maintainability of service oriented software," in *18th International Symposium on High Assurance Systems Engineering (HASE)*, 2017, pp. 88–95
S72	L. Kumar et al.	[108]	L. Kumar, S.K. Rath, and A. Sureka, "Using source code metrics to predict change-prone web services: A case-study on ebay services," in *IEEE Workshop on Machine Learning Techniques for Software Quality Evaluation – MaLTeSQuE*. IEEE, 2017, pp. 1–7
S73	R. Malhotra et al.	[109]	R. Malhotra and R. Jangra, "Prediction and assessment of change prone classes using statistical and machine learning techniques," *Journal of Information Processing Systems*, Vol. 13, No. 4, 2017, pp. 778–804

Table A5 continued

ID	Author	Ref.	Title
S74	G. Szoke et al.	[110]	G. Szőke, G. Antal, C. Nagy, R. Ferenc, and T. Gyimóthy, "Empirical study on refactoring large-scale industrial systems and its effects on maintainability," *Journal of Systems and Software*, Vol. 129, 2017, pp. 107–126
S75	Y. Gokul et al.	[111]	Y. Gokul and M. Gopal, "An authoritative method using fuzzy logic to evaluate maintainability index and utilizability of software," *Advances in Modelling and Analysis B*, Vol. 60, No. 3, 2017, pp. 566–580
S76	P. Hegedűs et al.	[112]	P. Hegedűs, I. Kádár, R. Ferenc, and T. Gyimóthy, "Empirical evaluation of software maintainability based on a manually validated refactoring dataset," *Information and Software Technology*, Vol. 95, No. 1, 2018, pp. 313–327
S77	L. Kumar et al.	[113]	L. Kumar and S. Ashish, "A comparative study of different source code metrics and machine learning algorithms for predicting change proneness of object oriented systems," *arXiv preprint arXiv:1712.07944*, 2018
S78	G. Scanniello et al.	[114]	G. Scanniello, C. Gravino, M. Genero, J.A. Cruz-Lemus, and G. Tortora, "On the impact of UML analysis models on source-code comprehensibility and modifiability," *ACM Trans. Softw. Eng. Methodol.*, Vol. 23, No. 2, 2014, pp. 13:1–13:26
S79	A.M. Fernández-Sáez et al.	[115]	A.M. Fernández-Sáez, M.R.V. Chaudron, M. Genero, and I. Ramos, "Are forward designed or reverse-engineered UML diagrams more helpful for code maintenance?: A controlled experiment," in *Proceedings of the 17th International Conference on Evaluation and Assessment in Software Engineering*. New York, NY, USA: ACM, 2013, pp. 60–71
S80	G. Scanniello et al.	[116]	G. Scanniello, C. Gravino, G. Tortora, M. Genero, M. Risi, J.A. Cruz-Lemus, and G. Dodero, "Studying the effect of UML-based models on source-code comprehensibility: Results from a long-term investigation," in *Proceedings of the 16th International Conference on Product-Focused Software Process Improvement*, Vol. 9459, New York, 2015, pp. 311–327
S81	A.M. Fernández-Sáez et al.	[117]	A.M. Fernández-Sáez, M. Genero, D. Caivano, and M.R. Chaudron, "Does the level of detail of UML diagrams affect the maintainability of source code?: A family of experiments," *Empirical Software Engineering*, Vol. 21, No. 1, 2016, pp. 212–259
S82	G. Scanniello et al.	[118]	G. Scanniello, C. Gravino, M. Genero, J.A. Cruz-Lemus, G. Tortora, M. Risi, and G. Dodero, "Do software models based on the UML aid in source-code comprehensibility? Aggregating evidence from 12 controlled experiments," *Empirical Software Engineering*, Vol. 23, No. 5, 2018, pp. 2695–2733

Table A6. Results of data extraction from nine databases

ID	Ref.	MQ1	MQ2		MQ3	MQ4	MQ5	MQ6	MQ8
		Publication year	Publication source	Publication channel	Research type	Empirical approach	Software application type	Dataset	Technique
S1	[44]	2000	CRE	Conference	SP	Ex	POA	ANSI C programs	PR
S2	[45]	2003	SEKE	Conference	ER	HbE	OOA	QUES, UIMS	GRNN, WNN
S3	[46]	2003	METRICS	Symposium	SP	Ex	OOA	Different OO application domains	MLR
S4	[47]	2004	QSIC	Conference	SP	Ex	OOA	Different OO application domains	DA
S5	[48]	2004	CSMR	Conference	SP	CS	WbA	Different WA Freeware applications	WF
S6	[49]	2005	IST	Journal	ER	HbE	OOA	QUES, UIMS	BN, RT, BE, SS
S7	[50]	2005	ICSM	Conference	SP	Ex	OOA	Different software student projects	MLR
S8	[51]	2005	SQJ	Journal	SP	Ex	OOA	C++ open source software	MLR
S9	[52]	2006	JSS	Journal	ER	HbE	OOA	QUES, UIMS	MARS, MLR, ANN, RT, SVR
S10	[19]	2006	AJCAI	Conference	SP	HbE	POA	Medical Imaging System	GMM, SVM-RBF, DT
S11	[53]	2007	ICACC	Conference	SP	Ex	NI	Different software data	FL
S12	[54]	2007	ESE	Journal	SP	Ex	OOA	Different OO application domains	MLR
S13	[55]	2007	ISPRDC	Symposium	SP	CS	NI	Different software projects	SM
S14	[56]	2008	IJCEACIE	Journal	ER	HbE	OOA	QUES, UIMS	ANN
S15	[57]	2008	ICSKG	Conference	SP	HbE	POA	Medical imaging System	AODE, SVM-LIN, NB, BN, RF, KNN, C4.5, OneR, RBF
S16	[58]	2008	WUJNS	Journal	SP	Ex	OOA	Java open source software	MLR
S17	[59]	2009	CISE	Conference	SP	CS	NI	Data from software design	CBR
S18	[60]	2009	ICCC	Conference	SP	CS	CbA	Billing system	FL
S19	[61]	2009	ICISE	Conference	ER	HbE	OOA	QUES, UIMS	PPR, ANN, MARS

Table A6 continued

ID	Ref.	MQ1	MQ2		MQ3	MQ4	MQ5	MQ6	MQ8
		Publication year	Publication source	Publication channel	Research type	Empirical approach	Software application type	Dataset	Technique
S20	[62]	2009	SIGSOFT	Conference	SP	HbE	POA	Aggarwal	FL
S21	[63]	2009	CSMR	Conference	ER	HbE	OOA	QUES, UIMS	TreeNet
S22	[64]	2010	JC	Journal	SP	HbE	OOA	BIS	MLR
S23	[65]	2010	IJCA	Journal	ER	HbE	OOA	QUES, UIMS	FFNN, FIS, ANFIS, GRNN, RBF
S24	[66]	2010	ICMIT	Conference	ER	Ex	OOA	Software developed by students	SVM
S25	[67]	2010	ICCIS	Conference	SP	CS	NI	Different software products	FET
S26	[68]	2010	JC	Journal	SP	HbE	OOA	QUES, UIMS	ELM
S27	[69]	2011	IJAES	Journal	SP	CS	OOA	ATM System	FL
S28	[70]	2012	IJCA	Journal	SP	Ex	OOA	OO software systems	FL
S29	[71]	2012	SIGSOFT	Conference	ER	HbE	OOA	QUES, UIMS	MLP
S30	[72]	2012	ICITeS	Conference	SP	CS	OOA	Real BIS	LR
S31	[73]	2012	IJERA	Journal	ER	HbE	OOA	UIMS	ANN
S32	[74]	2012	ICSESS	Conference	ER	HbE	OOA	QUES, UIMS	MFL
S33	[75]	2012	SEIJ	Journal	ER	HbE	OOA	QUES, UIMS	GMDH, GA, PNN
S34	[76]	2012	ICSM	Conference	SP	CS	OOA	Jedit, Log4j Java projects	PD
S35	[77]	2012	IJSCE	Journal	ER	HbE	OOA	UIMS	MLP
S36	[78]	2012	ASEA-DRBC	Conference	SP	Ex	OOA	Jedit, Industrial software product	LR, ANN, DT
S37	[79]	2012	IJCA	Journal	ER	HbE	OOA	QUES, UIMS	SVM-RBF
S38	[80]	2013	IWCANN	Conference	SP	HbE	OOA	QUES, UIMS	MLP, RBF, SVM, M5P, Ensemble
S39	[81]	2013	IWSQM	Workshop	SP	HbE	OOA	Jedit	PD
S40	[82]	2013	QR2MSE	Conference	SP	Ex	NI	Virtual maintenance system	FIT
S41	[83]	2013	QR2MSE	Conference	SP	Ex	OOA	C++ open source system	DT, BPNN, SVM, Bagging
S42	[84]	2013	ESSE	Journal	SP	HbE	OOA	QUES, UIMS	MFL, ANFIS, PNN, RBF, SVM

Table A6 continued

ID	Ref.	MQ1	MQ2		MQ3	MQ4	MQ5	MQ6	MQ8
		Publication year	Publication source	Publication channel	Research type	Empirical approach	Software application type	Dataset	Technique
S43	[85]	2013	ICTACT	Journal	ER	HbE	OOA	QUES, UIMS	SBLLM, ELM
S44	[3]	2013	IJSEKE	Journal	ER	HbE	OOA	QUES, UIMS	MLR, LMSR, PaceR, PPR, IR, RegByDisc, GPR, MLP, RBF, AR, GRNN, GMDH, SVR, FSC, DS, ANFIS, K*, M5P, LWL, Bagging, KNN, REPTree, RF, ES, CR, DTable, M5R
S45	[86]	2013	IJCT	Journal	ER	HbE	OOA	QUES	KMC, XMC
S46	[87]	2013	IST	Journal	SP	Ex	OOA	Different OO application domains	LgR
S47	[21]	2014	IJSAE	Journal	ER	HbE	OOA	FLM, EASY	GMDH, FF3LBPN, GRNN
S48	[88]	2014	ICRITO	Conference	SP	HbE	OOA	Lucence	SS, BE
S49	[14]	2014	ICDMIC	Conference	ER	HbE	OOA	Lucence, JHotdraw, JEdit, JTreeview	NB, BN, LgR, MLP, RF
S50	[89]	2014	ICICT	Conference	SP	Ex	OOA	Private software system	FL
S51	[90]	2014	IJARCSSE	Journal	SP	HbE	OOA	BIS	ML
S52	[91]	2014	–	Chapter	SP	HbE	OOA	FLM, EASY, ABP, SMS, IMS	MLR, BPNN, KN, FFNN, GRNN
S53	[92]	2014	ICCSA	Conference	SP	Ex	WbA	Different WA domains	LR
S54	[93]	2015	JSC	Journal	ER	HbE	OOA	QUES, UIMS, Vssplugin, PeerSim	MLP, RBF, SVM, M5P, DT, LgR, KMC, GEP, MV, NL, Boosting, Bagging, AVG, WT, BTE
S55	[94]	2015	IC2IT	Conference	SP	HbE	OOA	QUES, UIMS	ANN, Neuro-GA
S56	[95]	2015	ICISMTT	Conference	SP	HbE	OOA	UIMS	T2FLS
S57	[96]	2015	PCS	Journal	ER	HbE	OOA	QUES, UIMS	Neuro-GA
S58	[97]	2016	JSS	Journal	SP	HbE	OOA	QUES, UIMS	FGA, AFGA, FPSO, MFPSO, FCSA

Table A6 continued

ID	Ref.	MQ1	MQ2		MQ3	MQ4	MQ5	MQ6	MQ8
		Publication year	Publication source	Publication channel	Research type	Empirical approach	Software application type	Dataset	Technique
S59	[17]	2016	SCEECS	Conference	ER	HbE	OOA	jTDS, jWebUnit , jXLS, SoundHelix	GA, DTable, RBF, BN, SMO
S60	[98]	2016	IJICIC	Journal	ER	HbE	OOA	Drumkit, OpenCV, Abdera, Ivy, Log4j, JEdit , JUnit	LR, M5R, DT, SVM, K*, Bagging, JERN, BPNN, KN, PNN, GMDH, GRNN, GGAL
S61	[99]	2016	IJSAEM	Journal	ER	HbE	SOA	5 versions of eBay web service system	SVM-LIN, SVM-SIG, SVM-RBF
S62	[100]	2016	ICRITO	Conference	ER	HbE	OOA	jTDS, Jchess, ArtOfIllusion, OrDrumbox	GEP, DFT, SVM, LR, MLP, RBF
S63	[32]	2016	JSS	Journal	ER	HbE	OOA	Camel, JEdit, Tomcat, JHotDraw	MLR
S64	[101]	2016	ICACCI	Conference	SP	HbE	OOA	jtds	GdA
S65	[102]	2017	HASE	Symposium	ER	HbE	OOA	Eclipse software application	LR, NB, ELM-LIN, ELM-PLY, BTE, MV, SVM-SIG, ELM-RBF, SVM-LIN, SVM-RBF
S66	[103]	2017	ICACCI	Conference	ER	Ex	OOA	Apache Jackrabbit, Light Weight Java Game Library	LR, Cubist, Lasso, Elastic Net, RF
S67	[104]	2017	MSL	Journal	SP	Ex	NI	3 software products	FL

Table A6 continued

ID	Ref.	MQ1	MQ2		MQ3	MQ4	MQ5	MQ6	MQ8
		Publication year	Publication source	Publication channel	Research type	Empirical approach	Software application type	Dataset	Technique
S68	[16]	2017	ES	Journal	ER	HbE	OOA	Art of illusion, Camel, Eclipse, Free mind, Games, Gantt, Geoxygene, Ivy, Jabref, Jajuk, Jasper reports, Javaml, Jfree ant, Jfree chart, Jgap, Jmt, Jnetpcap, Lucene, Mallet, Pandora, POI, Sglj, Tree view, Ujac, Workzen, Xalan	MLR, MLP, SVR, M5P
S69	[105]	2017	IJSAEM	Journal	SP	HbE	OOA	UIMS, QUES	Neuro Fuzzy
S70	[106]	2017	SOCA	Journal	ER	HbE	SOA	5 versions of eBay web service	SVM-LIN, SVM-SIG, SVM-RBF
S71	[107]	2017	HASE	Symposium	ER	HbE	SOA	5 versions of eBay web services	MARS, MLR, SVM
S72	[108]	2017	MaLTeSQuE	Workshop	SP	HbE	SOA	5 versions of eBay web services	LSSVM-LIN, LSSVM-RBF, LSSVM-SIG
S73	[109]	2017	JIPS	Journal	ER	HbE	OOA	Art-of-Illusion, Sweet-Home-3D	LgR, RF, Bagging, AdaBoost, MLP, BN, NB, LogitBoost, J48, NNge
S74	[110]	2017	JSS	Journal	ER	Ex	OOA	Industrial systems	PD
S75	[111]	2017	AMSE/IIETA	Journal	SP	Ex	OOA	Private data sources	MFL
S76	[112]	2018	IST	Journal	SP	Ex	OOA	antlr4, junit, mapdb, mcMMO, mct, oryx, titan	Statistical

Table A6 continued

ID	Ref.	MQ1	MQ2		MQ3	MQ4	MQ5	MQ6	MQ8
		Publication year	Publication source	Publication channel	Research type	Empirical approach	Software application type	Dataset	Technique
S77	[113]	2018	arXiv	Journal	ER	HbE	OOA	Compare, webdav, debug, update, core, swt, team, pde, ui, jdt	LR, PR, LgR, DT, SVM-LIN, SVM-PLY, SVM-RBF, ELM-LIN, ELM-PLY, ELM-RBF, LSSVM-LIN, LSSVM-PLY, LSSVM-RBF, NGD, NGDM, BTE, NGDA, NNM, MVE, NDTF, NLM
S78	[114]	2014	TOSEM	Journal	SP	Ex	OOA	2 private systems (the selling of CDs/DVDs in a music shop and the booking of theater tickets)	Statistical
S79	[115]	2015	IST	Journal	SP	Ex	OOA	OO application domain (Sports center application which was created as part of the Master's Thesis of a student from the University of Castilla-La Mancha)	Statistical
S80	[116]	2015	PROFES	Conference	SP	Ex	OOA	A chunk of a system music shop software and a chunk of a theater ticket reservation system implemented in Java, JHotDraw	Statistical

Table A6 continued

ID	Ref.	MQ1	MQ2		MQ3	MQ4	MQ5	MQ6	MQ8
		Publication year	Publication source	Publication channel	Research type	Empirical approach	Software application type	Dataset	Technique
S81	[117]	2016	ESE	Journal	SP	Ex	OOA	2 systems (a library application from which a user can borrow books and a sport center application from which users can rent services)	Statistical
S82	[118]	2018	ESE	Journal	SP	Ex	OOA	OO application domains (music shop and theater ticket reservation applications)	Statistical

Acronyms used in table:

• **Research type acronyms**: Solution Proposal (SP), Evaluation Research (ER)

• **Empirical approach acronyms**: History-based Evaluation (HbE), Experiment (Ex), Case study (CS)

• **Software application type acronyms**: Object Oriented Applications (OOA), Procedural Oriented Applications (POA), Web-based Applications (WbA), Component-based Application (CbA), Service Oriented Applications (SOA), Not Identified (NI).

• **Publication source acronyms**: Conference on Reverse Engineering (CRE), International Conference on Software Engineering & Knowledge Engineering (SEKE), International Software Metrics Symposium (METRICS), International Conference on Quality Software (QSIC), European Conference on Software Maintenance and Reengineering (CSMR), Information and Software Technology (IST), International Conference on Software Maintenance (ICSM), Software Quality Journal (SQJ), The Journal of Systems & Software (JSS), International Conference on Advanced Computing and Communications (ICACC), Empirical Software Engineering (ESE), IEEE International Symposium on Pacific Rim Dependable Computing (ISPRDC), International Journal of Computer, Electrical, Automation, Control and Information Engineering (IJCEACIE),International Conference on Semantics, Knowledge and Grid (ICSKG), Wuhan University Journal of Natural Sciences (WUJNS), International Conference on Computational Intelligence and Software Engineering (CISE), Computing International Conference on Information Science and Engineering (ICISE), SIGSOFT Software Engineering Notes (SIGSOFT), Journal of Computing (JC), International Journal of Computer Applications (IJCA), International Conference on Multi-Media and Information Technology (ICMIT), International Conference on Computer and Information Science (ICCIS), International Journal of Advances in Engineering Sciences (IJAES), International Conference on Information Technology and e-Services (ICITeS), International Journal of Engineering Research and Applications (IJERA), International Conference on Software Engineering and Service Science (ICSESS), Software engineering: an international Journal (SEIJ), International Journal of Soft Computing and Engineering (IJSCE), International Workshop on Software Quality and Maintainability (IWSQM), International Conference on Quality, Reliability, Risk, Maintenance, and Safety Engineering (QR2MSE), Special Issue on Empirical Studies in Software Engineering (ESSE), ICTACT Journal on Soft Computing (ICTACT), International Journal of Software Engineering and Knowledge En-

gineering (IJSEKE), International Journal of Computers & Technology (IJCT), International Journal of System Assurance Engineering and Management (IJSAEM), International Conference on Reliability, Infocom Technologies and Optimization (ICRITO), International Conference on Data Mining and Intelligent Computing (ICDMIC), International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT), International Journal of Advanced Research in Computer Science and Software Engineering (IJARCSSE), International Conference on Computational Science and Its Applications (ICCSA), The Journal of Soft Computing (JSC), International Conference on Intelligent Software Methodologies, Tools, and Techniques (ICISMTT), Procedia Computer Science (PCS), International Work-Conference on Artificial Neural Networks (IWCANN), Computer Applications for Software Engineering, Disaster Recovery, and Business Continuity (ASEA-DRBC), Australasian Joint Conference on Artificial Intelligence (AJCAI), International Conference on Contemporary Computing (ICCC), International Conference on Computing and Information Technology (IC2IT), IEEE Students' Conference on Electrical, Electronics and Computer Science (SCEECS), International Journal of Innovative Computing, Information and Control (IJICIC), International Conference on Reliability Infocom Technologies and Optimization (ICRITO), International Conference on Advances in Computing, Communications and Informatics (ICACCI), International Symposium on High Assurance Systems Engineering (HASE), Management Science Letters (MSL), Evolving System: An Interdisciplinary Journal for Advanced Science and Technology (ES), Service Oriented Computing and Applications (SOCA), Workshop on Machine Learning Techniques for Software Quality Evaluation (MaLTeSQuE), Journal of Information Processing Systems (JIPS), arXiv Repository of electronic preprints (arXiv), ACM Transactions on Software Engineering and Methodology (TOSEM), International Conference on Product-Focused Software Process Improvement (PROFES), Association for the Advancement of Modelling and Simulation Techniques in Enterprises/International Information and Engineering Technology Association (AMSE/IIETA).

• **Technique acronyms**: Linear Regression (LR), Multiple Linear Regression (MLR), Logistic Regression (LgR), Backward Elimination (BE), Stepwise Selection (SS), Multiple Adaptive Regression Splines (MARS), Projection Pursuit Regression (PPR), Polynomial Regression (PR), Least Median of Squares Regression (LMSR), Pace Regression (PaceR), Isotonic Regression (IR), Regression By Discretization (RegByDisc), Additive Regression (AR), Gaussian Process Regression (GPR), Least Absolute Shrinkage and Selection Operator (Lasso), Probability Density function (PD), Gaussian Mixture Model (GMM), Discriminant Analysis (DA), Weighted Functions (WF), Stochastic Model (SM), Artificial Neural network (ANN), Multilayer Perceptron (MLP), Radial Basis Function Network (RBF), Probabilistic Neural Network (PNN), Group Method of Data Handling (GMDH), General Regression Neural Network (GRNN), Feed Forward Neural Network (FFNN), Back Propagation Neural Network (BPNN), Kohonen Network (KN), Ward Neural Network (WNN), Feed Forward 3-Layer Back Propagation Network (FF3LBPN), Extreme Learning Machines (ELM), Sensitivity Based Linear Learning Method (SBLLM), Neuro-Genetic algorithm (Neuro-GA), Functional Link Artificial Neural Network (FLAAN) with Genetic Algorithm (FGA), Adaptive FLANN-Genetic (AFGA), FLANN-Particle Swarm Optimization (FPSO), Modified-FLANN Particle Swarm Optimization (MFPSO), FLANN-Clonal Selection Algorithm (FCSA), ELM with linear (ELM-LIN), ELM with polynomial (ELM-PLY), ELM with Radial Basis Function kernels (ELM-RBF), ANN with Levenberg Marquardt method (NLM), GRNN with Genetic Adaptive Learning (GGAL), Jordan Elman Recurrent Network (JERN), ANN with normally Gradient descent method (NGD), ANN with Gradient descent with momentum (NGDM), ANN with Gradient descent with adaptive learning rate (NGDA) method, ANN with Quasi-Newton method (NNM), Kstar (K*), Locally Weighted Learning (LWL), k Nearest Neighbor (IBK or KNN), Nearest-Neighbor-like algorithm that uses non-nested generalized exemplars (NNge), Bayesian Networks (BN), Decision Tree (DT), Regression tree (RT), M5 for inducing trees of regression models (M5P), Random Forest (RF), Decision Stump (DS), Reduced Error Pruned Tree (REPTree), Decision Tree Forest (DFT), Genetic Expression programming (GEP), Case-Based Reasoning (CBR), Genetic Algorithm (GA), Greedy algorithm (GdA), Na?ve-Bayes (NB), Aggregating One-Dependence Estimators (AODE), Support Vector Machine (SVM), Support Vector Regression (SVR), Sequential Minimal Optimization (SMO), SVM with radial basis function kernel (SVM-RBF), SVM with linear kernel (SVM-LIN), SVM with sigmoid kernel (SVM-SIG), Least Square Support Vector Machine (LSSVM) with linear kernel (LSSVM-LIN), LSSVM with radial basis function kernel (LSSVM-RBF), SVM with Polynomial Kernel (SVM-PLY), LSSVM with

sigmoid kernel (LSSVM-SIG), LSSVM with Polynomial Kernel (LSSVM-PLY), Fuzzy Logic (FL), Adaptive Neuro-Fuzzy Inference Systems (ANFIS), Fuzzy Inference Systems (FIS), Type-2 fuzzy logic system (T2FLS), Mamdani-Based Fuzzy Logic (MFL), Fuzzy Entropy Theory (FET), Fuzzy Subtractive Clustering (FSC), Fuzzy Integral theory (FIT), Decision Table (DTable), Conjunctive Rule Learner (CR), M5 Rules (M5R), K-Means Clustering algorithm (KMC), X-Means Clustering algorithm (XMC), Ensemble Selection (ES), Average-based ensemble (AVG), Weighted-based ensemble (WT), Best-in-training-based ensemble (BTE), Majority-voting ensemble (MV), Non-linear ensemble (NL), Nonlinear Ensemble Decision Tree Forest (NDTF), Adaptive Boosting (AdaBoost).

• **Dataset acronyms**: UIMS (User Interface Management System), QUES (Quality Evaluation System), VSSPLUGIN (Visual Source Safe PLUGIN), PeerSim (Peer-to-Peer Simulator), MIS (Medical imaging system), FLM (File Letter Monitoring System), EASY (EASY Classes Online Services collection), SMS (Student Management System), IMS (Inventory Management System) and APB (Angel bill printing), Bank Information System (BIS).

Table A7. Distribution of predictors used as independent variables

Predictors	Supported studies	# of studies (percent)
Chidamber and Kemerer (C&K) measures, such as: coupling between object (CBO), depth of inheritance tree (DIT), number of children (NOC), weighted methods per Class (WMC), response for a class (RFC), lack of cohesion in methods (LCOM)	S2, S6, S8, S9, S14, S16, S19, S21, S23, S24, S26, S28, S29, S31, S32, S34, S33, S35, S37, S38, S41, S42, S43, S44, S45, S46, S47, S48, S49, S52, S54, S55, S56, S57, S58, S59, S60, S61, S62, S64, S65, S66, S68, S69, S70, S71, S72, S73, S75, S77	50 (61%)
Li and Henry (L&H) measures, such as: message passing coupling (MPC), data abstraction coupling (DAC), number of local methods (NOM), SIZE1 (LOC calculated by counting the number of semicolons in a class), SIZE2 (Number of properties including the attributes and methods defined in a class)	S2, S6, S9, S14, S19, S21, S23, S24, S26, S29, S31, S32, S33, S35, S37, S38, S42, S43, S44, S45, S46, S47, S48, S49, S54, S56, S57, S58, S61, S65, S69, S70, S71	33 (40%)
Class diagram measures, such as measures related to: – method level such as: number of methods, average number of methods per class, number of foreign methods accessed, number of local methods accessed, number of constructors, etc. – attribute level such as: number of attributes, average number of attributes per class, number of attributes added, etc. – class level such as: number of classes, classes changed, classes added, etc. – relationships such as: number of generalisations, number of associations, number of aggregations, number of dependencies.	S3, S4, S7, S8, S12, S16, S22, S27, S36, S39, S41, S46, S48, S49, S51, S59, S62, S65, S66, S68, S72, S73, S74, S77	24 (29%)
Source code size measures, different lines of code (LOC) measures, such as: source lines of code, logical lines of code, total lines of code, etc.	S7, S8, S10, S15, S20, S34, S36, S39, S41, S46, S48, S49, S62, S65, S66, S68, S73, S74, S76, S77	20 (24%)
McCabe complexity measure (McCabe)	S1, S20, S27, S34, S36, S39, S48, S49, S50, S52, S61, S70, S71, S72, S73, S74, S76	17 (21%)
Software quality attributes, such as: understandability, documentation quality, readability of source code, testability, etc.	S11, S17, S18, S25, S40, S50, S63, S67	8 (10%)
Martin's measures, such as: abstractness (A), the distance from the main sequence (D), and the normalized distance from the main sequence (Dn), etc.	S48, S49, S59, S60, S63, S66	6 (7%)
Halstead measures, such as: number of distinct operators, number of distinct operands, total number of occurrences of operators, total number of occurrences of operands, length (N) of a program, program volume (V), etc.	S8, S10, S15, S48, S49	5 (6%)
Brito e Abreu and Carapuça (BA&C) measures, such as: method hiding factor, polymorphism factor, etc.	S8, S16, S62, S64, S68	5 (6%)

Table A7 continued

Predictors	Supported studies	# of studies (percent)
Factors such as: origin of UML diagrams, level of detail of UML diagrams, method (analysis models and source code, and source code alone), models (software models plus source code without comment), ability, and source code without comments.	S78, S79, S80, S81, S82	5 (6%)
Coding rule measures, such as: number of serious coding rule violations, number of suspicious coding rule violations, number of coding style issues, etc.	S34, S36, S74	3 (4%)
QMOOD measures, such as: data access metric (DAM), measure of aggression (MOA), method of functional abstraction (MFA), etc.	S59, S60, S66	3 (4%)
Maintainability index (MI) measure	S7, S52	2 (2%)
Web-based application (WbA) measures, such as: total web page, server script, web page control structure, client page, web control coupling, server page, etc. class diagram measures	S5, S53	2(2%)
Jensen measures	S10, S15	2 (2%)
Effort measures, such as: coding effort, design effort, requirement effort, effort integration, ratio of requirement effort and design effort to coding effort (RDCRatio), etc.	S7, S30	2 (2%)
Module level measures, such as: percentage of modules changed, module level information flow, etc.	S1, S7	2 (2%)
Sequence diagram measures, such as measures related to: scenarios (number of scenarios), – messages (average number of return messages, weighted messages between objects, average number of the directly dispatched messages, etc. – conditions (average number of condition messages)	S4	1 (1%)
Lorenz and Kidd (L&K) measures, such as: average method size and coupling factor, etc.	S8	1 (1%)
Fault measures, such as: number of detected faults and number of corrected faults, etc.	S13	1 (1%)
Database measures, such as: number of data base connections and the schema complexity to comment ratio, etc.	S52	1 (1%)

Table A8. Acronyms of successful predictors

Acronym	Description	Acronym	Description
ACLOC	Average class lines of code	NAssoc	Number of associations
AIF	Attribute inheritance factor	NC	Number of classes
AMLOC	Average method lines of code	NClienP	Number of client pages
AVPATHS	Average depth of paths	NHD	Normalized Hamming distance
B	Number of bugs (Halstead)	NEWLCOM3	New lack of cohesion in methods 3
Ca	Afferent coupling	NPM	Number of Public methods
CAMC	Cohesion among methods in a class	NDep	Number of dependencies

Table A8 continued

Acronym	Description	Acronym	Description
CBO_IUB	Coupling between object (CBO) – Is used by attributes or methods of class	NGen	Number of generalisation
CBO_U	CBO – Using by the methods of class	NGenH	Number of generalisation hierarchies
CC	Cyclomatic complexity	NM	Number of methods
CDENS	Control DENSity	NOC	Number of children
Ce	Efferent coupling	NODBC	Number of data base connections
CLOC	Comments lines of code	NOM	Number of local methods
ClS	Client scripts	NPAVGC	Average number of parameters per method
COF	Coupling factor	NServP	Number of server pages
Coh	Cohesion	NWebP	Number of web pages
Command	Number of commands	OCmax	Maximum operation complexity
CONS	Number of constructors	OCMEC	Other class method export coupling
CSA	Average number of attributes per class	OL2	The average strength of the attribute
CSO	Average number of methods per class	OSAVG	Average complexity per method
Cyclic	Number of cyclic dependencies	OSmax	Maximum operation size
DAC	Data abstraction coupling	PCCC	Path connectivity class cohesion
DAM	Data access metric	POF	Polymorphism factor
DCd	Degree of cohesion-direct	PPPC	Percentage public/protected members
DCi	Degree of cohesion-indirect	Query	Number of query
Dcy	Number of dependencies	RFC	Response for a class
Dcy*	Number of transitive dependencies	RDCRatio	Ratio of Requirement Effort and Design Effort to Coding Effort
DIT	Depth of inheritance tree	SCCR	Schema complexity to comment Ratio
Inner*	Number of inner classes	SCOM	Class cohesion metric
LCC	Loose class cohesion	SIZE1/LOC	Line of code
LCOM	Lack of cohesion in methods	SIZE2	Number of Properties
LLOC	Logical lines of code	SLoc	Source lines of code
LSCC	Low-level design similarity-based class cohesion	SS	Server scripts
MaxDIT	Maximum depth of inheritance tree	STAT	Number of STATements
MI	Maintainability index	SWMC	Average weighted methods per class
MIF	Method inheritance factor	TCC	Total cyclomatic complexity
MOA	Measure of aggregation	TCC	Tight class cohesion
MPC	Message passing coupling	TL	Total languages
n	Vocabulary size (Halstead)	TLOC	Total LOC
N	Program length (Halstead)	TWP	Total web page
NA/NOA	Number of attributes	TWPDC	Total web page data coupling
NAA	Number of attributes added	TWPR	Total web page relationships
NAgg	Number of aggregations	V	Program volume (Halstead)
NAggH	Number of aggregations hierarchies	WMC	Weighted methods per Class
NAggR	Number of aggregation relationships	WO	WebObject

Table A9. Prediction techniques and accuracy criteria per dependent variable topics

Topic	ID	Prediction technique per category	Accuracy criteria
Change	S2	ANN	R-squared, R, MSE, MAE, MinAE, MaxAE
	S6	BN, DT, RA	MaxMRE, MMRE, Pred(25), Pred(30), sum Ab. Res, med. Ab. Res, Std. Ab. Res
	S9	RA, ANN, DR, SVM/R	MaxMRE, MMRE, Pred(25), Pred(30), sum ARE, Med. ARE, Std. ARE
	S10	GMM, SVM/R, DT	WAP, recall
	S14	ANN	MARE, R, MRE
	S15	ANN, SVM/R, DT, BN, CBR	WAP, WARec
	S19	ANN, RA	RMSE
	S23	ANN, FNF	MARE, MRE, R
	S24	SVM/R	MARE, MRE, R
	S26	ANN	MaxMRE, MMRE, Pred(25), Pred(30), Sum Ab. Res, Med. Ab. Res, Std. Ab. Res
	S29	ANN	R-squared, R, MAE, minAE, maxAE
	S30	RA	–
	S31	ANN	–
	S32	FNF	RMSE, NRMSE, MMRE
	S33	ANN, EA	MaxMRE, MMRE, MARE, Pred(25), Pred(30), Pred(75)
	S35	ANN	R, MAE
	S37	SVM/R	MARE, MRE, R
	S38	ANN, SVM/R, DT, EM	MMRE, Std. MRE, Pred(30)
	S42	FNF, ANN, SVM/R	NRMSE, MMRE, Pred(25), Pred(30).
	S43	ANN	MaxMRE, MMRE, Pred(25), Pred(30), Sum Ab. Res., Med. Ab. Res., Std. Ab. Res.
	S44	RA, ANN,SVM/R, DT, FNF, CBR, IRB	MaxMRE, MMRE, Pred(25), Pred(30), Sum ARE, Med. ARE, Std. ARE, RMSE
	S45	CM	Qout, Nit, cut-off
	S47	ANN	MMRE, Pred(25), Pred(30), % under, % over
	S48	RA	R-squared, R
	S49	RA, ANN, BN	Recall, Precision, ROC area
	S52	RA, ANN	MaxMRE, MMRE, Pred(25)
	S54	RA, ANN, SVM/R, DT, EA, CM	MMRE, Std. MRE, Pred(30), CCR, AUC
	S55	ANN	MMRE, MARE, MAE, RMSE, SEM
	S57	ANN	MAE, MARE, RMSE, SEM, MMRE, e, é
	S58	ANN	MAE, R, MMRE, e, é
	S59	ANN, SVM/R, BN, EA, IRB	MAE, RMSE

TableA9 continued

Topic	ID	Prediction technique per category	Accuracy criteria
Change	S60	RA, ANN, SVM/R, DT, FNF, CBR, IRB	MAE, RMSE, Pred(25), Pred(75)
	S61	SVM/R	Precision, Recall, F-measure, Specificity, Accuracy, AUC
	S62	RA, ANN, SVM/R, DT, EA	MAE, RMSE
	S64	EA	–
	S65	RA, ANN, SVM/R, EM, BN	Accuracy, AUC
	S66	RA	MAE, RMSE, Accuracy
	S69	FNF	Performance index
	S70	SVM/R	F-measure, Accuracy
	S71	RA, SVM/R	Accuracy, Recall, Precision
	S72	SVM/R	F-measure, Accuracy
	S73	RA, ANN, DT, BN, CBR	Sensitivity, Specificity, ROC, cutoff
	S74	PD	–
	S77	RA, ANN, SVM/R, DT	Accuracy, F-measure
Expert opinion	S11, S18, S20, S25, S27, S28, S50, S67	FNF	–
	S36	RA, ANN, DT	MAE
	S41	ANN, DT, SVM/R	TPR, FPR, Precision, Recall, F1 score, AUC
Maintainability index	S8	RA	R-squared, R, adjusted R-squared, Std. EE
	S16	RA	R-squared, MARE, MMRE
	S39	PF	R
	S40	FNF	–
	S48	RA	R-squared, R
	S68	RA, ANN, DT, SVM/R	AOC, StdMRE, MMRE, Pred(30)
	S75	FNF	–
	S76	Statistical	Rs
Maintainability level	S4	DA	Accuracy
	S22	RA	R-squared, R, adjusted R-squared, Std.EE,
	S51, S53	RA	Rs
	S78, S80, S82	Statistical	–
Maintainability time	S3	RA	MMRE, qMRE, Pred(30)
	S5	WF, DA	–
	S7	RA	R-squared
	S12	RA	–
	S17	CBR	–
	S78, S80, S82	Statistical	–

Accuracy criteria acronyms: Magnitude of Relative Error (MRE), Mean MRE (MMRE), quartiles of MRE distribution (qMRE), Standard Deviation of MRE (Std.MRE), Coefficient of correlation R, Coefficient of determination (R-squared), Percentage Relative Error Deviation (Pred(0.25), Pred(0.30), Pred(0.75)), Mean Absolute Error (MAE), Minimum AE (MinAE), Maximum AE (MaxAE), Coefficient of correlation (R), Root Mean Square Error (RMSE), Normalized RMSE (NRMSE), Standard Error of the Estimate (Std.EE),

Weighted Average Precision (WAP), Area Under Curve (AUC), True Positive Rate (TPR), False Positive Rate (FPR), Weighted Average Recall (WARec), Spearman's coefficient of correlation (Rs), Cut-off factors (cut-off), Mean Square Error (MSE), Mean Absolute Relative Error (MARE), Roc Area (ROC), Sum of Absolute Residual (Sum Ab. Res), Standard Deviation of Absolute Residual (Std. Ab. Res), Median of Absolute Residual (Med. Ab. Res), Standard Error of the Mean (SEM). Area Over Curve (AOC).