Received December 30, 2020, accepted January 15, 2021, date of publication January 26, 2021, date of current version March 26, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3054730

Software Product Quality Metrics: A Systematic Mapping Study

FATIMA NUR COLAKOGLU[1], ALI YAZICI[2], AND ALOK MISHRA[2,3], (Senior Member, IEEE)
[1]HAVELSAN, 06510 Ankara, Turkey
[2]Department of Software Engineering, Atilim University, 06830 Ankara, Turkey
[3]Department of Informatics and Digitalization, Molde University College—Specialized University in Logistics, 6410 Molde, Norway

Corresponding author: Alok Mishra (alok.mishra@himolde.no)

This work was supported by Molde University College—Specialized University in Logistics, Norway.

ABSTRACT In the current competitive world, producing quality products has become a prominent factor to succeed in business. In this respect, defining and following the software product quality metrics (SPQM) to detect the current quality situation and continuous improvement of systems have gained tremendous importance. Therefore, it is necessary to review the present studies in this area to allow for the analysis of the situation at hand, as well as to enable us to make predictions regarding the future research areas. The present research aims to analyze the active research areas and trends on this topic appearing in the literature during the last decade. A Systematic Mapping (SM) study was carried out on 70 articles and conference papers published between 2009 and 2019 on SPQM as indicated in their titles and abstract. The result is presented through graphics, explanations, and the mind mapping method. The outputs include the trend map between the years 2009 and 2019, knowledge about this area and measurement tools, issues determined to be open to development in this area, and conformity between conference papers, articles and internationally valid quality models. This study may serve as a foundation for future studies that aim to contribute to the development in this crucial field. Future SM studies might focus on this subject for measuring the quality of network performance and new technologies such as Artificial Intelligence (AI), Internet of things (IoT), Cloud of Things (CoT), Machine Learning, and Robotics.

INDEX TERMS Software quality, software product quality, metrics, systematic mapping.

I. INTRODUCTION

IEEE 1061:1998 defines measurement and software quality metric (SQM) as a function whose inputs are software data, while the output is a single numerical value that can be interpreted as the degree to which software possesses a given attribute that affects its quality [1]. Within the current competitive world, producing quality products has become a prominent factor that warrants the enduring success of competitors in business. In this regard defining and following the SPQM to be applied in the detection of the current quality state. Hence maintaining the continuous improvement of systems within the software industry has gained considerable significance.

Many international standards and models focusing on this need are in line with Tom DeMarco, who stated that "we cannot control and improve something that we haven't measured" [2]. Pursuing this motto, this paper sets out to analyze a specific set of articles and conference papers published in the last decade using the SM approach, which specifically focuses on SPQM as indicated in their titles and abstract sections.

The metrics and analyses of measurement results are good indicators for the quality of the products and/or organizational operations [3]. For instance, if the metrics spot a deviation from the threshold in a negative manner, a red flag will be raised immediately to implement an emergency plan for intervention [4]. In such cases, these metrics function as a safety belt for firms. In order to be able to compete in the industry, it is crucial to measure the current situation so that undesired scenarios can be prevented while continuous development is enhanced. This is so critical that if the road map is set with the wrong set of metrics, these meaningless and incorrect metrics will most probably mislead users [5]. In other words, if calibrated and validated metrics are not defined in an efficient way to reach the target, most likely

The associate editor coordinating the review of this manuscript and approving it for publication was Mario Luca Bernardi.

the pointer will show incorrect data. Such cases where, for example, the charts show operations to go smoothly, in reality, the company may have gone bankrupt.

In the 2018-2019 World Quality Report [6], the end-user satisfaction is shown to be only around 40% while the detection of software defects before delivery to the end-user rate is less than 40%. Additionally, an increase in the quality of software/product is around 40%. The mapping of SPQM is a strategically important process for the increase in the quality of products and processes. As indicated in the same report, the current situation of the SPQM is a topic that is treated with the utmost care in the international standards and software process models.

CMMI (Capability Maturity Model Integration) and PMBOK (Project Management Body of Knowledge) present detailed information about the plan, development and management of the projects and systems in software engineering. The new versions of CMMI v2.0 model and the PMBOK [7] are on agility and continuous improvement. The CMMI maturity model, and well-known standards such as IEEE 1061-Standard for a Software Quality Metrics Methodology, ISO 9001-Standard for Quality Management Systems, ISO 25010-Systems and Software Quality Requirements and Evaluation (SQuaRE), and AS9100-Standard for Quality Management Systems-Requirements for Aviation, Space and Defense Organizations all agree in their emphasis on the necessity of measurement and process analysis to increase the quality of product, process and projects. Moreover, in [8] it is highlighted that there is a need for guidance to evaluate the software quality of European projects. Ouhbi *et al.* explore the need for software product quality prediction by considering the software quality models and standards, especially ISO 25010 [9].

Based on these supporting arguments above, the main motivation of this paper is to conduct an SM study for the delineation of the state of the art in SPQM, serving to support future investigations on the subject. Undertaking an SM activity is a valuable experience providing both reusable research skills and a good overview of a research topic [10]. Additionally, such a study provides a systematic and objective procedure for identifying the nature and extent of the empirical study data that is available to answer a particular research question [11].

When the literature on SPQM was reviewed, a few SM studies related to the object-oriented software metrics only are found and there is no comprehensive study focusing on the SPQM with a general view which is crucial for SQA professionals and researchers. In addition, in the majority of the previous SM studies, it is observed that usually only a very specific and narrow field about SPQM is investigated. Recently Ouhbi *et al.* [9] also supported further study in this direction.

To this end, the main research question is to find out and investigate how the SPQM studies have been developed during the last decade, and what the patterns, trends, and gaps are in this area. This will be very helpful to researchers, Software Quality Assurance (SQA) professionals and software process and quality divisions in companies.

This study aims to extract the data for SPQM covering the years between 2009 and 2019, and to introduce the patterns, trends and gaps in this area with the help of the SM method [10]. This method guides both academia and the industry at all levels related to software engineering by presenting a trend map and revealing the relations among different terminologies. Further, the use of OpenAIRE digital library, and representing the trends with the mind mapping method are other two novel features of this study.

The remainder of the article is structured as follows. Section II gives some background information on software quality models and metrics, and data regarding the related papers on SPQM. Section III provides the research method and research questions (RQs), paper selection criteria, overview, quality assessment criteria, and data extraction of the selected studies. In Section IV answers to RQs are provided with visual graphics. The mind mapping of the SPQM trend and the discussion on these trends are included in Section V by comparing with other articles, standards and quality models. Section VI discusses the threats to the validity of the results by using four validity types. Section VII draws the conclusion, limitations, and suggestions for further research laid out by pointing to the current research scenario and trends in the last decade in this area and the improvement opportunities in SPQM.

II. BACKGROUND AND RELATED WORK

A. BACKGROUND

Quality is defined as the degree to which a set of solution characteristics fulfills the requirements in CMMI v2.0 [4]. Software Quality is defined in ISTQB as the totality of functionality and features of a software product that bear on its ability to satisfy stated or implied needs [12]. IEEE 1061:1998 defines software product metric as a metrics used to measure the characteristics of any intermediate or final product of the software development process [1].

ISO/IEC 25000 defines 'Quality Model' as a defined set of characteristics, and of relationships between them, which provide a framework for specifying quality requirements and evaluating quality [13]. Each one of these quality models consists of a set of quality characteristics and sub-characteristics, such as maintainability, reliability and so on. Therefore, it will be possible to contribute to the product quality by defining and using the metrics that serve the quality characteristics.

ISO 9126 Software Engineering Product Quality standard is revised by ISO 25010:2011 [14]. In comparison to these quality models, ISO 25010:2011 presents the newly categorized quality models in three areas, namely, Quality in Use (QinU), System/Software Product Quality (SPQ), and Data Quality (DQ) [15]. The definition of quality characteristics and sub-characteristics for each quality model are defined in ISO 25010 [15] and ISO 25012 [16].

Based on the definition of SQM given earlier, IEEE 1061:1998 standards define SPQM as a metric used to measure the characteristics of the documentation and code [1]. SPQM is the framework for software product quality assessment based on a set of quality sub-characteristics that are refined into a set of characteristics, which are further refined into a set of metrics [17].

The metric levels represent the classifications made according to the stages of measurement throughout the project life cycle. *Requirement level (Functional and Non-Functional), Code level (Class Level, Directory Level, File Level, Method Level, Variable Level), Design Level,* and *Test Level* are among the most common metric levels. Some SPQM examples based on the *code level* are [17]: Lack of Cohesion in Methods (LCOM) is defined as the number of non-intersecting sets of local methods. The cohesion of a class is characterized by how closely the local methods are related to the local instance variable in the class. Depth of Inheritance Tree (DIT) is a measure of how many ancestor classes can potentially affect this class. The node in the tree represent classes, the DIT metric is the length of the maximum path from the node to the root of the tree. LCOM and DIT are examples of *class level* metrics. Defect Density (DD) is the SPQM that evaluate the effectiveness of defect detection in project lifecycle [4]. Delivered Defect Density (DDD) metric is calculated by dividing the number of known defects by product size. DD and DDD are *test level* metrics.

In addition, examples of SPQM based on the quality attributes which are presented in ISO 25010 and ISO 25012 are as follows: Mean Time Between Failures (MTBF) is defined as a *Reliability* metric that is the average time between system breakdowns. MTBF is the example of *Reliability* quality attribute of SPQ and also can be categorized as a system level metric. Customer product satisfaction and perception trends are the example of *Satisfaction* quality attribute of QinU [4]. When *Availability, Portability* and *Recoverability* are the system-dependent data quality attributes; *Accuracy, Completeness, Consistency, Credibility,* and *Currentness* are related with inherent-data quality [16]. *Portability* metric given by 1-ET/ER, where ET is a measure of the resources needed to move the system to the target environment, and ER is a measure of the resources needed to create the system for the resident environment is an example of a system level metric [17].

In this study, SM is applied to analyze the SPQM in selected papers with reference to the most recent ISO 25010 series quality model (a list of papers is provided in the appendix).

B. RELATED WORK

SM and systematic literature review (SLR) studies related to SPQM and their content, scope, and differences are listed in Table 1. Although the SM and SLR studies in the table are related to software quality metrics, this study differs from the ones in this table in terms of scope and covered years.

Compared to these earlier studies in Table 1, in addition to the title and abstract fields of articles and conference papers about SPQM, our study focuses on the metrics applicable in all the development cycle that leads to software product quality for various types of application domains and programming types. In Table 1, only paper [18] is similar to the present study, but its scope is limited to object-oriented software's internal metrics only for the years 2004-2013.

Furthermore, paper [19] is related but not near to the present study. In [19], SM is conducted on software quality models focusing on encompassed model elements and supports to architecting quality. In the same study, RQs related to publication trends, types of research, common meta-models elements are considered in software quality models and architecture support of quality models. Also, there is no data about the SPQM as outlined in the present study. Thus their scope, approach and RQs are different from our study. The years between 2009 and 2019 have been selected for analysis in detail since the SPQMs have diverse and broad content. Therefore, the present study extends the RQs on SPQM and rigor.

After the release of the PROMISE repository, which is an online database provided by the School of IT and Engineering, University of Ottawa to share the datasets and models among software engineers [20], the publication rate of product quality metric-related papers increased significantly in the years 2008-2009. To analyze the condition of the progress and transition in this area during and after 2009, the period from 2009 to the current year is selected as the period of analysis for this SM study. Also, in the literature, there are some SLR and SM studies in 2008 which cover the analysis of quality metric-related data; therefore, the year 2008 was excluded from this study.

In the present study, the SM of SPQM was chosen because SPQM is one of the important knowledge areas in SWEBOK (Software Engineering Body of Knowledge) [21]. As indicated in the trends of World Quality Report, the SPQMs today are under a lot of attention, as a result of which there can be continuous improvement in these products based on international standards and models [6].

This study is a complementary study in the context of SPQM. Therefore, it is assumed that the readers have a degree of familiarity with software quality and SPQM terms to be able to follow the rationale herein.

III. RESEARCH METHOD

The related principles of SM as proposed in [22] are applied here as the research method. Accordingly, the process of this study has the following main steps: Developing the Research Method, Definition of Goal and RQs, Quality Assessment Criteria, Conducting Search and Paper Selection, Data Extraction, Data Synthesis and Results, Discussions, Threats to Validity, Conclusions, Limitations and Future Work.

A. DEVELOPING AND EVALUATING THE RESEARCH METHOD

The research method for this SM study is given in Fig.1. The process starts with the article and conference paper selection

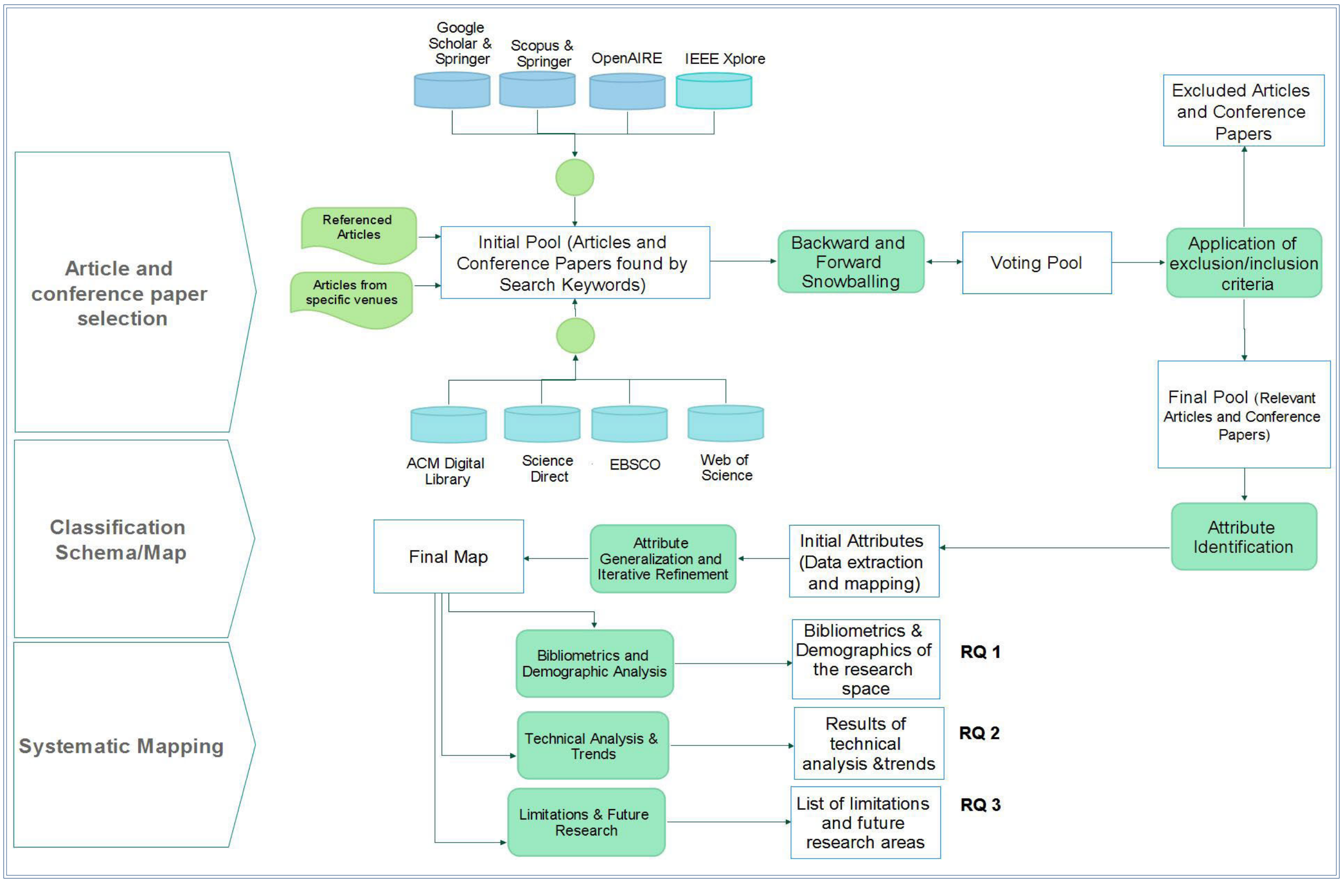

Google Scholar & Springer
Scopus & Springer
OpenAIRE
IEEE Xplore
Excluded Articles and Conference Papers
Referenced Articles
Articles from specific venues
Initial Pool (Articles and Conference Papers found by Search Keywords)
Backward and Forward Snowballing
Voting Pool
Application of exclusion/inclusion criteria
Article and conference paper selection
Final Pool (Relevant Articles and Conference Papers)
ACM Digital Library
Science Direct
EBSCO
Web of Science
Classification Schema/Map
Final Map
Attribute Generalization and Iterative Refinement
Initial Attributes (Data extraction and mapping)
Attribute Identification
Bibliometrics and Demographic Analysis
Bibliometrics & Demographics of the research space
RQ 1
Systematic Mapping
Technical Analysis & Trends
Results of technical analysis &trends
RQ 2
Limitations & Future Research
List of limitations and future research areas
RQ 3

FIGURE 1. **The overview of the research method.**

by specific search keywords and, then, proceeds with the filtering phase based on the inclusion/exclusion criteria. The second step of the process is the classification of the papers based on a set of criteria to answer the RQs. At the end of the data classification, data is baselined to generate synthesis for the SM study. In Fig.1, the utilized academic search engine databases (Google Scholar, Scopus, and OpenAIRE) appear in dark blue boxes. To the best knowledge of the authors, OpenAIRE has not been used before for software engineering-related SM studies.

Before starting the SM study, preliminary research was conducted to establish the infrastructure of the study to ensure that it will obtain healthy and verifiable results. For this purpose, several SM tools were investigated. After analyzing the SM tool comparison in [23], the version of the free web tool CADIMA (v2.1.3) is selected [24]. Paper elimination, including automatic duplicate papers elimination and no-full text item, is facilitated with the help of the CADIMA tool.

B. RESEARCH QUESTIONS

The research questions for SM study are classified into three groups, namely, (i) bibliometric and demographic analysis questions, (ii) technical evaluation and emerging trend questions and (iii) questions for future directions by analyzing limitations and gaps. Table 2 lists the RQs along with possible answers to each.

C. QUALITY ASSESSMENT CRITERIA

After the papers are selected based on the search keywords, quality assessment is applied for each paper. The quality assessment is realized based on the three questions to evaluate the completeness, consistency, and relevance of the selected study. The questions below are derived from the previous studies of Kitchenham *et al.* [34] and other SM studies.

Q1. Are the aims and scope of the study clearly stated?
Q2. Are all the study questions answered?
Q3. Are the data source, contexts, and conclusions described appropriately for future reference?

Each quality assessment question is answered based on the options proposed by Kitchenham *et al.* [34] as *Yes* $= 1$, *Somewhat* $= 0.5$ and *No* $= 0$. In this approach, each author assesses the study based on the quality questions by using the web-based CADIMA tool. The quality score for each study is computed by adding up all the scores of the answers to the questions. If there is a conflict between the score of authors, meetings and peer review sessions are held until reaching a joint-decision. The threshold for the acceptable quality rate is set as over 1.5 i.e., with a quality score greater

TABLE 1. List of related studies.

Published Year	Title of Article/Conference Paper/Thesis	# of Covered Studies	Covered Year	Scope / Differences
2012	SWOT Analysis of Software Quality Metrics for Global Software Development: A Systematic Literature Review Protocol [25]	N/A	Until June 2012	This study can be categorized as a preliminary study for SLR. This journal only defines research questions and inclusion/exclusion criteria and then searches the digital library to discover papers. There is no mapping, data analysis, and discussion data about the quality metrics for global software development.
2012	A Systematic Mapping Study On Dynamic Metrics And Software Quality [26]	60	1992-2011	This study mainly focuses on 'dynamic metrics' and 'run-time metrics/measurements' like coupling and cohesion. In our study, the covered year and search keywords are different. Our study doesn't only include the dynamic metrics, but also the other metrics that contribute to product quality.
2014	A Report on the Analysis of Metrics and Measures on Software Quality Factors – A Literature Study [27]	25	2007-2014	This study categorizes only object-oriented metrics based on the mix of ISO 9126 and earlier quality factors. In our study, ISO 25010, an updated version of ISO 9126 is used to categorize the metrics. Also, our study is not limited to the specific programming type and application domain to analyze the whole picture of the product metrics.
2015	The Visualization of Software Quality Metrics - A Systematic Literature Review [28]	18	2000-2014	The search string for the study was ("quality metric" AND "software" AND "visualization") between 2000 and 2014 [28]. The topic is searched using only three search engines. Our study focuses on the visualization methods of metrics based on the type of metrics. Therefore, the concept of this paper is totally different from our study.
2015	Predicting Software Product Quality: A Systematic Mapping Study [9]	69	1997-2013	This study focuses on the SM study on Software Product Quality estimation and prediction approaches between the years 1997 and 2013. This study analyzed papers related with software quality models and characteristics on the basis of SPQ approach. A more general approach is taken in our study to investigate the publications covering all aspects and models of SPQM. Also, the covered period of this paper is older.
2016	A preliminary mapping study of software metrics thresholds [29]	67	1970-2015	As seen from the title of the paper, it focuses on the threshold values of the metrics. This study is not completely an SM study and it is only a preliminary mapping. Only SCOPUS database is used. In our study, nine different digital databases are utilized for paper search. Additionally, in addition to SPQM, existence of a threshold value for software metrics is explored.
2016	Metrics and statistical techniques used to evaluate the internal quality of object-oriented software: A systematic mapping [18]	79	2004-2013	The search keywords of this study focus on the internal quality metrics for Object-Oriented software and only related to the ISO 25010 quality characteristics. The covered years and the scope of our paper are different than this paper; our study not only focuses on Object-Oriented software, it also deals with quality product metrics for various application domains and programming types.
2017	A Systematic Mapping Review of Software Quality Measurement: Research trends, model, and method [30]	42	2007-2017	This article aims to find the answer to these two research questions, (1) What is the most selected method for assessing software quality? and (2) What are the most selected quality models to measure software quality? It covers only 42 studies to determine the status by analyzing three search engines (IEEE, Scopus, and Science Direct). Our study seeks many RQs to analyze the status of SPQMs in-depth and in a multidirectional manner.
2017	A Systematic Mapping Study of Quality Assessment Models for Software Products [31]	31	2015	The search string for the study was ("software quality model" AND "metric" OR "measure") only for the year 2015 [31]. The main focus of the paper is the quality models studied for a single year. Our study covers 10 years.

TABLE 1. *(Continued.)* List of related studies.

2018	Coupling and Cohesion Metrics for Object-Oriented Software: A Systematic Mapping Study [32]	137	1991-2017	This study focuses on two specific types of metrics for only Object-Oriented Software. In our study, in addition to Object-Oriented software other types of software are included with metrics, such as complexity, size, and so on.
2019	Software Quality Models: A Systematic Mapping [19]	40	1976-2019	The search string for the study is related to software product quality models and maps the publication trend. The main focus of the paper is to analyze the coverage of the quality models with elements across the quality assessment phases [18].
2019	Software Quality Measurement in Software Engineering Project: A systematic literature review [33]	38	1984-2015	This article is identified as an article/conference paper only related to open-source development projects between the years 1984 and 2015. It covers 38 studies by analyzing compatibility with the ISO 9126 quality model. In our study, compatibility with ISO 25010 is analyzed.

than 1.5 (50% of the percentage score). Content analysis is realized by using Table 4 for different levels.

D. CONDUCTING SEARCH AND PAPER SELECTION

The population, intervention, comparison, and outcome (PICO) selection criteria [35] were used as follows to determine the search keywords. PICO method provides guidance for lots of research areas conducting SM. The CADIMA tool also supports the PICO criteria before the selection and elimination of the papers. By applying the PICO viewpoint, the search keywords were generated as shown in Table 3.

Table 4 shows the utilization of inclusion/exclusion criteria and application of the search level for each criteria to filter the papers in the pool.

The CADIMA tool has a feature that allows the implementation of inclusion/exclusion criteria based on the voting of the researchers. The voting scale for topic inclusion/exclusion criteria is between 0-3. "0" indicates the strong opinion for excluding the articles and conference papers, whereas "3" refers to the strong opinion for the inclusion of articles and conference papers. To increase the reliability of the voting mechanism and final results, the articles and conference papers were first voted by one author and then reviewed by the others. Based on the results of the joint voting, the excluded articles and conference papers were moved to the "Excluded" Excel spreadsheet in Google Docs system.

E. DATA EXTRACTION

Methods related to the synthesis of quantitative and qualitative data in software engineering, such as narrative synthesis, meta-analysis, cross-case analysis, thematic analysis, content analysis, case survey, and qualitative comparative analysis are categorized in [36].

While an SM study is conducted, statistical methods may also be used to analyze the present data. However, because of the heterogeneity of the primary studies in our SM repository, it was not possible to carry out a statistical analysis of the data. After assessing the applicability of the possible methods for this SM study, the most applicable synthesis method was determined to be the thematic analysis.

In this study, data synthesis is carried out through employing the thematic analysis method steps as outlined in the thematic synthesis checklist given in [37] and [38]. The detailed results and graphical representation of the thematic synthesis are explained in Section IV.

The overall process of selecting the relevant papers is shown in Fig.2. The related search strings along with the databases searched and the total number of search results can be found in Table 5. The total number of articles and conference papers, which were published in the designated period of this study is found to be 985. The backward and forward snowballing technique [39] was utilized to eliminate the risk of missing relevant papers. 54 additional papers are found as a result and shown in Fig. 2.

In total, our paper repository consists of 1039 papers. After elimination of non-English and duplicate papers, 612 articles and conference papers remain to screen. Unfortunately, 28 papers are written in Korean, Chinese, and Spanish, and only titles and abstract parts of these papers are written in English. Therefore, these papers are excluded from the repository.

After applying the inclusion and exclusion criteria on titles and abstracts, 612 articles and conference papers remain to be screened. These articles are categorized according to the PICO criteria given in Table 3, and the relevance level of the topic is judged from the title and abstract. After that, 263 papers remain. These articles are analyzed based on the inclusion and exclusion criteria in a full-text manner. After this elimination, the final paper pool size is 70 papers. In all, 70 papers amount to 719 pages in total. The final pool is published in an online repository on Google Drive [40].

A sample categorization of the article data screens on Google spreadsheet is given in the link [40]. Data extraction is completed using CADIMA and the online spreadsheet of Google.

TABLE 2. Research questions.

RQ Number	Attribute	Possible Types/Answers
RQ1.1	Which are the most cited papers between 2009 and 2019?	Citation Count Normalized Citation Count Normalized Citation Count of Paper vs Country
RQ1.2	Which are the top publishing venues?	The target of Paper
RQ2.1.	What type of research methods/facets are used in the papers?	{Analytical, Empirical, Others}
RQ2.2	What type of research approaches are used in the papers?	{AHP (Analytical Hierarchical Process), Novel, Fuzzy, Novel, GQM (Goal-Question-Metric), PSM (Practical Software Measurement), Others, N/A}
RQ2.3	What is the SPQM presented in the papers?	-
RQ2.3.1	Which metrics levels are commonly presented in the papers?	{Func-Req Level, Non-Func. Req Level, Class Level, Directory Level, File Level, Method Level, Variable Level, Design Level, Test Level, Others}
RQ2.3.2	Which metric levels can be applicable for which application domains?	{Embedded Software, Web Application, Mobile Application, Artificial Intelligence, Safety-Critical Application, Generic, Others, N/A}
RQ2.4	Which statistical model/method is used to validate or generate new metrics?	{Logistic Regression, Machine Learning, Meta-Analysis, Descriptive Analysis, Correlation Test, Genetic Algorithm, Swarm Optimization, N/A, Others}
RQ2.5	Which standards and process models have been used/referenced from papers to measure software product quality?	{ISO 15504, ISO\IEC\IEEE 12207, IEEE 1061:1998, CMMI, ISO/IEC 15939, Others (ITIL, COBIT, ISO 9001 etc.), N/A}
RQ2.6	Which quality models have been used in the papers to measure software quality?	{Mc Call, FURPS, ISO 9126, ISO 25010, Others, N/A}
RQ2.7	Which quality attributes of SPQ in ISO 25010 Quality Model are mostly measured in the papers?	{Functional Suitability, Performance Efficiency, Reliability, Usability, Compatibility, Security, Maintainability, Portability, Others}
RQ2.8	Which quality attributes of QinU in ISO 25010 Quality Model are mostly measured in the papers?	{Effectiveness, Efficiency, Satisfaction, Freedom from risk, Content Coverage, Others}
RQ2.9	Which quality attributes of DQ in ISO 25012 Quality Model are mostly measured in the papers?	{Accessibility, Currentness, Consistency, Credibility, Completeness, Accuracy, Traceability, Recoverability, Portability, Availability, Precision, Understandability, Efficiency, Confidentiality, Compliance, Others}
RQ 2.10	Which of the Software Development Life Cycle (SDLC) stages are mostly addressed in the data of the papers?	{Planning, Requirement, Design, Code, V&V, All, N/A}
RQ2.11	Which software programming types have been subject to SPQM in the articles?	{Object-Oriented Programming, Aspect-Oriented Programming, Scripting Programming, Basic Programming, Others, N/A}
RQ2.12	Are there any SPQMs related to SDLC model?	{Waterfall, Iterative, Agile, Product Line, Others, N/A}
RQ2.13	Is there a metric threshold value mentioned or not in the paper repository?	{Yes, No}

TABLE 2. ***(Continued.)*** **Research questions.**

RQ2.14	Is there any information about software product quality metrics/measurement management tools in our paper repository?	Metric Tool Support Data
RQ3	What are the limitations and future directions of current researches?	{Develop a New Tool, Improve The Tool, Develop a new Technique, Improve the Technique, Make More Detailed Research (New Case Studies), No Future Plan}

TABLE 3. **PICO selection criteria and search keywords.**

Key element	Definition	Criteria
Population **(P)**	What is the target population of the research?	Software, Product Quality
Intervention **(I)**	In which aspects do you wish to research to intervene in the population?	Metric(s)/Measure(ment), Product Quality, Metric(s)/Measure(ment)Tool
Comparator **(C)**	Is there any comparator/alternative to the intervention?	N/A
Outcome **(O)**	What are the outputs and settings of the intervention?	SM study for Software Product Quality Metrics
Keywords	Title and Abstract Field: ((“Software” AND “Product” AND “Quality”) AND (“Metric” OR “Metrics” OR “Measure” OR “Measurement”))	

IV. DATA SYNTHESIS AND RESULTS

The results of the SM study are presented below.

RQ1: Bibliometric and Demographics of the Publications

RQ1.1 Which are the most cited papers between 2009 and 2019?

The citation count of papers is taken from the Scopus database as of July 31, 2019. This database defines citation count as the total number of documents indexed that cite a document, group of documents, or researchers and exclude self-citations of all authors. The average normalized citation count (NCC) is 2.62, and there are 12 papers on average. We calculated *NCC per paper* for each country by using the formula below:

$$NCC\ per\ paper = \frac{Sum\ of\ (NCC)\ for\ all\ papers\ per\ country}{Number\ of\ papers\ for\ country} \quad (1)$$

The most cited papers are [S19], [S29], and [S66] with 245 citations (NCC = 30.63), 137 citations (NCC = 17.13), and 106 citations (NCC = 13.25) respectively.

When we analyze the number of papers with the normalized citation count, India is seen to be the leading country for quantitative criteria; whereas, the USA emerges as the leading country for quality criteria as shown in Fig.3. When we compared the number of papers among Germany, Brazil, and China, Germany and Brazil have the most paper in quantitative criteria, but China leads in the quality aspect.

RQ1.2 Which are the top publishing venues?

The top ten venues were determined through the calculation of the sum of the total citation count and the total number of papers, as presented in Fig.4. According to the inclusion/exclusion criteria in our paper pool, there are no papers presented in EUROSPI2, MetriKon, MetriSec and SAM. Between the years 2009 and 2019, there are no papers published in EUROSPI2, MetriKon, MetriSec, and SAM with keywords “software product quality metric /measure/ measurement” in the title or abstract parts. In these conferences or workshops, there are lots of metrics related to software, but there is no direct link between these metrics and the increase in software quality.

RQ2: SM questions related to technical issues and trends

RQ2.1 What type of research methods/facets are used in the papers?

Most papers generated by the empirical research method have been published in 2011 under the computer science field. Also, analytical papers have been mostly published in 2010 under this same category as presented in Fig.5.

The research method results of the selected papers in this study are categorized according to [41]. The results of this study show that 67% of the 42 papers were developed using the empirical research method. In our study, this ratio is found to be 74.3% (52 papers out of 70). An empirical research method is mostly utilized for the papers under consideration.

TABLE 4. Inclusion/exclusion criteria.

Criteria Definition	Criteria	Inclusion/Exclusion	Search Level/Scope
Publication Language	English	Inclusion	First Level/Title, Second Level/Abstract, Third Level/Full-text
Publication Language	Other than English	Exclusion	First Level/Title, Second Level/Abstract, Third Level/Full-text
Publication Type	Article and Conference Paper	Inclusion	First Level/Title
Publication Type	Books, Patent Documents and White Papers, Keynote and Tutorial Paper are out of scope.	Exclusion	First Level/Title
Relevant with Topic	Yes/No Selection	Inclusion/Exclusion	First Level/Title, Second Level/Abstract
Relevant to Research Questions	Yes/No Selection	Inclusion/Exclusion	First Level/Title, Second Level/Abstract
Accessible Electronically	Yes/No Selection	Inclusion/Exclusion	First Level/Title
Publication Date	01.01.2009-31.07.2019	Inclusion	First Level/Title
Publication has citations	For 2009-2018, If the citation count>0, include, For 2009-2018, if the citation count=0, exclude	Inclusion/Exclusion	First Level/Title
Publication has citations	For the year 2019, citation count 0 (zero) is acceptable because newly published papers cannot be cited yet.	Inclusion	First Level/Title
Scope of Covered Studies	Only product quality-related papers are included to analyze the articles/conference papers that directly emphasize the contribution of metrics to product quality.	Inclusion	First Level/Title, Second Level/Abstract, Third Level/Full-text
Scope of the Product	Software as a service (SaaS), Quality of Service (QoS), Quality of Experience (QoE) and Internet of Things (IoT) are out of scope.	Exclusion	First Level/Title, Second Level/Abstract

RQ2.2 What type of research approaches are used in the papers?

According to Fig.6, the most frequently used research approach is *Goal-Question-Metric (GQM)*, followed by the *Novel Approach* (this approach refers to a unique/new approach for known methods, approaches in qualitative research) [42]. 19 papers (evaluation or survey papers) do not mention their research methods in detail and consequently, these are not categorized. Finally, 6 papers were marked as *Others*, which include statistical approaches such as the Hidden Markov Model and systematic reviews.

One of the 19 papers marked as N/A in Fig. 6 uses the advanced version of GQM, with the name GQIM (Goal-Question-Indicator-Metric) instead. Besides, 3 papers use a combination of research approaches, namely, AHP (Analytical Hierarchical Process) with Novel Approach [43], AHP with Fuzzy Approach [44], GQM with PSM (practical software measurement) [45].

TABLE 5. Digital library search results.

#	Search string	Database Name	Results
1	(("Document Title": software metric(s)) OR ("Document Title": software measure(ment)) AND ("Document Title": product quality) OR ("Document Title": tool))	IEEE Explore	64
2	(("Document Title": software quality) OR ("Abstract": software product quality) OR ("Abstract": software) AND ("Abstract": metric(s)) AND ("Abstract": measure(ment)) AND ("Abstract": tool))	IEEE Explore	224
3	(("Document Title": software) AND "Document Title": measure) AND "Abstract": software product quality) OR ("Abstract": tool))	IEEE Explore	75
4	(TITLE (software AND metric(s)) OR TITLE (software AND measure(ment)) AND TITLE (quality) OR (tool)) AND DOCTYPE (ar OR cp) AND PUBYEAR>2008 AND LIMIT-TO (LANGUAGE , "English"))	SCOPUS &Springer, Google Scholar, Web of Science	90, 104,12
5	(TITLE (software AND metric) OR TITLE (software AND measurement) AND ABS (software AND quality AND product AND metric(s)) OR ABS (software AND quality OR Product AND measure(ment))) AND PUBYEAR>2008 AND ((LIMIT-TO (LANGUAGE, "English"))	SCOPUS, Springer	319
6	(+software +metric(s) +software +measure(ment)) AND record Abstract:(software product quality metric(s), software quality measure(ment), tool)	ACM	57
7	Title (+software + metric(s) + software + measure(ment)) AND Abstract:(software product quality metric(s), software quality measure(ment)) Title AND Abstract: (+software +quality+ metric(s)/measurement +tool))	OpenAIRE	68
8	TI Title and TI AB (software product quality and metric(s) or measurement or measure) TI Title and TI AB (quality tool and metric or measurement or measure) TI Title and TI AB:(quality metric tool or quality measurement tool)	EBSCO	16
9	Title AND Abstract:(software product quality metric(s), software quality measurement/measure) Title AND Abstract:(quality metric tool or quality measurement tool)	Science Direct	10

Although GQM is an old method, it is still the most popular approach in the SPQM area during the observed duration. Newly introduced methods are seen to be less utilized.

RQ2.3 What is the SPQM presented in the papers?

SPQMs data are analyzed in detail and documented for each 70 articles on the "Metric Details" spreadsheet of Online Repository on Google Drive [40]. Summary of this extracted data is presented under RQ 2.3.1 and RQ 2.3.2.

RQ2.3.1 Which metrics levels are commonly presented in the papers?

The summary of level based quality metrics is given below:

- McCabe's and Halstead's metrics represent quality at the method-level, while Chidamber and Kemerer's (CK) metrics suite represents quality at a class-level.
- Brito e Abreu's MOOD metrics and Bansiya-Davis's QMOOD metrics are the ones that have been most widely used in the papers. Michael Howard's measurement method has been used for the measurement of the security of Microsoft's Security Development Lifecycle.
- Sizing metrics are mostly related to fault proneness.
- Correlation between the modularity metrics and ANMCC (Average Number of Modified Components per Commit) in the Java projects has been proposed by [S3].
- The papers mostly mentioned the usage of coupling and cohesion metrics per measure for the maintainability, understandability, and reusability quality attributes, and inheritance and complexity-related metrics per measure of the flexibility, understandability, and reusability quality attributes.
- There are some special quality attributes for object-oriented software related metrics as given in [S61] and [S70].

$$\begin{aligned} &\mathit{Extendibility}\\ &\quad = 0.5 \times (\mathit{abstraction} - \mathit{coupling} + \mathit{inheritance}\\ &\qquad + \mathit{polymorphism}) \end{aligned} \tag{2}$$

$$\begin{aligned} &\mathit{Effectiveness}\\ &\quad = 0.2 \times \begin{pmatrix} \mathit{abstraction} + \mathit{encapsulation}\\ + \mathit{composition}\\ + \mathit{inheritance} + \mathit{polymorphism} \end{pmatrix} \end{aligned} \tag{3}$$

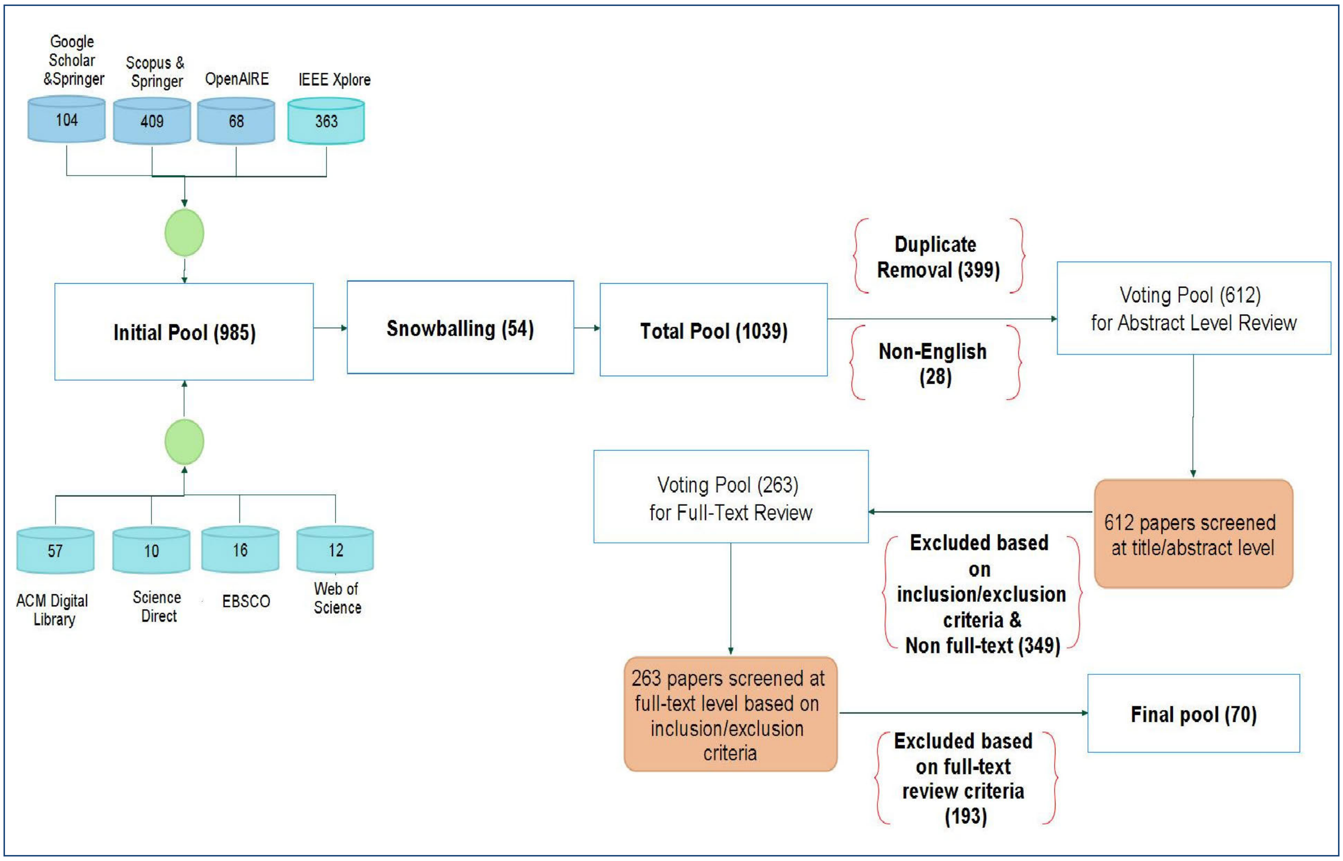

Google Scholar &Springer
104
Scopus & Springer
409
OpenAIRE
68
IEEE Xplore
363
Initial Pool (985)
Snowballing (54)
Total Pool (1039)
Duplicate Removal (399)
Non-English (28)
Voting Pool (612) for Abstract Level Review
612 papers screened at title/abstract level
Voting Pool (263) for Full-Text Review
Excluded based on inclusion/exclusion criteria & Non full-text (349)
263 papers screened at full-text level based on inclusion/exclusion criteria
Final pool (70)
Excluded based on full-text review criteria (193)
57
ACM Digital Library
10
Science Direct
16
EBSCO
12
Web of Science

FIGURE 2. Flowchart diagram of the search results.

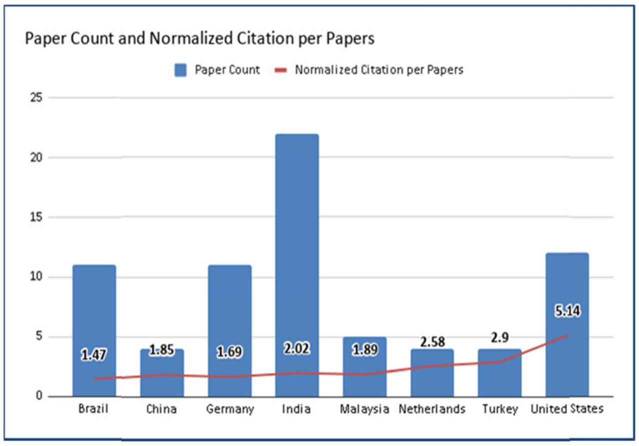


FIGURE 3. Normalized citation count per paper vs. country.

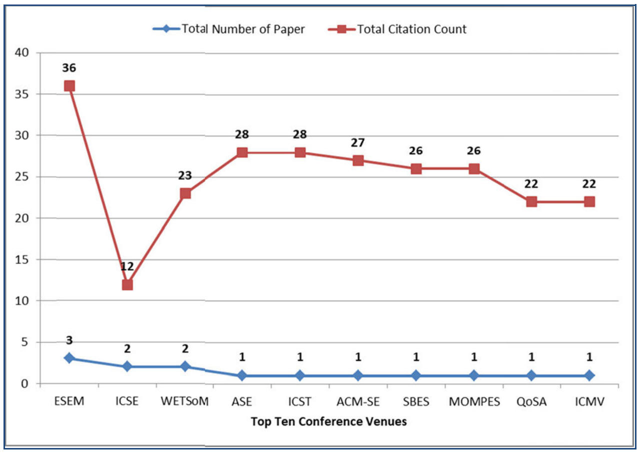


FIGURE 4. Top ten conference venues.

$$\begin{aligned} &\textit{Flexibility} \\ &\quad = 0.25 \times (\textit{encapsulation} - \textit{coupling} + \textit{composition} \\ &\qquad + \textit{polymorphism}) \qquad (4) \end{aligned}$$

$$\begin{aligned} &\textit{Understandability} \\ &\quad = 0.33 \times \begin{pmatrix} -\textit{abstraction} + \textit{encapsulation} \\ -\textit{coupling} + \textit{cohesion} - \textit{polymorphism} \\ -\textit{compexity} - \textit{designsize} \end{pmatrix} \qquad (5) \end{aligned}$$

$$\begin{aligned} &\textit{Functionality} \\ &\quad = (0.12 \times \textit{cohesion}) \\ &\qquad + \left(0.22 \times \begin{pmatrix} \textit{polymorphism} + \textit{messaging} \\ + \textit{design size} + \textit{hierarchies} \end{pmatrix}\right) \qquad (6) \end{aligned}$$

- The other Design Level metrics are interface density, easy learning (average time for learning/master the usage for the component), clearness of error messages, the number of configurable metrics for the interface,

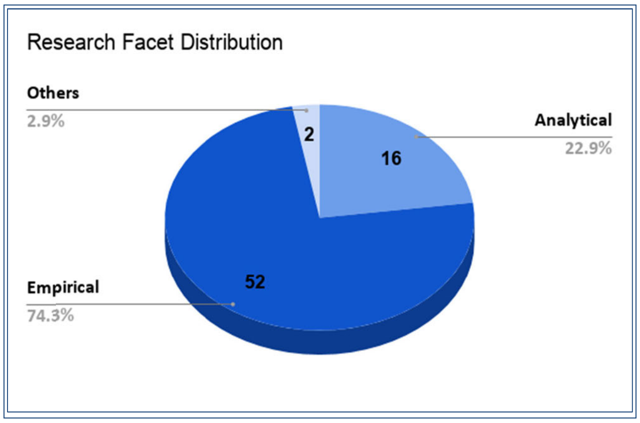


FIGURE 5. Research facets distribution.

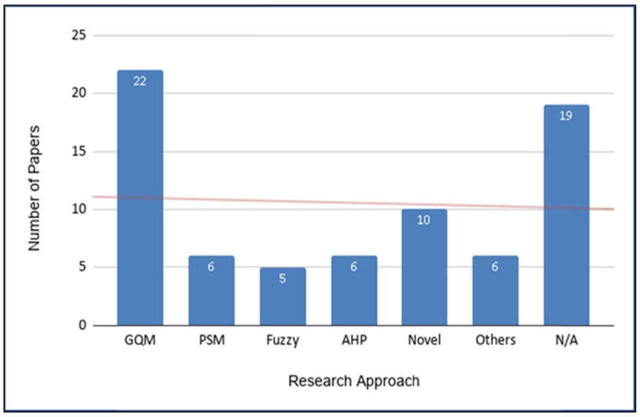


FIGURE 6. The research approach distribution of papers.

IPCI (Index of Package Changing Impact), Gang of Four (GoF) design patterns related metrics, IIPE (Index of Inter-Package Extending), IIPU (Index of Inter-Package Usage), IIPUD (Index of Inter-Package Usage Diversion), IIPED (Index of Inter-Package Extending Diversion), APMH (Average number of alternate paths in multiple hierarchies).

- The mostly used Project Management Level metrics are effort variance, productivity, COCOMO, Schedule Variance, Schedule Performance Index (SPI), Earned Value Analysis (EVA), Cost Performance Index (CPI), Return on Security Investment (ROSI), Cost of Quality, average effort spend for maintainability and Energy Consumption.
- The mostly used requirement level metrics are RSCR (Requirement Specification Change Request), RV (Requirements Volatility), traceability metrics to analyze the consistency between business requirements and product requirements.
- The mostly used Test Level Metrics are a number of test cases, the number of bugs in unit tests, integration tests, and regression tests. Other metrics are like branch and line coverage ratio, test density per class, test-growth-ratio relation to source code and NPATH (Number of execution path through functions), DRE (Defect Removal Efficiency), DDD (Delivered Defect Density), IDD (Internal Defect Density), RE (Review Efficiency).
- The mostly used system level, which consists of one more software, product metrics are as follows: Customer-Found Defects and Regressions, Customer Perception Calculate, MTTF (Mean Time to Failure), MTTR (Mean Time to Repair), MTBF (Mean Time Between Failures), ROCOF (Rate of Occurrences of Failure) and POFOD (Probability of Failure on Demands).

RQ2.3.2 Which metric levels can be applicable for which application domains?

10 out of the 70 papers mention the quality measurement of the web application, 4 papers per safety-critical applications, 3 papers per mobile application, 2 papers for embedded software applications, and finally, 1 paper is related to new technology and artificial intelligence.

In our paper repository, the common metrics used for Web application domains were observed to be Comments to Code Ratio, Number of Data Base Connections, WMC (Weighted Methods per Class) and LCOM [S20]. MOOD and QMOOD metrics are applicable to the Object-Oriented and Aspect-Oriented approaches.

The following metrics were suggested in the papers to measure the application domains for directory and class levels:

- The directory-level metrics: total LOC, physical LOC, Number of Statements, Number of Blank Lines, Number/Percentage of Comment Lines, Non-comment Non-blank (NCNB), and Executable Statements.
- The class-level metrics are mostly given as WMC, RFC, LCOM, CBO (Coupling between Objects), DIT, Dynamic CBO, MC (Method/Message Complexity), CWC (Coupling Weight for a Class), AC (Attribute Complexity), CLC (Class Complexity), AMCC (Average Method Complexity per Class), ACC (Average Class Complexity), ACF (Average Coupling Factor) and AAC (Average Attributes Per Class).
- McCabe's and Halstead's metrics represent quality at a method-level, while CK metrics represent quality at the class-level.
- Generic Source Code Metrics were suggested as LOC, NOM (Number of Methods), NOF (Number of Fields), CC (Cyclomatic Complexity), LCOM, Number Lines of Comment, Percentage Comment, CA (Afferent Coupling), CE (Efferent Coupling), and ABC (Association Between Class) [S60].
- In [S20], the following metrics were proposed as suitable for Object-Oriented and Aspect-Oriented Programs: Maintainability Index (MI), DIT, CBO, and LOC.
- In [S21], the following metrics are suggested as Class Level, Object-Oriented metrics: CA, CE, DIT, LCOM, MLOC, NBD (Nested Block Depth), NOC, NOF, NOM, NORM, NSC (Number of Children), NSF, NSM-PAR

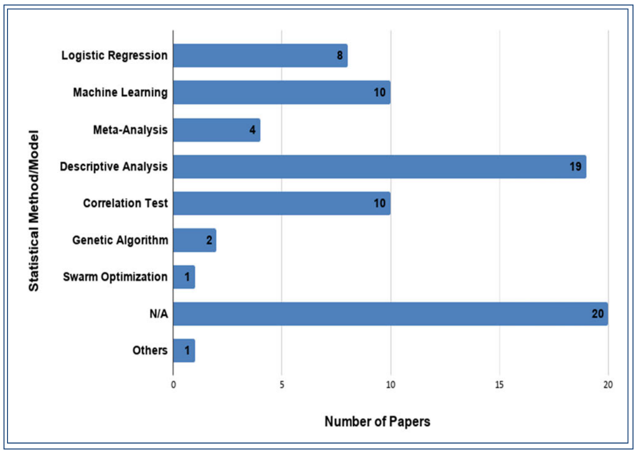


FIGURE 7. The number of papers for statistical method/model.

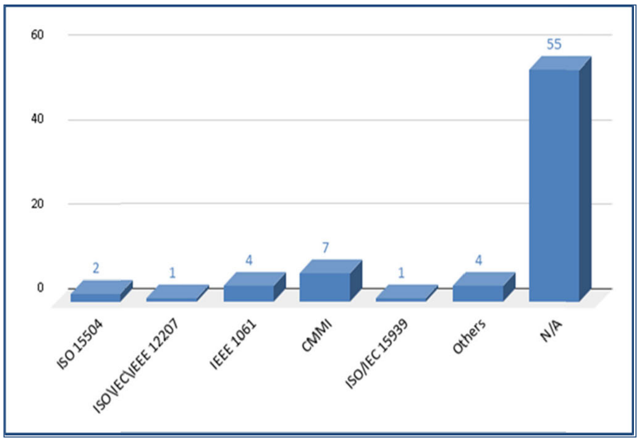


FIGURE 8. Distribution of papers content related to standards, model.

(Number of Parameters), RMD, WMC (Weighted Methods per Class), and McCabe Complexity.

- In [S30] mentioned the DIT and LCOM metrics, which are unsuitable for Object-Oriented design to measure quality and complexity. NOC cannot be used to predict fault proneness in all circumstances.
- In [S67] mentioned that the low CBO values show that most of the classes refer to few other classes. Low values of DIT and NOC strongly suggest that reuse through inheritance may not be fully adopted in the design of class libraries.
- As shown in Table 6, maintainability-related metrics, presented as a common quality characteristic, was used to measure the quality of all types of applications, except for artificial intelligence. Furthermore, the security metric was another common quality characteristic for all application domains except for embedded applications. There is a need for choosing the best combination of metrics for each different application domain for the given application context. Moreover, the selection of metric parameters can be automated by using prediction functions and different techniques [S28].

RQ2.4 Which statistical model/method is used to validate or generate new metrics?

As shown in Fig. 7, a genetic algorithm is used by only two papers, [S28] in 2012 and [S58] in 2013. Also, swarm optimization is used in only [S26] in 2014. Only one paper [S44] used a statistical extrapolation method to forecast future data relying on historical data. The most popular statistical method utilized is descriptive analysis. Machine learning methods show better results compared to statistical methods for predicting the relationship between Object-Oriented metrics and change proneness.

Between 2014 and 2015, in the future plan of the two papers [S25] and [S38], the authors mention that they wanted to replicate their study with a larger data set by using the less explored learning techniques, such as genetic algorithms and colony optimization, to obtain more adequate results. However, when our search keywords are applied, in this respect, no answer emerges. This means that there has been no example of the usage of genetic algorithms and swarm optimization in the last five years. Another SM study may be carried out by entering keywords that directly include the name of the statistical method/algorithm and the type of SPQM to analyze the trend of using these algorithms. As mentioned [S33], there is a need for finding the most appropriate statistical method/algorithm to come up with fault predictions in early project phases by matching them with quality attributes.

RQ2.5 Which standards and process models have been used/referenced from papers to measure software product quality?

74% of the papers, while proposing the product quality metrics, do not reference any standards or models. On the other hand, the most popular model seems to be *CMMI*. In Fig. 8, 4 papers marked as *Others* mention ITIL (Information Technology Infrastructure Library), ISO/IEC 14598, ISO/IEC 14764, and Software Component Maturity Model (SCCM). [S55] generated the Component Quality Characteristic (CQC) Model by analyzing and merging the quality requirement components with the traditional quality models. This model consists of six quality characteristics and 23 sub-quality characteristics including reliability, portability, maintainability, usability, functionality, and efficiency.

None of the papers mentions ISO 9001, COBIT (Control Objectives for Information and Related Technology), and ISO/IEC 33001 during their measurement and analysis process.

RQ2.6 Which quality models have been used in the papers to measure software quality?

The most frequently used quality model remains ISO 9126 [14], although the ISO 25010 standard was published in 2011. Only two papers are referring to the Mc Call model and FURPS. *Others* are marked as a Design Quality Model (DQM), Quality Model for Object-Oriented Design (QMOOD), Pragmatic Software Quality Model.

Questions RQ2.7-RQ2.9 below aims at understanding the level of contribution of the quality attributes (SPQ, DQ, and QinU) of ISO 25010 Quality Model) attained by the papers under investigation. A list of all papers about the quality

TABLE 6. Paper references for application domains.

Quality Characteristics	Application Domain	Citation Count	References for Papers
Maintainability, Performance Efficiency, Reliability	Embedded Software	2	[S57], [S65]
Effectiveness, Efficiency, Performance Efficiency, Freedom from Risk, Functional Suitability, Maintainability, Usability, Reliability, Security	Web Application	10	[S8], [S15], [S20], [S50], [S53], [S54], [S56], [S63], [S65], [S66]
Functional Suitability, Maintainability, Usability, Performance Efficiency Security	Mobile Application	3	[S36], [S50], [S53]
Security, Usability Performance Efficiency, Functional Suitability	Artificial Intelligence	1	[S50]
Context Coverage, Consistency, Efficiency, Effectiveness, Freedom from Risk, Maintainability, Usability, Performance Efficiency, Portability, Precision, Reliability, Satisfaction, Security	Safety Critical Application	4	[S12], [S16], [S22], [S44]
Maintainability, Reliability, Security	Others	4	Database [S7], Procedure Oriented Type Enforcing Language used in Telecommunications [S29], GUI-Driven Applications [S40], Real-Time System Applications [S62]

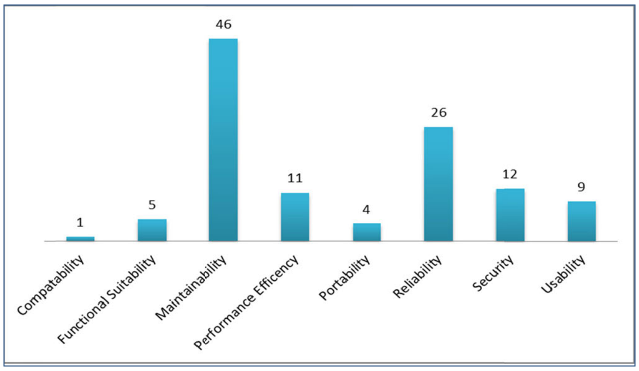


FIGURE 9. The distribution of quality characteristics for the SPQ model.

characteristics is given in the Online Repository on Google Drive [40].

RQ2.7 Which quality attributes of SPQ in ISO 25010 Quality Model are mostly measured in the papers?

As presented in Fig. 9, most of the SPQM are related to *Maintainability* and *Reliability* quality attributes. Mostly used quality attributes are the same since 2015 when we compared our results with those of the article [18]. Traceability between numbers of papers based on each quality attribute can be reached from the Online Repository on Google Drive [40].

RQ2.8 Which quality attributes of QinU in ISO 25010 Quality Model are mostly measured in the papers?

As presented in Fig. 10, the maximum number of metrics that can be observed in QinU is *Freedom from Risk, Effectiveness* and *Satisfaction.*

RQ2.9 Which quality attributes of DQ in ISO 25012 [16] Quality Model are mostly measured in the papers?

As shown in Fig. 11, there is no paper related to the usage of other data quality models such as *Credibility, Correctness, Accessibility, Compliance, Confidentiality, Efficiency, Availability, Portability,* and *Recoverability*. The most often

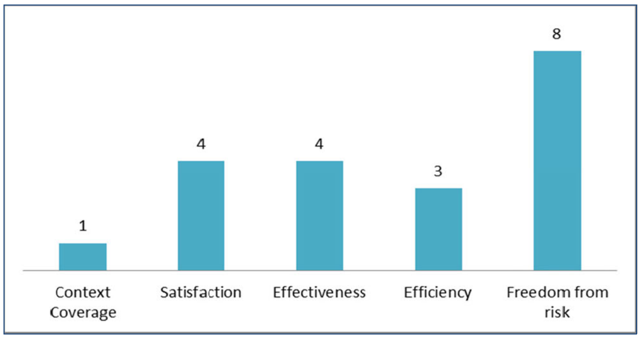


FIGURE 10. The distribution of quality characteristics for the QinU model.

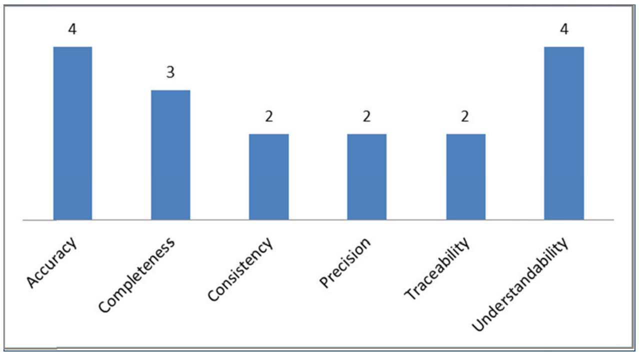


FIGURE 11. The distribution of quality characteristics for DQ model.

used data quality characteristics are *Understandability* and *Accuracy*.

The highlights of the measurement of the quality attributes/sub-attributes mentioned in the papers are listed below:

1. The coupling and cohesion metrics are mostly used in the measurement of the maintainability, understandability, and reusability quality attributes.
2. The inheritance and complexity metrics are mostly used to measure the Flexibility, Fault Proneness, Understandability, and Reusability quality attributes.
3. [S24] presents a WMC, LOC, and RFC metric combination as applicable to many various types of software projects for predicting the change-prone classes to reduce the cost of testing.
4. [S20] proves that there is a strong correlation between the NODBC, CCR metrics, and maintainability for data-intensive applications.
5. [S30] asserts that the five metrics of MOOD suite are appropriate for quality measurement and that Coupling Factor is not suitable for quality measurement.
6. The maintainability metric of the product can be measured as mentioned in [S32] by giving different weights to each of the quality attributes for analyzability, changeability, stability, and testability.
7. In the paper pool of this study, four new metrics were offered as listed in Table 7.

TABLE 7. Lists of new metrics.

Quality Characteristics	Definition of Metrics
Satisfaction, Usability, Performance Efficiency, Freedom from Risk, Security, Reliability	Desirability Metric for Requirement: By modifying the parameters of the desirability functions, the quality and priority of requirements can be evaluated as necessary for different software domains [S12].
Maintainability	The maintainability index (MI) Metric to predict maintainability for data-intensive and Object-Oriented Software: It correlates the CCR (Comments to Code Ratio) and NODBC (# of Data Base Connections) [S20].
Maintainability	The new metric for Predicting Software Change (CI): Metric is derived by using CK metrics and Gene Expression Algorithm [S25]: CI = ((((0.24 * RFC) + (2 * SLOC)) – WMC) – 59.62) – LCOM
Security	The new security metric parameters (based on representative weaknesses of the system): the percentage of the security weaknesses that occurred, frequency of occurrences, and risk degree [S54].

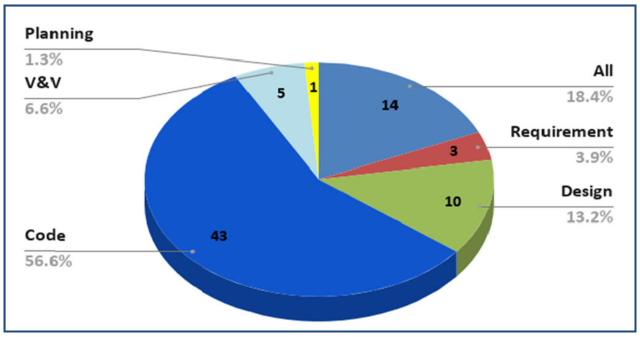


FIGURE 12. The distribution of SDLC phase for papers.

RQ2.10 Which of the Software Development Life Cycle (SDLC) stages are mostly addressed in the data of the papers?

As shown in Fig. 12, the most of the SPQMs focus on the code and design phase to predict and avoid bugs in a proactive manner. Moreover, 6.6% of the papers are related to the test phase, aiming to eliminate bugs of software before release. To become more proactive, more metrics for the requirement phase can be considered.

RQ2.11 Which software programming types have been subject to SPQM in the articles?

The SDLC phase of 43 out of 70 papers was revealed to be at the code level. As shown in Fig. 13, 35 of the 43 papers have been subject to increasing the quality of Object-Oriented programming. Three papers are related to Aspect-Oriented programming metrics and also three of them are related to *Basic Programming* and one is related to *Script*

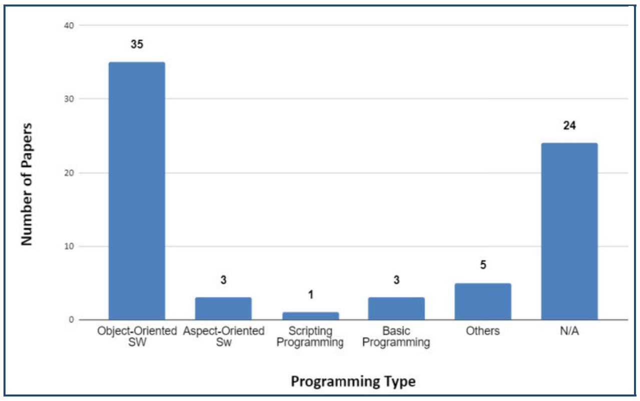


FIGURE 13. The number of papers for programming types.

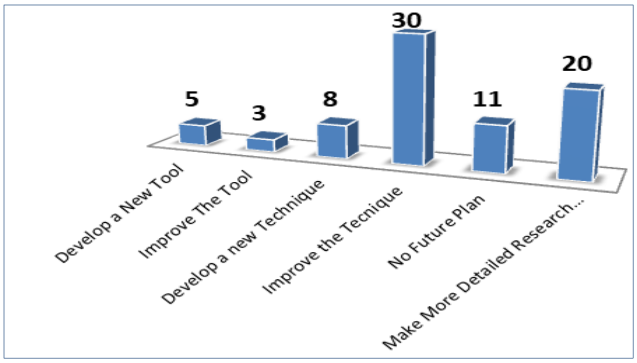


FIGURE 14. Future plan offered by the papers.

Programming for the code phase. Finally, five of them are marked as *Others* which include Component-Based Programming or Language-Independent programs.

The conclusions of [S20] are usable only for Object-Oriented, and medium-size systems. There is a need for metrics for Aspect-Oriented, Service-Oriented, and Component-Based application development to predict the maintainability.

RQ2.12 Are there any SPQMs related to SDLC model?

Only [S9] and [S63] mention the relationship between the metrics and the Agile model. The SDLC phase and steps were not used as a parameter of metric in most of the papers. These articles have used generic development metrics instead, but these have been re-evaluated according to the activities of the agile methods. There exist a few studies that consider quality in agile methods. In [S9] and [S36], generic development metrics have been used, but they have been re-evaluated by using the 3C (Continuous Integration, Continuous Measurement, Continuous Improvement) model for activities of the agile model.

RQ 2.13 Is there a metric threshold value mentioned or not in the paper repository?

Adding the threshold to the metric data has tremendous importance to control deviation from the goal of the product quality. If the metric data exceed the threshold, the system will immediately issue a warning to implement an emergency plan. Therefore, threshold value-related data has been searched in this study.

Twenty-two of the 70 papers mention the threshold values only concerning their study or in a general manner, often in brief comments only. The proposed threshold values in the literature are mostly based on the experiences and coding standards [S52]. Therefore, these threshold values might appear to be lower or higher, depending on the structure and scope of the source code.

To generate relative thresholds for different types of source code metrics, RTTOOL (relative threshold tool) is implemented by the authors of [S52]. The tool has been validated by using the Qualitas Corpus dataset, which is an open-source software system collection for empirical studies related to code products. This tool is publically available at http://aserg.labsoft.dcc.ufmg.br/rttool/.

The calculation method of RTTOOL for extracting the relative threshold value and penalty rate when it exceeds the threshold is given in [S52]. [S40] proposes that their study can be a starting point for determining threshold values for design metrics to improve design practices.

RQ2.14 Is there any information about software product quality metrics/measurement management tools in our paper repository?

Although there is a lack of integrated quality metric/measurement management tools, which collect and analyze the metrics to measure different quality characteristics, various tools are available to measure different types of quality metrics as given in the following: ASSIST [46], Analizo [47], RAF [48], RTTool [49], iPlasma Tool [50], MUSE (Muse Understand Scripting Engine) [51], VizzMaintenance [52] and EMERALD (Enhanced Measurement for Early Risk Assessment of Latent Defects) [53]. When we analyzed the name of the software quality datasets and metric tools on the Internet, we encountered the QSM (Quantitative Software Management) tool [54] that supports full measurement processes from beginning to the end for large projects. The tool includes 10, 000 projects from the industry, with validated data for use by users with a license or as a demo version. Unfortunately, there is no paper in our pool that uses QSM.

RQ3: What are the limitations and future directions of current researches?

Most of the papers plan the next step and give suggestions, which relate them to ''Improve the Technique'' (39%) and ''Make More Detailed Research'' (26%) by using new case studies, as shown in Fig. 14. In this sense, 39% of the papers in the pool mentioned that techniques could be improved through the implementation of different languages and applications to the different phases of the projects.

Based on this, future directions and limitations can be categorized into three groups as follows.

1) STANDARDS/MODELS

A majority of the papers mention low understandability levels and a lack of examples of quality models like ISO 25010 and ISO 9126. To increase the usability of the quality models to generate more quality metrics, additional documents can be prepared to address clarities in different research areas. Practically, the implementation of quality attributes and integration of multiple quality models can be the new area for researchers [S39].

Also, there is a difference between the metric ontology and quality models/standards. This prevents the common usage of the standards in industry and papers. The difference in ontology is one of the biggest obstacles for continuous improvement of standards. Therefore, conferences and workshops might be productive grounds where the discrepancy observed between the standards and academic articles can be discussed.

The authors of [S55] propose new quality characteristics to measure the marketability quality characteristics with sub-characteristics: Development Time, Cost, and Time to Market, Targeted Market, and Affordability. Also, the authors of [S37], propose new quality attributes to measure the attribute of developer/tester by using parameters such as culture, behavior, and point of view. The usage and contribution of these new quality characteristics can be analyzed and included in the new version of the ISO 25010.

With the help of machine learning and AI techniques, the software development processes and software quality model can be modelled to guide the developers/testers. With this new intelligent model, new attributes associated with quality will be automatically included in the system. Besides, this model can also guide developers/testers about activities and with the help of this model, the error-rate of the developers could be reduced [S4].

2) METRICS/MEASUREMENTS/TECHNIQUES

There is a need for choosing the best combination of metrics for each different application domain for the given application context. Moreover, the selection of metric parameters can be automated with the tool by using prediction functions and combining different techniques [S28].

Further research is required to detect the defects of the COSMIC (Common Software Measurement International Consortium) functional size metrics for various application domains by applying broader measurements belonging to different application domains [S65]. There is a need for more complicated metrics with more diversified parameters, such as the complexity of attributes with software cost, effort, and time factors.

There is a need for the new product quality metrics for new technology trends such as AI, IoT, CoT, Machine Learning, and Robotics [S50].

Although there are many security-related metrics in the literature, we cannot say that the entire system is secured by analyzing the metric for only one security attack type. Therefore, there is a need for more reliable security metrics/measurements to quantify different aspects of security (unintended threats) for specific goals with different combinations [S19]. There is a need to ensure the minimization of security vulnerabilities for technology products [55]. Therefore, it is estimated that the need for security-related product quality metrics may increase.

CMMI v2.0 [4] and PMBOK [7] emphasize the adoption of the processes following the agile approach. Unfortunately, only two papers propose metrics related to Agile. Moreover, an SM study by [56] emphasizes that there is a limited study that mentions the contribution of DevOps (Development and Operations), which is a method for increasing the agility of development and operations for fast delivery to customers. Therefore, more metrics are required for measuring the quality of agile projects.

Due to the lack of knowledge about software metric thresholds, the performance of software engineering metrics is affected in a negative manner [57]. Unfortunately, little threshold information is detected in our paper repository. Therefore, there is a need for producing new threshold values according to the different application domains, programming languages, size of the projects, quality characteristics, and SDLC phase, methods.

EFQM defined Benchmarking, as a systematic comparison of approaches with other relevant organizations, offers insights that will help the organizations to improve performance [58]. Therefore, the usage of similar metric formula needs to be encouraged among software companies working in the same sector. In this way, they can use the opportunity to improve their processes.

3) TOOLS

Calculation of metrics at different tool platforms and then bringing them together to calculate another metric is a hard and time-consuming task. There are many product quality metric tools to measure different types of quality metrics. Instead of using lots of tools for measurement, using one tool to measure multi-purpose metrics would increase the reliability, traceability, and completeness of the metric data. As a result, there is a need for an integrated and multi-dimensional measurement tool that would automatically calculate the different levels of metrics and generate relative thresholds for them by using third-party statistical and machine learning tools [S66]. In this way, instead of spending time on metric calculations, the time can be used more efficiently for locating the root cause and offering solutions for the metrics which exceed the threshold values. Thus, researchers can have access to metrics at any time they wish to assess the current status of the project [59].

There is also a need for a tool to measure and test the correctness of the functional size measurement for different types like IFPUG, UniFSM, EFES (Effort Estimation Methodology), and so on. Additionally, the E-Quality tool can be extended for object-oriented languages, like C++, C#, and PHP. Also, the tool can be extended to visualize the design phase by extracting design metrics and relations from UML diagrams [S34].

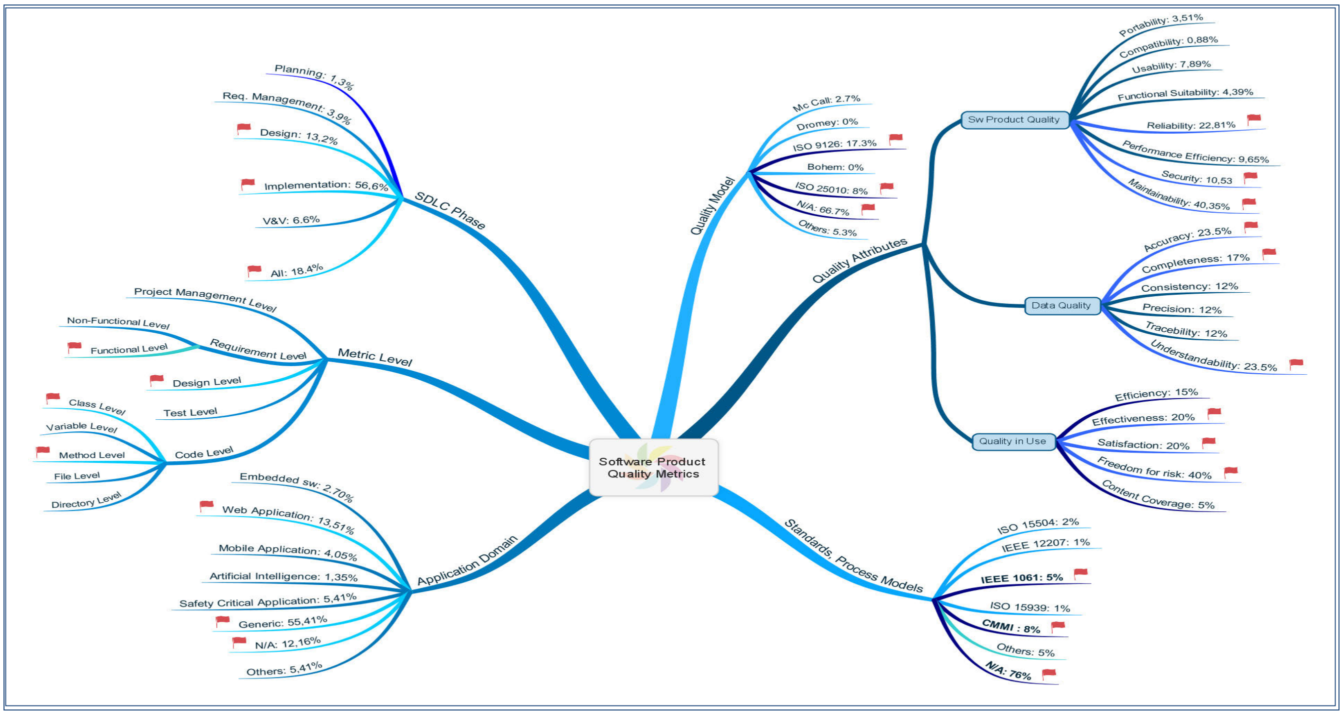


FIGURE 15. Mind map of software product quality metrics trend.

V. DISCUSSIONS

The SM results of SPQM-related articles and conference papers for the 2009 and 2019 years are presented with the Mind Mapping technique [60], which allows us to see the big picture by visualizing and accelerating the process of gap analysis. With the help of this technique, we have a chance to analyze the current situation and improvement opportunities related to SPQM. The mind map shown in Fig.15, was drawn by using the iMindMap v10.1.1. By analyzing this figure, researchers will get an opportunity to obtain some general information about the trends of the SPQM in the literature. Trends for each categorization are shown through percentage values. Besides, the highest percentage in each category is marked with a flag and a different color. The percentages are calculated with the following formula:

$$\begin{aligned} &\mathit{Distribution\ of\ Categories\ per\ RQs}(x\%) \\ &\quad = \left(\frac{\mathit{Sum\ of\ Category}(x)}{\mathit{Sum\ of\ Category}(x, y, z)}\right) \times 100 \qquad (7) \end{aligned}$$

For example, Distribution of Maintainability Characteristics per RQ 2.7 = (Sum of maintainability metrics related paper (46)) / (Sum of other categories (1 + 5 + 46 + 11 + 4 + 26 + 12 + 9)) × 100, (46/114) × 100 = 40.35 %.

The discussions and remarks about the SPQM trends for the last ten years are listed as follows based on Fig. 15.

As we compared our study results with those of [41] the result for the selection of a quality model for use appears to be the same. The usage of ISO 9126 (an earlier version of ISO 25010) is still more popular than ISO 25010 between the years 2006 and 2019. Therefore, it can be said that a majority of the articles and conference papers published within this duration has not realized a transition to the new standard yet, as shown in Fig. 15. Although the ISO 25010 quality model was released in 2011, the ISO 9126 is still more popular. To increase the usage of the new version of the quality model, additional guidelines can be prepared to give more explanations and examples to clarify certain issues.

The most popular quality characteristics to measure Software Product Quality are Maintainability, Reliability, and Security. Also, the most popular quality characteristics to measure the QinU are Freedom from Risk, Effectiveness, and Satisfaction. Lastly, the most popular quality characteristics to measure DQ are Understandability, Accuracy, and Completeness. Whereas Understandability, Reliability, and Reusability quality factors were top in the study that examined the articles and conference papers written in 2007 and 2014 [27], it was Maintainability, Reliability, and Security in the present study. It is still observed that reliability is an important quality factor for the measurement of product quality. There is a need for metrics to measure the data quality in the following aspects of quality characteristics: Credibility, Correctness, Accessibility, Compliance Confidentiality, Efficiency, Availability, Portability, and Recoverability. There is no quality attribute within the standards or quality models for the metrics which are related to the culture, attribute, attitude, and point of view of the developers, testers, and managers to quality attributes, standards, and process models [S37].

In our paper pool, while presenting the quality metrics, the most referenced standard was IEEE 1061:1998 and the referenced process model was CMMI. Countries that have the highest number of CMMI certificates made a greater contribution to SPQM. There is a direct link between the rate of the usage of the quality standards and process models and those of SPQMs-related articles and conference papers. Encouraging the usage of these standards and models will contribute to the literature on SPQM. The textbook "Software Engineering: Modern Approaches" [3] and CMMI [4] have proposed a set of metrics as listed here: a measure of size, effort, variance in cost, number of defects by severity, percentage of the missing and defective requirements found per hour in inspection, rework percentage before and after delivery, customer/developer satisfaction index, productivity, reliability (Mean Time to Failure), maintainability (software down/error time), percentage of system vulnerabilities, test coverage rate, CC, fault density, number of entries and exits per module and the percentage of comment lines. There is a consistency between the metric sets given in the CMMI model and books with the metrics suggested in this paper repository.

Although GQM was introduced to the software industry in the 1990s, it is still the most popular approach in the SPQM area during and after 2009. Still, transition rates to new approaches remain low.

As shown in Fig. 15, there are a few metrics related to non-functional requirements, which can affect the performance efficiency and quality of the systems negatively. For example, non-functional requirements related to security can affect critical software at medical centers. Therefore, there is a need for more quality metrics as non-functional requirements to predict and eliminate the bugs in a proactive manner. Also, it is essential to be able to extract coupling and cohesion metrics from UML representations to predict bugs in the previous phases of development.

A majority of the metrics are related to the quality of Object-Oriented software for medium-sized systems. There is a need for new metrics for other types of programming languages and large and small-sized systems. Further research is needed to calculate the metrics for service-oriented software development and component-based application development. These metrics should be tailored by with the company size.

The majority of the papers have validated their proposed metrics by using medium/large data. To do so, more open-source code data is needed because, after a while, the open-source codes at hand may be insufficient to prevent diversity. The NASA MDP datasets were collected from many different projects developed by many developers. However, there is no information about the number of developers and their experiences. As such we cannot measure defect prediction by using the personal data of developers. Therefore, there is a need for more open-source and publically accessible project databases to test the validity of new or existing metric set combinations. Moreover, there is also a need for the developer/tester- related data to measure their effectiveness on metrics such as the number of developers, culture, attributes, points of view to the processes and standards, and so on. Through enhancing the collaboration between the computer engineering/science, mathematics and statistics departments of universities, the use of statistical models/methods can be increased. These methods/models can be used to propose new metrics, validate the results of the metrics and also predict the bugs in the next phase of the projects.

VI. THREATS TO VALIDITY

The threats-to-validity part of quality assessment is realized for this SM study by inspecting the title, keywords, abstract and, full-text of the articles and conference papers to analyze the relevancy level of papers with SPQM. If sufficient information was not available in these sources, the table of contents and some parts of the articles were explored in depth.

A. INTERNAL VALIDITY

To prevent the loss of any data about SPQM, alternative keywords and different combinations were searched from search engines. The inclusion and exclusion criteria were applied to the final pool of papers. To remove subjectivity in the selection of the papers, the authors ranked the papers, and then the results were compared. If the authors' scores were different, the issue would be resolved and a unanimous decision is taken. The voting scale for the topic inclusion/exclusion criteria is between 0-3. "0" indicates a strong opinion for excluding the articles and conference papers, whereas "3" refers to the strong opinion for the inclusion of articles and conference papers. Based on the results of joint voting, the excluded articles and conference papers were moved to an "excluded" spreadsheet in the Google Docs system. If this SM study was repeated, the selected set of primary papers might have deviated in a small amount, but we believe that the big picture of the result given in this paper would not change much.

B. CONSTRUCT VALIDITY

After the final paper pool was constructed, the data was systematically mapped following the research questions by using the GQM approach. GQM approach helps to minimize risks of construct validity by providing traceability between goal and questions. Questions are answered based on the categorization schema. This categorization schema is determined for each RQ based on the knowledge in literature about SQM and finalized after several iterative improvement processes. Additionally, with the help of peer review in a crosscheck manner, meetings and evidence-based data production, the reliability and validity of this study took its final form.

C. CONCLUSION VALIDITY

As presented in Section IV, graphs, trends, and analyses have been generated directly from the raw and synthesized data to ensure the validity of the conclusion. By using this raw data, this SM study can be replicated by following the same steps defined in this study. As stated before, the selected set of primary papers might have deviated in small amounts, but the authors believe that the bigger picture related to the results here do not change significantly.

D. EXTERNAL VALIDITY

There is an external limitation that this study included, limiting the number of papers available in the selected digital library databases to English only; merely the titles and abstracts of some papers are written in English and the rest are not in English. As a result, even though these papers were highly related to our topic, they could not be included in the SM repository. These papers, which contain information about the evaluation of existing metrics or the introduction of new metrics, couldn't be read and studied widely due to language barriers. The results of this study are discussed and compared with books, other papers, and quality standards/models. It was found that the results of our study are sufficient to reflect the know-how in the literature about SPQMs reported by researchers and professionals.

Additionally, this study serves as a starting point for researchers in the SPQM area, SQA professionals, and software process and quality divisions in a company, implying that there is no intention to generalize these results.

VII. CONCLUSION, LIMITATIONS, AND FUTURE WORK

This study will hopefully function as a starting point for other studies related to SPQM. Additionally, the results of this survey can be used by the industry by applying the most-often-used product quality metrics to measure quality attributes and using threshold values to compare and analyze the results. By comparing their results by thresholds, corrective and preventive actions can be taken to eliminate possible threats in the projects.

SaaS, QoS, and QoE are out of the scope of this SM study. Future SM or SLR studies might focus on this subject for measuring the quality of network performance. Moreover, future SM studies can observe the trend and gap analysis of the new SPQM that come with new technologies such as AI, IoT, CoT, Machine Learning, and Robotics [S50]. CMMI v2.0 [4] and PMBOK [7] emphasize the adoption of the processes following Agile. Unfortunately, only two papers propose metrics related to Agile. It may be useful to conduct an SM or SLR study in this area.

Although there are many security-related metrics in the literature, we cannot say that the entire system is secure by analyzing the metric for only one security attack type. Therefore, there is a need for more reliable security metrics/measurements to quantify different aspects of security (unintended threats) for specific goals with different combinations [S19] [55]. Therefore, it is estimated that the need for security-related product quality metrics may increase in the future, thus setting the ground for additional studies.

APPENDIX

List of papers, which used to extract data for this SM study, is given as follows:

[S1] P. Rotella and S. Chulani, "Implementing quality metrics and goals at the corporate level," in *Proc. 8th Working Conf. Mining Softw. Repositories*, 2011, pp. 112–113.

[S2] R. Plösch, J. Bräuer, C. Körner, and M. Saft, "MUSE: A framework for measuring object-oriented design quality," *J. Object Technol.*, vol. 15, no. 4, pp. 1–29, 2016.

[S3] Z. Li, P. Liang, P. Avgeriou, N. Guelfi, and A. Ampatzoglou, "An empirical investigation of modularity metrics for indicating architectural technical debt," in *Proc. 10th Int. Sigsoft Conf. Qual. Softw. Archit.*, 2014, pp. 119–128.

[S4] J. Yahaya and A. Deraman, "Measuring the unmeasurable characteristics of software product quality," *Int. J. Adv. Comput. Technol.*, vol. 2, no. 4, pp. 95–106, Oct. 2010.

[S5] G. Nair and V. Suma, "Estimation of characteristics of a software team for implementing effective inspection process through inspection performance metric," 2011, *arXiv:1107.3201*. [Online]. Available: https://arxiv.org/abs/1107.3201

[S6] D. I. K. Sjøberg, B. Anda, and A. Mockus, "Questioning software maintenance metrics: A comparative case study," in *Proc. IEEE Int. Symp. Empirical Softw. Eng. Meas.*, 2012, pp. 107–110.

[S7] A. Gosain, S. Nagpal, and S. Sabharwal, "Quality metrics for conceptual models for data warehouse focusing on dimension hierarchies," *ACM SIGSOFT Softw. Eng. Notes*, vol. 36, no. 4, p. 1, 2011.

[S8] I. Atoum, C. Bong, and N. Kulathuramaiyer, "Towards resolving software quality-in-use measurement challenges," *J. Emerg. Trends Comput. Inf. Sci.*, vol. 5, no. 11, pp. 877–885, 2014.

[S9] R. L. Nord, I. Ozkaya, H. Koziolek, and P. Avgeriou, "Quantifying software architecture quality report on the first international workshop on software architecture metrics," *ACM SIGSOFT Softw. Eng. Notes*, vol. 39, no. 5, pp. 32–34, Sep. 2014.

[S10] K. Jinzenji, T. Hoshino, L. Williams, and K. Takahashi, "Metric-based quality evaluations for iterative software development approaches like agile," in *Proc. IEEE 23rd Int. Symp. Softw. Rel. Eng. Workshops*, Nov. 2012, pp. 54–63.

[S11] K. Lochmann and L. Heinemann, "Integrating quality models and static analysis for comprehensive quality assessment," in *Proc. 2nd Int. Workshop Emerg. Trends Softw. Metrics*, 2011, pp. 5–11.

[S12] C. E. Otero, E. Dell, A. Qureshi, and L. D. Otero, "A quality-based requirement prioritization framework using binary inputs," in *Proc. 4th Asia Int. Conf. Math./Anal. Modeling Comput. Simulation*, 2010, pp. 187–192.

[S13] L. Ping, "A quantitative approach to software maintainability prediction," *Proc. Int. Forum Inf. Technol. Appl.*, vol. 1, 2010, pp. 105–108.

[S14] S. Misra, A. Adewumi, L. Fernandez-Sanz, and R. Damasevicius, "A suite of object oriented cognitive complexity metrics," *IEEE Access*, vol. 6, pp. 8782–8796, 2018.

[S15] V. A. S. Velan S, and C. Babu, "Evaluation of reusability in aspect oriented software using inheritance metrics," in *Proc. IEEE Int. Conf. Adv. Commun., Control Comput. Technol.*, May 2014, pp. 1715–1722.

[S16] K. Z. Sultana, B. J. Williams, and A. Bosu, "A comparison of nano-patterns vs. Software metrics in vulnerability prediction," in *Proc. 25th Asia–Pacific Softw. Eng. Conf. (APSEC)*, Dec. 2018, pp. 355–364.

[S17] S. Husein and A. Oxley, "A coupling and cohesion metrics suite for object-oriented software," in *Proc. Int. Conf. Comput. Technol. Develop. (ICCTD)*, vol. 1, 2009, pp. 421–425.

[S18] M. Abdellatief, A. B. M. Sultan, A. A. A. Ghani, and M. A. Jabar, "A mapping study to investigate component-based software system metrics," *J. Syst. Softw.*, vol. 86, no. 3, pp. 587–603, Mar. 2013.

[S19] P. K. Manadhata and J. M. Wing, "An attack surface metric," *IEEE Trans. Softw. Eng.*, vol. 37, no. 3, pp. 371–386, May 2011.

[S20] R. Malhotra and A. Chug, "An empirical study to redefine the relationship between software design metrics and maintainability in high data intensive applications," *Eng. Comput. Sci.*, vol. 1, pp. 61–66, Dec. 2013.

[S21] B. L. Sousa, M. A. S. Bigonha, and K. A. M. Ferreira, "An exploratory study on cooccurrence of design patterns and bad smells using software metrics," *Softw., Pract. Exper.*, vol. 49, no. 7, pp. 1079–1113, May 2019.

[S22] J. Dong, "An improved fuzzy synthesis evaluation algorithm for software quality," *Proc. Int. Conf. Inf. Manag. Innov. Manag. Ind. Eng.*, vol. 2, 2009, pp. 565–569.

[S23] A. Tahir and R. Ahmad, "An AOP-based approach for collecting software maintainability dynamic metrics," in *Proc. 2nd Int. Conf. Comput. Res. Dev.*, 2010, pp. 168–172.

[S24] S. Eski and F. Buzluca, "An empirical study on object-oriented metrics and software evolution in order to reduce testing costs by predicting change-prone classes," in *Proc. IEEE 4th Int. Conf. Softw. Test., Verification Validation Workshops*, Mar. 2011, pp. 566–571.

[S25] R. Malhotra and M. Khanna, "A new metric for predicting software change using gene expression programming," in *Proc. 5th Int. Workshop Emerg. Trends Softw. Metrics*, 2014, pp. 8–14.

[S26] R. A. Coelho, F. D. R. N. Guimaraes, and A. A. A. Esmin, "Applying swarm ensemble clustering technique for fault prediction using software metrics," in *Proc. 13th Int. Conf. Mach. Learn. Appl.*, Dec. 2014, pp. 356–361.

[S27] P. Meirelles, C. Santos, Jr., J. Miranda, F. Kon, A. Terceiro, and C. Chavez, "A study of the relationships between source code metrics and attractiveness in free software projects," in *Proc. Brazilian Symp. Softw. Eng.*, Sep. 2010, pp. 11–20.

[S28] D. Westermann, J. Happe, R. Krebs, and R. Farahbod, "Automated inference of goal-oriented performance prediction functions," in *Proc. 27th IEEE/ACM Int. Conf. Automat. Softw. Eng.*, Sep. 2012, pp. 190–199.

[S29] K. Gao, T. M. Khoshgoftaar, H. Wang, and N. Seliya, "Choosing software metrics for defect prediction: An investigation on feature selection techniques," *Softw.-Practice Exper.*, vol. 41, no. 5, pp. 579–606, 2011.

[S30] M. Bansal and P. Agrawal, "Critical analysis of object oriented metrics in software development," in *Proc. Int. Conf. Adv. Comput. Commun. Technol.*, 2014, pp. 197–201.

[S31] Ö. F. Arar and K. Ayan, "Deriving thresholds of software metrics to predict faults on open source software: Replicated case studies," *Expert Syst. Appl.*, vol. 61, pp. 106–121, Nov. 2016.

[S32] L. B. L. de Souza and M. de Almeida Maia, "Do software categories impact coupling metrics?" in *Proc. IEEE Int. Work. Conf. Min. Softw. Repos.*, May 2013, pp. 217–220.

[S33] A. Kaur, P. S. Sandhu, and A. S. Bra, "Early software fault prediction using real time defect data," in *Proc. 2nd Int. Conf. Mach. Vis.*, 2009, pp. 242–245.

[S34] U. Erdemir, U. Tekin, and F. Buzluca, "E-quality: A graph based object oriented software quality visualization tool," in *Proc. 6th Int. Workshop Visualizing Softw. Understand. Anal. (VISSOFT)*, Sep. 2011, pp. 1–8.

[S35] M. P. Barcellos, R. D. Falbo, and A. R. Rocha, "Establishing a well-founded conceptualization about software measurement in high maturity levels," in *Proc. 7th Int. Conf. Qual. Inf. Commun. Technol.*, Sep. 2010, pp. 467–472.

[S36] J. Feigenspan, S. Apel, J. Liebig, and C. Kästner, "Exploring software measures to assess program comprehension," in *Proc. Int. Symp. Empir. Softw. Eng. Meas.*, 2011, pp. 127–136.

[S37] M. Umarji and C. Seaman, "Gauging acceptance of software metrics: Comparing perspectives of managers and developers," in *Proc. 3rd Int. Symp. Empirical Softw. Eng. Meas.*, Oct. 2009, pp. 236–247.

[S38] A. Tripathi and K. Sharma, "Improving software quality based on relationship among the change proneness and object oriented metrics," in *Proc. Int. Conf. Comput. Sustain. Glob. Dev. (INDIACom)*, 2015, pp. 1633–1636.

[S39] S. Srivastava and R. Kumar, "Indirect method to measure software quality using CK-OO suite," in *Proc. Int. Conf. Intell. Syst. Signal Process. (ISSP)*, 2013, pp. 47–51.

[S40] A.-J. Molnar, A. Neamţu, and S. Motogna, "Longitudinal evaluation of software quality metrics in open-source applications," in *Proc. 14th Int. Conf. Eval. Novel Approaches Softw. Eng.*, 2019, pp. 80–91.

[S41] T. Ruhroth, H. Voigt, and H. Wehrheim, "Measure, diagnose, refactor: A formal quality cycle for software models," in *Proc. 35th Euromicro Conf. Softw. Eng. Adv. Appl.*, 2009, pp. 360–367.

[S42] P. K. Kapur, G. Singh, N. Sachdeva, and A. Tickoo, "Measuring software testing efficiency using two-way assessment technique," in *Proc. 3rd Int. Conf. Rel., Infocom Technol. Optim.*, Oct. 2014, pp. 1–6.

[S43] R. Plösch, J. Bräuer, C. Körner, and M. Saft, "Measuring, assessing and improving software quality based on object-oriented design principles," *Open Comput. Sci.*, vol. 6, no. 1, pp. 187–207, Dec. 2016.

[S44] Y. Shi, M. Li, S. Arndt, and C. Smidts, "Metric-based software reliability prediction approach and its application," *Empirical Softw. Eng.*, vol. 22, no. 4, pp. 1579–1633, Aug. 2017.

[S45] C. Santos, T. Novais, M. Ferreira, C. Albuquerque, I. De Farias, and A. Furtado, "Metrics focused on usability ISO 9126 based," in *Proc. Iber. Conf. Inf. Syst. Technol. Cist.*, Jul. 2016, pp. 1–3.

[S46] S. U. Farooq, S. Quadri, and N. Ahmad, "Metrics, models and measurements in software reliability," in *Proc. IEEE 10th Int. Symp. Appl. Mach. Intell. Informat. (SAMI)*, Jan. 2012, pp. 441–449.

[S47] S. Ragab and H. Ammar, "Object oriented design metrics and tools a survey," in *Proc. 7th Int. Conf. Inform. Syst.*, 2010, pp. 1–7.

[S48] V. Gupta and J. K. Chhabra, "Package level cohesion measurement in object-oriented software," *J. Brazilian Comput. Soc.*, vol. 18, no. 3, pp. 251–266, Sep. 2012.

[S49] U. Pooja, "Prediction of software defects using object-oriented metrics," vol. 9, no. 1, pp. 889–899, 2018.

[S50] D. Chhillar and K. Sharma, "Proposed T-model to cover 4S quality metrics based on empirical study of root cause of software failures," *Int. J. Electr. Comput. Eng.*, vol. 9, no. 2, pp. 1122–1130, 2019.

[S51] J. S. Challa, A. Paul, Y. Dada, V. Nerella, and P. R. Srivastava, "Quantification of software quality parameters using fuzzy multi criteria approach," in *Proc. Int. Conf. Process Autom., Control Comput.*, Jul. 2011, pp. 1–6.

[S52] P. Oliveira, F. P. Lima, M. T. Valente, and A. Serebrenik, "RTTool: A tool for extracting relative thresholds for source code metrics," in *Proc. IEEE Int. Conf. Softw. Maintenance Evol.*, Sep. 2014, pp. 629–632.

[S53] M. Aniche, C. Treude, A. Zaidman, A. Van Deursen, and M. Gerosa, "SATT: Tailoring code metric thresholds for different software architectures," in *Proc. IEEE 16th Int. Work. Conf. Source Code Anal. Manip. (SCAM)*, Dec. 2016, pp. 41–50.

[S54] J. Wang, H. Wang, M. Guo, and M. Xia, "Security metrics for software systems," in *Proc. 47th Annu. Southeast Reg. Conf. (ACM-SE)*, vol. 47, 2009, pp. 1–6.

[S55] A. Tiwari and P. S. Chakraborty, "Software component quality characteristics model for component based software engineering," in *Proc. IEEE Int. Conf. Comput. Intell. Commun. Technol.*, Feb. 2015, pp. 47–51.

[S56] G. Lajios, "Software metrics suites for project landscapes," in *Proc. 13th Eur. Conf. Softw. Maintenance Reeng.*, 2009, pp. 317–318.

[S57] M. F. S. Oliveira, R. M. Redin, L. Carro, L. Lamb, and F. Wagner, "Software quality metrics and their impact on embedded software," in *Proc. 5th Int. Workshop Model-based Methodol. Pervas. Embedded Softw.*, Apr. 2008, pp. 68–77.

[S58] K. Punitha and S. Chitra, "Software defect prediction using software metrics," in *Proc. Int. Conf. Inf. Commun. Embed. Syst. (ICICES)*, 2013, pp. 555–558.

[S59] J. Chen and X. Liu, "Software maintainability metrics based on the index system and fuzzy method," in *Proc. 1st Int. Conf. Inf. Sci. Eng.*, Dec. 2009, pp. 5117–5120.

[S60] G. D. Pereira Moreira, R. P. Mellado, D. Á. Montini, L. A. V. Dias, and A. M. da Cunha, "Software product measurement and analysis in a continuous integration environment," in *Proc. 7th Int. Conf. Inf. Technol., New Generat.*, 2010, pp. 1177–1182.

[S61] P. S. Silveira, K. Becker, and D. D. Ruiz, "SPDW+: A seamless approach for capturing quality metrics in software development environments," *Softw. Qual. J.*, vol. 18, no. 2, pp. 227–268, Jun. 2010.

[S62] M. K. Debbarma, N. Kar, and A. Saha, "Static and dynamic software metrics complexity analysis in regression testing," in *Proc. Int. Conf. Comput. Commun. Informat.*, Jan. 2012, pp. 1–6.

[S63] A. Janus, R. Dumke, A. Schmietendorf, and J. Jager, "The 3C approach for agile quality assurance," in *Proc. 3rd Int. Workshop Emerg. Trends Softw. Metrics (WETSoM)*, Jun. 2012, pp. 9–13.

[S64] G. Rakic, Z. Budimac, and K. Bothe, "Towards a 'Universal' software metrics tool: Motivation, process and a prototype," in *Proc. 5th Int. Conf. Softw. Data Technol.*, vol. 2, 2010, pp. 263–266.

[S65] G. Yilmaz, S. Tunalilar, and O. Demirors, "Towards the development of a defect detection tool for COSMIC functional size measurement," in *Proc. Joint Conf. 23rd Int. Workshop Softw. Meas. 8th Int. Conf. Softw. Process Product Meas.*, Oct. 2013, pp. 9–16.

[S66] I. Chowdhury and M. Zulkernine, "Using complexity, coupling, and cohesion metrics as early indicators of vulnerabilities," *J. Syst. Archit.*, vol. 57, no. 3, pp. 294–313, Mar. 2011.

[S67] U. Kulkarni, Y. Kalshetty, and V. Arde, "Validation of CK metrics for object oriented design measurement," in *Proc. 3rd Int. Conf. Emerg. Trends Eng. Technol. (ICETET)*, 2010, pp. 646–651.

[S68] M. Rudolph and R. Schwarz, "A critical survey of security indicator approaches," in *Proc. 7th Int. Conf. Availability, Reliab., Secur. (ARES)*, 2012, pp. 291–300.

[S69] P. Sandhu, A. Brar, R. Goel, J. Kaur, and S. Anand, "A model for early prediction of faults in software systems," in *Proc. 2nd Int. Conf. Comput. Autom. Eng. (ICCAE)*, vol. 4, 2010, pp. 281–285.

[S70] M. Thirugnanam and J. N. Swathi, "Quality metrics tool for object oriented programming," *Int. J. Comput. Theory Eng.*, vol. 2, no. 5, pp. 712–717, 2010.

REFERENCES

[1] *IEEE Standard for a Software Quality Metrics Methodology*, Standard 1061-1998, 1998.

[2] T. Demarco, *Controlling Software Projects: Management, Measurement, and Estimates*. London, U.K.: Pearson, 1982.

[3] E. Braude and M. Bernstein, *Software Engineering: Modern Approaches*, 2nd ed. Hoboken, NJ, USA: Wiley, 2011, pp. 722–753.

[4] *CMMI Development Full Model v2.0*, CMMI Institute, Pittsburgh, PA, USA, 2018.

[5] K. Johnson and M. Kulpa, "Measurement within the CMMI," in *Proc. 16th Int. Conf. Softw. Eng. Process Group*, FL, USA, Mar. 2004.

[6] S, M. Cap Gemini. (Sep. 2018). *World Quality Report-2018-2019*. Accessed: Sep. 30, 2019. [Online]. Available: https://community.microfocus.com/t5/Application-Delivery-Management/The-World-Quality-Report-2018-19-is-now-available/ba-p/1666131/page/2

[7] *A Guide to the Project Management Body of Knowledge (PMBOK GUIDE)*, Project Management Institute (PMI), Newtown Square, PA, USA, 2017.

[8] G. Casale, C. Chesta, P. Deussen, E. Di Nitto, P. Gouvas, S. Koussouris, V. Stankovski, A. Symeonidis, V. Vlassiou, A. Zafeiropoulos, and Z. Zhao, "Current and future challenges of software engineering for services and applications," *Procedia Comput. Sci.*, vol. 97, pp. 34–42, 2016.

[9] S. Ouhbi, A. Idri, J. L. Fernández-Alemán, and A. Toval, "Predicting software product quality: A systematic mapping study," *Comput. Sistemas*, vol. 19, no. 3, pp. 547–562, Oct. 2015, doi: 10.13053/CyS-19-3-1960.

[10] B. Kitchenham, P. Brereton, and D. Budgen, "The educational value of mapping studies of software engineering literature," in *Proc. 32nd ACM/IEEE Int. Conf. Softw. Eng.*, vol. 1, 2010, pp. 589–598, doi: 10.1145/1806799.1806887.

[11] D. Budgen, M. Turner, P. Brereton, and B. Kitchenham, "Using mapping studies in software engineering," *Ppig*, vol. 2, pp. 195–204, Dec. 2008.

[12] ISTQB. *International Software Testing Qualifications Board (ISTQB) Glossary*. Accesses: Aug. 1, 2019. [Online]. Available: https://glossary.istqb.org/en/search/

[13] *Systems and Software Engineering—Systems and Software Quality Requirements and Evaluation (SQuaRE)—Guide to SQuaRE)*, Standard ISO/IEC 25000: 2014. Accessed: Mar. 1, 2020. [Online]. Available: https://iso25000.com/index.php/en/iso-25000-standards

[14] *Information Technology—Software Product Quality—Part 1: Quality Model*, Standard ISO/IEC 9126-1, 2001.

[15] *Systems and Software Engineering—Systems and Software Quality Requirements and Evaluation (SQuaRE)—System and Software Quality Models*, Standard ISO/IEC 25010, 2011. Accessed: Mar. 1, 2020. [Online]. Available: https://iso25000.com/index.php/en/iso-25000-standards/iso-25010

[16] *Software Engineering—Software Product Quality Requirements and Evaluation (SQuaRE)-Data Quality Model*, Standard ISO, ISO/IEC 25012, 2008. Accessed: Mar. 1, 2020. [Online]. Available: https://iso25000.com/index.php/en/iso-25000-standards/iso-25012

[17] N. Fenton and S. Pfleeger, *Software Metrics: A Rigorous and Practical Approach*, 2nd ed. London, U.K.: PWS, 1997, pp. 245–322.

[18] M. Santos, P. Afonso, P. H. Bermejo, and H. Costa, "Metrics and statistical techniques used to evaluate internal quality of object-oriented software: A systematic mapping," in *Proc. 35th Int. Conf. Chilean Comput. Sci. Soc. (SCCC)*, Oct. 2016, pp. 1–11, doi: 10.1109/SCCC.2016.7836021.

[19] P. Nistala, K. V. Nori, and R. Reddy, "Software quality models: A systematic mapping study," in *Proc. IEEE/ACM Int. Conf. Softw. Syst. Processes (ICSSP)*, May 2019, pp. 125–134.

[20] *PROMISE Software Engineering Database*, School Inf. Technol. Eng., University of Ottawa, Ottawa, ON, Canada, 2020. Accessed: Mar. 31, 2020. [Online]. Available: http://promise.site.uottawa.ca/SERepository/

[21] P. Bourque and R. Fairley, "Guide to the software engineering body of knowledge: SWEBOK," IEEE Comput. Soc., Washington, DC, USA, Tech. Rep. ISO/IEC TR 19759, May 2014.

[22] B. Kitchenham and S. Charters, "Guidelines for performing systematic literature reviews in software engineering," *Tech. Rep.*, vol. 2, no. 3, pp. 1–57, 2007.

[23] C. Kohl, E. J. McIntosh, S. Unger, N. R. Haddaway, S. Kecke, J. Schiemann, and R. Wilhelm, "Online tools supporting the conduct and reporting of systematic reviews and systematic maps: A case study on CADIMA and review of existing tools," *Environ. Evidence*, vol. 7, no. 1, pp. 1–17, Dec. 2018, doi: 10.1186/s13750-018-0115-5.

[24] *CADIMA SLR Tool*. Accessed: Feb. 29, 2020. [Online]. Available: https://www.cadima.info/index.php

[25] S. Rehman, "Swot analysis of software quality metrics for global software development: A systematic literature review protocol," *IOSR J. Comput. Eng.*, vol. 2, no. 1, pp. 1–7, 2012, doi: 10.9790/0661-0210107.

[26] A. Tahir and S. G. MacDonell, "A systematic mapping study on dynamic metrics and software quality," in *Proc. 28th IEEE Int. Conf. Softw. Maintenance (ICSM)*, Sep. 2012, pp. 326–335, doi: 10.1109/ICSM.2012.6405289.

[27] N. Vanitha and R. ThirumalaiSelvi, "A report on the analysis of metrics and measures on software quality factors—A literature study," *Int. J. Comput. Sci. Inf. Technol.*, vol. 5, no. 5, pp. 6591–6595, 2014.

[28] A. Dur and S. Titang, "The visualization of software quality metrics—A systematic literature review," M.S. thesis, Dept. Comput. Sci. Eng., Gothenburg, Sweden, 2015.

[29] E. Ronchieri and M. Canaparo, "A preliminary mapping study of software metrics thresholds," in *Proc. 11th Int. Joint Conf. Softw. Technol.*, 2016, pp. 232–240, doi: 10.5220/0005988402320240.

[30] T. Wahyuningrum and K. Mustofa, "A systematic mapping review of software quality measurement: Research trends, model, and method," *Int. J. Electr. Comput. Eng.*, vol. 7, no. 5, pp. 2847–2854, 2017, doi: 10.11591/ijece.v7i5.pp2847-2854.

[31] M. Yan, X. Xia, X. Zhang, L. Xu, and D. Yang, "A systematic mapping study of quality assessment models for software products," in *Proc. Int. Conf. Softw. Anal., Test. Evol. (SATE)*, Nov. 2017, pp. 63–71.

[32] S. Tiwari and S. S. Rathore, "Coupling and cohesion metrics for object-oriented software: A systematic mapping study," in *Proc. 11th Innov. Softw. Eng. Conf.*, Feb. 2018, pp. 1–11.

[33] A. J. Suali, S. S. M. Fauzi, M. H. N. M. Nasir, W. A. W. M. Sobri, and I. K. Raharjana, "Software quality measurement in software engineering project: A systematic literature review," *J. Theor. Appl. Inf. Technol.*, vol. 97, no. 3, pp. 918–929, 2019.

[34] B. Kitchenham, O. Pearl Brereton, D. Budgen, M. Turner, J. Bailey, and S. Linkman, "Systematic literature reviews in software engineering—A systematic literature review," *Inf. Softw. Technol.*, vol. 51, no. 1, pp. 7–15, Jan. 2009, doi: 10.1016/j.infsof.2008.09.009.

[35] (Jul. 27, 2019). *Systematic Reviews-Research Guide*. Accessed: Mar. 31, 2020. [Online]. Available: https://libguides.murdoch.edu.au/systematic/PICO

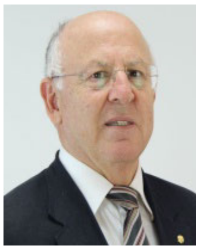
[36] B. Kitchenham, D. Budgen, and O. P. Brereton, "Using mapping studies as the basis for further research: A participant-observer case study," *J. Inf. Softw. Technol.*, vol. 53, no. 6, pp. 638–651, Jun. 2011, doi: 10.1016/j.infsof.2010.12.011.

[37] D. S. Cruzes and T. Dybå, "Recommended steps for thematic synthesis in software engineering," in *Proc. Int. Symp. Empir. Softw. Eng. Meas.*, 2011, vol. 7491, pp. 275–284, doi: 10.1109/ESEM.2011.36.

[38] V. Clarke and V. Braun. *Guidelines for Reviewers and Editors Evaluating Thematic Analysis Manuscripts*. Accessed: Jul. 14, 2019. [Online]. Available: https://www.psych.auckland.ac.nz/en/about/thematic-analysis.html
[39] C. Wohlin, "Guidelines for snowballing in systematic literature studies and a replication in software engineering," *Assoc. Comput. Mach.*, vol. 7, pp. 1–10, May 2014, doi: 10.1145/2601248.2601268.
[40] F. N. Colakoglu and A. Mishra. *Online Paper Repository for Systematic Mapping Study on Software Product Quality Metrics*. Accessed: Oct. 15, 2020. [Online]. Available: https://tinyurl.com/yxvzgqh4
[41] D. Santos, A. Resende, P. Junior, and H. Costa, "External quality metrics for object oriented software—A systematic literature review," *Duke Law J.*, vol. 20, no. 3, pp. 1–18, doi: 10.1017/CBO9781107415324.004.
[42] B. Everitt and A. Skrondal, *The CAMBRIDGE Dictionary of Statistics*, 4th ed. Cambridge, U.K.: Cambridge Univ. Press, 2010.
[43] C. E. Otero, E. Dell, A. Qureshi, and L. D. Otero, "A quality-based requirement prioritization framework using binary inputs," in *Proc. 4th Asia Int. Conf. Math./Analytical Model. Comput. Simulation*, 2010, pp. 187–192, doi: 10.1109/AMS.2010.48.
[44] J. Chen and X. Liu, "Software maintainability metrics based on the index system and fuzzy method," in *Proc. 1st Int. Conf. Inf. Sci. Eng.*, Dec. 2009, pp. 5117–5120, doi: 10.1109/ICISE.2009.1073.
[45] G. D. Pereira Moreira, R. P. Mellado, D. Á. Montini, L. A. V. Dias, and A. Marques da Cunha, "Software product measurement and analysis in a continuous integration environment," in *Proc. 7th Int. Conf. Inf. Technol., New Generat.*, 2010, pp. 1177–1182, doi: 10.1109/ITNG.2010.85.
[46] B. Keser, T. Iyidogan, and B. Ozkan, "ASSIST: An integrated measurement tool," in *Proc. 23rd Int. Work. Softw. Meas.*, 2013, pp. 237–242, doi: 10.1109/IWSM-Mensura.2013.41.
[47] P. Meirelles, C. Santos, J. Miranda, F. Kon, A. Terceiro, and C. Chavez, "A study of the relationships between source code metrics and attractiveness in free software projects," *Proc. 24th Brazilian Symp. Softw. Eng.*, 2010, pp. 11–20, doi: 10.1109/SBES.2010.27.
[48] B. L. Sousa, M. A. S. Bigonha, and K. A. M. Ferreira, "An exploratory study on cooccurrence of design patterns and bad smells using software metrics," *Softw., Pract. Exper.*, vol. 7, pp. 1079–1113, May 2019, doi: 10.1002/spe.2697.
[49] P. Oliveira, F. P. Lima, M. T. Valente, and A. Serebrenik, "RTTool: A tool for extracting relative thresholds for source code metrics," in *Proc. IEEE Int. Conf. Softw. Maintenance Evol.*, Sep. 2014, pp. 629–632, doi: 10.1109/ICSME.2014.112.
[50] L. B. L. de Souza and M. de Almeida Maia, "Do software categories impact coupling metrics?" in *Proc. 10th Work. Conf. Mining Softw. Repositories (MSR)*, May 2013, pp. 217–220, doi: 10.1109/MSR.2013.6624030.
[51] R. Plösch, J. Bräuer, C. Körner, and M. Saft, "MUSE: A framework for measuring object-oriented design quality," *J. Object Technol.*, vol. 15, no. 4, pp. 1–29, 2016, doi: 10.5381/jot.2016.15.4.a2.
[52] A.-J. Molnar, A. Neamáu, and S. Motogna, "Longitudinal evaluation of software quality metrics in open-source applications," in *Proc. 14th Int. Conf. Eval. Novel Approaches Softw. Eng.*, 2019, pp. 80–91, doi: 10.5220/0007725600800091.
[53] K. Gao, T. M. Khoshgoftaar, H. Wang, and N. Seliya, "Choosing software metrics for defect prediction: An investigation on feature selection techniques," *Softw., Pract. Exper.*, vol. 41, no. 5, pp. 579–606, Apr. 2011, doi: 10.1002/spe.1043.
[54] QSM. *Industry Database (Quantitative Software Management Tool)*. Accessed: Mar. 31, 2020. [Online]. Available: https://www.qsm.com/luminosity/QSM_Demo/
[55] Cynet. (Apr. 2020). *COVID-19 Cyber Attack Analysis*. Accessed: Mar. 2020. [Online]. Available: https://tinyurl.com/y69bnmd4
[56] A. Mishra and Z. Otaiwi, "DevOps and software quality: A systematic mapping," *Comput. Sci. Rev.*, vol. 38, Nov. 2020, Art. no. 100308, doi: 10.1016/j.cosrev.2020.100308.
[57] K. A. M. Ferreira, M. A. S. Bigonha, R. S. Bigonha, L. F. O. Mendes, and H. C. Almeida, "Identifying thresholds for object-oriented software metrics," *J. Syst. Softw.*, vol. 85, no. 2, pp. 244–257, Feb. 2012, doi: 10.1016/j.jss.2011.05.044.
[58] *EFQM User Guide: EFQM Benchmarking Guidelines*, EFQM, Brussels, Belgium, 2013.
[59] K. Valenca, E. D. Canedo, and R. M. Da Costa Figueiredo, "Construction of a software measurement tool using systematic literature review," in *Proc. IEEE Int. Conf. Internet Things*, Jul. 2018, pp. 1349–1354, doi: 10.1109/Cybermatics_2018.2018.00308.
[60] T. Buzan, *Mind Map Mastery: The Complete Guide to Learning and Using the Most Powerful Thinking Tool in the Universe*. London, U.K.: Watkins Media, 2018.

FATIMA NUR COLAKOGLU received the B.S. degree in computer science and information system (CTIS) from Bilkent University, Ankara, Turkey, in 2011, and the M.S. degree in software engineering from Atilim University, Ankara, in 2019.

She worked at AYESAŞ, as a Software Quality Specialist in avionics projects. From 2014 to 2019, she worked as a Quality Assurance and Process Management Specialist at TUBITAK Bilgem Iltaren. Since 2019, she has been a Quality Specialist at HAVELSAN, Ankara. She has eight years of experience in software quality assurance and is the coauthor of two published articles related to quality process and process tailoring. She owns the certificates of ISO 9001:2015 Lead Auditor, AS 9100, CMMI v2.0, EFQM, ISO 10002:2018, ISO 27001, ISO 14001, ISO 45001, ISO 20000, ISO 31000, ISO 19011, and DO-178. She has received the highest-ranked student and excellent academic success awards from the CTIS Department, Bilkent University.



ALI YAZICI received the B.S. and M.S. degrees in mathematics from Middle East Technical University (METU), Ankara, Turkey, in 1972 and 1974, respectively, and the Ph.D. degree from the Computer Science Department, Waterloo University, Waterloo, ON, Canada, in 1983.

He is currently the Chairperson of the Software Engineering Department, Atilim University, Ankara. His research interests include parallel and distributed computing, cloud computing, big data programming, database systems, symbolic computing, and e-Topics. In the last 40 years, he has been affiliated as a full-time Academic Staff with Middle East Technical University, Yarmouk University, Jordan, Sultan Qaboos University, Oman, TOBB University of Economics and Technology, and Atilim University, Turkey. He is the writer of many scientific articles, books, and research reports in the field of computing and informatics.

Prof. Yazici is a Founding Member of Turkish Mathematics Foundation (since 1990) and Turkish Informatics Foundation (since 1990), and an Associate Member (since 1981) and an Honorary Board Member of Informatics Association of Turkey (2017–2021).

ALOK MISHRA (Senior Member, IEEE) is currently Professor in informatics and digitalization at Molde University College (A Specialized University in Logistics), Norway. He is also a Professor in software engineering with Atilim University, Turkey. His areas of research interests include software engineering, information systems, information technology, and artificial intelligence.

He is an editorial board member of many reputed journals, including *Computer Standards and Interfaces* (Elsevier), *Journal of Universal Computer Science*, *Computing and Informatics*, and *Data Technologies and Applications* journal. He is actively involved in editing special issues of reputed journals in his areas of research interest. He had also extensive experience in online education related to computing and management disciplines. In teaching, he has received Excellence in Online Education Award by U21Global Singapore, while in research; he has been awarded by the Scientific and Research Council of Turkey and the Board of Management of University for outstanding publications in *Science Citation Index* and *Social Science Citation Index* (Thomson Reuter) journals. He was a recipient of many scholarships, international awards, and research projects.