F. N. ÇOLAKOĞLU

SOFTWARE QUALITY METRICS: A SYSTEMATIC LITERATURE REVIEW

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
ATILIM UNIVERSITY

FATIMA NUR ÇOLAKOĞLU

A MASTER OF SCIENCE THESIS
IN
THE DEPARTMENT OF SOFTWARE ENGINEERING

SEPTEMBER 2019

ATILIM UNIVERSITY 2019

SOFTWARE QUALITY METRICS: A SYSTEMATIC LITERATURE REVIEW

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
ATILIM UNIVERSITY

BY

FATIMA NUR ÇOLAKOĞLU

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE IN
THE DEPARTMENT OF SOFTWARE ENGINEERING

SEPTEMBER 2019

Approval of the Graduate School of Natural and Applied Sciences, Atılım University.

_____

Prof. Dr. Ali KARA
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of **Master of Science in Software Engineering, Atılım University**.

_____

Prof. Dr. Ali YAZICI
Head of Department

This is to certify that we have read the thesis SOFTWARE QUALITY METRICS: A SYSTEMATIC LITERATURE REVIEW submitted by FATIMA NUR ÇOLAKOĞLU and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

_____

Prof. Dr. Ali YAZICI
Supervisor

Examining Committee Members:

Prof. Dr. Alok MISHRA
Software Engineering Department,
Atılım University _____

Prof. Dr. Ali YAZICI
Software Engineering Department,
Atılım University _____

Prof. Dr. Erdoğan DOĞDU
Computer Engineering Department,
Çankaya University _____

Date: 02.09.2019

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Fatıma Nur ÇOLAKOĞLU

Signature :

ABSTRACT

SOFTWARE QUALITY METRICS: A SYSTEMATIC LITERATURE REVIEW

Çolakoğlu, Fatıma Nur

M.S., Department of Software Engineering

Supervisor: Prof. Dr. Ali YAZICI

September 2019, 128 pages

Context:

Within the current competitive world we inhabit, producing quality products has become a prominent factor that warrants the enduring success of competitors in business. Along with it, defining and following the software quality metrics to be used in the detection of the current quality situation, and hence maintaining the continuous improvement of systems within the software industry, gained tremendous importance. Many international standards and models focusing on this need would definitely agree with Tom DeMarco, who stated that "we cannot control and improve something that we haven't measured". Deriving from this motto, this thesis sets out to analyze a specific set of articles and conference papers published in last ten years, which specifically focus on software quality metrics as indicated in their titles and abstract sections.

Goal:

In the initial literature review conducted for this study, any classification of software quality criteria was not encountered. For this reason, it was necessary to investigate the current studies in the field of software quality criteria, which would allow for the analysis of the current situation as well as enabling us to make predictions regarding the future research areas. For this aim, this study classifies software quality metrics related articles and conference papers published in the last 10 years (2009-2019) with the aim of both analyzing the active research areas of software quality metrics and at the same time revealing the maturity level of software quality metric in the software engineering sector.

Method:

This thesis is based on a set of documents consisting of articles and conference papers, since their preparation and publication takes less time in comparison to the books. The title and abstract parts of the articles and conference papers, which focus on software quality metrics/measurement, were examined according to the selection criteria and a systematic mapping study was conducted. As a result of this initial classification, four ensuing research questions, which are related to the objective of determining the maturity and gap analysis of the information in this area, are identified. In order to produce valid answers to these research questions, information on the document set was synthesized and "systematic literature review" was carried out.

Outputs:

A list of articles and conference papers this study focuses on were published between 2009 and 2019, and they are publicly accessible on the web. Systematic literature review method was applied in total to 70 articles and conference papers. Main outputs of the study are presented through graphics and explanations, and in order to facilitate understanding, the results are categorized through mind mapping method. Other outputs of this thesis are as follows: 1) Trend map between the years 2009 and 2019, 2) Issues determined to be open to development in this area, 3) Knowledge about the software quality metrics and measurement tools, 4) Compliance status between conference papers and articles and internationally valid quality models.

Results:

Trends regarding improvement in the software engineering sector are presented on software quality metrics. The results and findings obtained from this study may thus serve as input to the studies of the future researchers who aim to contribute to the development in this field.

Keywords: software quality, metric, measurement, systematic mapping, systematic literature review

ÖZ

YAZILIM KALİTE METRİKLERİ:
SİSTEMATİK LİTERATÜR İNCELEMESİ

Çolakoğlu, Fatıma Nur
Yüksek Lisans, Yazılım Mühendisliği
Tez Yöneticisi: Prof. Dr. Ali Yazıcı

Eylül 2019, 128 sayfa

Bağlam:

Piyasada rekabet arttıkça, ürünün kalitesi, ürünü rekabet yarışında öne çıkaran bir unsur haline gelmiştir. Yazılım sektöründe kalite açısından mevcut durumun tespitinin yapılarak sürekli iyileştirme sağlanması için yazılım kalite metriklerinin tanımlanması ve takibi hususu daha önemli bir hal almıştır. Tom DeMarco' nun ifade ettiği ve ayrıca birçok uluslararası standardın ve modellerin de vurguladığı gibi "Ölçmediğimiz bir şeyi kontrol edemeyiz ve iyileştiremeyiz". Bu ilke söz ile yola çıktığımız tez çalışmamızda son on yılda yayınlanan makale ve konferans bildirilerindeki başlık ve öz kısımlarında ana odak noktası yazılım kalite metrikleri olanlar analiz edilmiştir.

Amaç:

Literatür taraması sonucunda herhangi bir yazılım kalitesi ölçütlerini sınıflandırma çalışmasına rastlanmamıştır. Bu nedenle yazılım kalitesi ölçütleri alanındaki güncel çalışmaların belirlenmesi, analiz edilmesi, mevcut durumun haritalanması ve gelecek çalışma alanlarının belirlenmesi amaçlanmıştır. Bu çalışma, son 10 yılda (2009-2019) yayınlanan yazılım kalitesi ölçütleriyle ilgili makaleleri ve konferans bildirilerini sınıflandırmak ve böylelikle makale ve konferans bildirilerinde yazarlarının perspektifine dayanarak yazılım kalitesi ölçütlerinin aktif alanlarını

analiz etmek ve ayrıca yazılım mühendisliği sektöründe yazılım kalitesi ölçütlerinin olgunluk düzeyini ortaya çıkarmaktır.

Yöntem:

Bu tez çalışmasında hazırlanma, yayınlanma ve literatür havuzuna katılma hızı kitaplara göre daha yüksek olan makale ve konferans bildirilerinden oluşan doküman seti temel alınmıştır. Bunlardan başlık ve öz kısmında yazılım kalite metrikleri konusu çalışanlar seçim kriterlerine göre incelenerek öncelikle "sistematik haritalama" çalışması yapılmıştır. Ardından bu alandaki mevcut bilgi olgunluğunu ve açık noktaları tespit hedefi ile ilişkili görülen dört adet araştırma sorusu tanımlanmıştır. Bu araştırma sorularına cevap bulmak için doküman setindeki bilgiler sentezlenerek "sistematik literatür incelemesi" gerçekleştirilmiştir.

Çıktılar:

Bu tez kapsamında gerçekleştirilen SLR çalışması sonucunda çıkan veriler genel erişime açık olacak şekilde web üzerinden paylaşılmıştır. Sistematik literatür taraması yöntemi 2009-2019 yılları arasında yayınlanan 70 adet makale ve konferans bildirisinde uygulanmıştır. Çıktılar grafikler ve açıklamalar yoluyla verilmiş olup sonucun kolaylıkla görülmesi ve analiz edilebilmesi için zihin haritalama yöntemi ile sonuçlar kategorize edilerek sunulmuştur. Çalışmanın başlıca diğer başlıca çıktıları: 1) 2009-2019 yılları arasındaki eğilim haritası, 2) Bu alanda gelişmeye açık olduğu belirlenen hususlar, 3) Yazılım Kalite Metrikleri ve ölçüm araçları hakkında oluşan bilgi birikimi, 4) Uluslararası alanda geçerli olan kalite modelleri ile bildiriler ve makaleler arasındaki uyum durumu

Sonuçlar:

Yazılım kalite metrikleri konusunda yazılım mühendisliği sektöründeki eğilimler ve geliştirmeye açık alanlar sunulmuştur. Elde edilen bilgi ve bulgular bu alandaki gelişime katkı sağlamak isteyen araştırmacıların çalışmalarına girdi teşkil edecektir.

Anahtar Kelimeler: yazılım kalitesi, ölçütler, ölçümler, sistematik haritalama, sistematik literatür incelemesi

To My Family

ACKNOWLEDGMENTS

I would like to express deep gratitude to my supervisor Prof. Dr. Ali YAZICI, who has guided me by sharing his valuable experience, suggestions, encouragement and time.

Besides my supervisor, I would like to thank the rest of my thesis committee:
Prof. Dr. Alok MISHRA and Prof. Dr. Erdoğan DOĞDU for their contributions, constructive and insightful comments.

I would like to express my gratitude to my medical doctors: Prof. Dr Ali AYHAN, Assoc. Prof. Dr. Mutlu DOĞAN, Dr. Nazife ÇOLAKOĞLU, Prof. Dr. Hatice BODUR, Prof. Dr. Hürrem BODUR and Dr. Ali ÇOLAKOĞLU who have helped me beat cancer and restore my health. Only after getting healthy, I had a chance to continue my master's program.

I am also thankful to my family: Assoc. Prof. Dr. Mustafa ÇOLAKOĞLU, Refika ÇOLAKOĞLU, M. Eren ÇOLAKOĞLU and Att. M. Kemal ÇOLAKOĞLU for their patience and support in encouraging me to complete my thesis.

I am also thankful to my friends: Dr. Gökçe ONUR, Öykü ÖZDEN, Ebru KORKMAZ and Çağla ATAGÖREN for their help and supports to complete my thesis.

Finally, I would like to thank my managers and colleagues at HAVELSAN (Air Electronic Industry-Turkey) for their contribution and understanding during my thesis study.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS/ABBREVIATIONS

ACM	Association for Computing Machinery
BSI	British Standards Institution
CA	Afferent Coupling
CBO	Coupling between Objects
CC	Cyclomatic Complexity
CE	Efferent Coupling
CMMI	Capability Maturity Model Integration
DIT	Depth of Inheritance Tree
IEEE	Institute of Electrical and Electronics Engineers
COBIT	Control Objectives for Information and Related Technology
DQ	Data Quality
EFQM	European Foundation for Quality Management
ISO	International Standardization Organization
LOC	Line of Code
LCOM	Lack of Cohesion of Methods
MLOC	Method Lines of Code
N/A	Not Applicable
NBD	Nested Block Depth
NOC	Number of Classes
NOF	Number of Fields
NOM	Number of Methods
NORM	Number of Overridden Methods
NSC	Number of Children
NSM-PAR	Number of Parameters
MOOD	Metrics for Object Oriented Design
MPM	Managing Performance and Measurement
OO	Object Oriented

QA	Quality Assurance
PICO	Population Intervention Comparator Outcome
PMBOK	Project Management Body of Knowledge
QinU	Quality in Use
RQ	Research Question
SDLC	Software Development Life Cycle
SE	Software Engineering
SLR	Systematic Literature Review
SM	Systematic Mapping
SS	Search String
SPQ	Software Product Quality
SQM	Software Quality Metrics
WMC	Weighted Methods per Class

CHAPTER 1

INTRODUCTION

1.1. Software Quality Concepts

1.1.1. Definitions

This section aims to present an introductory definition and explanation of the Software Quality Metric related definitions and measurement processes.

CMMI v2.0 defines quality as “the degree to which a set of solution characteristics fulfills the requirements.” [1]

- Quality management is a two phased process that includes quality control and quality assurance. In general, quality assurance refers to all planned and systematic activities that are carried out within the quality system to ensure adequate standards in order to meet the requirements for quality of a unit (project, division, work, etc.). Noticeably, while quality assurance applies to the full life cycles; quality control can only be applied to the testing phase of the project. Therefore, quality assurance aims to prevent defects before they occur whereas quality control activities are related mostly to the detection and removal of the defects [2].
- According to ISTQB: Software Quality is defined as the totality of functionality and features of a software product that bear on its ability to satisfy stated or implied needs [3].
- There are different software quality definitions, but broadly software quality can be defined as the compliance between features offered by software and customer needs.

- IEEE1061-1992 defines measurement and software quality metric 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 processes a given attribute that affects its quality [4].
- Measurement is the act or process of assigning a number or category to an entity to describe an attribute of that entity. A figure, extent, or amount obtained by measuring [4].
- Quality Factor is a management and user-oriented attribute of software that contributes to its quality [4].

1.1.2. Software Quality Metric Framework

A framework provides a ground for analyzing, selecting, synthesis and mapping of more effective, traceable and meaningful metrics by presenting the breakdown approach.

The breakdown structure presented in Figure 1.1 suggests an approach through which the quality metrics might be determined:

1. Firstly, define quality requirements of the system in accordance with quality attributes,
2. After baseline, the quality requirements, the definition of quality factors, are derived from the quality requirements by managers and users,
3. Then set the quality factors, if there is a need, divide quality factors into sub-branches according to information received from technical personnel,
4. Generate metrics from sub-factors and direct metrics from main quality factors to measure the quality of product, process and project.

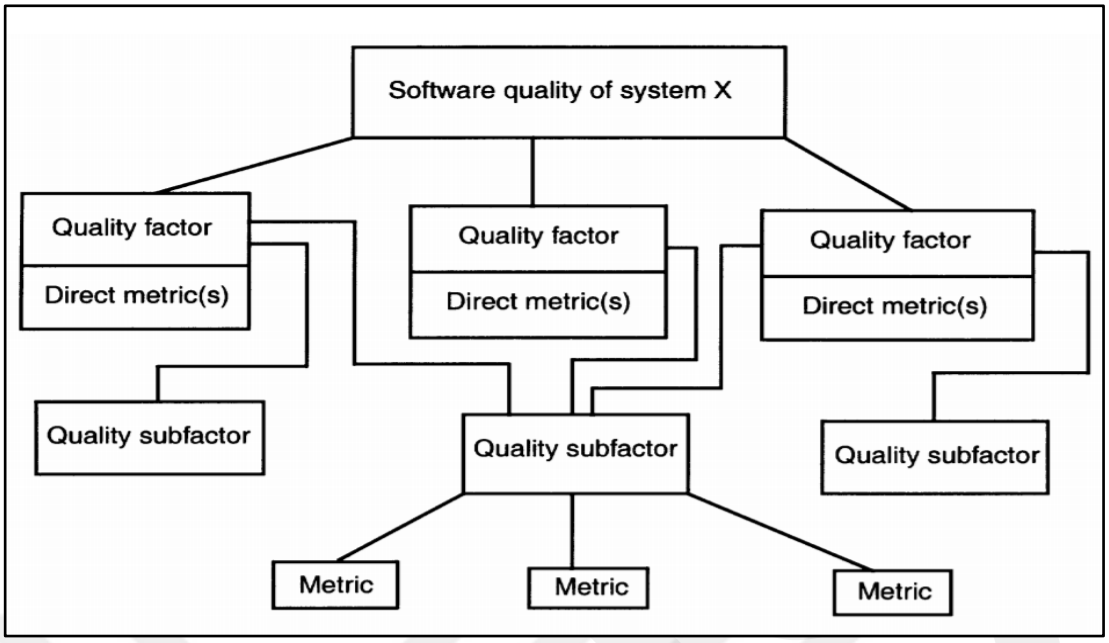


Figure 1.1 Software Quality Metrics Framework [4]

1.1.3. Quality Model

ISO/IEC 25000:2005 defines Quality Model as defined set of characteristics, and of relationships between them, which provides a framework for specifying quality requirements and evaluating quality [5].

Each one of these quality models consists of quality characteristics. Therefore, metrics related to the quality model can represent the status of quality characteristics metrics.

ISO9126:1998 standard is revised by ISO 25010:2011. ISO 9126 represented two main quality models; Internal/External Quality and Quality in Use [6]. In comparison to these quality models, ISO 25010:2011, presents the newly categorized quality models in three areas: Quality in Use, System/Software Product Quality and Data Quality as shown in Figure 1.2, Figure 1.3 and Figure 1.4 respectively.

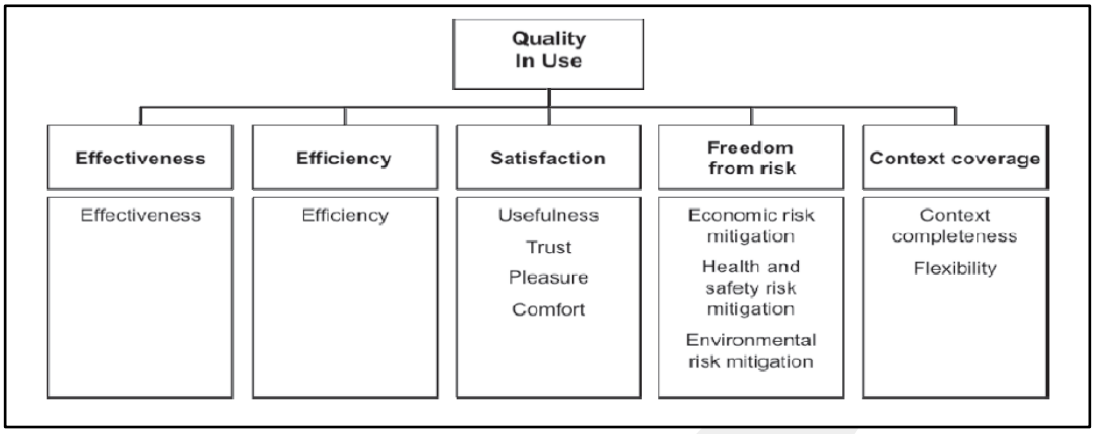


Figure 1.2 Quality in Use Model [7]

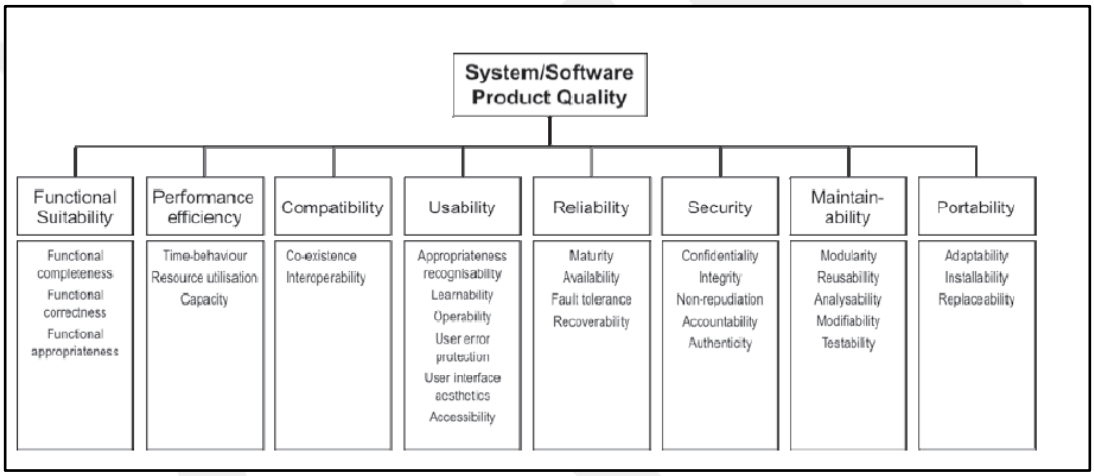


Figure 1.3 System/Software Product Quality Model [7]

Data Quality Model grouped under three headings: System Dependent (S), Inherent (I), Both (I/S).

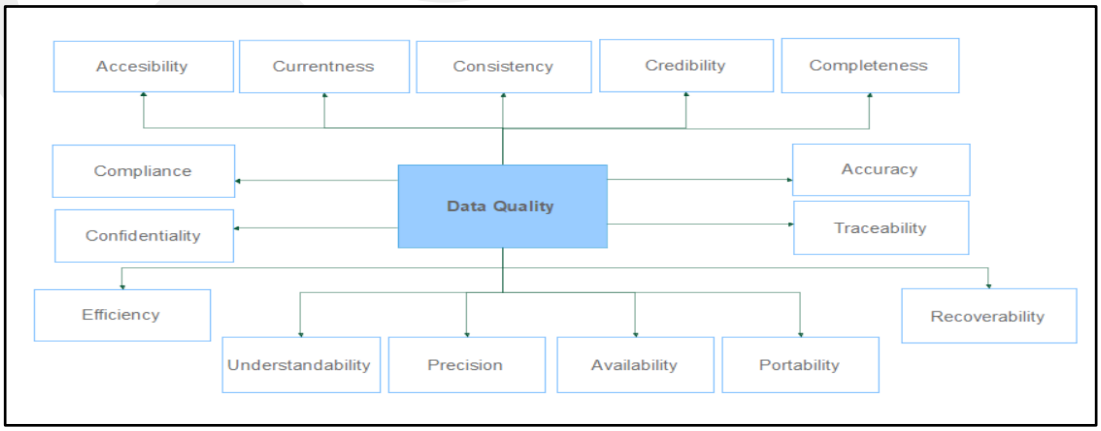


Figure 1.4 Data Quality Model [8]

1.2 Motivation

World Quality Report has been prepared by CapGemini, Sogeti and Micro Focus in the last decade. In the 2018-2019 report, the executive management objectives with QA and testing trend in IT industry for the years 2016-2018 were laid out, as shown in Figure 1.5.

This report was prepared through interviews with 1700 executives across 10 different sectors and 32 countries. The countries are mainly from North America, Western Europe, UK Ireland, Benelux, Eastern EU, Australia, China and Nordics.

This survey was carried out in accordance with eight executive management objectives shown in Figure 1.5. A decreasing trend in 2017 can be observed in the below graphics. Even though the percentages in the graph increased in 2018, quality related objectives are still considerably low. For example, end-user satisfaction is only around 40% while the detection of software defects before delivery to the end-user rate is less than 40%. Finally, an increase in the quality of software is around 40%. Therefore, the mapping of software quality metrics is a strategically important process for the increase in the quality of products and processes.

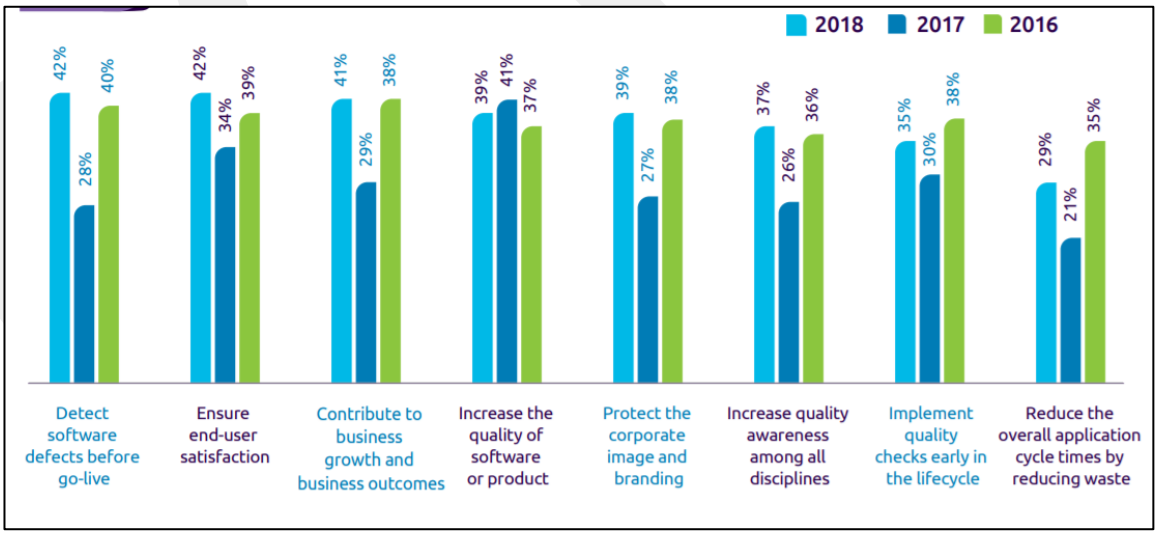


Figure 1.5. Executive Management Objectives with QA and Testing [9]

As in every field, competition for companies doing business in the field of software engineering has accelerated as a result of the rapid change and development brought about by the advances in information technology. With the increased competitiveness, the importance of agile working method has also increased.

In order to be able to compete in the industry, it is crucial to measure the current situation so that undesired situations can be abruptly prevented while at the same time continuous development can be enhanced. CMMI and PMBOK are the two internationally accepted guides in software engineering industry. The main focus of new versions of CMMI v2.0 model and the 6th edition of the PMBOK [10] (published in 2018), is the agility and continuous improvement. For this end, CMMI, IEEE, ISO and Aerospace related quality standards and process models agree in their emphasis on the necessity of measurement and process analysis to increase the quality of product, process and projects. For example, CMMI v2.0 includes a "Managing Performance and Measurement (MPM)" practice area which aims to manage performance by using measurement and analysis, in order to achieve business objectives. Within this context, CMMI v2.0 has defined the contribution of MPM process as maximizing business return on investment by focusing on management and improvement efforts on cost, schedule, and quality performance [1].

1.2.1 Necessity of Metrics and Measurement

Metrics and analysis of measurement results are good indicators for the progress of the projects, process and products and/or organizational processes. For instance, if metrics spot a deviation from the threshold in a negative manner, the flag will be raised abruptly to implement an emergency plan for intervention. In such cases, metrics function as life jackets for firms. They are so important that if we set the road map with the wrong set of metrics, these meaningless and incorrect metrics will most probably mislead us during the trip. In other words, if calibrated metrics are not defined in an efficient way to reach the target, most likely the pointer will show the incorrect data. Such cases where for example while the charts show things to be

going well, in reality the company may have gone bankrupt, are likely to occur under such conditions.

Besides, metrics also facilitate evidence-based decision-making about critical issues. Finally, as indicated in the trends World Quality Report presents, the current situation of the Software Quality Metrics is a topic that is treated with utmost importance in the international standards and process models. Given all these features, with its focus on systematic literature review of software quality metrics, this thesis aims to fill a gap in this area through mapping the quality metrics data and representing maturity level.

First step of the study is formulating systematic mapping and then analyzing the results obtained from them to generate SLR data. Definitions of the SM and SLR are given below:

- Systematic Mapping (SM) (also call as scoping study) is defined as a broad review of primary studies in a specific topic area that aims to identify what evidence is available on the topic [11].
- Systematic Literature Review (SLR) is defined as a review of a clearly formulated question that uses systematic and explicit methods to identify, select, and critically appraise relevant research, and to collect and analyze data from the studies that are included in the review. Statistical methods (meta-analysis) may or may not be used to analyze and summarize the results of the included studies [12].
- SLR study has emerged and spread from the field of medicine. Therefore, definition of the SLR is taken from the Cochrane Collaboration which is the organization dealing with the health interventions faced by health professionals.

1.3 Organization of the Thesis

The thesis is organized as follows:

Chapter 1 offers an introduction to the topic through focusing on the definition of software quality concepts, the motivation behind the selection of this topic and organization of thesis.

Chapter 2 focuses on the background data regarding the related papers, SLR and SM studies which are related to this thesis.

Chapter 3 provides the definition of the research method employed to conduct SM and SLR study.

Chapter 4 presents the results of this study with graphical representation.

Chapter 5 ensures the validity of this study by analyzing four (4) validity types: internal validity, construct validity, conclusion validity and external validity.

Finally, Chapter 6 is the conclusion part of this thesis where suggestions for further research are laid out by pointing to the gaps and improvement opportunities in the software quality metrics researches.

The References section was organized in a way that aimed to allow easy tracking of the sources. In order to avoid the confusion the readers might have regarding the main references of this thesis and other references used in SLR, references part only includes the main resources, list of standards and process models used in this research. Moreover, accessibility of the internet resource links was checked lastly on August 1, 2019; therefore, date of access for the entire internet sources is the same.

This study contains three appendices: Appendix A offers general information and access link to the systematic literature review repository, Appendix B includes the list of sources used in SLR study and Appendix C includes the Curriculum Vitae (CV) of the author of this thesis.

CHAPTER 2

BACKGROUND AND RELATED WORK

2.1 Related Work and Secondary Studies in Software Quality Metrics

To begin with, it should be noted that the literature review about software quality metric related studies yields no significant results. There are not any thesis works that are directly related to our thesis topic. It is possible to encounter some articles and conference papers on software quality metrics, mostly related to secondary studies like SM and SLR study.

The secondary study is defined as a study that reviews all the primary studies relating to a specific research question, with the aim of integrating/synthesizing evidence, related to a specific research question [11].

Software quality metrics related to SM and SLR studies and their content, scope and limitations are listed as below:

- In 2006, Parthasarathy and Anbazhagan [13] published a paper entitled "Analyzing the Software Quality Metrics for Object-Oriented Technology". This paper focuses only on object-oriented software related quality metrics. However, in this thesis we concentrate on the software quality metrics in longitudinal manner by focusing on the application domain and programming types, and hence analyzing the big picture instead of just a specific type of technology.
- In 2012, Rehman and Khan [14] published a paper entitled "SWOT Analysis of Software Quality Metrics for Global Software Development". This paper is about the search keywords, number of papers at the end of database research and process of SM study. This can be categorized as a preliminary study for SM and SLR.

- In 2016, Santos, Resende, Junior and Costa [15] published a paper entitled "Attributes and Metrics of Quality that Impact the External Quality of Object-Oriented Software: A Systematic Literature Review". Since it is written in Spanish, content and search keywords of the article cannot be analyzed. However, this paper analyzed quality attributes by using ISO\IEC 9126. In this thesis, all the quality models related to software quality metrics were searched in papers. Also, quality attributes mentioned in papers have been traced with the most up-to-date quality model ISO 25010 and quality attributes.
- Other SM and SLR studies related to software quality metrics are listed at the Table 2.1 and details are provided in the "SLR-SM Analysis" tab of the SLR repository, excel link for which is shared in Appendix A of this thesis.

Table 2.1 List of Secondary Studies

Type of Secondary Study	Title of Article/Conference Paper	# of Covered Studies	Covered Year	Ref
SM	A preliminary mapping study of software metrics thresholds	67	1970-2015	[16]
SLR	A systematic literature review for software sustainability measures	16	1/1/1992 - 31/12/2012	[17]
SLR	A systematic literature review on software measurement programs	65	1997-2014	[18]
SM	A systematic mapping review of software quality measurement: Research trends, model, and method	42	2007-2017	[19]
SLR	A systematic review of software maintainability prediction and metrics	15	1985-11/11/2008	[20]
SLR	A Tools Integration for Supporting Software Measurement: A Systematic Literature Review	12	Until 2015	[21]
SLR	Approaches for software metrics threshold derivation: A preliminary review	14	Until 2017	[22]
SM	Aspect-oriented software maintenance metrics: A systematic mapping study	138	1992-01/06/2011	[23]
SLR	A Report on the Analysis of Metrics and Measures on Software Quality Factors – A Literature Study	25	2007-2014	[24]
SLR	Construction of a Software Measurement Tool Using Systematic Literature Review	27	2007-2017	[25]
SLR	Design Quality Measurement for Service Oriented Software on Service Computing System: A Systematic Literature Review	15	2005-2019	[26]
SLR	Evaluation and measurement of software process improvement (SPI): A systematic literature review	148	1991-2008	[27]

SLR	External Quality Metrics for Object Oriented Software: A Systematic Literature Review	10	Until April 2015	[28]
SM	Measuring Software Process: A Systematic Mapping Study	462	1984-September 10, 2017	[29]
SM	Metrics and statistical techniques used to evaluate internal quality of object-oriented software: A systematic mapping	79	2004-2013	[30]
SLR	Metrics for analyzing variability and its implementation in software product lines: A systematic literature review	29	2007-10.01.2017	[31]
SM	Metrics for Software Reliability: An Initial Systematic Mapping Study	128	1980-2017	[32]
SM	Predicting change using software metrics: A review	21	1998-2009	[33]
SLR	Scalability , elasticity , and efficiency in cloud computing : a systematic literature review of definitions and metrics	20	2005-April 2015	[34]
SLR	Software fault prediction metrics: A systematic literature review	106	1991-2016	[35]
SM	Software metrics for measuring the understandability of architectural structures - A systematic mapping study	25	1990-June 2013	[36]
SLR	Software quality measurement in software engineering project: A systematic literature review	38	1984-2015	[37]
SLR	State-of-the-Art in Empirical Validation of Software Metrics for Fault Proneness Prediction: Systematic Review	29	January 1995-December 2012	[38]
SM	Systematic mapping study of quality attributes measurement in service oriented architecture	53	2007-2012	[39]
SLR	Using metrics in Agile and Lean software development - A systematic literature review of industrial studies	30	2000-December 2013	[40]

Depending on the average number of covered articles and papers in above mentioned studies shown in Table 2.1, it can be argued that in comparison to approximately 62 papers that were analyzed, the number of papers this study analyzes is 70.

In the majority of the SM and SLR studies, it is observed that usually only a very specific and narrow field about software quality metrics is investigated. Also, this study differs from others with its focus on the period between 2009 and 2019.

Overall, it can be concluded that all of these studies share a similarity with the subject of this thesis, but all are more specific studies related to systematic mapping study. The common focus of these articles and papers is on the specific fields of the software quality metrics as indicated in the titles. Compared to these earlier studies, our study focuses specifically on the title and abstract fields of articles and

conference papers about software quality metrics, with the aim of analyzing a larger scale of data. Research in the above table shows that the studies carried out in this area are on the rise at 2008. Moreover, after the release of PROMISE repository, which is an online data repository provided by School of Information Technology and Engineering, University of Ottawa, Canada in order to share datasets and models between software engineers [41], publication rate of metric related papers increased significantly between the years 2008 and 2009. To analyze the condition of the progress in this area after 2008, the period from 2009 to the current year is selected as the period of analysis for this SLR study.

Articles and conference papers, which were published in the last decades, related to software quality metrics were searched in our study. After the final version of the articles and conference papers repository was established, draft classification schema was developed and refined step by step. After classification schema and data extraction process, systematic mapping and literature review are conducted in more detailed manner. Significant results and expectations regarding the future are highlighted on the paper and also this data is imported both to the SLR Repository in excel format and to Google Drive.

SM data analysis and synthesis process were carried out to reach SLR results.

In view of these, at the end of this study following contributions will be provided:

- A SM and literature review of the software quality metrics/measurement
- The link of the online article and conference papers which includes the raw data of our study.

This study is a secondary study in the context of Software Quality Metrics. Therefore it assumes that readers have a degree of familiarity with the software quality and software quality metrics terms to be able to follow the ensuing discussion originating from this study. For this end, brief information about the software quality, software quality metrics and their relation to the research questions of this study will be provided in the following chapter.

CHAPTER 3

RESEARCH METHOD

In the following part firstly an overview of our research method and secondly the goal and research questions of our study are presented. The SLR process principles which are proposed by Kitchenham and Charters are applied here as the research method, [42].

The SLR process has the following main steps:

1. Overview
2. Definition of Goal and Research Questions
3. Conduct Search and Paper Selection
4. The final pool of articles and the online repository
5. Data Extractions
6. Data synthesis

Application of these steps to Software Quality Metrics SLR study is outlined in the following section.

3.1. Overview

Tool selection criteria and methods for installing SLR study infrastructure are presented in Section 3.1.1. Following this, developing and evaluation of review protocol establishment data is presented in Section 3.1.2. Study will be conducted and reported step by step as offered in following outline:

3.1.1. SLR Tool Research and Selection

Before starting the SM and SLR study, preliminary research was conducted to establish the infrastructure of study and hence ensure that the study will produce

healthy and verifiable results. There are several software tools for conducting SLR. A comparison of these tools in accordance with various criteria is given in Figure 3.1.

SLR tool selection checklist for this study has eight (8) criteria, listed as follows:

1. Availability (Open Source)
2. Ease of installation (online tool, downloadable exe installation file)
3. Intended research field: Software Engineering
4. Offline Work Support, 7/24 works
5. Multiple authorship, opportunity to work simultaneously
6. If search keyword is enquired in different search engines, an overlap of papers is quite likely to occur. To avoid the inclusion of duplicate papers in the SLR paper pool, there is a need for automatic elimination of duplicate data.
7. Text Mining Capability
8. Presentation of results with diagrams

According to tool selection criteria and web search literature results, three (3) tools are selected for test before use. MDX SLR Tool is obtained through internet sources. CADIMA and ALARCOS SLR Tool are analyzed with the help of the information in paper [43] and comparison table in Figure 3.1. Other free SLR Tools are MDX SLR Tool, ALARCOS SLR Tool and CADIMA have been tested before carrying out this study.

Software name	Setting up the review	Scoping/pilot study	Literature searching	Duplicate checking	Article screening	Data coding	Critical appraisal	Synthesis	Documentation
E.g.	Facilitation of question formulation and/or stakeholder engagement	Protocol development, PICO* elements specified	Software integrated with publication databases	Automated marking of duplicates	For study selection	Tagging and extraction to support meta-analyses	Risk of bias assessments	Facilitates quantitative/ qualitative syntheses of results	Output of text, figures or tables to assist with report writing
CADIMA									
Colandr									
Covidence									
DistillerSR									
EROS									
EPPI-Reviewer 4									
HAWC									
METAGEAR package for R									
PARSIFAL									
Rayyan									
REviewER									
RevMan 5									
RevMan Web							Data unavailable		
SESRA									
SLR-Tool									
SLuRp									
SRDB.PRO									
SRDR									
StArt									
SUMARI									
SWIFT-Review									
SyRF									
TOTAL	5	10	13	11	20	19	12	15	13

Figure 3.1 SLR-SM Tools Comparisons for Some Criteria [43]

MDX SLR Tool is available on the web to everyone, but there is a huge disadvantage that stems from the fact that the tool is not stable and not reachable all the time. On the other hand, ALARCOS SLR Tool, developed by ALARCOS with Java, is a downloadable desktop application. This tool has some constraints about system requirements. For instance, it only works under Windows XP or Vista operating system and MySQL Server 5.1. So this tool is not available for all platforms and it is not up-to-date. Also, there is some missing data about the system compatible database version of MySQL Server in user manual of the tool. When it was downloaded to Windows XP on the virtual machine, the tool could be opened, but database connection could not be maintained in an efficient way. Besides, there is no data about the sub-version of the database. Therefore, these tools are eliminated because of the problems of reliability, availability and portability issues. CADIMA, an online SLR-SM analysis tool and Google Drive combination are employed for this SLR search. Last version (v2.1.1, May 2019) of the CADIMA tool is accessible

from the web and it is free. In comparison to other SM-SLR software, CADIMA has lots of advantages such as automatic elimination of duplicated papers, bulk uploading to the software and automatic matching to facilitate reaching and screening papers, thus allowing criteria and sub-criteria definition to synthesize papers and finally extract the data as excel and flow diagram. Moreover, this software allows multiple authors to reach the SLR Repository at the same time, so two authors can simultaneously screen and score papers. After that, results of scored papers are combined by the software.

After these tests of SLR tools and the reading of the explanations in Figure 3.1 were evaluated, CADIMA tool and Google Datasheet, Google Drive are selected for the infrastructure of this thesis study. With the help of the CADIMA tool, work flow compatibility with SM-SLR study protocol is maintained since CADIMA tool enforces users to comply with these steps to complete the study.

Selected infrastructure and usage purpose of tools are given as follows:

- CADIMA: Uploading the list of selected pool after the first search, eliminating duplicate papers, eliminate papers based on title and abstract, then bulk uploading all PDF for remaining papers, eliminating papers based on full text, entering SM criteria and realizing SM study following the paper elimination, generating flow diagram of paper elimination process as seen at
- Figure **3.9**.
- Google Data Sheet: After carrying out SM study to produce SLR study, excel is generated from CADIMA tool and this excel is added into Google Drive with the format of Google sheet.
- Google Drive: Results of SM and SLR study, SLR repository, and downloaded e-source of article and conference papers will be shared with everyone. In this way, anyone who wishes to learn about the trend in recent years or work on it can have an easy access.

Some screenshots from CADIMA tool are given in Figure 3.2 and Figure 3.3.

In Figure 3.2, bulk upload of PDF files to the tool is shown as the tool is automatically matching the PDF with the title of the paper. In this way, the need for searching and finding the document from the computer for each study, is resolved.

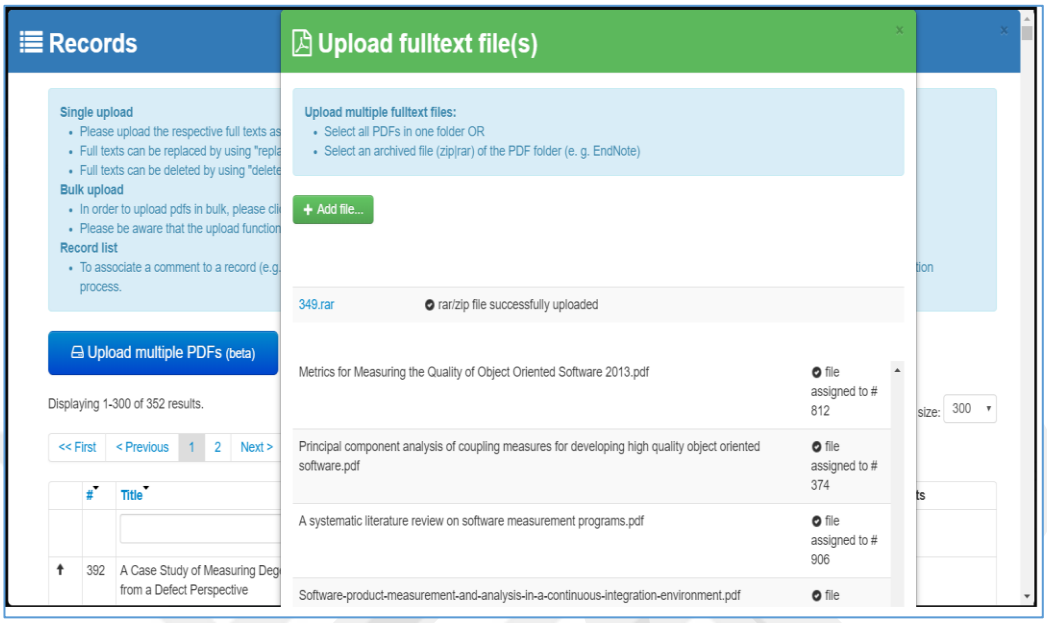


Figure 3.2 The Screenshot of Upload File Module of CADIMA Tool

Paper elimination-selection activity according to full-text has been realized as shown in Figure 3.3.

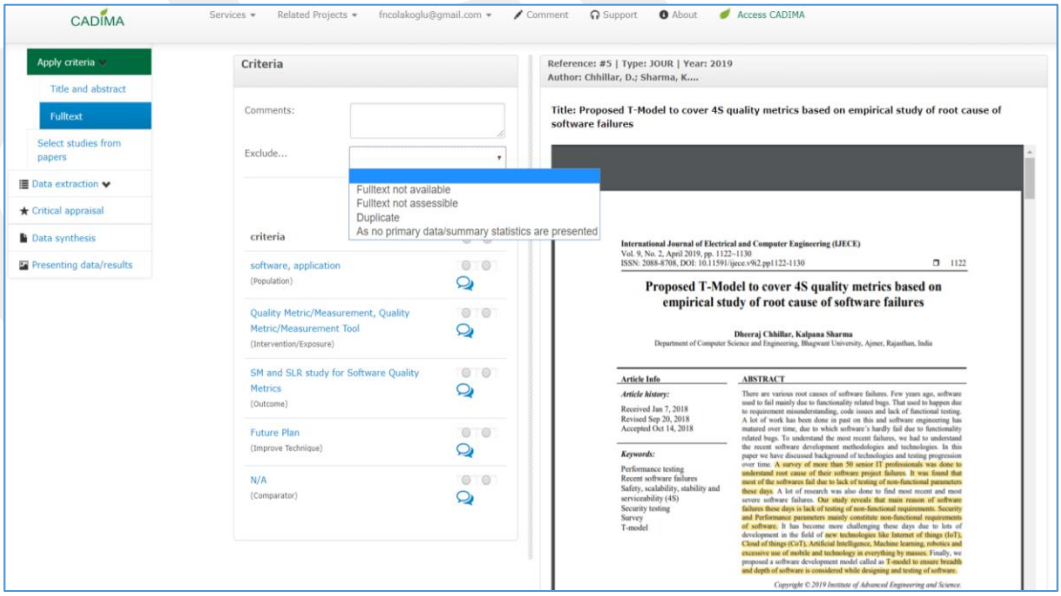


Figure 3.3 The Screenshot of CADIMA Tool

While carrying out the SM study with the help of this free tool, some improvement opportunities or minor bugs that were observed are listed below:

- In the elimination of the duplicate papers, there are three filters such as title, year, and author. More filters could be added.
- When “eliminate duplicate papers” button is clicked, tool eliminates both papers, as a result of which duplicate papers have to be added to the study paper pool manually. This is very time consuming especially when the study includes great number of papers.
- Sometimes, abstracts of the papers cannot be imported in the tool automatically, so abstract of the paper needs to be added manually, too. When abstract is added, some characters such as “n”, “u” are lost. These characters are the shortcuts of the tool. In such cases, some of the statements become unreadable.
- After having defined paper selection criteria or eliminate the paper options once, there is no chance to update the paper selection criteria again. Also, if you eliminate the paper inadvertently, there is no chance to turn back and take paper into selected paper pool again.
- Screen of the tool sometimes freezes for a short while.
- Comment adding field does not have a scrollbar, so reading comments is not easy.

These findings are reported and shared via e-mail (poststelle@julius-kuehn.de) with the Julius Kühn-Institute Federal Research Centre which is the developer of the tool. If this Institute corrects these findings, this will certainly contribute to the studies of future researchers to conduct their studies in a more effective and easier way, and these findings will work to enhance the human-computer interface of the tool for the positive.

3.1.2. Developing and Evaluating the Review Protocol

In Figure 3.4 The Review Protocol for this SLR study, mainly used by academic search engine databases (Google Scholar, Scopus and OpenAIRE) is marked with dark blue to distinguish it from the others. Other academic search databases were

also searched to ensure that there is no missing paper related to this SLR study. Figure 3.4 is drawn with iMindMap tool v10, which is the tool for generating Mind Mapping, Brainstorming and Project Planning Software, used for illustrating the software quality metrics methodology [44].

Following the identification of the need for the reviews of this SLR study, research questions for SM and SLR study are generated. The review protocol given in Figure 3.4 is used for conducting SM and SLR studies. Software engineering adapted systematic mapping process in paper [45] is used as a reference in the drawing of the review protocol for this study. The process starts with the article and conference paper selection by using search keywords and inclusion/exclusion criteria. Second step of the process is the classification of the papers based on selected categorization to answer the research questions. At the end of the data classification, SM data is baselined to generate synthesis for SLR study.

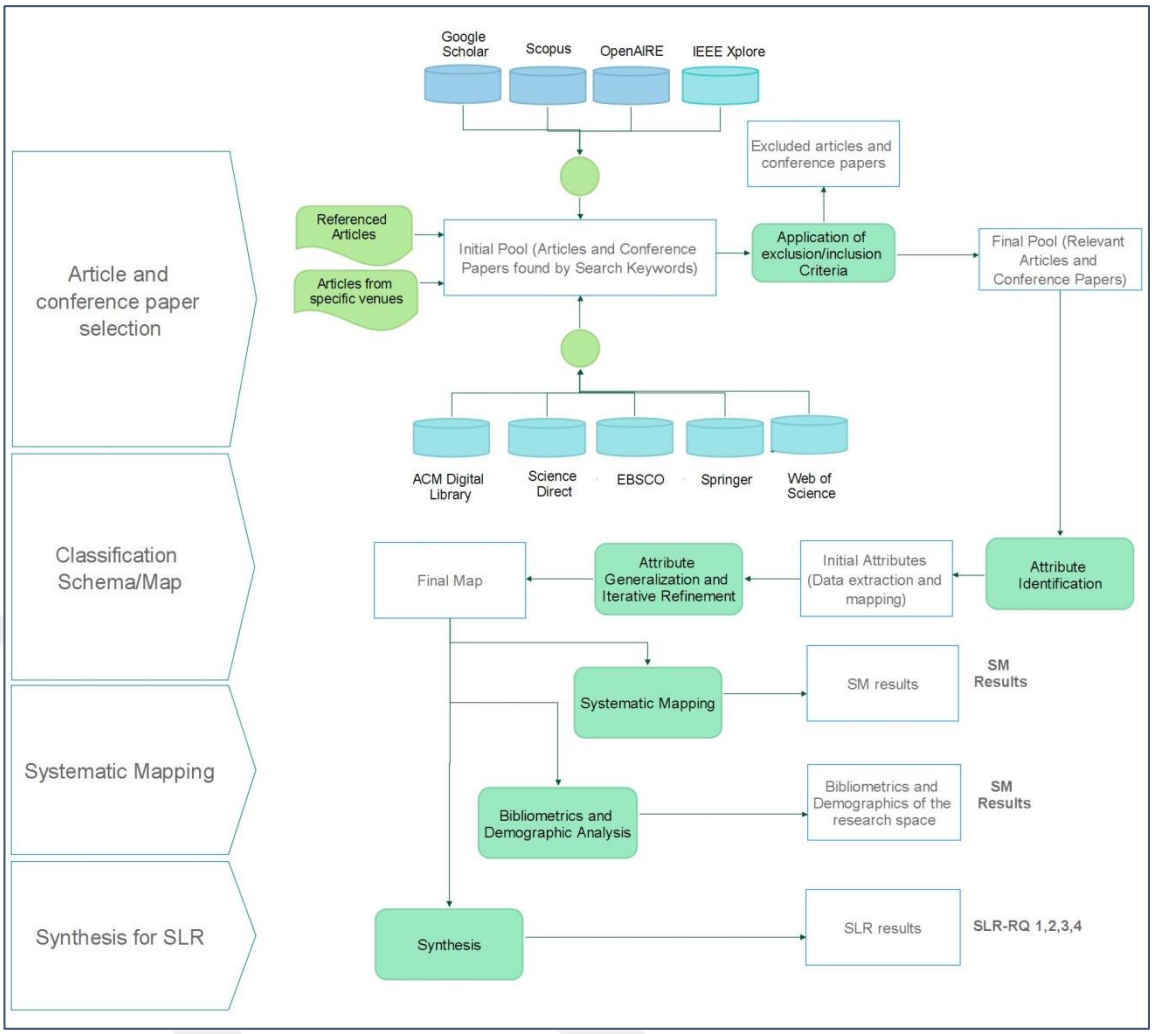


Figure 3.4 The Review Protocol for this SLR Study

3.2. Definition of Goal and Research Questions

The main goal of our study is to extract and analyze statistical data from the collection of software quality metrics published in the last decade. To reach this goal, we classified our questions into two main research questions for SM study:

1) Systematical review questions related to technical issues and trend

2) Bibliometric and demographic analysis questions

SM-RQ 1: Bibliometric and Demographics of the Publications

SM-RQ 1.1. What is the number of papers published per year?

SM-RQ 1.2. What is the type of the papers published?

SM-RQ 1.3. What is the affiliation of authors?

SM-RQ 1.4.Under which subject area are papers published or categorized by the publishers and search engines?

SM-RQ 1.5. What is the article distribution over countries? What are the most active countries in software quality metrics studies?

SM-RQ 1.6. Which are the top cited papers in the last decade?

SM-RQ 1.7. What is the relation between citation count and download count of the papers?

SM-RQ 1.8. What is the average number of the authors who conduct research and publish the papers?

SM-RQ 1.8. Who are the most active researchers in the software quality metrics area?

SM-RQ 1.10. What is the name of the top publishing venues?

SM-RQ 2: Systematic Mapping: Trends and Technical Data

How are papers in Software Quality Metrics designed and reported?

SM-RQ 2.1: What type of research methods/facets are used in the papers?

SM-RQ 2.2: What type of research approaches are used in the papers?

SM-RQ 2.3: Whether authors of papers used case study or not, to explain the metrics, method, research results, tool usages etc.? Which papers explain the metrics with case study by using which datasets or projects?

SM-RQ 2.4: What is the level of ease of understanding of the software quality metrics presented in the papers?

SM-RQ 2.5: Which statistical model/methods are used to validate or generate new metrics?

SM-RQ 2.6: What are the threats to the validity of the results of the papers?

What are the trends of software quality metric related papers in the last decade?

SM-RQ 2.7: Which standards and process models have been used / referenced from papers to measure software quality? What are the most popular software quality standards and process models mentioned in the literature in the last ten years?

SM-RQ 2.8: Which quality models have been used in papers to measure software quality? What are the most popular quality models mentioned in the literature in the last ten years?

SM-RQ 2.9: Which types of process, product, and project metrics are mostly focused on measuring quality?

SM-RQ 2.10: In which level software quality metrics have been measured?

SM-RQ 2.11: How many papers mentioned the tool usage for measuring software quality?

SM-RQ 2.12: Which quality attributes of "Software Product Quality (SPQ)" in ISO 25010 Quality Model is mostly measured in the papers?

SM-RQ 2.13: Which quality attributes of "Quality in Use (QinU)" in ISO 25010 Quality Model is mostly measured in the papers?

SM-RQ 2.14: Which quality attributes of "Data Quality (DQ)" in ISO 25012 Quality Model is mostly measured in the papers?

SM-RQ 2.15: Which of the SDLC stages are mostly referred in the data in the papers?

SM-RQ 2.16: Which application domains have been subject to software quality metrics in articles?

SM-RQ 2.17: Which software programming types have been subject to software quality metrics in articles?

SM-RQ 2.18: Are there any software quality metrics specific to SDLC model?

SM-RQ 2.19: Whether a threshold value is mentioned or not? Which papers mentioned the threshold values of metrics?

SM-RQ 2.20: What are the future directions of current researches?

Table 3.1 gives the information about the classification schema for each RQ by presenting the list of options per each RQ, permission of multiple/single selection from option lists and presentation type of the results like string, integer etc.

Table 3.1 Classification Scheme developed in our study

RQ Number	Attribute	Possible Types	(M)ultiple / (S)ingle Selection
SM-RQ 1: Bibliometric and Demographics of the Publications			
SM-RQ 1.1	Published paper count per year	Count: Integer	-
SM-RQ 1.2	Paper Types	{Article, Conference Paper}	S
SM-RQ 1.3	Affiliation of Authors	{Academic, Industry, Both}	S
SM-RQ 1.4	Subject Area of Papers	{Computer Science, Engineering, Computer Engineering, All, Others}	M
SM-RQ 1.5	Active countries (Top 10 country)	Countries: String[]	-
SM-RQ 1.6	Top-Cited Papers (Most popular article and conference paper titles)	Citation Count: Integer or Double Normalized Citation Count: Integer or Double Normalized Citation Count per Papers: Integer or Double	-
SM-RQ 1.7	Citation Count vs. Download Count of Papers	Count: Integer	-
SM-RQ 1.8	Size of the author team	Team Size: Integer	-
SM-RQ 1.9	Most Active Researchers	Researchers: String[]	-
SM-RQ 1.10	Top Venues	The target of Paper: String[]	-
SM-RQ 2: Systematic Mapping Questions related to technical issues and trends			
SM-RQ 2.1.	Research Facets/Methods	{Analytical, Empirical, Others}	M
SM-RQ 2.2	Research Approach	{AHP (Analytical Hierarchical Process), Fuzzy, Novel, GQM (Goal-Question-Metric), PSM (Practical Software Measurement), Others, N/A}	M
SM-RQ 2.3	Case Study	{Yes, No}	S
SM-RQ 2.4	Ease of Measurement	{Easy: Simple to counting (basic calculations, all variables-methods determined). Medium: Requires some effort to understand and calculate metrics Difficult: Difficult for understand and calculate; because of the complexity of the terms and concepts.}	S
SM-RQ 2.5	Statistical Model/Method	{ Logistic Regression, Machine Learning Meta-Analysis, Descriptive Analysis, Correlation Test, Genetic Algorithm,	M

		Swarm Optimization, N/A, Others}	
SM-RQ 2.6	Metric and Technique Validation Data	{0: No validation data 1:self-constructed examples used, 2: validated by small data 3: validated by medium/large data and/or validate by tool 4: Proposed by one author and validate by another author or research team and/or automated tool with huge data.}	S
SM-RQ 2.7	Standards, Process Models	{ISO 15504, ISO\IEC\IEEE 12207, IEEE 1061, CMMI, ISO/IEC 15939, Others (ITIL, COBIT, ISO 9001 etc.), N/A}	M
SM-RQ 2.8	Quality Model	{Mc Call, Bohem, Dromey, ISO 9126 ISO 25010, Others, N/A}	M
SM-RQ 2.9	Type of Metric	{Product, Process, Project, All}	M

RQ Number	**Attribute**	**Possible Types**	**(M)ultiple / (S)ingle Selection**
SM-RQ 2.10	Metric Level	Func-Req, Non-Func. Req, Class Level, Directory Level, File Level, Method Level, Variable Level, Design, Test, Project Mang. Level, Process Level, Others	M
SM-RQ 2.11	Metric Tool Usage Data	{NTS: No Metric Tool Support, TS: Metric Tool Support, Others (mentioned about tool but not used, awareness of tool usage)} {Name of the Metric Tools}	S
SM-RQ 2.12	Software Product Quality (ISO 25010 Quality Model)	{Functional Suitability, Performance Efficiency, Reliability, Usability, Compatibility, Security, Maintainability, Portability, Others}	M
SM-RQ 2.13.	Quality in Use (ISO 25010 Quality Model)	{Effectiveness, Efficiency, Satisfaction, Freedom from risk, Content Coverage, Others}	M
SM-RQ 2.14	Data Quality (ISO 25012 Quality Model)	{Accessibility, Currentness, Consistency, Credibility, Completeness, Accuracy, Traceability, Recoverability, Portability, Availability, Precision, Understandability, Efficiency, Confidentiality, Compliance, Others}	M
SM-RQ 2.15	SDLC Phase	{Planning, Requirement, Design, Code, V&V, Maintenance, All, N/A}	M
SM-RQ 2.16	Application Domain	{Embedded Software, Web Application, Mobile Application, ,Artificial Intelligence Application, Safety Critical App, Generic, Others, N/A}	M
SM-RQ 2.17	Software Programming Type	{Object-Oriented Programming., Aspect-Oriented Programming, Scripting Programming, Basic Programming, Others, N/A}	M
SM-RQ 2.18	SDLC Model	{Waterfall, Iterative, Agile, Product Line, Others, N/A}	M
SM-RQ 2.19	Threshold Info/Value	{Yes, No}	S
SM-RQ 2.20	Future Plan	{Develop a New Tool, Improve The Tool, Develop a new Technique, Improve the Technique, Make More Detailed Research (New Case Studies), No Future Plan}	M

In this study, papers are categorized according to ISO 25010 series quality model since it is the latest quality model updated and published in accordance with the necessities and comments of the users. In the cases where authors of the papers used a different quality model, then, ISO 25010 quality attribute categorization is employed via analyzing the specialties and contribution of metrics to the quality attributes.

SM study provides input for the SLR study. SM provides a categorization of conference papers and articles which are related to SQM. With the help of SM, researches can visualize a large mapping at a single glance to initiate their study on software quality metrics. After mapping software quality metrics in the related conference papers and articles, we conduct the SLR study. SLR study provides reliable data for decision making as well as serving as the starting point for the other studies related to the topic.

Based on this research goal we derived four (4) SLR research questions as follows:

SLR-RQ 1: What is the software quality metrics presented in the papers?

> SLR-RQ 1.1. Which metrics are presented under which metric levels and which are commonly mentioned in the papers?
>
> SLR-RQ 1.2. Which metric levels can be applicable for which application domain?
>
> SLR-RQ 1.3. What metrics can be applicable for which programming type?

SLR-RQ 2- Which quality characteristic of ISO/IEC 25010 Quality Model measured mostly for satisfying Product Quality (SPQ), Quality in Use (QiU), and Data Quality (DQ)?

SLR-RQ 3 - Which software quality metrics/measurement management tools are focused on the articles and conference papers?

SLR-RQ 4 - Which areas in software quality metrics/measurement require further research?

3.3. Conducting Search and Paper Selection

Paper selection activity consists of the following steps:

- Source selection and search keywords
- Application of exclusion and inclusion criteria.

These steps will be explained in the following section.

3.3.1 Source Selection and Search Keywords

3.3.1.1. Source Selection

Based on the SLR guidelines, the following nine (9) major electronic academic search engines, often referred to by the software engineers, are selected (Table 3.2):

Table 3.2 The List of Search Sources

Search Sources	1	Google Scholar	https://scholar.google.com/
	2	ACM Digital Library	https://dl.acm.org/
	3	IEEExplore	https://ieeexplore.ieee.org/Xplore/home.jsp
	4	Springer	https://link.springer.com/
	5	ScienceDirect	https://www.sciencedirect.com/
	6	Scopus	https://www.scopus.com/search/form.uri?display=basic
	7	OpenAIRE	https://explore.openaire.eu
	8	EBSCO	http://web.a.ebscohost.com/ehost/search
	9	Web of Science	http://apps.webofknowledge.com/

One can use various numbers of search engines to search for a scientific study related to a thesis subject. A scientific document may be provided by more than one content provider, and in such cases searching with multiple keywords from multiple search engines becomes very time consuming.

There are many search engines for academic studies. Firstly, the search engines which are mostly related to software engineering, computer engineering and

computer science were utilized. When we analyzed the computer science related academic journals, theses, conference papers, systematic literature reviews, we found out that the nine (9) databases that are listed above were widely referred.

Consequently, we needed to make a second elimination to accelerate our study. In the second elimination, the main criterion was the coverage of the database. Many of the extensive databases already cover all major academic search databases. For example; Scopus and Google Scholar cover Springer, ACM Digital Library, IEEExplorer, ScienceDirect, EBSCO, and therefore it is possible to have a detailed search by using Scopus, Google Scholar and OpenAIRE.

Academic search engines selection criteria were determined in accordance with the type of this thesis (SM and SLR) as well as the goal, scope, and technical data needed for this thesis.

Academic search databases were selected in accordance with the following criteria:

- **Focus field:** Given the main focus of this study, naturally we aimed to reach academic search engines related to "Software Engineering, Computer Engineering and Computer Science"
- **Publication year coverage:** Publication year coverage is quiet important in this respect because it allows the tracking of the evolution of the data.
- **Systematic Literature Review:** Due to the stylistic qualities of this thesis, similar studies about SLR and SM were searched in order to analyze the most frequently used search engines for literature review.
- **Publication Type:** Due to the inclusion criteria formed for this thesis, the scope is limited with conference papers and articles related to software quality metrics. Hence, priority was given to the search engines which among their publication types include conference papers and articles. Also, in order to extend the scope, further attention was diverted to interviews, reviews, reports and books.
- **A number of Accessible Sources:** In order to ensure the validity of SLR and prevent missing data about software quality metrics, it was crucial to include significant number of sources.

- **Open Accessibility:** One of the important inclusion and exclusion criteria in this thesis is “Are articles and conference papers electronically available?”. For detailed information about inclusion and exclusion criteria of this thesis, refer to Section 3.3.2.
- **Covered Database/Library:** Reaching accurate, consistent and the largest dataset needs to be time-saving, too; and therefore, it is important to work with an academic search database which encompasses several, authenticated databases.

According to Table 3.3 and the review presented to evaluate the suitability of Google Scholar as a source of scientific information [46], it can be observed that Google Scholar has significantly expanded coverage rate of databases through years. This makes Google Scholar an authoritative database that can be consulted for the scholarly research. Unfortunately, in Google Scholar database, duplicated records and source titles appear. Therefore, a number of citation counts and sources given by Google Scholar do not reflect the reality. Considering these, Scopus and OpenAIRE databases are also investigated in addition to the Google academic database, to eliminate any drawbacks that might stem from this inadequacy [46].

Scopus and Elsevier are scientific peer-reviewed journal abstract and citation databases [47]. According to the agreement between Elsevier and Google Scholar, it is possible to search an article in the Elsevier database in Google Scholar, but it cannot be downloaded since it requires payment, which means Google Scholar does not contain the content. Furthermore, since there is no agreement between Scopus and Google Scholar, one cannot search for an article scanned by Scopus citation database in Google Scholar [48]. In order to overcome these inadequacies, we employed both Google Scholar and Scopus during this study.

OpenAIRE has not been used before for software engineering related SLR and SM. When the efficiency of OpenAIRE database was inquired, the following information was obtained:

OpenAIRE (https://explore.openaire.eu) is a database which was funded by the EU Horizon 2020 Research and Innovation Program under Grant Agreement No.777541.

This database contains 25.962.283 Publications, 1.130.919 research data and 2.753.668 project's documents. Those are provided by 14.156 content providers and 18 funders [49]. Due to this richness, OpenAIRE database was also included in this study. The list of content providers for journals can be reached from the link https://explore.openaire.eu/search/journals. Other content providers' number is 1.016 the link is https://explore.openaire.eu/search/content-providers. The number of content providers as registries is 30; the link of their list is https://explore.openaire.eu/search/entity-registries. The link of data providers that has 15.293 providers is https://explore.openaire.eu/search/find/dataproviders.

Table 3.3 Academic Search Engines Selection (May,26 2019)

SEARCH DATABASE / CRITERIA	**Google Scholar**	**OpenAIRE**	**Scopus**
Focus Field	Software Engineering, Computer Engineering, Computer Science	Software Engineering, Computer Engineering, Computer Science	Software Engineering, Computer Engineering, Computer Science
Publication Year Coverage	Theoretically all available electronically	Theoretically all available electronically	1970-Present
SLR, SM References	Yes	No	Yes
Publication Type	The article, Thesis, Conference Papers, Books, Interview, Report	The article, Thesis, Conference Papers, Books, Interview, Report, Projects, Newspaper and weekly magazine, visual material	The article, Thesis, Conference Papers, Books, Interview, Review, Tools, Short Surveys
Number of Accessible Sources	Unknown (Huge number of source accessibility but duplicate artifacts also included)	25.962.283 Publications, 1.130.919 research data, 2.753.668 project's documents	Over eight million conference papers, Over 21,950 peer-reviewed journals, "Articles-in-Press" from over 8,000 journals
Open Access	Yes	Yes	Yes
Updated Daily	Unknown	Yes	Yes
Covered Database/ Library	Elsevier, Google Books, IEEExplore, JSTOR,	IEEExplore, ACM Digital, EBSCO, arXiv.org, Research Gate, Elsevier	IEEExplore, Elsevier, Science Direct, Wiley Online Library, Springer, Research Gate, ACM

	SpringerLink, Wiley Online, Library, Citeseer Library, ACM Digital Library, ResearchGate		Digital Library, Oxford University Press, Cambridge University Press

Scopus, Google Scholar and OpenAIRE database analyze results according to the selected criteria, given at Table 3.3.

In addition, we have to cross-check to ensure the reliability, quality and coverage rate of Google Scholar, Scopus and OpenAIRE. Therefore, we searched other six (6) databases with the search string given in Section 3.3.1.2 to ensure that these three databases include all conference papers and articles about software quality metrics. When we compared the search results between Google Scholar, Scopus, OpenAIRE and other six search sources (Table 3.2) for the given search string, we found out that there are some minor differences. Only 1 additional article is found in the EBSCO database, 12 conference papers in IEEEXplore and 4 articles in ScienceDirect. These papers were also added into our paper pool.

3.3.1.2 Search Keywords

SLR study is used mostly in the medical research area, so we used PICO selection criteria (Table 3.4) before searching dataset to understand the target population, problem, intervention method, alternative methods for intervention and expected outcomes following the application of the methods [50]. CADIMA tool required the PICO criteria before the selection and elimination of the papers.

Table 3.4 PICO Selection Criteria

Key element	Definition	Criteria
Population (P)	What is the target population for research?	Software
Intervention (I)	In which aspects, do you wish to research to intervene to the population?	Quality Metric(s)/Measure(ment), 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 and SLR study for Software Quality Metrics

In order to conduct keyword search, a search string was generated as shown in Table 3.5.

Table 3.5 Search Keywords

Keywords	Title and Abstract Field: ((“Software” AND/OR “Quality”) AND (“Metric” OR “Metrics” OR “Measure” OR “Measurement”))
	Name of tools calculation and analyzing Major Software Quality Metric

3.3.2 Application of Inclusion/Exclusion Criteria

We apply the following inclusion/exclusion criteria (Table 3.6) to the initial pool:

1. **Topic Relevance:** Are articles and conference papers relevant to Software Quality Metric?
2. **International Language:** Are articles and conference papers in the English language?
3. **Article and conference paper:** Are articles and conference papers electronically available?

Table 3.6 Inclusion and Exclusion Criteria

Publication Language	English
Publication Type	Article and Conference Paper
Relevant with Topic	Yes/No Selection
Relevant with Research Questions	Yes/No Selection
Open Access (Full text is available electronically)	Yes/No Selection
Publication Date	01.01.2009-31.07.2019
Publication has citations.	2009-2018, citation count>0 For year 2019, citation count 0 (zero) is acceptable; because newly published papers can be get citation yet.

We inspected the title, keywords, abstract of the articles and conference papers to analyze the relevance level of articles with software quality metrics. If sufficient information was not available in these sources, table of contents and some parts of the articles were explored in depth. CADIMA tool has a feature of allowing the

implementation of inclusion/exclusion criteria based on the voting of the researchers. Voting scale for topic inclusion/exclusion criteria is between 0-3. “0” point indicates the strong opinion for excluding the articles and conference papers, whereas giving “3” points refer to the strong opinion for the inclusion of articles and conference papers. To increase the reliability of the voting articles and conference papers mechanism and final results, articles and conference papers were first voted by the researcher and then reviewed by the supervisor. Based on the results of the joint voting, excluded articles and conference papers were moved to “excluded” excel spreadsheet in Google Docs system. Also, downloadable links of the e-articles and conference papers were given in the SLR paper pool repository in Google Drive.

The SLR synthesis table can be reached via the link given in Appendix A and several screenshots of the table are given in Figure 3.5-Figure 3.8.

The bibliometric and the demographics data of the papers were determined in accordance with the SM-RQ 1.1 to SM-RQ 1.10 questions as shown in the Figure 3.5.

Number	Name of Article/Conference Paper	Link	Venues	Year	Article or Conference Paper?	Subject Area	Authors	# of authors	Author 's affiliation	Authors' Countries	Citation Count	Full Text Views	Venues Abb.
1	A Coupling and Cohesion Metrics Suite for Object-Oriented Software	https://ieeexplore.ieee.org/document/5359706	International Conference on Computer Technology and Development	2009	Conference paper	Computer Science	S. Husein; A. Oxley	2	Academic	Malaysia	8	117	ICCTD
2	A Critical Survey of Security Indicator Approaches	https://ieeexplore.ieee.org/document/6329197	International Conference on Availability, Reliability and Security	2012	Conference paper	Engineering	M. Rudolph; R. Schwarz	2	Academic	Germany	7	425	ARES
3	A Quality-Based Requirement Prioritization Framework Using Binary Inputs	https://ieeexplore.ieee.org/search/searchresult.jsp?newsearch=true&queryText=10.1109%2Fams.2010.48&SID=Elsevier:Scopus	Asia International Conference on Mathematical/Analytical Modelling and Computer Simulation	2010	Conference paper	Others	C. E. Otero; E. Dell; A. Qureshi; L. D. Otero	4	Academic	United States	13	346	AMS
4	A Quantitative Approach to Software Maintainability Prediction	https://ieeexplore.ieee.org/document/5635174	International Forum on Information Technology and Applications	2010	Conference paper	Computer Science	L. Ping	1	Academic	China	18	537	IFITA
5	A Suite of Object Oriented Cognitive Complexity Metrics	https://ieeexplore.ieee.org/document/8253447	Journal of IEEE Access	2018	Article	All	S. Misra; A. Adewumi; L. Fernandez-Sanz; R. Damasevicius	4	Academic	India, Litvania, Spain, Nigeria	7	704	

Figure 3.5 Screenshot for Bibliometric and Demographics Data

Research facet, research approach, standards, process models, quality model, etc. of the papers were determined as Figure 3.6 to answer questions SM-RQ 2.1, 2.2, 2.7, 2.8, 2.9, 2.10 and 2.11.

Number	Name of Article/Conference Paper	Research Facet	Research Approach	Standards, Process Model	Quality Model	Type of Metric	Metric Level	Metric Tool Usage?	Tool Name
6	A mapping study to investigate component-based software system metrics	Others	GQM	N/A	N/A	Product	Class Level	Others	No
7	A model for early prediction of faults in software systems	Empirical	N/A	N/A	N/A	Project	Method Level	NTS	No
8	A new metric for predicting software change using gene expression programming	Empirical	Novel	N/A	N/A	Product	Class Level	TS	Understand for C++
9	A study of the relationships between source code metrics and attractiveness in free software projects	Empirical	N/A	N/A	N/A	Project	Method Level	TS	Analizo, CCC, Cscope, Eclipse-Metrics, Macxim/Spago4Q
10	An AOP-based approach for collecting software maintainability dynamic metrics	Analytical	AHP	N/A	N/A	Project	Class Level	NTS	No

Figure 3.6 The Screenshot of Technical Data Table-1

The technical data of papers based on eleven criteria are determined as shown in Figure 3.7 and Figure 3.8 to answer questions SM-RQ 2.3, 2.4, 2.5, 2.6, 2.12, 2.13, 2.14, 2.15, 2.16, 2.17, 2.18, 2.19 and 2.20.

Number	Name of Article/Conference Paper	RQ 1										
		SPQ	QinU	DQ	Type of Quality Attribute	Validation	Ease of Measurement	Case Study	Statistical Method/Model	SDLC Phase	Application Domain	Programming Type
11	An Improved Fuzzy Synthesis Evaluation Algorithm for Software Quality	Yes	Yes	No	Context Coverage, Security, Efficiency, Usability, Realiability, Maintainability, Portability	2	Medium	Yes	N/A	All	Safety Critical App.	N/A
12	An empirical investigation of modularity metrics for indicating architectural technical debt	Yes	No	Yes	Maintainability, Completeness, Tracebility	4	Easy	Yes	Correlation Test	Design	Generic	Object-Oriented SW
13	An empirical study on object-oriented metrics and software evolution in order to reduce testing costs by predicting change-prone classes	Yes	No	No	Maintainability	2	Medium	Yes	N/A	Code	Generic	Object-Oriented SW
14	An empirical study to redefine the relationship between software design metrics and maintainability in high data intensive applications	Yes	No	No	Maintainability	3	Difficult	Yes	Machine Learning	Code	Web Application	Object-Oriented SW

Figure 3.7 The Screenshot of Technical Data Table-2

The technical data of papers related to five criteria for RQ 1 and six criteria for RQ 2 are determined as shown in Figure 3.8.

Number	Name of Article/Conference Paper	Case Study	Statistical Method/Model	SDLC Phase	Application Domain	Programming Type	Further Research Require?	SLDC Models	Measurement Types	Threshold Info/Value	Explanation for Future Research	Future Plan
15	Applying swarm ensemble clustering technique for fault prediction using software metrics	No	Swarm Optimization	Code	Generic	Object-Oriented SW	Yes	N/A	Y	No	comparision of presented methods in paper with those found in the literature	Make More Detailed Research (New Case Studies)
16	Automated inference of goal-oriented performance prediction functions	Yes	Genetic Algorithm, Machine Learning	Code	Generic	Object-Oriented SW	Yes	N/A	Y	Yes	Investigation of why different combinations of methods worked better in one scenario than the other: - classification of scenarios to choose best combinations of methods to be applied to a given application context. - automate our approach, automatically selects those system parameters that should be included in a prediction function using a combination of techniques.	Improve the Tecnique
17	Critical Analysis of Object Oriented Metrics in Software Development	No	N/A	Code	Generic	Object-Oriented SW	Yes	N/A	Y	No	validation of the identified metric suite against prevalent metric suites using Fuzzy Logic and Neuro-Fuzzy technique.	Make More Detailed Research (New Case Studies)
18	Deriving thresholds of software metrics to predict faults on open source software: Replicated case studies	Yes	Logistic Regression	Code	Generic	Object-Oriented SW	Yes	N/A	Y	Yes	thresholds can be tested and analyzed in a wider range of datasets including different programming Languages The same threshold derivation model can be applied to method-level metrics	Improve the Tecnique

Figure 3.8 The Screenshot of Technical Data Table-3

3.4. Final Pool of Articles and the Online Repository

The final pool of selected articles and conference papers, consisting of 70 papers, are saved under Google Drive online repository, and a list of the final pool is published as Google Docs. After the completion of thesis study, Google Drive repository will be accessible to everyone via the link shared in Appendix A.

Data extraction is completed through using CADIMA and online spreadsheet of Google.

3.5. Data Extractions

The number of articles for each database found in the search of databases and further sources is shown in Table 3.7. In the first search, total number of articles and conference papers, which were published between 01.01.2009 and 31.07.2019 in English language, is 1039. After applying inclusion and exclusion criteria, 640 articles and conference papers are left. 640 articles are categorized according to PICO criteria given in the Table 3.4 and relevancy level of the topic judged from the

title and abstract. 70 papers amount to 719 pages in total. Overall process of selecting the relevant papers is shown in

Figure **3.9**.

Table 3.7 Literature Search Results

# SS	Search string	Database Name	Results
SS-1	(("Document Title": software metric(s)) OR ("Document Title": software measure(ment)) AND ("Document Title": quality) OR ("Document Title": tool))	IEEEE Explore	64
SS-2	(("Document Title": software quality) OR ("Abstract": software quality) OR ("Abstract": software) AND ("Abstract": metric(s)) AND ("Abstract": measure(ment)) AND ("Abstract": tool))	IEEEE Explore	204
SS-3	(("Document Title": metric(s)) OR ("Document Title": measurement) AND ("Document Title": tool) AND ("Abstract": software measurement))	IEEEE Explore	37
SS-4	(("Document Title": software) AND "Document Title": measure) AND "Abstract": software quality) OR ("Abstract": tool))	IEEEE Explore	58
SS-5	(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 (EXCLUDE (SUBJAREA , "MATH") OR EXCLUDE (SUBJAREA , "DECI") OR EXCLUDE (SUBJAREA , "BUSI") OR EXCLUDE (SUBJAREA , "ENER") OR EXCLUDE (SUBJAREA , "PHYS") OR EXCLUDE (SUBJAREA , "MATE") OR EXCLUDE (SUBJAREA , "MEDI") OR EXCLUDE (SUBJAREA , "MULT")) AND (EXCLUDE (LANGUAGE , "Chinese") OR EXCLUDE (LANGUAGE , "German") OR EXCLUDE (LANGUAGE , "Portuguese") OR EXCLUDE (LANGUAGE , "Russian") OR EXCLUDE (LANGUAGE , "Spanish") OR EXCLUDE (LANGUAGE , "Turkish") OR EXCLUDE (LANGUAGE , "Undefined")) AND (EXCLUDE (SUBJAREA , "SOCI") OR EXCLUDE (SUBJAREA , "BIOC")) AND (EXCLUDE (SRCTYPE , "k"))	SCOPUS, Google Scholar, Web of Science	90, 116
SS-6	(TITLE (software AND metric) OR TITLE (software AND measurement) AND ABS (software AND quality AND metric(s)) OR ABS (software AND quality AND measure(ment))) AND DOCTYPE (ar OR cp) AND PUBYEAR > 2008 AND (LIMIT-TO (SUBJAREA , "COMP") OR LIMIT-TO (SUBJAREA , "ENGI")) AND (LIMIT-TO (LANGUAGE , "English"))	SCOPUS	319
SS-7	(+software +metric(s) +software +measure(ment)) AND recordAbstract:(software quality metric(s), software quality measure(ment), tool)	ACM	57
SS-8	Title (+software +metric(s) +software +measure(ment)) AND Abstract:(software quality metric(s), software quality measure(ment)) Title AND Abstract : (+software +quality+ metric(s)/measurement +tool))	OpenAIRE	68
SS-9	TI Title and TI AB (software quality and metric(s) or measurement or measure) TI Title and TI AB (quality tool and metric or	EBSCO	16

	measurement or measure) TI Title and TI AB:(quality metric tool or quality measurement tool)		
SS-10	Title AND Abstract:(software quality metric(s), software quality measurement/measure) Title AND Abstract:(quality metric tool or quality measurement tool)	Science Direct	10

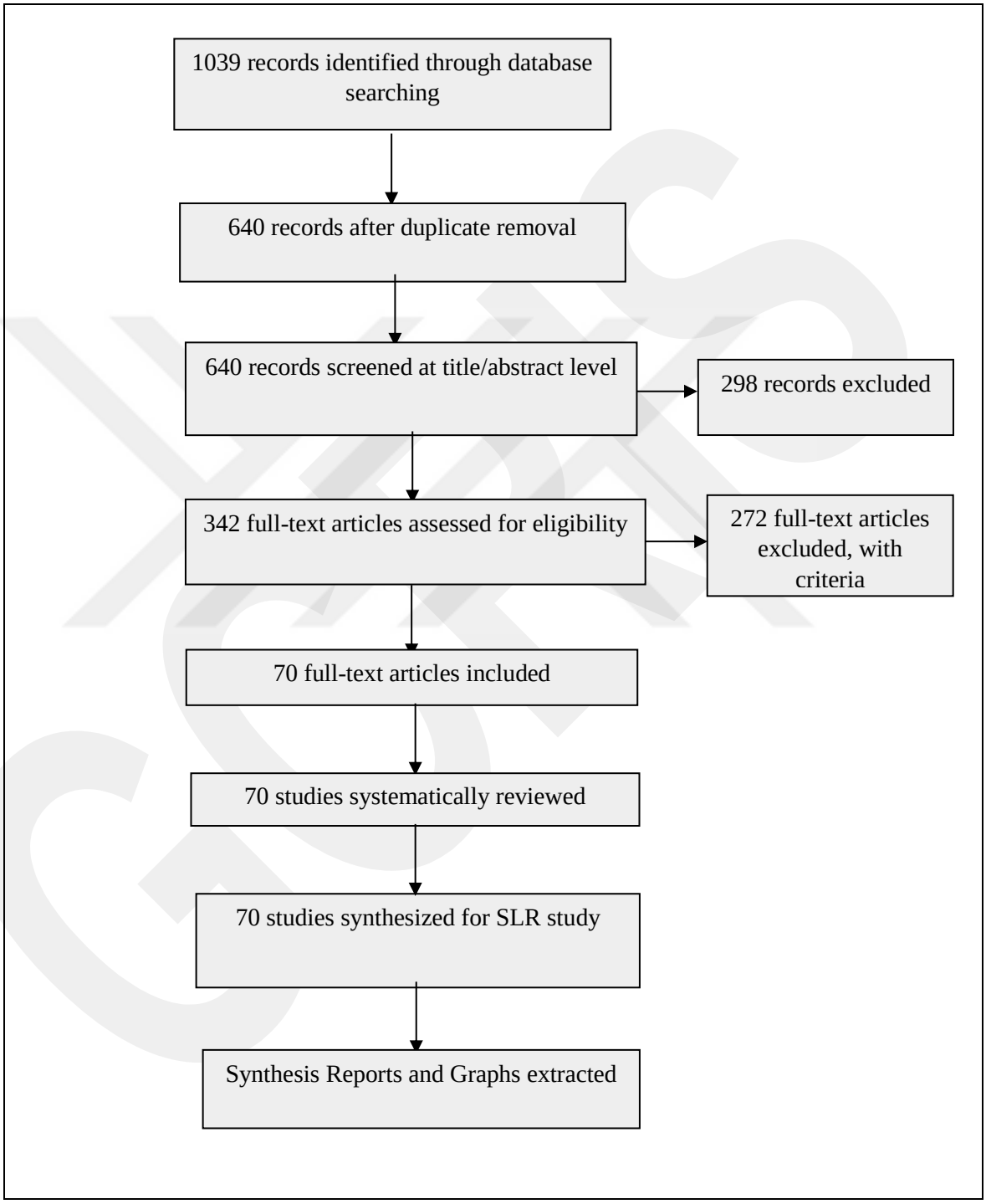


Figure 3.9 Flowchart Diagram of Search Results

3.6 Data Synthesis

In order to reply the SM-RQ 1 and SM-RQ 2 questions, SM is conducted and after the synthesis of the SM, studies are used for answering the SLR-RQ 1, 2, 3 and 4.

Methods related to the synthesis of quantitative and qualitative data in software engineering, like narrative synthesis, meta-analysis, cross-case analysis, thematic analysis, content analysis, case survey, qualitative comparative analysis are categorized in paper [51],

While a SLR study is being conducted, statistical methods might be used to analyze the present data. Due to the heterogeneity of the primary studies in our SLR repository, it was not possible to carry out a statistical analysis of the data. After assessing the applicability of possible methods for this SLR study, the most applicable synthesis method was determined to be the thematic analysis. Thematic analysis consists of the following steps [52]:

a. Extract demographic, bibliometric and technical data from primary studies.
b. Detect and code data categories, findings and trends
c. Translate code into coherent and consistent themes
d. Generate relations between themes by giving clear descriptions for each theme
e. Evaluate the reliability of the synthesis by answering research questions

Data synthesis is carried out in this study through employing the thematic analysis method steps as outlined in the thematic synthesis checklist given in the following chapter [52].

The detailed results and graphical representation of the thematic synthesis are explained in Chapter 4.

CHAPTER 4

RESULTS

4.1 Results of SM and SLR Study

Following the importation of the title of 70 papers to the Wordle tool (http://www.wordle.net/), graphics shown in Figure 4.1 were generated. This graph chart is important for this analysis, since it gives the opportunity to identify the keywords that help authors and readers to form general ideas about the topic. Overall, this presentation shows us that software quality metric related papers mostly include the keywords like software, metrics, quality, early prediction, object-oriented, empirical, maintainability, tool, etc.

Figure 4.1 Word Count Presentation of Paper Titles

China and Korea deserve a particular attention in this regard. Academic Search Engines identify the papers listed below, but unfortunately only titles and abstracts of papers are written in English and rest of the articles are in Korean and Chinese, as a result of which even though these papers were highly related to our topic, they could

not be included in SLR repository. Given the fact that these articles, which contain information about the evaluation of existing metrics and/or the introduction of new metrics cannot be read and studied widely due to language barriers, it is a significant loss for literature.

A list of these Chinese and Korean Journals is given below. Some example source titles from these journals are:

- Research and Application of Software Quality Metric to an Aspect-oriented Digital Video Monitoring System
- The Study of quality measurement plan for software reliability
- Software-based Quality Measurement of Mobile VoIP Services
- A Software for Subscriber-Oriented IPTV Service Quality Measurement
- Security Quality Measurement Model of M-Commerce software
- A Study of Smart Healthcare Services Software Quality Satisfaction Rating System based on QoS(Quality of Service) Measurement Model
- Model-Based Effort and Reliability Measurement for Software Quality Assurance
- Design of Quality Measurement Metrics for Mobile Software
- Quality measurement evaluation implement in aerospace software

The results of the SM study are presented in the following parts:

SM-RQ 1: Bibliometric and Demographics of the Publications

SM-RQ 1.1. What is the number of papers published per year?

As shown in Figure 4.2; there is an increase in the number of publications between the years 2009 and 2011. After that, the number of papers published decreases with an exception in 2014. During the ten year period, the year in which minimum number of documents has been published is 2017. For 2019, nine (9) papers can be considered satisfying because data includes only first seven months of the year. In the remaining part of the year, number of published papers will possibly surpass the number of papers published between 2010-2011.

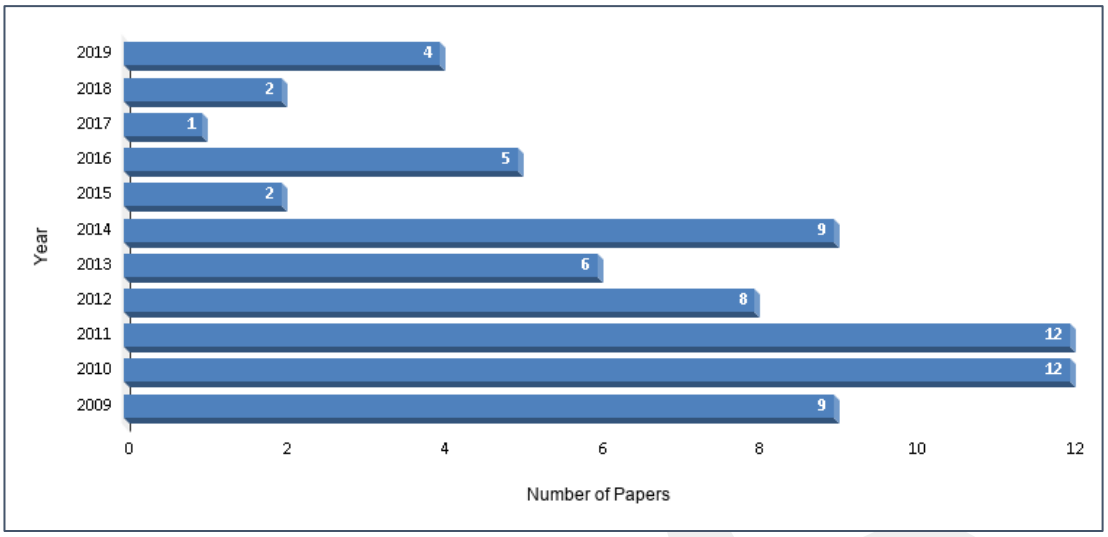


Figure 4.2 Numbers of Papers Published by the Years

Having an "article in press" section, Scopus provides us with a list of articles, conference papers, books and book chapters, which are coming forth. This indicates that documents have been accepted for publication but have not been allocated for a specific journal issue yet. Following is the list of forthcoming articles from the September 2019 to September 2020:

1. An article titled "Measurement of service quality of a public transport system, through agent-based simulation software" is accepted for publication as Computer Intelligence related book chapter by Springer at 2020. It is written by four (4) academics from Colombia and Slovakia, and currently, only abstract is available for this paper. As deduced from the abstract, this research aims to present the agent based modeling software which measures the service quality based on rapidity and comfortability of public transportation system. When analyzed the articles and conference papers, which were published between 2009-2019 in the area of software quality metric/measurement, we have pointed out that "Agent-Based Software Engineering" topic is greatly popular among scholars.

2. Another article titled "Evaluation of Design Pattern Utilization and Software Metrics in C# Programs" will be published as a conference paper on Intelligence Conference on Dependability and Complex Systems, and it will appear as a book

chapter in 2020. It is written by two (2) scholars from Poland. There is lot of experimental research related to the association between different software metrics and design patterns which can be grasped easily by through analyzing source code. Unfortunately, almost none of them are related to the C# programs. This paper focuses on the measurement of design patterns from the C# programs to analyze the impact of software quality. Authors predict that outcome of this paper will be used for the improvement of activities related to reverse engineering, refactoring and maintenance.

3. "Threshold estimation from software metrics by using evolutionary techniques and its proposed algorithms, models" titled article is planned to be published in 2019 as a part of Evolutionary Intelligence Journal. This paper is written by three (3) scholars from India. As presented in the abstract part, this research presents how to generate the threshold values from software metrics with the help of novel evolutionary intelligence techniques. Thresholds play an important role for analyzing the software metrics meaningfully. If thresholds are exceeded, managers and technical leaders can raise the flag for taking corrective action after the location of the root cause of problem.

SM-RQ 1.2. What is the type of papers?

As shown in Figure 4.3, out of the total number of 70 papers, 20 are articles and 50 are published as conference papers. The year which witnesses the highest number of article is 2011 whereas the year with the highest number of conference paper is 2010. In the year 2011, number of the articles and conference papers is the same. Number of conference papers in our selected paper pool made a peak in year 2010. Trend has decreased after 2011 and started to increase slightly at the beginning of 2019.

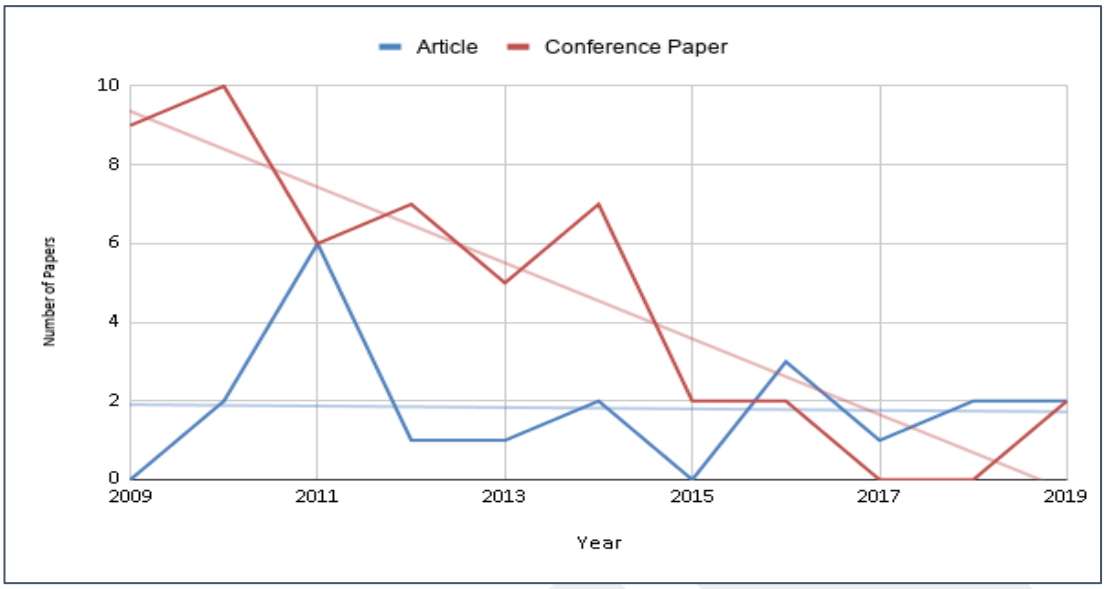


Figure 4.3 The Change in Number of Articles, Conference Papers by Years

SM-RQ 1.3. What is the affiliation of authors?

When articles and conference papers were analyzed, it was found out that they were mostly produced by authors with academic affiliation, in total; three (3) from industry and twelve (12) from mixed affiliation. The distribution of 70 papers is shown in below Figure 4.4.

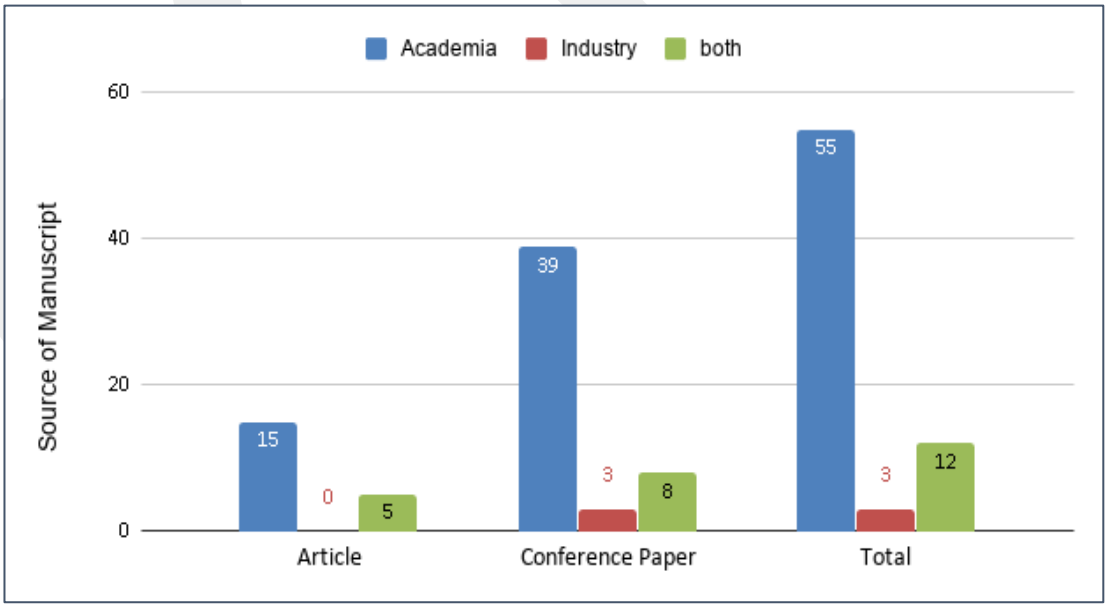


Figure 4.4 The Number of Article and Conference Papers as Source

Most of the software quality metric related papers are written in the academia, which should alert us to the need for collobration between academia and industry in order to combine technical data with life practices. Also, real development projects are useful in that sense for testing the validation of the metrics. As mentioned in the future plan of paper [S14] and [S43], there is a need to validate and to ensure that the metrics, tools or methods used are compatible with the dynamics of the projects from different environments in industry.

SM-RQ 1.4.Under which subject area papers are published or categorized by the publishers and search engines?

The data for the subject areas of each paper is retrieved from the Scopus database. The results show that 16 of 20 articles and 46 of 50 conference papers are categorized under Computer Science as shown in Figure 4.5. Only 2 of the conference papers are categorized under engineering and 2 of the articles under computer engineering.

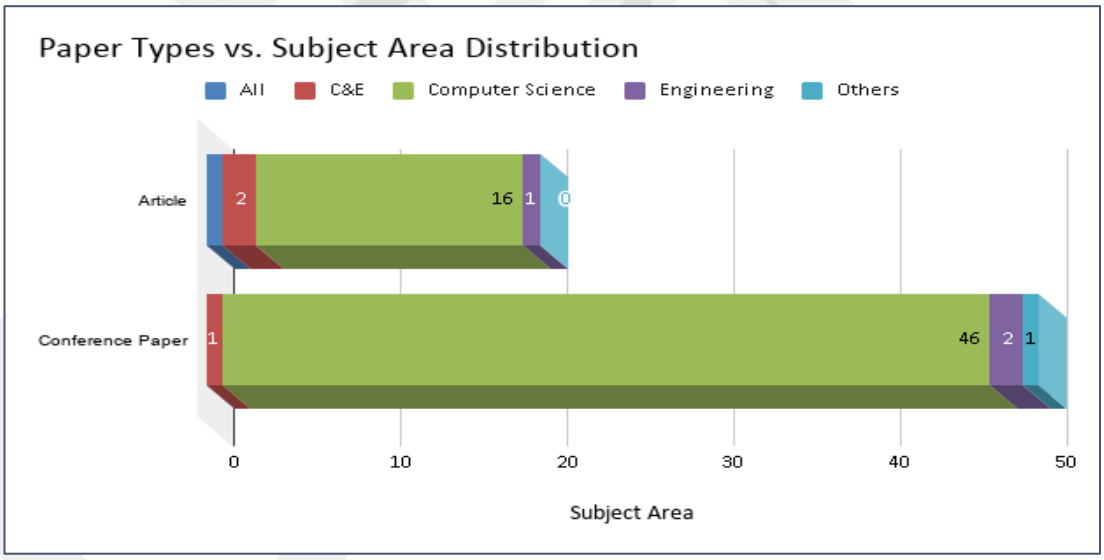


Figure 4.5 Paper Types vs. Subject Area Distribution

When given below and data repository are analyzed, it can be observed that the researchers mostly tend to prefer the papers which refer to industry experiences or the collaboration of Computer Engineering/Science and Mathematics and Statistics departments. Moreover, it was also observed that as the citation count of paper increases, the number of download count gets escalates, too. Besides, they also get a

higher number of full text views. This download data is taken from the publisher web-sites of the papers at the end of July 2019.

SM-RQ 1.5. What is the article distribution over countries? What are the most active countries in software quality metrics studies?

The distribution of papers by number according to the author's affiliation is given in Figure 4.6. India, followed by the United States, Brazil and Germany are the four countries with highest number of papers as shown in Figure 4.7.

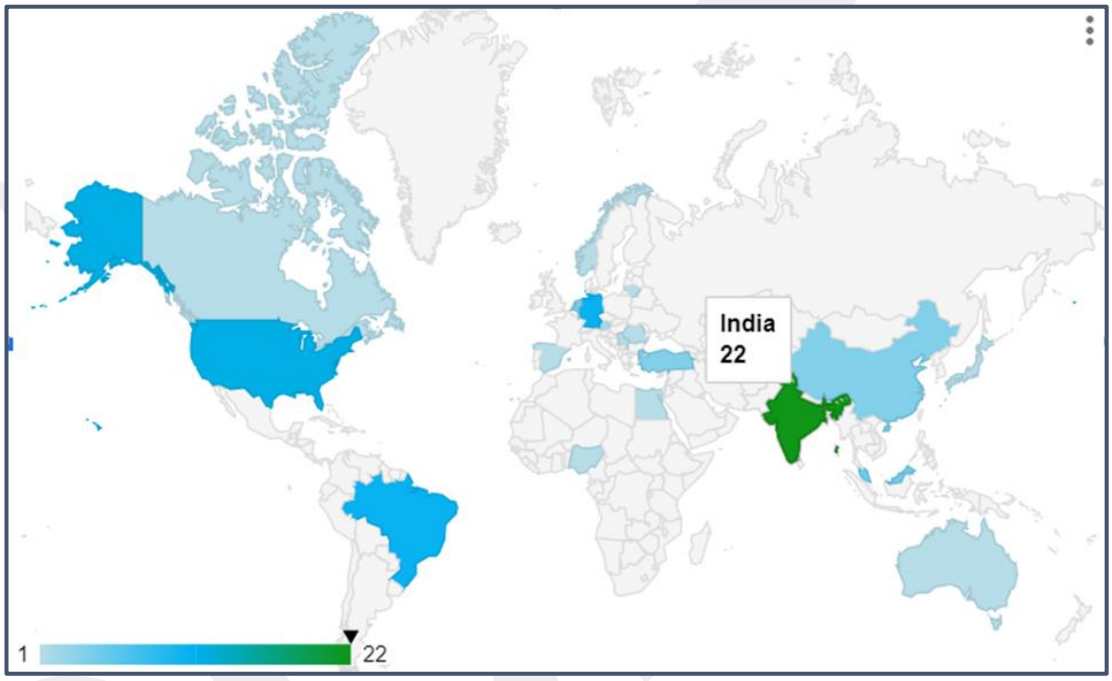


Figure 4.6 The Distribution of Paper Numbers for Countries

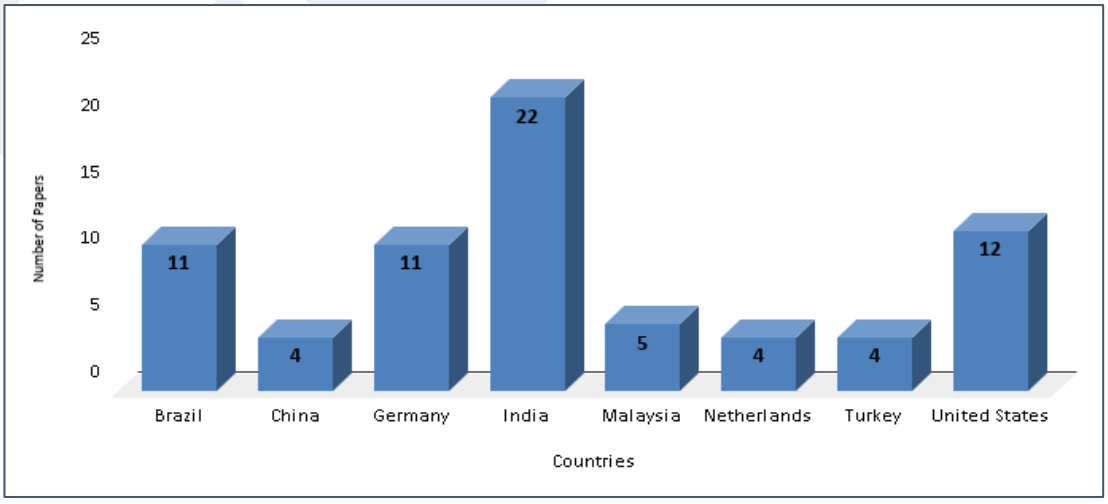


Figure 4.7 The Number of Papers for Countries

In our paper pool, papers were chosen from twenty (20) different countries. Comparing it to all countries in the world, which have 154 countries based on the report of United Nations, we can concluded that country coverage rate of our selected paper pool is 12%. After preliminary elimination of papers in the pool, number of remaining papers was 640. With this dataset, still USA and India were the leading countries followed by Brazil, Germany, Malaysia, China and Turkey, respectively.

Some countries have only one paper that meets our search criteria, so these countries were excluded in the analysis. List of these countries: Australia, Austria, Canada, Egypt, Japan, Litvania, Luxembourg, Nigeria, Norway, Romania, Serbia, Spain. The number of the papers that have been written by authors from more than one country is eleven (11).

Largest percentage of the papers in our paper pool is produced in India (30%), then United States with 16% and a slice of 15% belongs to Brazil and Germany as shown in Figure 4.8.

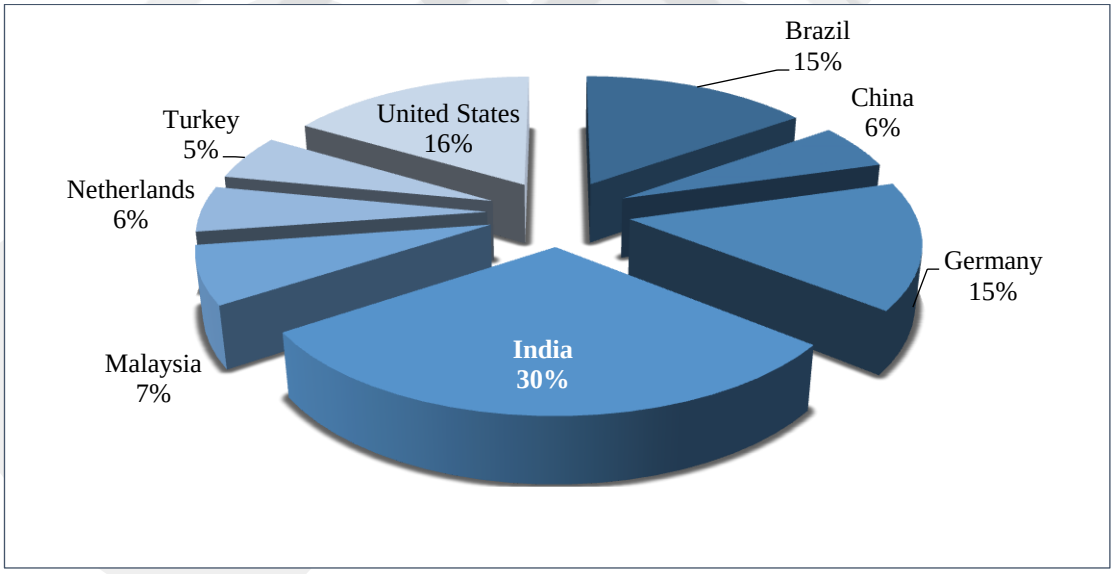
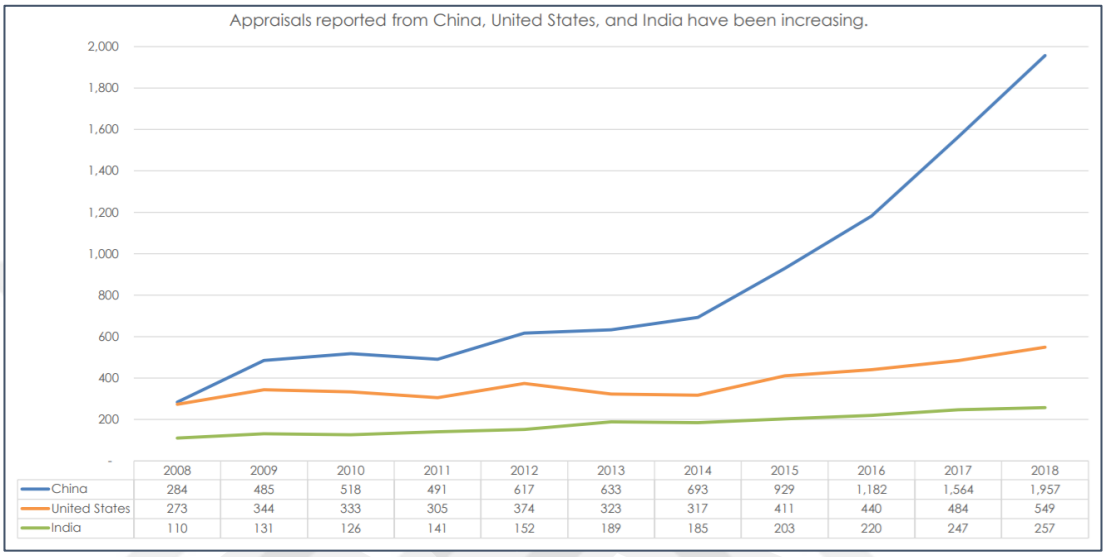


Figure 4.8 The Pie Chart of Papers by Country

Is there a link between the CMMI model, which recommends measurement and analysis process to improve quality, and the countries that wrote the majority of the articles on software quality metrics? To answer this question, last published

(December 2018) and available version of the CMMI adoption graph was analyzed. Number of appraisals for CMMI by country between the years 2008 and 2018 is given in Figure 4.9. The number of appraisals for India is increasing slowly, but for China a sharp rise in number from 2014 onwards was noticed. Mostly CMMI adopted countries are India, United States and also biggest share of our paper pool belongs to the same countries.



	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018
China	284	485	518	491	617	633	693	929	1,182	1,564	1,957
United States	273	344	333	305	374	323	317	411	440	484	549
India	110	131	126	141	152	189	185	203	220	247	257

Figure 4.9 Highest Numbers of Appraisals By Country [53]

SM-RQ 1.6. Which are the top cited papers in the last decade?

Citation and Normalized Citation count data is given in Figure 4.10 and Figure 4.11. Citation count of papers is taken from Scopus database as of July 31, 2019. The count for the most cited papers increased and peaked in 2011. Normally citation count and normalized citation count of papers which were published in 2019 is 0 (zero). The most cited papers are given in Table 4.1.

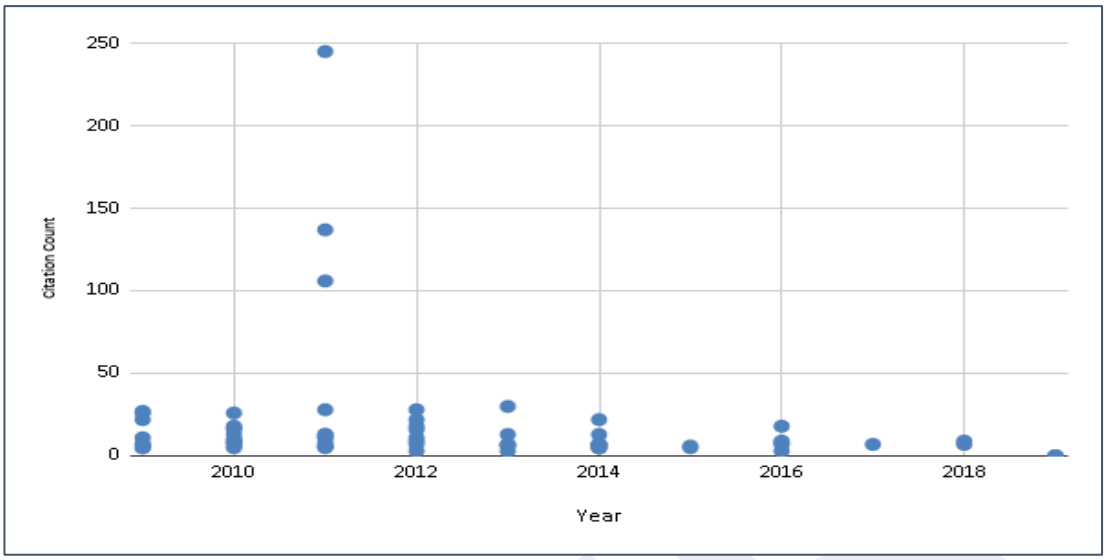


Figure 4.10 Citation Count as Year Published

Normalized citation formula is given below with an example:

$$Normalized\ Citation = \frac{Citations\ (as\ of\ July, 2019)}{Paper\ Age} \quad (1)$$

$$Normalized\ Citation\ [S48]\ = \frac{22}{2019 - 2012} = 3{,}14$$

Average of the normalized citation count is 2.62; and there were 12 papers over the average: [S14], [S18], [S19], [S24], [S28], [S29], [S31], [S43], [S44], [S48], [S49], [S66].

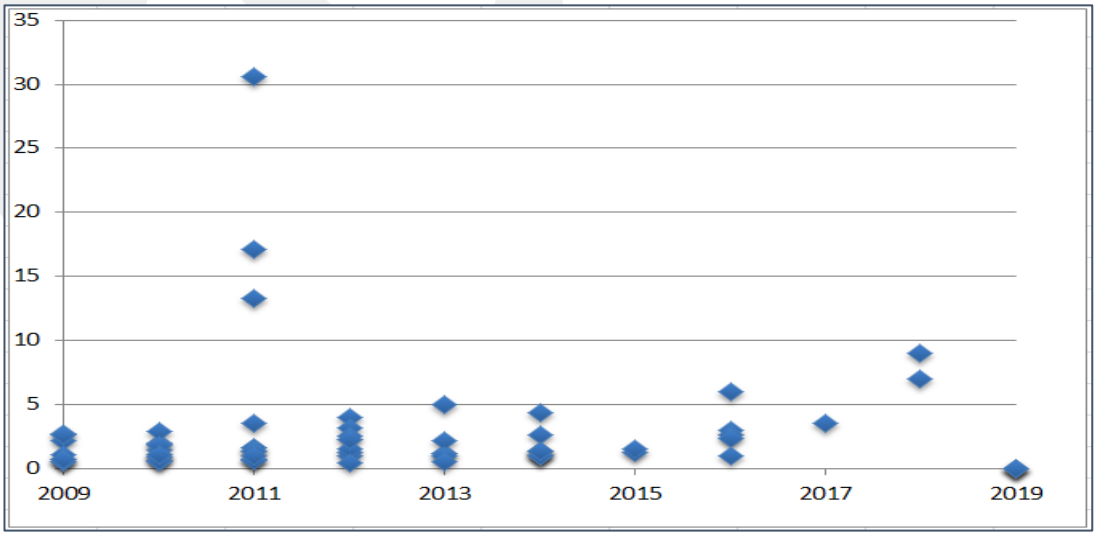


Figure 4.11 Normalized Citation Count of Papers

In addition to calculation of the quantity of paper per each country, popularity of the papers for each country is also another significant metric. To measure this, we calculated the normalization citation count per paper for each country by using below the formula:

$$Normalized\ Citation\ Count\ per\ Papers = \frac{Sum\ of\ (Normalized\ Citation\ Count)\ for\ all\ papers\ per\ country}{Number\ of\ Papers\ for\ Country} \quad (2)$$

When papers are categorized, India is seen to be the leading country, followed by USA. On the other hand, when we analyze the number of papers with normalized citation count, India is seen to be the leading country for quantitative criteria whereas, USA emerges as the leading country for quality criteria as shown in Figure 4.12.

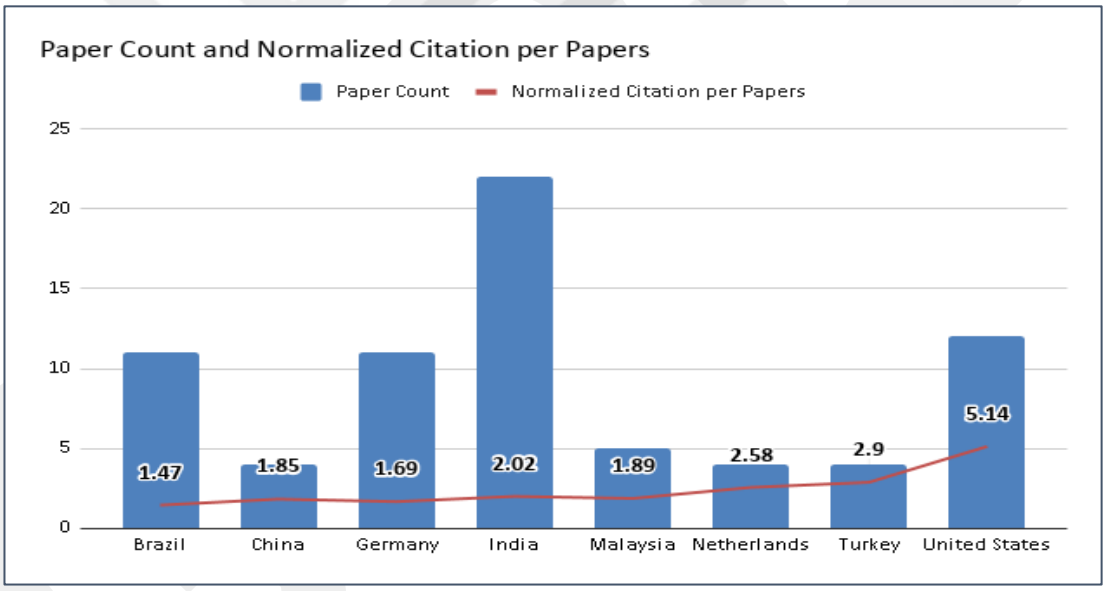


Figure 4.12 Normalized Citation Count per Papers vs Country

SM-RQ 1.7. What is the relation between citation count and download count of the papers?

Paper download rates are taken from the publisher of the paper. The ratio of the citation data and download count graph is presented in Figure 4.13.

Maximum citation count is 245 and maximum number of download for full text views is 3453. Full text view download data of this paper is taken from IEEE XPlore Digital Library which is the publisher of the article.

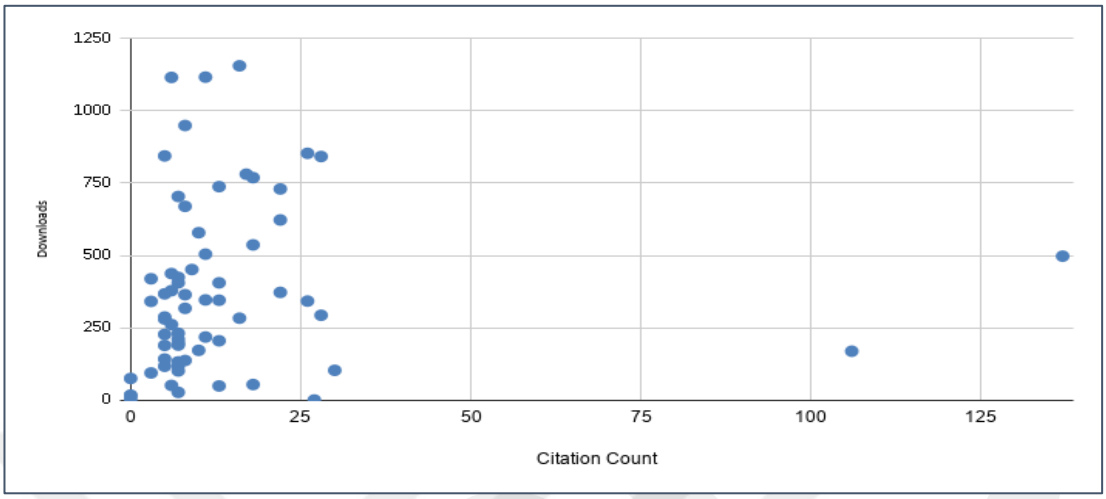


Figure 4.13 Citation Count vs. Downloads

The analysis shows that the papers with the highest citation count value, also have a higher number of download counts as presented in Table 4.1. Papers which were downloaded in equal numbers and more than 100 times, have been cited at least 3 times.

SM-RQ 1.8. What is the average number of the authors who conduct research and publish the papers?

Most of the papers in paper pool are written by approximately three (3) authors. Only 3 ([S13], [S22], [S56]) out of 70 papers are produced by a single author. The maximum number of authors is six (6) and this data is normal because of authors' background is industry and academia. There are 2 papers with the highest number of authors in paper pool, with following citation counts: [S27], year 2010, citation count: 26 and [S45], year 2016, citation counts: 6.

SM-RQ 1.9. Who are the most active researchers in the software quality metrics area?

To answer this question, authors of most cited papers are taken into account. The name of the authors is listed in Table 4.1. There are not any other works produced by the same authors.

We have listed the most suitable articles and conference papers below according to citation counts. Citation counts are taken from Scopus database. In the following table, first three (3) rows include information about venue, country, affiliation, number of authors, name of the authors and work place of authors of the included articles. Also, remaining four (4) rows present the data about the most cited conference papers.

Table 4.1 The list of Most Cited Papers

Venue	Article Name	# of Citation*	# of Downloads*	Normalized Citation*	Authors	Work Place
IEEE Transactions on Software Engineering (Publisher: IEEE)	An Attack Surface Metric [S19]	245	3453	30,63	P. K. Manadhata; J. M. Wing	Member, IEEE, USA
Software - Practice and Experience (Publisher: Wiley Online Library)	Choosing software metrics for defect prediction: An investigation on feature selection techniques [S29]	137	498	17,13	Gao, K.; Khoshgoftaar, T.M.; Wang, H.; Seliya, N.	Department of Mathematics and Computer Science, Eastern Connecticut State University, U.S.A. Computer and Electrical Engineering and Computer Science, Florida Atlantic University, U.S.A. Computer Science, Western Kentucky University, U.S.A. Computer Engineering, University of Michigan, U.S.A.
Journal of Systems Architecture (Publisher: Science Direct)	Using Complexity, Coupling, and Cohesion Metrics as Early Indicators of Vulnerabilities [S66]	106	170	13,25	Chowdhury, I., Zulkernine, M.	Department of Electrical and Computer Engineering, Queen's University, Kingston, Canada School of Computing, Queen's University, Kingston, Canada
International Conference on Software Testing, Verification and Validation Workshops	An Empirical Study on Object-Oriented Metrics and Software Evolution in order to Reduce Testing Costs by Predicting	28	842	3,50	Eski, S.; Buzluca, F.	Computer Engineering Department, Istanbul Technical University, Turkey

Venue	Article Name	# of Citation*	# of Downloads*	Normalized Citation*	Authors	Work Place
(ICST)	Change-Prone Classes [S24]					Center of Research for Advanced Technologies of Informatics and Information Security TUBITAK-BILGEM, Turkey
IEEE/ACM International Conference on Automated Software Engineering (ASE)	Automated Inference of Goal-Oriented Performance Prediction Functions [S28]	28	294	4,00	D. Westermann; J. Happe; R. Krebs; R. Farahbod	SAP Research, Germany
Brazilian Symposium on Software Engineering (SBES)	A Study of the Relationships between Source Code Metrics and Attractiveness in Free Software Projects [S27]	26	343	2,88	Meirelles, P.; Santos Jr., C.; Miranda, J.; Kon, F.; Terceiro, A.; Chavez, C.	Department of Computer Science Federal University of Bahia, Brazil Institute of Mathematics and Statistics University of São Paulo, Brazil
International ACM SIGSOFT Conference on Quality of Software Architectures (QoSA)	An Empirical Investigation of Modularity Metrics for Indicating Architectural Technical Debt [S3]	22	373	4,40	Li, Zengyang; Liang, Peng; Avgeriou, Paris; Guelfi, Nicolas; Ampatzoglou, Apostolos	Department of Mathematics-Computer Science University of Groningen Nijenborgh, Netherlands State Key Lab of Software Engineering, School of Computer Wuhan University, China Computer Science and Communications Research Unit University, Luxembourg

SM-RQ 1.10. What is the name of the top publishing venues?

Trend regarding the software quality metrics can be easily observed in the following conferences list. Top ten venues were determined through calculation of the sum of the total citation count and total number of papers, as presented in Figure 4.14:

- ESEM: ACM/IEEE International Symposium on Empirical Software Engineering and Measurement: [S6], [S36], [S37]
- ICSE: International Conference on Software Engineering: [S1], [S11]
- WETSoM: International Workshop on Emerging Trends in Software Metrics: [S25], [S63]
- ASE: IEEE/ACM International Conference on Automated Software Engineering: [S28]
- ICST: International Conference on Software Testing, Verification and Validation Workshops: [S24]
- ACM-SE: Annual Southeast Regional Conference: [S54]
- SBES: Brazilian Symposium on Software Engineering: [S27]
- MOMPES: International Workshop on Model-Based Methodologies for Pervasive and Embedded Software: [S57]
- QoSA: International ACM Sigsoft Conference on Quality of software architectures: [S3]
- ICMV: International Conference on Machine Vision [S33]

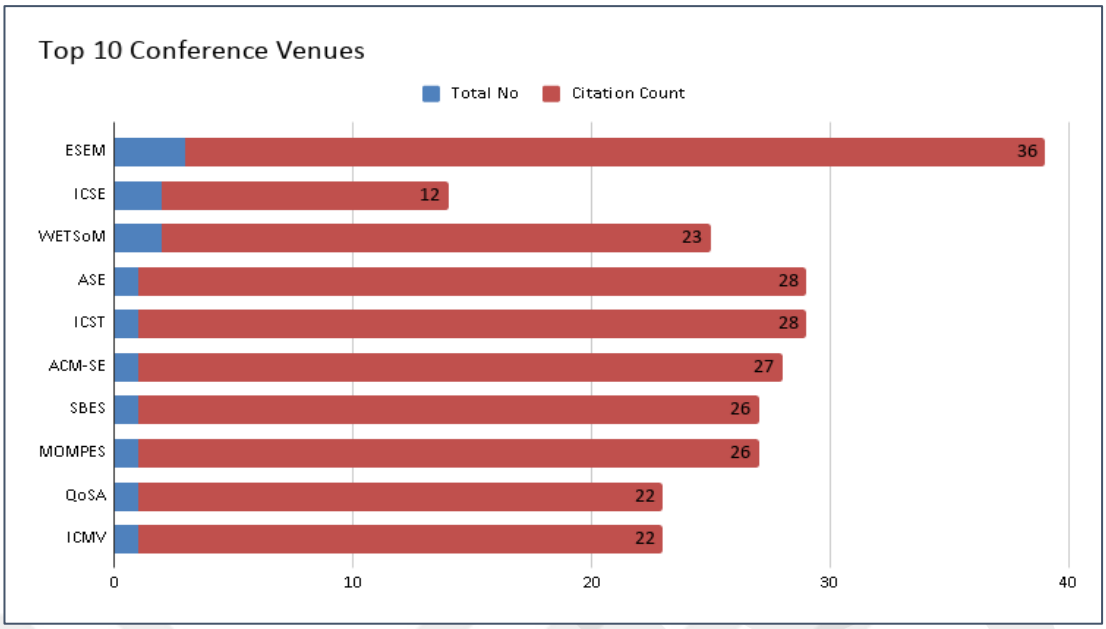


Figure 4.14 Top Ten Conference Venues

Software quality metrics is a broad topic and can be applicable to different domains so venues from which papers emerged are quite are widespread. So, there are lots of other venue names in our paper pool such as:

- IWSM-MENSURA: International Workshop on Software Measurement and International Conference on Software Process and Product Measurement
- MSR: Working Conference on Mining Software Repositories
- ARES: International Conference on Availability, Reliability and Security
- SCAM: IEEE International Working Conference on Source Code Analysis and Manipulation
- ISSREW: IEEE International Symposium on Software Reliability Engineering Workshops
- ICRITO: International Conference on Reliability, Infocom Technologies and Optimization
- ITNG: IEEE International Conference on Software Maintenance and Evolution

Moreover, the conference list, consisting of the conferences that were held between 2009 and 2018, and which specially include software metric and measurement in

their title is presented below. Following is a list of conferences hold by consortiums formed by IEEE with different organizations in different countries.

Table 4.2 The Conference List

Conference Name (which are available for 2009-2018 years)	Organization by	Web Link	Starting Year	Period
EUROSPI2 : European System, Software & Service Process Improvement & Innovation	Software Process Improvement Networks from different European countries and ECQA.org	https://nqa2.iscn.com/index.php/conferences-link	1994	Annually
International Workshop on Emerging Trends in Software Metrics (WETSoM)	Agile group	http://www.agilegroup.eu/	2009	Annually
International Workshop on Security Measurements and Metrics (MetriSec)	Institute of Electrical and Electronics Engineers (IEEE)	https://www.ieee.org/	2007	Annually
IEEE/ACM International Workshop on Software Architecture and Metrics (SAM)	IEEE and The Association of Computing Machinery (ACM)	https://ieeexplore.ieee.org/document/7203147	2014	Annually
International Conference of Software Measurement CSM (in joint with International Workshop of Software Process and Product Measurement)	IEEE	https://ieeexplore.ieee.org/xpl/conhome/6693157/proceeding	2006	Annually
International Symposium on Empirical Software Engineering and Measurement (ESEM)	IEEE, ACM	http://eseiw2019.com/	2007	Annually
MetriKon (The Software Metric Congress)	DASMA: The German Software Metrics Association	https://www.innovations-report.com/	2006	Annually

According to the inclusion/exclusion criteria this study uses, there is no paper presented in the following venues in our paper pool: EUROSPI2, MetriKon, MetriSec, SAM. Between the years 2009 and 2018; there are no papers which have

been issued in EUROSPI2, MetriKon, MetriSec, SAM with keywords “software 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.

SM-RQ 2.1: What type of research methods/facets are used in the papers?

Empirical Research: Any study in which the results are obtained from concrete and verifiable evidences [54]. In experimental researches; authors who are the users of the systems, carry out and report their studies by including their observation, opinion, lab-based studies, interviews, field study and experiments [55].

Analytical Research: In analytical researches; authors who are experts like academicians, practitioners and technical experts, realize and report their studies by including techniques like logical and mathematical calculations of experimental work, analysis of evaluation group results, heuristic evaluation, usability inspection and cognitive walkthrough [55].

Percentage and distribution number for each research facet is presented in Figure 4.15. Research method of 52 out of 70 papers is defined as Empirical, while 16 of them are analytical, and 2 are marked as others (Evaluation Papers with data analysis).

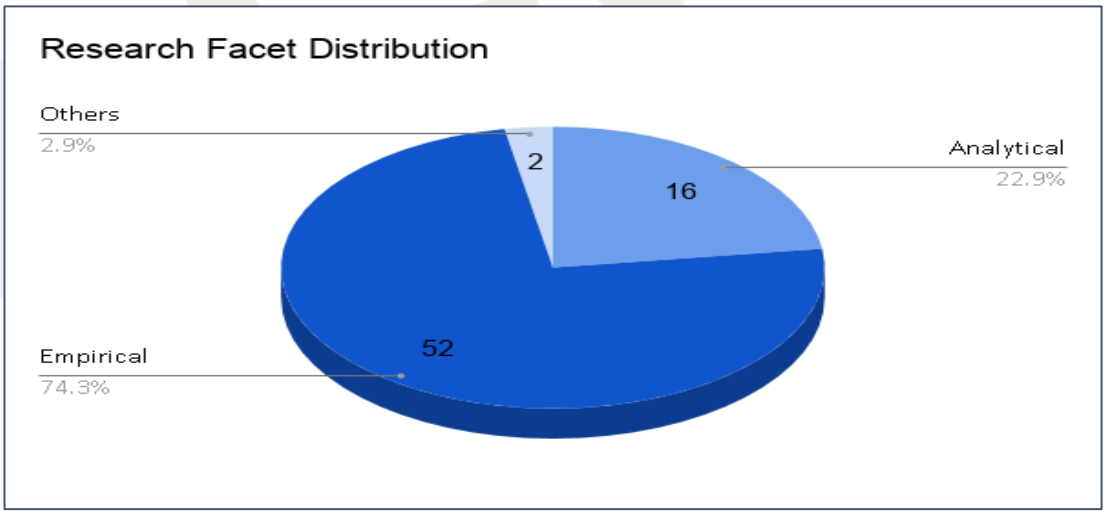


Figure 4.15 Research Facets Distribution

Most papers, which are 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 computer science field.

Research method results of paper pool in this thesis study, have been compared with the article titled “A systematic mapping review of software quality measurement: Research trends, model, and method [19]”. This systematic review examines 42 papers (publication years: 2007-2017) through two perspectives: Distribution based on research method and software quality model. Papers in article [19] are selected from the IEEExplore , Scopus, Science Direct search engines. Results has shown that 67% of the 42 papers were developed through empirical research method. Paper pool in this thesis includes 70 papers for the years between 2009-2019, and the resarch method results is approxiametly same, with ratio 74.3%.

SM-RQ 2.2: What type of research approaches are used in the papers?

Research Approaches: The framework of a set of philosophies in a research study.

- Fuzzy Logic: An approach to computing based on "degrees of truth" rather than the usual "true or false" (1 or 0) [56].
- Novel Approach: This approach refers to the methodological approaches in qualitative research, a unique approach in a qualitative research [56].
- GQM (Goal-Question-Metric): This technique has been suggested by Basili and Weiss, and it can be defined as a technique to identify meaningful metrics for the measurement process by establishing the measurement goal and defining questions to reach this goal and identifying the metrics to answer defined questions [57].
- PSM (Practical software measurement): A popular approach to measurement for process management. It is based on the collective experience of a working group representing universities, industry, and government. PSM have been documented as an international standard and incorporated into the Capability Maturity Model [58].

- Analytic Hierarchy Process (AHP): Proposed by Thomas Saaty in 1980. AHP is a multi-criteria decision making method widely used in decision making and help for decision maker to choose best selection [59].

As presented in bar chart in Figure 4.16, most frequently used research approach is Goal-Question-Metric, followed by the Novel Approach. 19 papers (evaluation or survey papers) do not mention their research methods in detail; so these papers were not categorized. Finally, 6 papers were marked as others; Statistical approaches like Hidden Markov Model and systematic reviews.

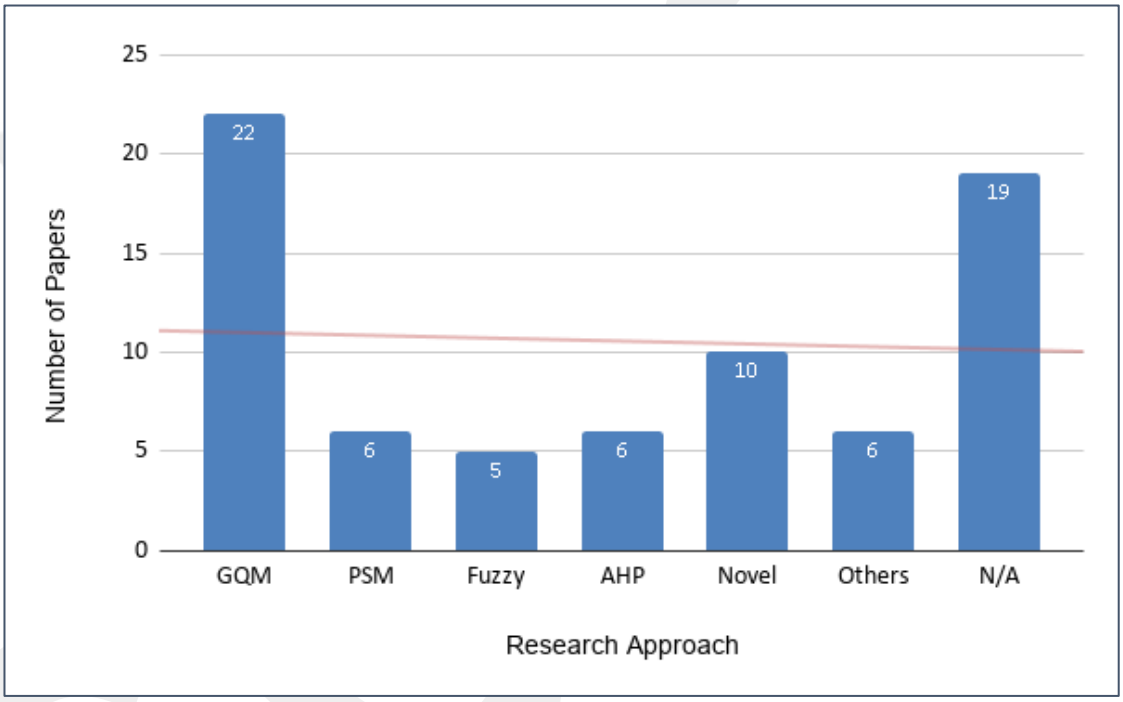


Figure 4.16 The Research Approach Distribution of Papers

One of the 19 papers uses the advanced version of GQM with the name of GQIM (Goal-Question-Indicator-Metric). Also, 3 papers use a combination of research approaches: [S12]: AHP with Novel Approach, [S59]: AHP with Fuzzy Approach [S60]: GQM with PSM.

Following table shows the trend in research facet and research method combination in the software quality metric area in the last decade. According to numbers in Table 4.3, researchers may choose to use the favorite research method/facet combination

for new studies in this area; however, researchers may also prefer to use less commonly used methods to reach new and unprecedented results.

Table 4.3 Research Facet Details

Research Facet	Research Method Detail
Empirical Research	1 AHP, 3 Fuzzy, 19 GQM, 6 Novel, 4 PSM
Analytical Research	5 AHP, 2 Fuzzy, 3 Novel, 2 GQM, 2 PSM, 1 others

SM-RQ 2.3: Whether authors of papers used case study or not to explain the metrics, method, research results, tool usages etc.? Which papers explain the metrics with case study by using which datasets or projects?

Approximately, 74% of papers used case study to explain their recommended metrics, methods, tool usage flow diagrams and process improvement suggestions. As shown in Figure 4.17, case study usage could not be spotted in 26% of papers. Case studies mostly help the readers to understand the mechanism and method presented in the papers. Using case studies thus facilitates comprehension. Level of the metrics in our selected paper pools has been calculated as easy Figure 4.18.

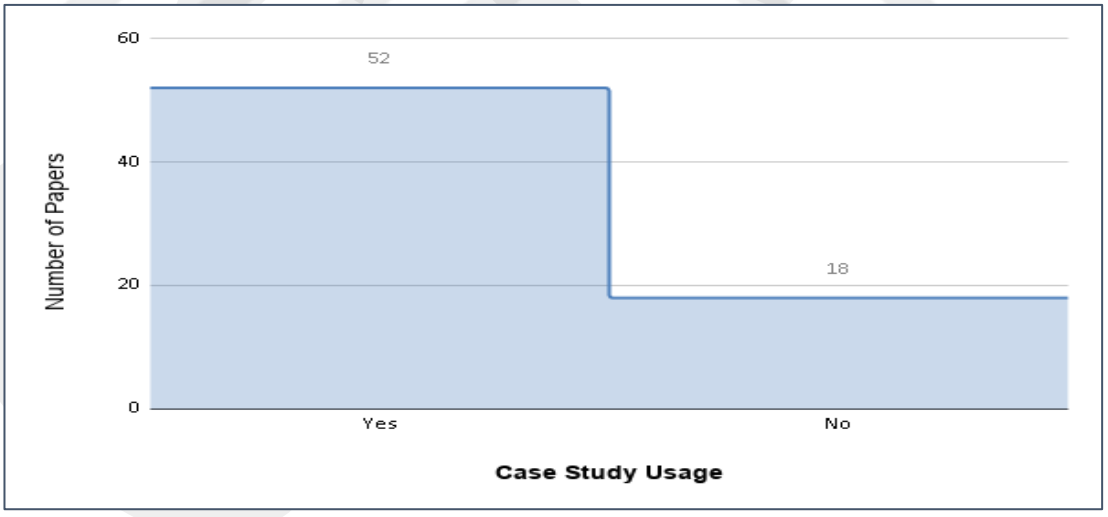


Figure 4.17 Numbers of Papers Including Case Study

Papers presented metrics, frameworks or processes by using case study as listed in Table 4.4:

Table 4.4 Papers with Case Study

Papers with Case Study	[S1], [S2], [S3], [S4], [S5], [S6], [S8], [S10], [S11], [S12], [S14], [S15], [S16], [S19], [S20], [S21], [S22], [S23], [S24], [S28], [S29], [S31], [S32], [S33], [S34], [S35], [S36], [S37], [S38], [S39],[S40] ,[S41], [S42], [S43], [S44], [S48], [S49], [S51], [S52], [S53], [S54], [S56], [S57], [S59], [S60], [S61], [S62], [S63], [S64], [S65], [S66], [S67].

SM-RQ 2.4: What is the level of ease of understanding of the software quality metrics presented in the papers?

Easy: Simple to count/measure (basic calculations, all variables-methods determined).

Medium: Requires some effort to understand and calculate metrics

Difficult: Difficult for understand and calculate; because of the complexity of the terms and concepts.

This question is answered based on the subjective data which is generated due to the education of the author of this thesis has been through. Results obtained from this data can be interpreted in the way that if readers have the education about software engineering or computer science, data in the 43 of the 70 papers will be understandable and meaningful as presented in Figure 4.18.

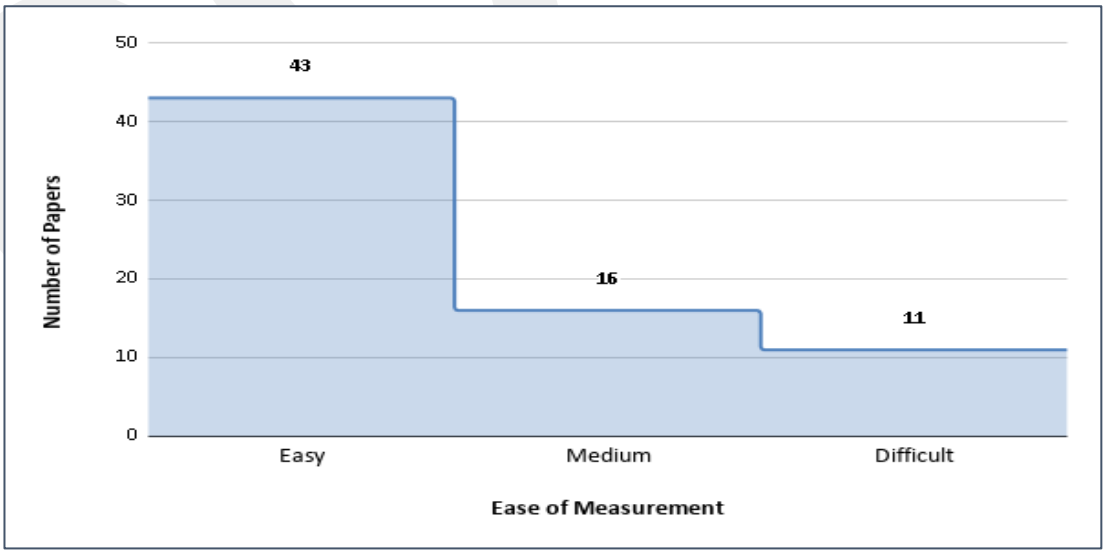


Figure 4.18 The Number of Papers for Ease of Measurement

SM-RQ 2.5: Which statistical model/methods are used to validate or generate new metrics?

Genetic Algorithm: “Genetic algorithm solves smooth or non-smooth optimization problems with any types of constraints, including integer constraints. It is a population-based algorithm that searches randomly by mutation and crossover among population members.” [60]

Swarm optimization: “Particle swarm solves bound-constrained problems with an objective function that can be non-smooth. If search does not work satisfactorily, this pattern can be used.” [61].

Optimization methods: “Procedures for finding the maxima or minima of functions of, generally, several variables. Most often encountered in statistics in the context of finding maximum likelihood estimation, where such methods are frequently needed to find the values of the parameters that maximize the likelihood.” [62].

Exploratory data analysis is defined as: **“**a philosophy and strategy of research which puts the primary focus of the researcher on using the data as the starting point for understanding the matter under research. This is distinct from the use of data as a resource for checking the adequacy of theory.” [63].

Descriptive statistics: “wide variety of techniques that allow us to describe the general characteristics of the data we collect. The central tendency (typical score) may be assessed by the mean, median or mode. The shape or spread of the distribution of scores can be presented graphically (using histograms, for example).” [63].

Machine Learning: “A term that literally means the ability of a machine to recognize patterns that have occurred repeatedly and to improve its performance based on past experience. In essence this reduces to the study of computer algorithms that improve automatically through experience. It is closely related to pattern recognition and artificial intelligence and is widely used in modern data mining.”[56]

Logistic regression: “A technique is used to determine which predictor variables are most strongly and significantly associated with the probability of a particular category in the criterion variable occurring.” [63].

Meta-analysis: “A collection of techniques whereby the results of two or more independent studies are statistically combined to yield an overall answer to a question of interest. The rationale behind of this approach is to provide a test with more power than is provided by the separate studies themselves.” [63].

As shown in Figure 4.19, genetic algorithm is used by only two papers; [S28] in year 2012, and [S58] in 2013. Also, swarm optimization is used in only paper [S26] in year 2014. An empirical study shows that Swarm Optimization can come in handy to detect the defect prevention of software.

Only 1 paper [S44] was used a “Statistical Extrapolation” method, which method is out of our statistical method options, author have been used this statistical method to forecast future data rely on the historical data.

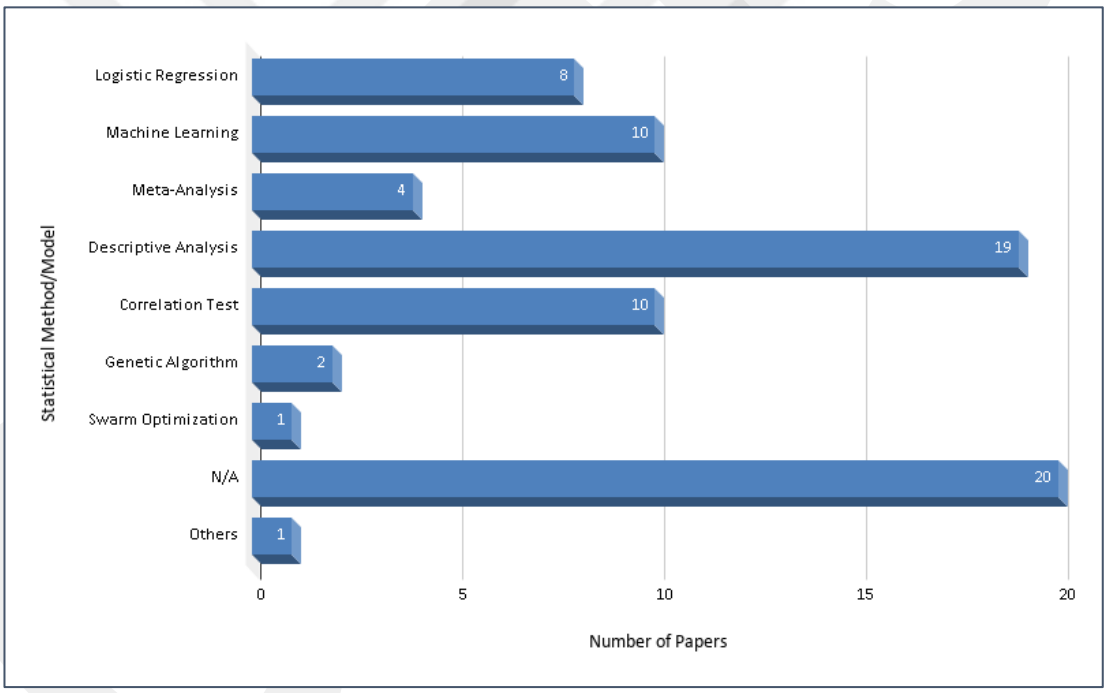


Figure 4.19 The Number of Papers for Statistical Method/Model

Machine learning methods show better results compared to statistical methods for predicting the relationship between OO metrics and change proneness.

Between 2014-2015, in the future plan part of the two papers, [S25] and [S38]; authors mentioned that they wanted to replicate their study with huge data by using

the unexplored learning techniques like genetic algorithms and colony optimization to obtain more adequate results. Despite the future plans of some researchers, when our search keywords are applied, no answer emerges. So, there is still no example about the usage of genetic algorithm and swarm optimization in the last five years. Another SM or SLR study may be carried out by entering keywords that directly include the name of the statistical method/algorithm and type of software quality metrics to analyze the trend of using these algorithms. As mentioned in the paper [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.

SM-RQ 2.6: What are the threats to the validity of the results of the papers?

Scale of validation data is taken from the systematic mapping study [36] and tailored in accordance with the content and scope of this SLR study.

0: No validation data, 1:self-constructed examples used, 2: validated by small data, 3: validated by medium/large data and/or validate by tool, 4: Proposed by one author and validate by another author or research team and/or automated tool with huge data.

Approximately 24% of the papers validated in level 2 and 27% of papers validated in level 3 as shown in Figure 4.20.

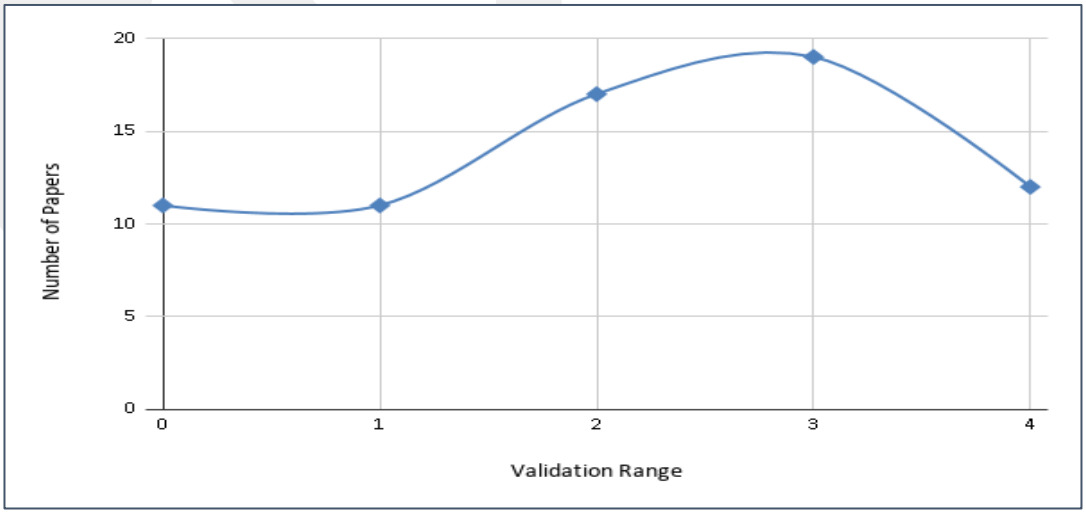


Figure 4.20 The Validation Range of Papers

When the data validation level is compared with the results of the paper [36], which evaluates the validity of data for architectural structures related papers generated in 1990-2013, results show that both studies show the level 2 and level 3 as validation ranges. Validation of the metrics has been actualized by using the following datasets of projects in general:

Table 4.5 Validation Datasets and Projects

Datasets and Projects	Definition/Properties
NASA MDP (Metric Data Program) and PROMISE repositories (NASA IV&V Metrics Data Program)	• Project data sets are publicly available at http://promise.site.uottawa.ca/SERepository/datasets-page.html • MDP repository covers data of 13 projects method tested by using real time defect datasets of the NASA software projects. Goals: • Predicting defects by using software metrics and data mining. • Analyzing empirically the relationship between fault proneness and metric thresholds.
SAP and SPEC benchmarks	• Goal: Analyzing functional dependencies between the performance relevant system parameters (such as configuration or workload parameters) and performance metrics. • Dataset is used as case study to evaluate suggested metric and statistical method combination.
Apache, Java.net, Google Code, Sourceforge.net	• 18.826 projects are collected.
Qualitas Corpus	• Open source Java system • Collection of software systems intended to be used for empirical studies of code artifacts.
Spring MVC and Android applications	Collection of 120 Spring MVC and 301 Android systems in Github has been used by authors validate the metrics. Also interviews with 6 different experts are conducted.
HP Brazil	• CMMI-3 level company data set is used for validation of metrics.
Mozilla Firefox 2 6.7 Microsoft IE 6 8.7 Microsoft IE 7 7.9 Apache Tomcat 4 4.0 Apache Tomcat 5 Amazon and CNET Websites Apache Tomcat, Apache CXF and Stanford Securibench Dataset	• To measure the security quality attributes of the web engines, most common search engines were used by the authors.

CUBIT	• Developed by METU-Informatics Institute (Turkey). • Web-Based tool, that provides easy means to measure software projects, generate and save measurement reports. • Includes the historical data about the functional size measurements of different software projects.
National Vulnerability Database (NVD)	• NVD is the US government repository of software vulnerability data.

SM-RQ 2.7: Which standards and process models have been used / referenced from papers to measure software quality? What are the most popular software quality standards and process models mentioned in the literature in the last ten years?

Figure 4.21 shows the distribution of the standards and models referred in the papers in our pool. According to this analysis, the most popular model is CMMI with 7 papers and standard is IEEE 1061 with 4 papers. 74% of papers have not referenced any standards or models while proposing the quality metrics.

The standards and models offered given in Figure 4.21 are listed below:

- ISO\IEC 15504: Software Process Improvement and Capability Determination
- ISO\IEC\IEEE 12207: Systems and Software Eng.- Software Life Cycle Processes
- IEEE 1061: IEEE Standard for a Software Quality Metrics Methodology
- CMMI: Capability Maturity Model Integration, SE Institute
- ISO/IEC 15939: Systems and Software Engineering-Measurement Process

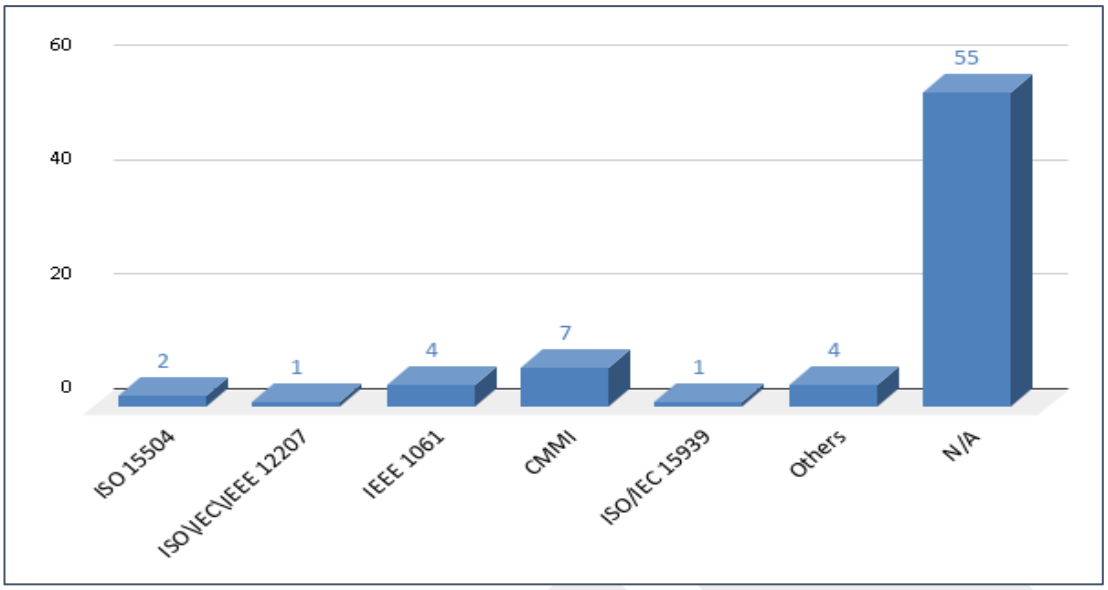


Figure 4.21 Distribution of Papers Content Related to Standards, Model

- **Others:** ITIL (Information Technologies Infrastructure Library), ISO\IEC 14598: Information Technology-Software Product Evaluation, ISO\IEC 14764: ISO/IEC 14764 Software Engineering-Software Life Cycle Processes, 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. As mentioned before, for these reasons, researchers work on the Software Component Quality Characteristics Model (SCCM).

Authors of paper [S55] mention that based on the proposed CQC Model, SCMM will be established for certification levels via certifying the quality components.

None of the papers mention the following standards and models during their measurement and analysis process:

- ISO 9001 Quality Management Systems Standard, Section 8 Measurement, Analysis and Improvement
- COBIT (Control Objectives for Information and Related Technologies) is a set of best practices for information technology management that have been

prepared by the collaboration of ITGE (IT Governance Institute) and ISACA (Information Systems Audit and Control Association) [64].

- ISO\IEC 33001: IT Process Assessment-Concepts and Terminology

SM-RQ 2.8: Which quality models have been used in papers to measure software quality?

66% of papers have no data regarding the quality model used while developing or improving quality metrics as shown in Figure 4.22. Still most frequently used quality model is ISO 9126, although ISO 25010 standard was published in 2011. Others are marked as Design Quality Model (DQM), QMOOD, Pragmatic Software Quality Model, FURPS.

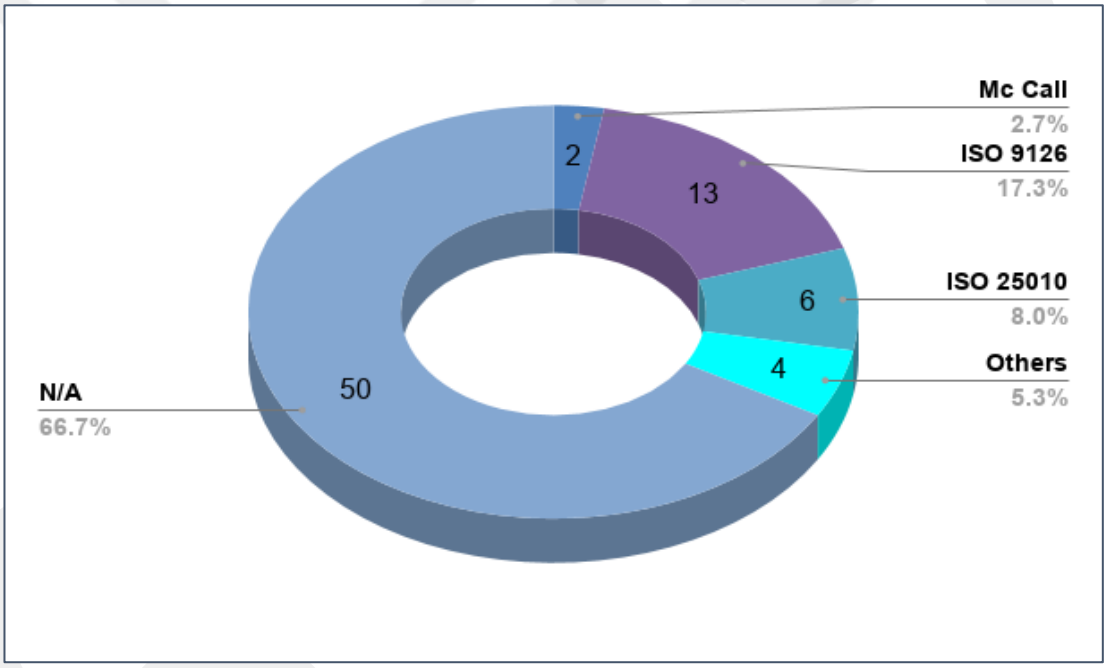


Figure 4.22 Distribution of Papers Content Related with Quality Models

SM-RQ 2.9: Which types of process, product, and project metrics are mostly focused on measuring quality?

The answer to the question of which papers mostly focused on which type of metrics to increase quality, is presented in Figure 4.23. 45 of the metrics are used to measure the quality of a product; with 15 of them for measuring the process, and 33 of them

for the projects. Most of the papers mention the Product-Project quality metrics, too, since project metric automatically affects the product metric, or vice versa.

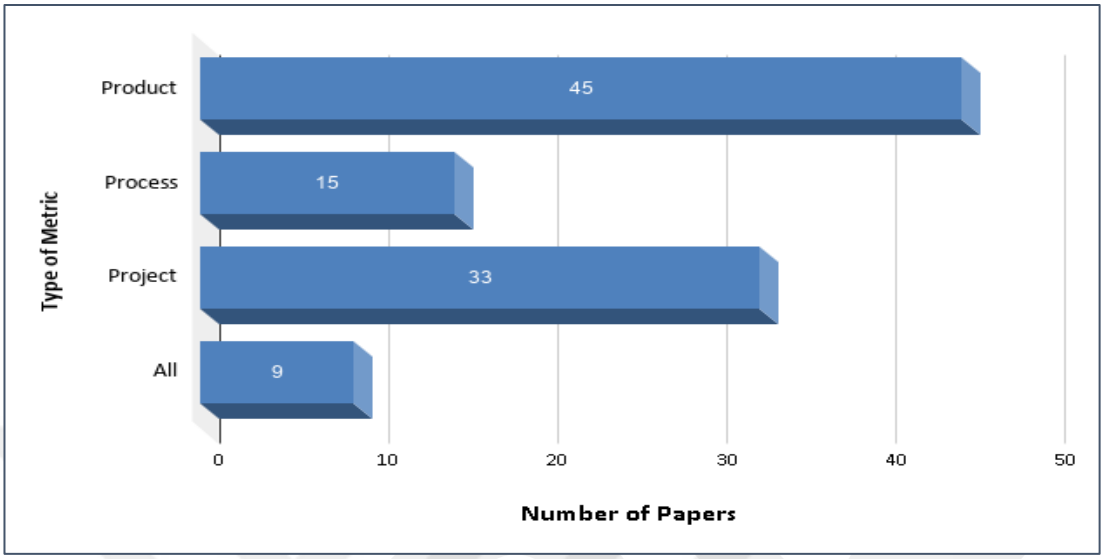


Figure 4.23 Numbers of Papers for Type of Metric

SM-RQ 2.10: In which level software quality metrics have been measured?

Class Level, Method level and Process Level metrics have been developed, suggested, validated and improved by the authors between 2009 and 2019 as presented in Figure 4.24**.** Only two papers refer to the SDLC model Agile and rest of the papers do not mention SDLC phases. 21 out of 70 papers suggest or mention the importance of threshold value for metric analysis.

The list of metrics extracted from the 70 papers in this research pool can be seen in detail from SLR Repository in Google Drive, the link to which can be found in given at Appendix A. Also, there are two new metrics that are generated in through the novel approach manner.

RFC and WMC are significant indicators of change proneness metrics as they are selected after applying the feature subset selection method CFS.

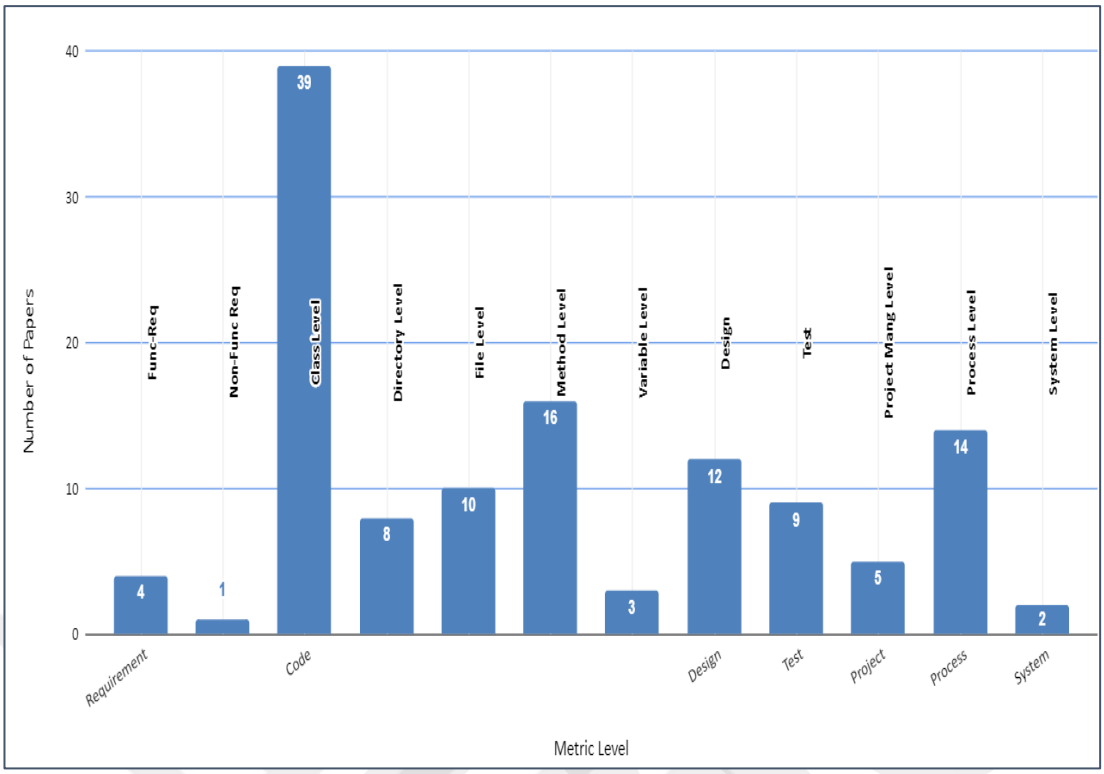


Figure 4.24 The Number of Papers for Metric Level

SM-RQ 2.11: How many papers mentioned the tool usage for measuring software quality?

NTS: No tool support, TS: Tool support, Others: Mentioned tool usage but not used (Awareness of tools)

52,9% of papers have been found to use tool supports while measuring data and 41,4% of them neither used or nor mentioned tool support as presented in Figure 4.25. 5,7% of papers mentioned the importance of tool usage for measuring data. In total, tool usage awareness rate for measuring software quality metrics is 58,6%.

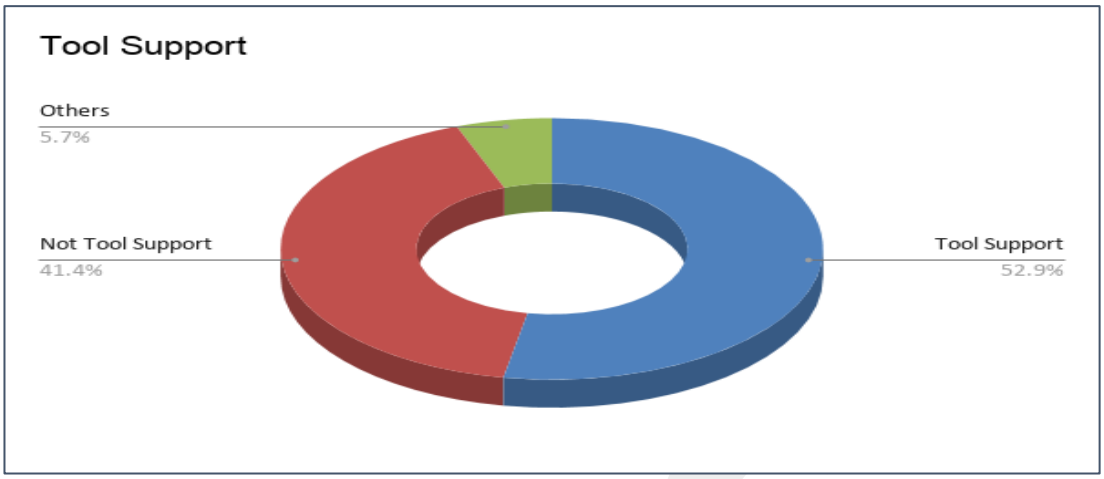


Figure 4.25 The Distribution of the Number of Tool Supports Studied in Papers

Figure 4.26 shows the distribution of the Data Quality (DQ), Quality in Use (QinU) and Software Product Quality (SPQ) quality attributes of ISO 25010 Quality Model in our paper pool. 71% (66 papers) of the papers designate the metrics related to Software Product Quality.

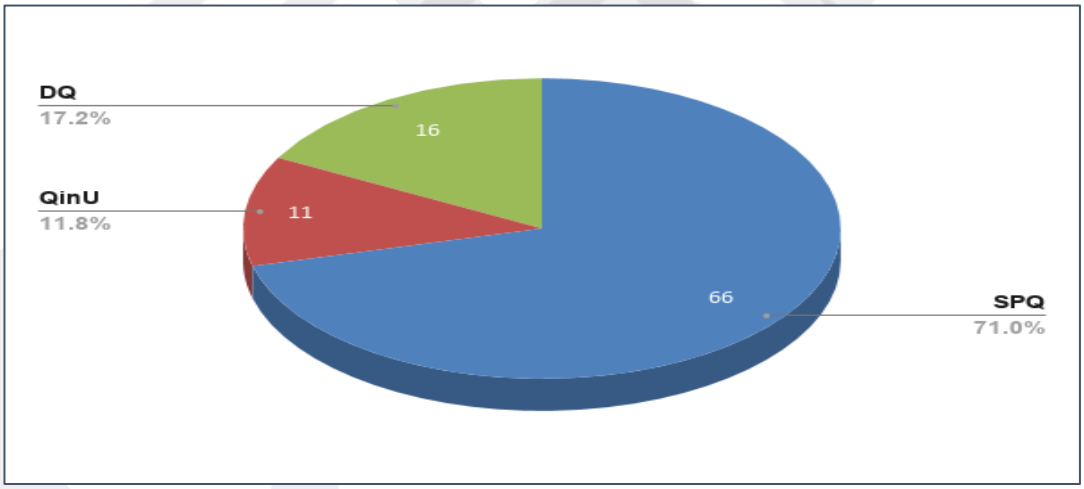


Figure 4.26 The Distribution of metrics for Quality Attributes

SM-RQ 2.12: Which quality attributes of "Software Product Quality (SPQ)" in ISO 25000 Quality Model is mostly measured in the papers?

As presented in Figure 4.27, most of the metrics used to measure Software Product Quality are related to Maintainability and Reliability quality attributes.

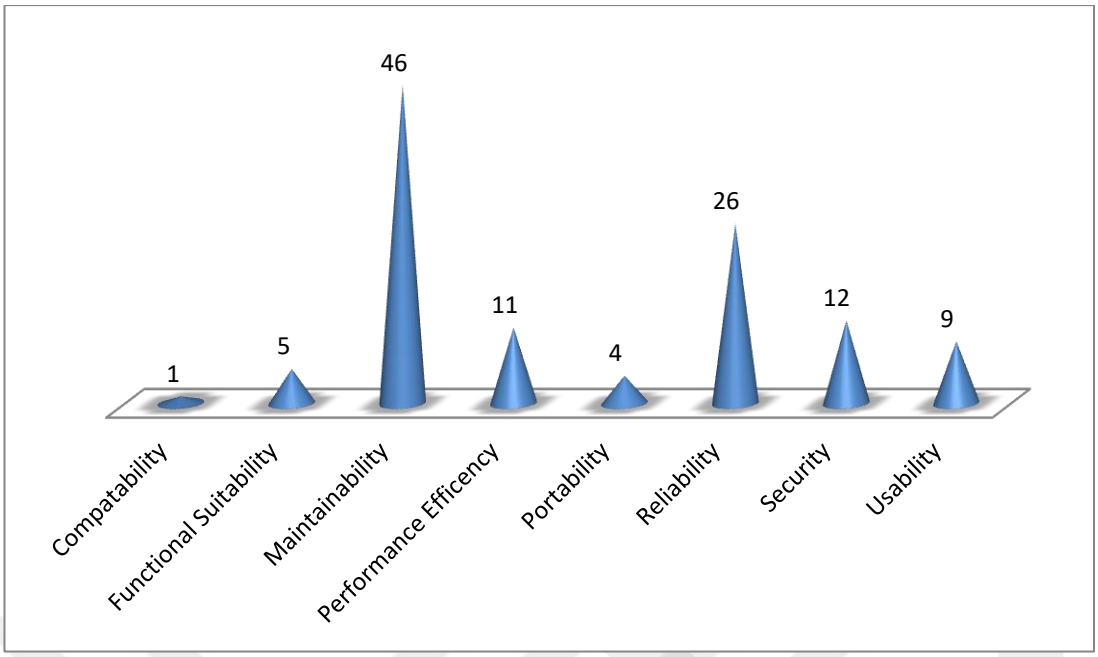


Figure 4.27 The Distribution of Quality Characteristics for SPQ Model

Traceability between SPQ attributes is presented in Table 4.6:

Table 4.6 Paper References for SPQ Characteristics

Name of Quality Characteristics	Citation Count	References of Papers
Compatibility	1	[S51]
Functional Suitability	5	[S4], [S11], [S50], [S51], [S60]
Maintainability	46	[S1], [S2], [S3], [S4], [S6], [S7], [S9], [S11], [S13], [S14], [S15], [S17], [S18], [S20], [S21], [S22], [S23], [S24], [S25], [S27], [S30], [S32], [S34], [S36], [S38], [S39], [S40], [S41], [S43], [S44], [S46], [S47], [S48], [S49], [S52], [S53], [S57], [S58], [S59], [S60], [S61], [S62], [S64], [S67], [S68], [S70]
Performance Efficiency	11	[S4], [S5], [S12], [S28], [S39],[S42], [S50], [S51], [S57], [S61], [68]
Portability	4	[S2], [S4], [S22], [S51]
Reliability	26	[S1], [S4], [S9], [S10], [S12], [S18], [S22], [S26], [S29], [S33], [S39], [S40], [S42], [S43], [S44],

		[S46], [S47], [S51], [S55], [S56], [S58], [S61], [S62], [S63], [S65], [S69]
Security	12	[S4], [S9], [S12], [S16], [S19], [S22], [S40], [S50], [S51], [S54], [S66], [S68]
Usability	9	[S4], [S12], [S22], [S45], [S50], [S51], [S52], [S53], [S67]

SM-RQ 2.13: Which quality attributes of "Quality in Use (QinU)" in ISO 25000 Quality Model is mostly measured in the papers?

As presented in the figure, top number of the metrics can be observed in sections Quality in Use is Freedom from Risk, Effectiveness and Satisfaction. Traceability between numbers of papers based on each quality attribute is given in Figure 4.28.

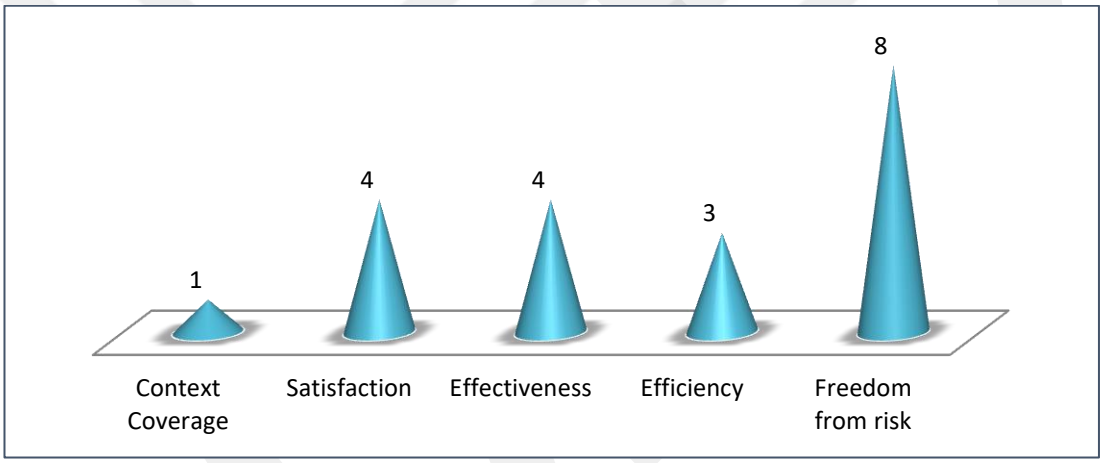


Figure 4.28 The Distribution of Quality Characteristics for QinU Model

Traceability between QinU attributes presented in papers is given in Table 4.7:

Table 4.7 Paper References for QinU Characteristics

Name of Quality Characteristics	Citation Count	References of Papers
Context Coverage	1	[S22]
Satisfaction	4	[S1], [S12], [S27], [S61]
Effectiveness	4	[S2], [S8], [S44], [S68]
Efficiency	3	[S8], [S22], [S44][S45]
Freedom from Risk	8	[S1], [S8], [S12], [S31], [S42], [S47], [S52], [S69]

SM-RQ 2.14: Which quality attributes of "Data Quality (DQ)" in ISO 25012 Quality Model is mostly measured in the papers?

Most often used data quality characteristics are Understandability and Accuracy as shown in Figure 4.29. There is no paper related to the usage of other data quality models like: Credibility, Currentness, Accessibility, Compliance Confidentiality, Efficiency, Availability, Portability, Recoverability.

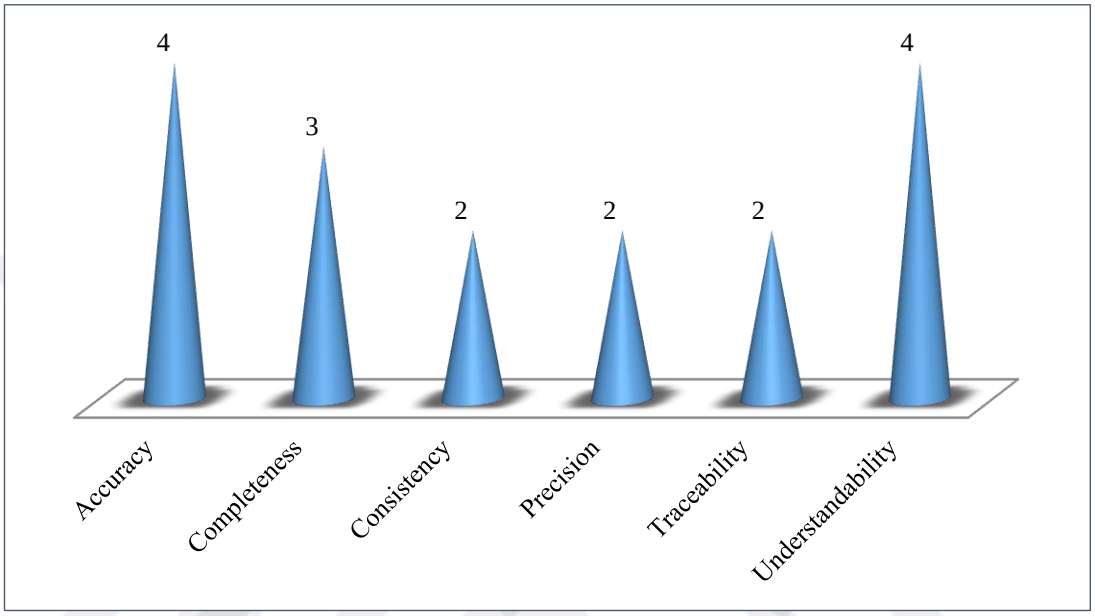


Figure 4.29 The Distribution of Quality Characteristics for DQ Model

Traceability between DQ attributes presented in papers is given in Table 4.8:

Table 4.8 Paper References for DQ Characteristics

Name of Quality Characteristics	Citation Count	References of Papers
Accuracy	4	[S26], [S31], [S35], [S58]
Completeness	3	[S3], [S11], [S39]
Consistency	2	[S16], [S64]
Precision	2	[S16], [S21]
Traceability	2	[S3], [S11]
Understandability	4	[S2], [S34], [S41], [S48]

SM-RQ 2.15: Which of the SDLC stages are most referred in the data in the papers?

Most of the software quality related metrics focus on code (implementation) and design phase to predict and avoid bugs in a proactive manner. In our paper pool, as shown in Figure 4.30, 6.6% of the papers have been found to be related to the test phase, aiming to eliminate bugs of software before its release to the users.

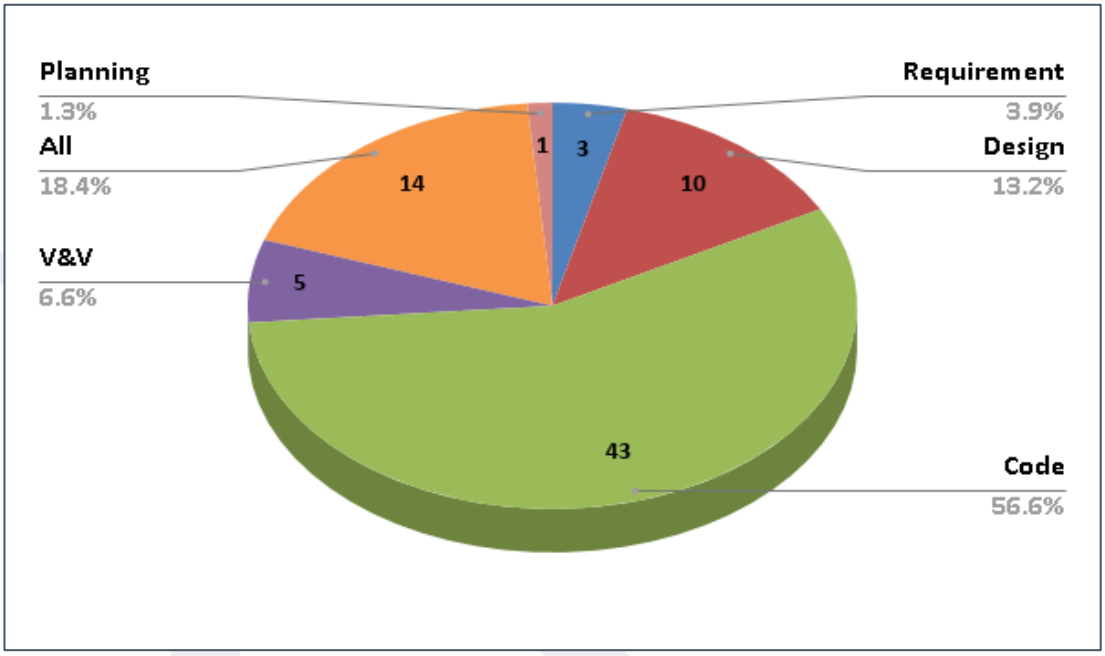


Figure 4.30 The Distribution of SDLC Phase for Papers

SM-RQ 2.16: Which application domains have been subject to software quality metrics in articles?

"Generic purposes" is the most referred metric with 41 papers, followed by 10 papers related to Mobile Application. 9 papers do not mention the application domain at all as presented in Figure 4.31.

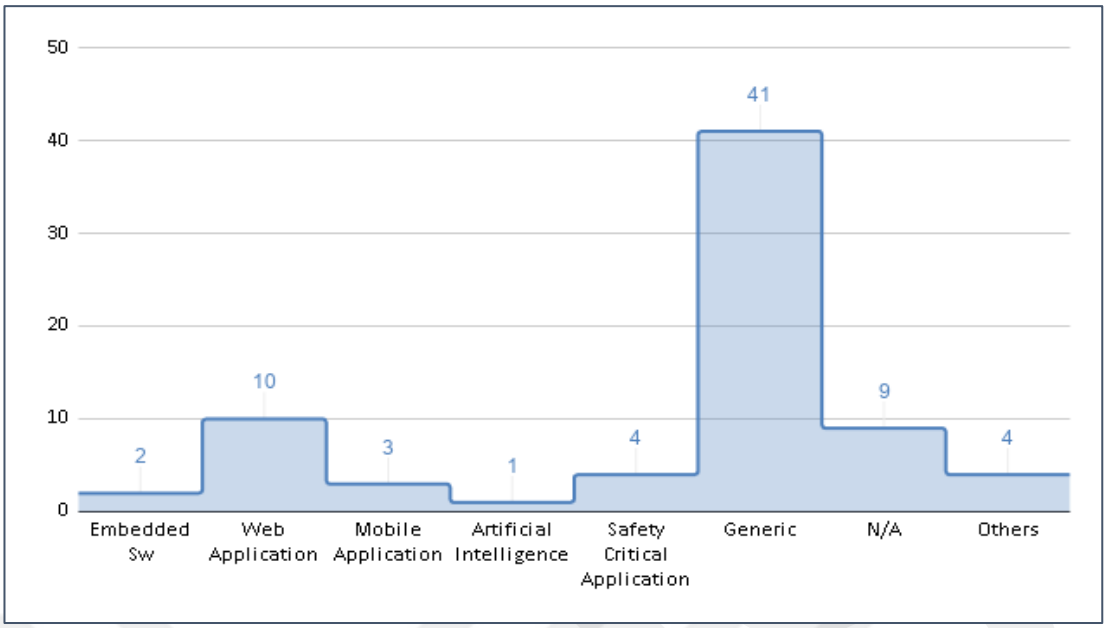


Figure 4.31 Application Domain vs. Number of Papers

The data regarding which quality attributes were used to measure the quality of which application domain is presented in Table 4.9. Also, paper references for metrics related application domains are given in below table.

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. As shown in Table 4.9, maintainability related metrics, presented as a common quality characteristic, were used to measure the quality of all type of applications except for the artificial intelligence. Furthermore, security metric was another common quality characteristic for all application domains except for embedded applications.

Table 4.9 Paper References for Application Domains

Quality Characteristics	**Application Domain**	**Citation Count**	**References of 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]

SM-RQ 2.17: Which software programming types have been subject to software quality metrics in articles?

SDLC Phase of 43 out of the 70 papers was revealed as in code level. 35 of the 43 papers have been subject to increasing the quality of Object-Oriented programming as extracted in Figure 4.32. 3 of them are related to Aspect-Oriented programming metrics, 3 of them are related to Basic Programming and 1 is related to Scripting

Programming for code phase. Finally, 5 of them are marked as others like Component Based Programming or Language Independent programs.

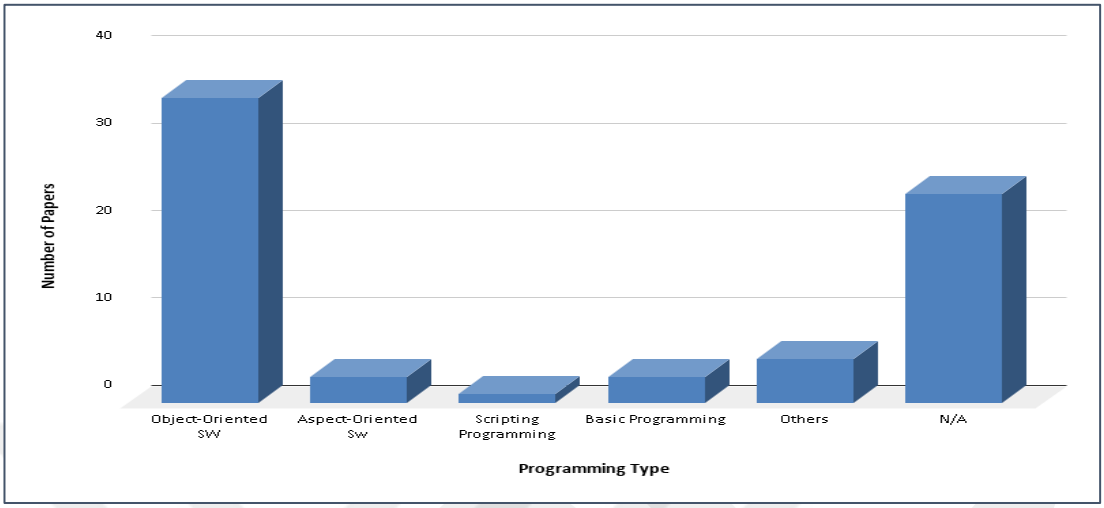


Figure 4.32 The Number of Papers for Programming Types

SM-RQ 2.18: Are there any software quality metrics specific to SDLC model?

Only, two following papers [S63] and [S10] mention the relation between metrics and Agile model. SDLC phase and steps were not used as a parameter of metric in most of the papers.

[S63] measures the quality of agile project with the following metrics:

- **Test Level:** Number of Tests, the Test-Coverage Rate and Number of Broken Builds.
- **Test Coverage Ratio**: (Branch, Line Coverage Types)= Code Covered By Tests / Complete Code: Brach and Line Coverage Types
- **Test-Growth-Ratio:** The number of tests must be expanded when the source code rises. Moreover, application of refactoring method should be taken into account, because this method might result in a decrease in the size of source code.

- **Broken Builds:** In agility, testing and fixing the broken builds has the highest priority. The number of integration or build fails shows the quality requirement has not been encountered.
- **Process Level:** 3C: In the process side, 3C (Continuous Integration, Continuous Measurement (Measure changes in code), Continuous Improvement) approach should be implemented in sequel.
- **Source metrics:** Total Lines (Java), Effective LOC (Java) without empty lines or brackets, Total Lines of the Java Server Pages, Number of the JUnit Tests Check style error/warning/info.

[S10] measures the quality of agile project with the following metrics:

Generic development metrics have been used but these have been re-evaluated according to activities of the agile model. Traditional metrics have been modified to be used for the agile model: None of test cases were calculated at the end of test case finalization. If bug origin is not eliminated, bug is not counted. If user story is eliminated, regression test in integration and system phase is not counted.

SM-RQ 2.19: Whether a threshold value is mentioned or not? Which papers mentioned the threshold values of metrics?

22 of the 70 papers mention the threshold values only in relation to their study or in a general manner, often in passing comments, or when they needed to refer to the necessity of the usage of thresholds. Proposed threshold values in the literature, are mostly based on the experiences and coding standards [S52]. Therefore, these threshold values might appear to be low or high, 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 the paper [S52]. Tool has been validated by using the Qualitas Corpus dataset which is the open source software systems collection for empirical studies for code products. Tool is publically available at the link: http://aserg.labsoft.dcc.ufmg.br/rttool/.

Calculation method of RTTOOL for extracting the relative threshold value and penalty rate when it exceeds the threshold is given in the paper [S52].

[S40] propose that their study can be a starting point for determining threshold values for design metrics to improve design practices.

SM-RQ 2.20: What are the future directions of current researches?

The discussion regarding the Future Plan is gathered under five road maps as

shown in Figure 4.33 developing a new tool, improving the current tool, developing a new technique, improving the current technique, and conducting research in a more detailed manner and finally, no future plan [65] .

Most of the papers planned the next step and give suggestions, which relates them to "Improve the Technique (39%), Make More Detailed Research (26%) by using new case studies." Further information regarding the details about the future plans laid out in papers is discussed more extensively in SLR-RQ 4 as well as in the future work section of this thesis.

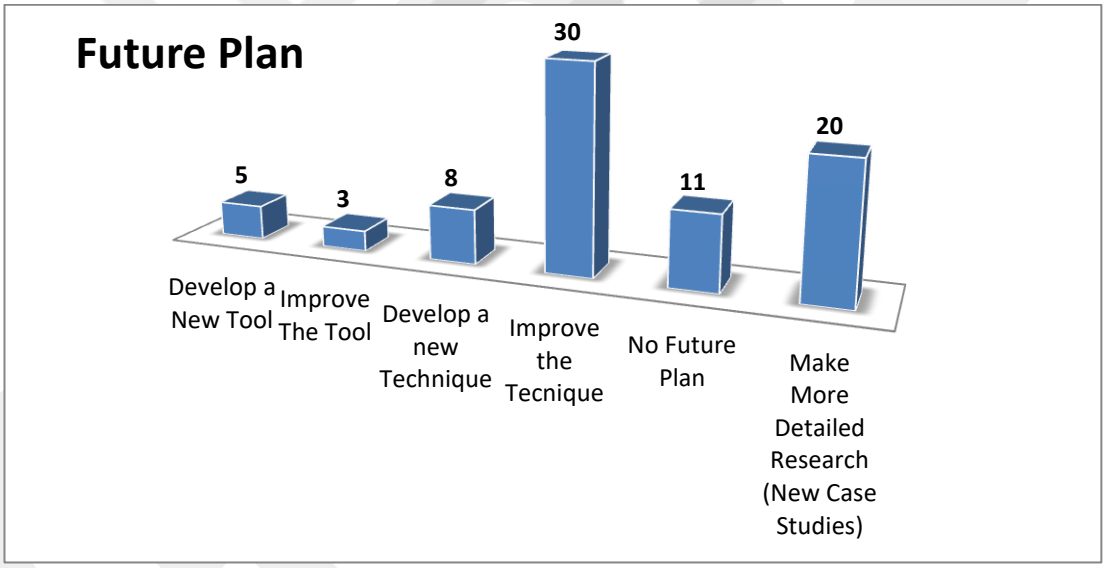


Figure 4.33 Future Plan Offered by the Papers

SLR-RQ 1: What are the software quality metrics presented in the papers?

SLR-RQ 1.1. Which metrics are presented under which metric levels and which are commonly mentioned in the papers?

Metric definitions per each metric level are presented in Table 4.10-Table 4.16

and summary of the 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 class-level.
- Brito e Abreu's MOOD metrics and Bansiya-Davis's QMOOD metrics are the ones that have been most often 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 paper [S3].
- Papers mostly mentioned the usage of coupling and cohesion metrics per measure the maintainability, understandability and reusability quality attributes and inheritance and complexity related metrics per measure the flexibility, understandability and reusability quality attributes.

Table 4.10 Code Level Metrics

Code Level Metrics		
Metric Type	Metric Definition	Paper References
Coupling	DAC (Data Abstraction), MPC (Message Passing Coupling) CM (Class-Method Interaction), GC (Global Coupling) IC (Inheritance Coupling), CBO, Ca (Afferent couplings), Ce (Efferent couplings), (I)Instability : Ce/Ca+Ce, CBM (Coupling Between Methods), ACF (Average Coupling Factor), AAC (Average Attributes Per Class), DCBO(Dynamic Coupling Between Objects), Dep_Out (# of elements on which this class depends), IC_Attr (# of attributes in the class having another class or interface as their type), Change propagation probability, IC_Par (# of parameters in the class having another class or interface as their type), Dep_In (# of elements that depend on this class),	[S13], [S14], [S17], [S18], [S23], [S31], [S32], [S44], [S47], [S48], [S57]

	EC_Attr (# of times the class is externally used as attribute type), EC_Par (# of times the class is externally used as parameter type)	
Cohesion	RCI (Ratio of Cohesive Interaction), H (The connected components formed by the classes and interfaces of the package), Conn Comp (This is the average number of internal relationships per class/interface/package), LCOM, TCC (Tight Class Cohesion), OMMIC (call to methods in an unrelated class)	[S6], [S17], [S18], [S20], [S28], [S31], [S34], [S44], [S47], [S57]
Size	LOC, Total LOC, NumAttr (# of attributes in the class), NumPubOps (# of public operations in a class), SZV – Size Variance (Original and Revised Baselines – SZVOB, SZVRB), MLOC, NOA(# of Attributes), NOC (# of Classes), NOI (# of Interfaces), NOM (# of Methods), NOPK (# of Packages), NOP (# of Parameters), NOSA (# of Static Attributes), NOSM (# of Static Methods), NAM (# of Attributes and Methods), NOC (# of Children), NOF (# of Fields), NOM (# of Methods), NORM (# of Overridden Methods, NOSF (# of Static Fields), NOSM (# of Static Methods), NOVM (# of Overridden Methods)	[S1], [S10], [S13], [S18], [S20], [S27], [S31], [S34], [S36], [S39], [S44], [S47], [S56], [S57], [S61], [S66]
Complexity	• Cyclomatic Complexity, Halstead complexity, McCabe Cyclomatic Complexity, Modified Cyclomatic Complexity, Strict Cyclomatic Complexity, Essential Cyclomatic Complexity, CountPath Complexity, Nesting Complexity, Nested Block Depth, Modified Complexity, Struct Complexity, • MsgSelf (The number of messages sent to instances of the same class), • R (The number of relationships between classes and interfaces in the package), • MC (Method Complexity), CWC (Coupling Weight For A Class), AC (Attribute Complexity), CLC (Class Complexity), CC (Code Complexity), AMC (Average Method	[S1], [S9], [S13], [S14], [S16],[S20], [S26],[S31], [S36], [S39], [S44], [S47], [S56] [S57], [S62], [S66], [S69],

	Complexity), AMCC (Average Method Complexity Per Class), ACC (Average Class Complexity), AVG_CC (Average Cyclomatic Complexity), MAX_CC (Maximum Cyclomatic Complexity)	
CK Metrics	RFC (Response for a Class), WMC (Weighted Methods per Class), NOC (# of Children), DIT (Depth of Inheritance Tree), CBO (Coupling Between Object classes), Lack of Cohesion in Methods (LCOM)	[S14], [S18], [S20], [S24], [S25], [S30], [S31], [S34], [S38], [S39], [S40], [S48], [S49], [S53], [S56], [S60], [S62], [S67]
MOOD and Other Metrics	• MOOD Metrics • Kid Suite Metrics • Chen & Lui Metrics • QMOOD Metrics: ANA (Average Number of Ancestors), CAM (Cohesion), NOM (Complexity), MOA (Composition), DCC (Direct Class Coupling), DAM (Data Access Metric), NOH (# of Inherited Class), MFA (Measure of Functional Abstraction), CIS (# of Services provided by Class), NOP (Change one object with another performance at runtime)	[S2], [S30]
Comment	CP (Comment Percentage), CCR (Comments to Code Ratio)	[S20], [S39]
Defect	BLOC (Bugs per Line of Code), DD (Defect Density), FD (Fault Days Number), Fault-Prone Metric, FR (Failure Rate), DRE (Defect Removal Efficiency), DDD (Delivered Defect Density), IDD (Internal Defect Density)	[S29], [S44], [S61], [S69]
Inheritance	NOC (# of children of the class), NumDesc (# of children of the class), NumAnc (# of parents of the class), AttrInh (# of inherited attributes), OpsInh (# of inherited operations), AdIP (Advice Inheritance Factor), PIP (Pointcut Inheritance Factor), AttIF (Attribute Inheritance Factor), AlF (Aspect Inheritance Factor), OIF (# of inherited operations divided by the total number of available operations)	[S15], [S47], [S48]

Table 4.11 Requirement Level Metrics

Requirement Level Metrics		
Metric Type	Metric Definition	Paper References
Functional, Non-Functional Requirement	Desirability Metric-Requirement, NASA MDP projects requirement metrics, RT Data (Requirement Traceability), RSCR (Requirement Specification Change Request), RV (Requirements Volatility), Traceability metrics between Business Requirements, Project and Security Requirements	[S12], [S33], [S44], [S50], [S61]

Table 4.12 Design Level Metrics

Metric Definition	Paper References
• Readability of the Project: Proportion of functional elements with meaningful names • Comprehensibility Interface: Proportion of functional elements utilized without errors • Understandability of incoming and outgoing messages: Proportion of correctly understood exceptions/returned values/arguments • Easy learning: Average time for learning/master the usage of the component, • Customization interface: Configurable parameters for the interface • Quality of error messages: Error messages by functional element density, Clearness of error messages • Interface Density: Operations/Events interface density • Design quality metrics and design quality model was proposed based on the fundamentals of following design principles: Single Responsibility Principle (SRP), Separation of Concern Principle (SOC), Information Hiding Principle (IHI), Don't Repeat Yourself Principle, Open Closed Principle (OCP), Acyclic Dependency Principle (ADP), Interface Segregation Principle (ISP), Liskov Substitution Principle, Self-Documentation Principle (SDOP), Favour Composition over Inheritance (FCOI), Interface Separability (ISE), Stable Dependencies Principle, Law of Demeter, Command Query Separation (CCS), Common Closure Principle (CCP). • ANMCC (Average Number of Modified Components per Commit), IPCI (Index of Package Changing Impact), IPGF (Index of Package Goal Focus), Index of Inter-Package Usage (IIPU), Index of Inter-Package Extending (IIPE), Index of Inter-Package Usage Diversion (IIPUD), Index of Inter-Package Extending Diversion (IIPED) • CEG (Cause and Effect Graphing),	[S3], [S7], [S9], [S21], [S43],[S44], [S45],[S47], [S51],[S57], [S61], [S70]

• SDC (System Design Complexity), Ease of Change in Design • Gang of Four (GoF) design patterns and conviction metrics • NMH (Number of multiple hierarchies), APMH (Average number of alternate paths in multiple hierarchies), ALMH (Average number of levels in multiple hierarchies), SMH (Number of shared multiple hierarchies), ADMH (Average number of dimensions participating in shared multiple hierarchies)	

Table 4.13 Test Level Metrics

Metric Definition	Paper References
• Unit Test: Percentage of Test Coverage, Brach and Line Coverage Ratio, # of test cases, # of tested classes, test density per class, • Integration Test: # of bugs, # of test cases, • Regression Test: # of bugs, # of test cases, • System Test: # of bugs, # of test cases, • Security, performance, functionality and usability test metrics to measure the quality of requirements, user interface and source code, • Test Case Sufficiency Metric: measure the sufficiency of test cases in relation to size of the source code, • NPATH: metric counts the number of execution path through a functions, • Test-Growth-Ratio: measure the growth of the test in relation to the growth of the source code, • Identification of test modules and testing effort according to risk metric levels of tests,	[S1], [S10], [S19], [S44], [S50], [S51], [S60], [S61], [S62], [S63]

Table 4.14 Project Management Level Metrics

Project Management Level Metrics		
Metric Type	Metric Definition	Paper References
Attributes of Users	Project Manager' Quality based on cost, schedule pressure, cycle time of the project with relative to the project size metrics, Developer's Perspective Quality based on Portability, Maintainability, Efficiency, Functionality metrics, User's Perspective Quality based on the Reliability, Usability Metrics. To analyze the input of metrics and calculations in detailed, please look at the paper references.	[S6], [S18], [S24], [S44], [S51], [S57], [S65], [S68]
Effort	Average effort (hours) spend for maintainability,	

	Predictor of quality= Average # of Change, EV – Effort Variance (Original and Revised Baselines, EVOB, EVRB), PR – Productivity	
Performance	Energy Consumption Measurement: Worst Case & Best Case Energy Consumption, Worst Case & Best Case for Data Memory	
Schedule	SV – Schedule Variance (Original and Revised Baselines – SVOB, SVRB), SPI – Schedule Performance Index	
Progress	EVA (earned value analysis) metrics to monitor activities, frequency and latency of the measurement data, Estimation of effort, cost and budget of software projects.	
Budget/Cost	• Budgeted Cost of Work Performed (BCWP), Actual Cost of Work Performed (ACWP), COCOMO, Function Point, • Attack Cost (Security),Counter measure Cost (Security), Loss/Damage Cost (Security),Return on Security Investment (ROSI) (Security), • Reduce Testing Cost: by Predicting Change-Prone Classes at early phases of the life cycle, • Cost of metric collection, • CV (Cost Variance) , CQ (Cost of Quality), CVE (Cost Variance Earned), SVE (Schedule Variance Earned), CPI (Cost Performance Index)	

Table 4.15 System Level Metrics

System Level Metrics		
Metric Type	Metric Definition	Paper References
Internal Metrics	Individual engineering practices: static analysis, code review, requirements management, unit test, defect density	[S1], [S46]
Customer Experience	CFD (Customer-Found Defects), CFR (Customer-Found Regressions) , CFD MTTR, SWDPMH (Software defects per million hours usage per month)	
Customer Perception	Cisco software and hardware, customer perception metrics and other forms of customer feedback ratings and comments.	
Reliability	Probability of failure-free operation over a specified	

	time interval, MTTF (Mean time to failure), Expected number of failures per unit time interval (failure intensity), MTTR (Mean Time To Repair), MTBF (Mean Time between Failures), POFOD (Probability of Failure on Demand), ROCOF (Rate of Occurrences of Failure)	

Table 4.16 Process Level Metrics

Metric Definition	Paper References
• To increase the vocabulary diversity in quality standards and models, Unified Foundational Ontology is presented for the measurement procedure related to parameters like standards, models, procedure, organization scope, analytical method, operational definition of method. • Inspection Process Effectiveness Metric: DI = Number of defects captured by inspection process (Ni)/Total number of defects captured by both inspection and testing (Td) • Inspection Process Performance Metric: Inspection performance metric (IPM) = Number of defects captured by inspection process (Ni)/Inspection effort (IE) • User Awareness Metrics: Measurement of the difference between perspective and awareness level of the developer, user and managers about metrics and measurement processes. • User Perspective Metrics: o Managements/Developers perspective to highlight the importance of various attributes of testing; o Testers/Team Members perspective, to assess functioning and working of each of the attribute • Pragmatic Quality Factor Measurement Parameters: consists of four main components: 1) behavioral attributes, 2) impact attribute, 3) responsibility and measurement of metrics and 4) classification of attributes and weight factors. • SDLC (Waterfall, Iterative, Spiral, and Unified) model, phase and breakdown structure (Iterations, Phases, and Activities) related metrics (paper 48) • New quality characteristic proposed to measure the Marketability with sub-characteristics: Development Time, Cost, and Time to Market, Targeted Market, and Affordability. • Newly added sub characteristics under ISO9126-product quality attributes for component based software:	[S4], [S5], [S8], [S29], [S35], [S37], [S42], [S50], [S51], [S53], [S55], [S61], [S64]

○ maintainability: diagonoseability, repairability, expandability, recoverability ○ usability: configurability, monitorability ○ efficiency: repeatability • Proposed framework to measure ISO QinU quality attributes, • T model was proposed to design and test software to ensure full test coverage of software application by focusing on the Safety, Scalability, Stability, Serviceability. T-Model parameters consists from the followings: definition of roles related to metrics for each personnel in team, definition of performance and security, defect measurement tool efficiency, analyze architecture, database and scope of application, verify deployment and measurement tools, all plans related to scope, cost, schedule, effort, resources, risk.	

SLR-RQ 1.2. Which metric levels can be applicable for which application domains?

Data regarding the relationship between the metric levels and which quality characteristics were measured in each application domains is given in Table 4.17:

Table 4.17 Metric Levels per Application Domains

Application Domain	Quality Characteristics	Metric Level	Programming Languages
Safety Critical Systems	Satisfaction, Usability, Performance Efficiency, Freedom from Risk, Security, Reliability, Context Coverage, Maintainability, Portability, Reliability, Effectiveness, Consistency, Precision	Directory, Functional, Non-Functional Requirement, Design, Test, Project Management, Class, File Level,	Object Oriented, N/A
Embedded Software Application	Maintainability, Performance Efficiency, Reliability	Class, File Level, Method, Design, Variable, Package, Directory, Project Management Level	Object Oriented, N/A
Web Application	Maintainability, Usability, Security, Reliability, Effectiveness, Freedom from Risk, Performance Efficiency, Functional Suitability	Class, Method, Process, File, Directory, Test, Project Management Level	Object and Aspect Oriented, N/A
Mobile Application and Artificial Intelligence	Maintainability, Usability, Security, Performance Efficiency, Functional Suitability	Class, Method Process, File, Test Level	Scripting Programming, N/A

In our paper repository, common metrics used for Web Application domains were observed to be as follows: CCR (Comments to Code Ratio), NODBC (Number of Data Base Connections), WMC (Weighted Methods per Class) and LCOM [S20].

MOOD and QMOOD metrics are applicable for the Object Oriented and Aspect Oriented.

Following metrics were suggested in the papers to measure application domains for directory and class level:

- The directory level metrics: total LOC, physical LOC, Number of Statements, Number of Blank Lines, Number/Percentage of Comment Lines, Count of all lines Non-comment, Non-blank (NCNB) and Executable Statements (EXEC).
- Class level metrics are mostly given as WMC, RFC, LCOM, CBO, 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), AAC (Average Attributes Per Class).
- McCabe's and Halstead's metrics represent quality at a method-level, while Chidamber and Kemerer's (CK) metrics represent quality at class-level.

SLR-RQ 1.3. What metrics can be applicable for which programming type?

- Generic Source Code Metrics were suggested as; LOC, NOM, NOF, CC, LCOM, Number Lines Of Comment, Percentage Comment, CA, CE and ABC (Association Between Class) [S60].
- In paper [S20], following metrics were proposed as suitable for Object-Oriented and Aspect-Oriented Programs: Maintainability Index (MI), DIT, CBO and LOC.
- In paper [S21], following metrics are suggested as Class Level, Object-Oriented metrics: CA, CE, DIT, LCOM, MLOC, NBD, NOC, NOF, NOM, NORM, NSC, NSF, NSM, PAR, RMD, WMC and McCabe Complexity.
- Paper [S30] mentioned the DIT and LCOM metrics, which are unsuitable for OO design. NOC cannot be used to predict the fault proneness in all circumstances.
- 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 being fully adopted in the design of class libraries [S67] .

SLR-RQ 2- Which quality characteristic of ISO/IEC 25010 Quality Model is used for satisfying Product Quality (SPQ), Quality in Use (QiU), and Data Quality (DQ)?

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

- Coupling and Cohesion metrics are mostly used in the measurement of the maintainability, understandability and reusability quality attributes.
- Inheritance and Complexity metrics are mostly used to measure the Flexibility, Fault Proneness, Understandability and Reusability quality attributes.
- Each project has different scope and dynamics, so there is a need for the historical data of the projects. If the projects have version history, the combination of more accurate and suitable quality metrics can be found by analyzing the link between the historical metric data sets and results [S24].
- Paper [S24] presents the WMC, LOC and RFC metric combination as applicable to many various types of the software projects for predicting the change-prone classes to reduce cost of testing activity.
- Paper [S20] proved that there is a strong correlation between the NODBC (Number of Data Base Connections), CCR (Code to Comments Ratio) metrics and maintainability for data intensive applications.
- Paper [S30] laid out that five metrics of MOOD suite are appropriate for quality measurement and CF (Coupling Factor) metric is not suitable for quality measurement. Coupling Factor can be measured by the ratio of the possible number of couplings in the software to the actual number of couplings.
- Maintainability metric of the product can be measured as mentioned in paper [S32] by giving different weights for each of quality attributes for analyzability, changeability, stability and testability.

- In paper pool of this study, 4 (four) new metrics were offered. The relations regarding which metrics were proposed in which paper to measure which quality attribute is given in Table 4.18.
- Examples for quality characteristics related metrics were presented in Table 4.19. In ISO 9126 quality model, "Security" was the sub-characteristics under the "Functionality" quality characteristic. In 2017, popular restaurant searching platform Zomato, 99 hospitals and IT Companies in USA, Russia, China, UK and India suffered from security breach and were hacked due to security vulnerability [S50]. Because of the security problem in software systems of critical organizations, people's lives might be affected negatively. With the increasing importance of the security, "Security" is defined as one of the quality characteristics in ISO 25010 quality model, which is the updated version of ISO 9126.
- List of the security and maintainability metrics is presented in Table 4.18 and Table 4.19.
- List of metrics based on levels, were given in Table 4.10-Table 4.16. Traceability of quality attributes in each paper is presented in Table 4.6-Table 4.8. By examining these two set of tables given above, researchers can select the most appropriate metrics for their projects or research.

Table 4.18 Lists of New Metrics

Ref.	Quality Characteristics	Definition of Metrics
[S12]	Satisfaction, Usability, Performance Efficiency, Freedom from Risk, Security, Reliability	Desirability Metric for Requirement: By modifying the parameters of the desirability functions, quality and priority of requirements can be evaluated via the consideration of prioritized quality attributes that are necessary for different software domains.
[S20]	Maintainability	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).

[S25]	Maintainability	Change Index (CI): Metric is derived by using CK metrics and GEP Algorithm (Gene Expression Algorithm) CI = ((((0.24 * RFC) + (2 * SLOC)) – WMC) – 59.62) – LCOM
[S54]	Security	New Security metric parameters based on representative weaknesses of the system: percentage of the security weaknesses occurred, frequency of occurrences and risk degree.

Table 4.19 Quality Metrics Examples

Quality Characteristics	Metric Definition	Paper References
Security, Reliability	• Access Vector (AV), Access Complexity (AC), Authentication (Au), Confidentiality Impact (CC), Integrity Impact (IC), Availability Impact (AC), • Howard's measurement method, Attack Probability, Frequency of Attack, Vulnerability Count Per File • Attack Surface Metric Parameters as given below: o Entry and exit points of applications (config files, user interface, server, log files): entry/exit point framework to identify the relevant resources that contribute to a system's attack surface. o System Channels to connect in entry and exit point, acts of persistent data items, access rights and privileges for method and channel.	[S19], [S54], [S66]
Satisfaction, Maintainability, Reliability, Performance Efficiency	Extendibility formula = (0.5*Abstraction)-(0.5*coupling) + (0.5*inheritance) + (0.5*polymorphism) Effectiveness formula= (0.2*abstraction) + (0.2*encapsulation) + (0.2*composition) + (0.2*inheritance) + (0.2*polymorphism) Functionality formula= (0.12*cohesion) + (0.22*polymorphism) + (0.22*messaging) + (0.22*design size) + (0.22*hierarchies)	[S61], [S70]

	Understandability formula= (-0.33*abstraction) + (0.33*encapsulation)- (0.33*coupling) + (0.33*cohesion)- (0.33*polymorphism)-(0.33*complexity)-(0.33*design size) Flexibility formula= (0.25*encapsulation)-(0.25*coupling) + (0.5*composition) + (0.5*polymorphism)	

SLR-RQ 3 - Which software quality metrics/measurement management tools are focused on the articles and conference papers?

Although there is a lack of the 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 list below.

List of quality metric/measurement tools mentioned in paper repository is given in Table 4.20.

Table 4.20 The List of Metric Tools

#	Name of Tool	Properties	Development Style
1	ASSIST [66]	Integrated tool was developed by CMMI Level 3 organization in Turkey. This tool was designated based on the GQ(I)M approach to measure project management, issue tracking and enterprise resource planning (ERP) related metrics.	In-House Tool
2	CCMetrics [S17]	Tool was developed by authors of papers which calculate package level and class level metrics for object-oriented software automatically.	In-House Tool
3	Security Metric Tools [S68]	These tools mostly operate at a syntactic level, with very narrow capabilities in respect to security logic. Tools are only available for software products, but not for process improvement.	Commercial, In-House Tool
4	Analizo [S27]	Analizo is a multi-language source code analysis and metric extraction tool which was designed to support source code parsing in different languages and to report useful information about it. Calculates 14 source code metrics by extracting from C, C++ and Java source code files and generates dependency graphs.	Free http://www.analizo.org/

5	ModularityCalculator, CommitAnalyzer [S3]	These tools were developed by authors to calculate the modularity metrics and ANMCC (Average Number of Modified Components per Commit).	Free
6	No Name- In-House Tool [S20]	Tool was developed on visual studio to collect four metrics namely Code to Comments Ratio (CCR), Number of Data Base Connections (NODBC), Weighted Methods per Class (WMC) and Lack of Cohesion of Methods (LCOM).	In-House Tool
7	RAF Tool [S21]	Tool realizes the identification of methods, classes and packages with anomalous measurements of object-oriented software metrics. Tool generates report by including the metrics for the classes or methods fit the detection strategy and bad smells detection data.	Free
8	ckjm Tool [S31]	Tool calculates CK metrics for object-oriented software.	Free, https://www.spinellis.gr/sw/ckjm/
9	iPlasma Tool [S32]	Tool is an integrated platform for Quality Assessment of Object-Oriented Design. This tool analyzes the large-scale projects with huge LOC.	Free, http://loose.utt.ro/iplasma/iplasma.zip
10	E-Quality [S24]	Software quality visualization tool generates a graph based on object-oriented which extracts quality metrics and class relations from Java source code automatically. Also, it visualizes them on a graph-based interactive visual environment and permits users to realize the big picture by bringing class metrics and class relations together.	In-House Tool
11	Aspect Oriented Software Reusability Measurement (AOSRM) Tool [S15]	Tool was designed and developed by the authors of paper to measure inheritance metrics and compare results across versions.	In-House Tool
12	CISCO Web Based Software Quality Dashboard [S1]	Tool was developed by Cisco NSSTG (Network Software and Systems Technology Group) to analyze software quality metrics with business units (BU). Tool includes the 16 different dashboards; including software views at the organization, product, release, and component levels. Moreover, several other related views, like: BU quality plans, cost of poor quality, hardware quality, software engineering practices adoption and effectiveness and customer satisfaction.	In-House Tool

13	Understand (SCI) Tool [S38]	Collects software code metrics.	Commercial https://scitools.com/
14	ConQAT [S11]	Integrated external analysis tool for building quality dashboards based on the Quamoco model.	Free https://www.cqse.eu/en/blog/conqat-end-of-life/
15	MUSE (Muse Understand Scripting Engine) [S2], [S43]	Tool is a Perl library designed to measure object-oriented design by classifying violations of design best practices based on meta-information from source code.	Open-Source, Free
16	RMC [S41]	Tool support the execution of quality measurements and refactoring as an IDE for Eclipse.	Commercial
17	TestMaster (Teradyne TestMaster) [S44]	Tool helps to develop functional tests and also provide metric collection supportive. This tool began to be replaced by the using some popular commercial modeling tools which uses EFSM (Extended finite state machine model) construction.	This tool is no maintained by the company nor publicly accessible.
18	SD (Software Development) Metric Tool [S47]	Tool supports the measurement of several UML diagrams (activity diagrams and use case diagrams). Moreover, it supports the measurement of changes between two versions of UML models developed for the same system.	Commercial https://www.sdmetrics.com/Demo.html
19	ES2 [S47]	Tool can be used for gathering interface object-oriented metrics. These metrics calculates from interface specifications, which are available at the design stage of a project. Tool works for the C++ and Java languages.	Free
20	SAAT (Software Architecture Analysis Tool) [S47]	Calculates the rules and metrics automatically given an UML model. To identify defects of UML models, it checks the consistency and completeness rules to identify defects in UML models.	Free
21	SARA (Software Architecture Risk Assessment) [S47]	Provides software engineers and developers the ability to compute and analyze different architectural risk factors of software architectures modeled using UML. These risk factors include maintainability-based risk, reliability based risk, performance-based risk, and requirement-based risk.	In-House Tool
22	Quality Metrics Tool [S70]	Tool was developed by the authors of papers to assess the quality of the java applications.	In-House Tool

23	RTTool [S52]	Open source and publically available tool that generate relative thresholds for the source code metrics.	Open Source http://aserg.labsoft.dcc.ufmg.br/rttool
24	SonarQube, PMD [S53]	Tools which are static code analysis tool to increase code quality based on coding standards and some thresholds.	Open Source https://www.sonarqube.org/ https://pmd.github.io/
25	CA Clarity, HP PPM, IBM Rational ClearQuest, HP Quality Center, Bugzilla, Mantis, Borland CaliberRM, IBM Rational RequisitePro [S61]	Project management: CA Clarity, and HP PPM for project schedule, duration, effort, and cost Defect control and tracking solution: IBM Rational ClearQuest, HP Quality Center, Bugzilla and Mantis Requirements: Borland CaliberRM, IBM Rational RequisitePro	Commercial
26	SPADE, Cinderella, [S57]	SPADE: extracts physical metrics from C/C++ source code. Cinderella: estimates performance metrics from compiled C code.	Cinderella: Free, SPADE: In-House Tool,
27	Cobertura [S63]	Tool was developed to measure the test coverage metrics.	Free http://cobertura.github.io/cobertura/
28	Continuous Integration Engine Cruise Control [S63]	Tool reports the number of number of tests, Broken Builds and Test-Coverage metrics.	Free http://cruisecontrol.sourceforge.net/
29	Cockpit [S63]	Tool puts the measurement into graphs to see the changes of the measures over time.	In-House Tool
30	VizzMaintenance Tool [S40]	Software metrics which related with product quality is extracted by this tool.	Free
31	R-Cover [S65]	A tool which automates size measurement estimation for COSMIC functional size measurement method. The tool takes the business process models as input and creates estimation of COSMIC functional size measurements.	In-House Tool
32	EMERALD (Enhanced Measurement for Early Risk Assessment of Latent Defects) [S29]	Tool was developed by BNR, NORTEL, and Bell Canada to integrate software measurements. Software measurements were recorded using the software metrics analysis tool, which includes software-measurement facilities and software quality models. Tool provides recommended project-specific thresholds.	Free
33	In-House tool [S66]	The tool is basically a collection of Python scripts, which currently generates text output; automatically extract the analysis of vulnerability data per files. Retrieve source code file and parse it, then	In-House Tool

		get Bugzilla entries for each vulnerability, gets trace code-change for bug fix after these inputs, map vulnerabilities to files.	
34	SoCfeS [S4]	Tool was developed by Malaysian authors to realize continuous assessment tool to measure the two aspects of quality (technical and non-technical (human) facet quality) during the software life cycle. Tool automates the processes which required for realizing by user efficiently. In this way, tool enables the users to do assessment at any time throughout software life cycle.	In-House Tool

When we analyzed the name of the software quality datasets and metric tools on the internet; we encountered QSM (Quantitative Software Management) tool. This tool supports full measurement process from beginning to the end of the large projects. Moreover, tool includes 10, 000 projects from industry, with validated data. Thus, this database gives a chance to firms to compare metrics to improve their processes or helps to determine threshold values according to historical data [67].

To use this tool, users need to be a licensed user or limited demo version is available. Unfortunately, there is no paper in our pool, which uses QSM.

SLR-RQ 4 - Which areas in software quality metrics/measurement require further research?

39% of the papers in the pool, mentioned that techniques could be improved through implementation of different languages, different applications to the different phases of the projects. Moreover, suggestions for further research have been summarized in in Section 6.

Summary of the future works that appear in the papers can be categorized under seven (7) headings as given below:

1) More Metrics Generation:

a. Need for new modularity metrics to improve accuracy, or to take less effort for predicting bugs. Some of the metrics are complicated so it might be useful to simplify them to minimize the effort spent on them.
b. Need for more emphasis on objective metrics rather than subjective ones.

c. Need for analyzing the relationship between security vulnerabilities with other Complexity, Coupling and Cohesion metrics in service-oriented architecture.
d. Need for focusing on a code-change process instead of on the code properties to analyze the relation between software development process complexities and coupling [S66].
e. Need for more complicated metrics with more diversified parameters like usage of the complexity attributes with software cost, effort and time factors.
f. Need for choosing the best combination of metrics for each different application domain for the given application context. Moreover, selection of metric parameters can be automated with the tool by using prediction functions with the help of the combination of techniques [S28].

2) Different Programming Languages/Application Domains:

a. Conclusions of [S20] are usable only for Object-Oriented (OO) Systems and medium size systems, so authors would like to apply findings of these papers to large systems and OO systems. Also, testing the capability of findings by generating scenario for Aspect-Oriented, Service-Oriented and Component-Based application development to predict maintainability.
b. These metrics can be a good starting point to propose new/extended metrics for the other types of application domains like mobile applications, safety related applications, artificial intelligence applications, service-oriented applications and not user interface dominant applications.

3) Different Granularity Analysis:

a. There is need for more metrics related to design phase by calculating the coupling and cohesion metrics for UML representations of software.
b. Further research is required to detect defects of COSMIC metrics for various types of application domains by applying the broader measurements belonging to different application domain types [S65].

4) **Tool Automatization:**

a. Need for the development of plug-in for IDE tools to calculate modularity (minimum effect of change in one component to others) metrics.

b. E-Quality tool can be extended for object oriented languages, like C++, C# and PHP. Also, tool can be extended to visualize the design phase by extracting design metrics and relations from UML diagrams [S34]. Integration of the quality metrics for software product in an automated tool, which performs the selected optimization action.

c. Need for tool to measure and test the correctness of the functional size measurement for different types like IFPUG, UniFSM, EFES etc.

d. By applying artificial intelligence and machine learning techniques new metrics and measurement tools can be suggested.

e. There are lots of quality metric tools to measure different types of quality metrics. Instead of using lots of tools for measurement, using one tool to measure multipurpose metrics would increase the reliability, traceability and completeness of the metric data. Moreover, integrated tool usage would also decrease the spent time for measurement activity. So, there is need for the integrated quality metrics/measurements management tools to collect and analyze the metrics in one pool.

5) **Quality Attributes Metrics:**

a. Need for the extension of the metrics for measurement of the quality characteristics such as usability, integrity, satisfaction and context coverage.

b. Need for the metrics which use the process perspective and testing attributes of the employers as a parameter for measurement of quality attributes.

6) **Threshold Value Generation:**

a. Metrics do not have any importance or meaning unless results of the metrics are analyzed in accordance with the threshold values. Unfortunately, there are only a few threshold values in the literature. Therefore, there is a need on behalf the threshold values to produce different application domains,

programming languages, size of the projects, quality characteristics, SDLC phase and methods.

b. Threshold values can be tested and analyzed in a wider range of datasets and metric levels including different programming languages.

7) Process Improvement:

a. To reach greatest quality levels, it is usually necessary to generate metrics and procedures of the companies by concentrating on common set of goals, strategic plans and business practices of the projects/companies. Generating metrics from strategic plan and business practices, through putting forward more beneficial and meaningful metrics helps to improve the progress of the companies.

b. To increase the quality level, collection metric data and reporting method of the results of quality metrics/measurements are important. So, there is a need to continue to improve data mining and data reporting capabilities when the metrics are measured and analyzed. Without a thorough, accurate, and well-timed data of metrics, process improvement program would be largely ineffective.

c. With the help of the machine learning and artificial intelligence 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 included in the system automatically. Besides, this model can also guide developers/testers in relation to in which order activities will take place. With the help of this model, error making rate of the developers could be decreased [S4].

4.2 Trends and Discussions

The SM and SLR results of Software Quality Metrics related articles and conference papers for the last 10 years are presented with Mind Map technique, which allows us to see the big picture. With the help of this technique, we have a chance to analyze

the current situation and improvement opportunities related to software quality metrics. By analyzing Figure 4.34, researchers will have a chance to obtain general information about the trends of the software quality metrics in the literature.

Mind map technique which was invented by Tony BUZAN [68], provides invaluable benefits to researchers as presented follow:

- Visualizes your subjects, works to offer clearly drawn opinions about them
- Accelerates the process of learning and analysis.
- Provides initial point to planning the next step, so it is also a planning tool.
- Provides a big picture of a related issue, situation
- Provides clear traceability between data, issues, status
- Stirs your imagination to find more creative solutions
- Helps you to ensure your work is on the right track or not.

Following mind map was drawn by using the tool named as iMindMap (Ultimate edition, version 10.1.1). A mind map diagram for presenting the technical trends data of software quality metrics is shown in Figure 4.34. Trends for each categorization are shown through percentage values. Besides, the highest percentage in each category is marked with flag and a different color.

Discussions and remarks about the software quality metrics trends for the last ten years are listed as follows:

- As shown in Figure 4.34, the most popular quality model in the paper pool of this study is ISO 9126 quality model. Although the ISO 25010 (updated version of ISO 9126) quality model has been released in 2011, still usage rate of the ISO 9126 is popular. To increase the usage of the new version of quality model, additional documents can be prepared to give more explanations and examples to clarify the unclear issues.
- In our paper pool, while presenting the quality metrics, mostly referenced standard was IEEE 1061 and process model was CMMI. Countries, which

have the highest numbers of CMMI certificate, made greater contribution to software quality metrics. There is a direct proportion between rate of the usage of the quality standards, process models and rate of software quality metrics related articles and conference papers. Encouraging the usage of these standards and models, will definitely contribute to the literature on software quality metrics/measurements.

- Through enhancing the collaboration of the Computer Engineering/Science, Mathematics and Statistics departments of the universities, usage of the statistical model/methods can be increased. Statistical method/models can be used to propose the new metrics, validate the results of the metrics and also predict the bugs in the next phase of the projects.

- Most popular quality characteristics to measure the SPQ are Maintainability, Reliability and Security. Most popular quality characteristics to measure the QinU are Freedom from Risk, Effectiveness and Satisfaction. Most popular quality characteristics to measure the DQ are Understandability, Accuracy and Completeness.

- There is a need for metrics to measure the data quality in following aspects of quality characteristics: Credibility, Currentness, Accessibility, Compliance Confidentiality, Efficiency, Availability, Portability, Recoverability.

- Software Engineering: Modern Approaches book [69] and CMMI [1] have been proposed the set of metrics as listed as follow: measure of size, effort, variance in cost, # 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), maintanability (software down/error time), percentage of system vulnerabilities, test coverage rate, CC, fault density, number of entries and exits per module, percentage of comment lines. There is a consistency between the metric sets given in CMMI model and books with the metrics suggested in the this paper repository.

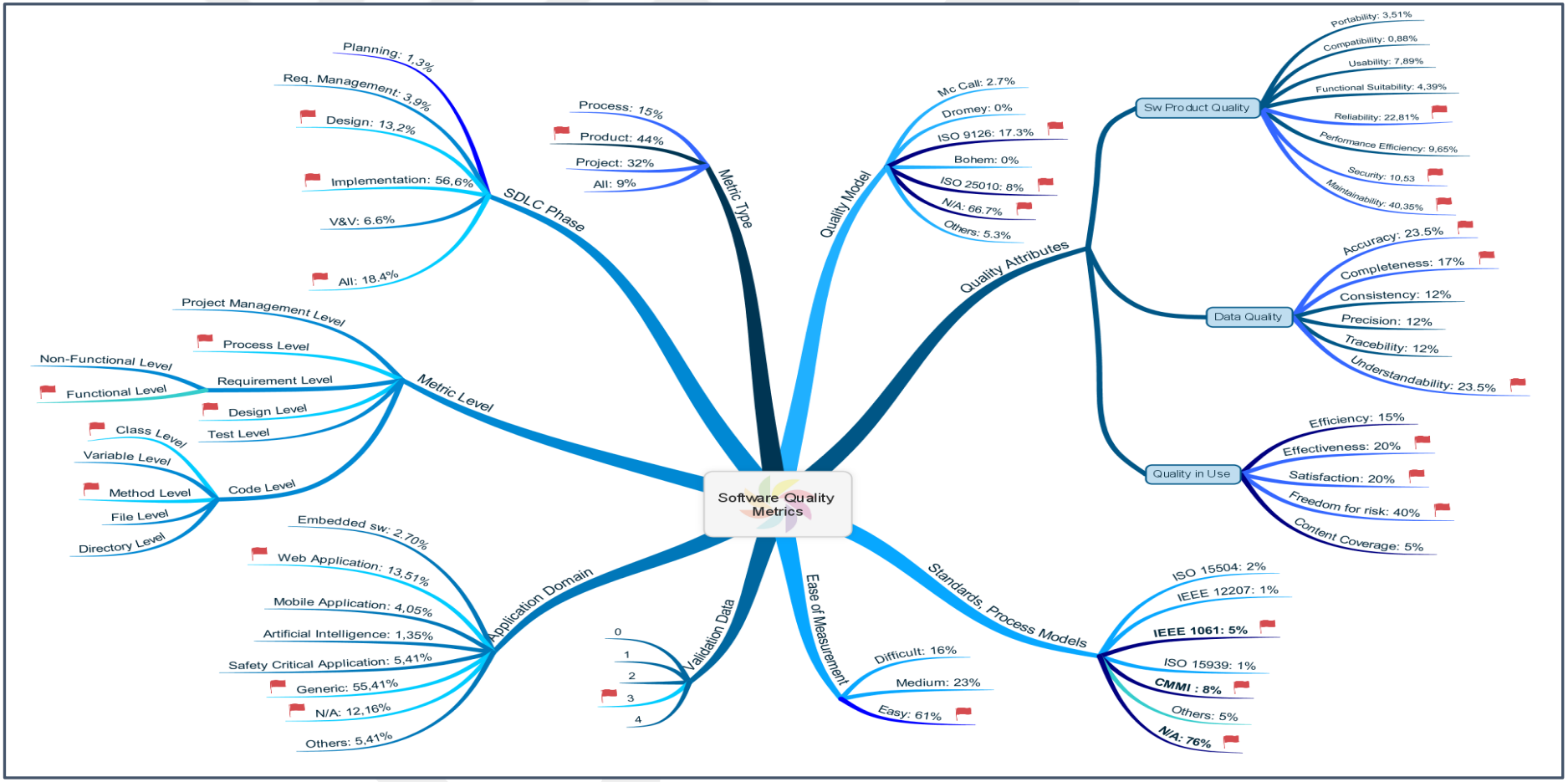


Figure 4.34 Mind Map of Software Quality Metrics Trends

CHAPTER 5

VALIDITY THREATS

The parameters of SM and SLR study that may affect the result of the study are database selection, selected search keywords, selected time frame, researchers, and primarily selected pool of papers. Similarly, validity threats can also be analyzed and discussed under four different headings: internal validity, construct validity, conclusion validity and external validity to maximize validity and minimize risk.

5.1. Internal Validity

Preliminary studies related to the internal structure of the study were carried out to reach complete, accurate and consistent data at the end of the study. By analyzing the example SLR and SM studies, we extracted the most commonly used databases to reach consistent and comparable data. Other details of source selection activities are given in Section 3.3.1.1.

In order to prevent the loss of any data pertaining to software quality metrics, alternative keywords and different combinations were searched through in the search engines. Inclusion and exclusion criteria, on which the final pool of papers is formed, are given in Section 3.3.2. The possibility of the presence of a threat to the study is based on two things:

- The judgment and experience of researchers
- Presentation method of data in the articles and conference papers.

The understandability, consistency and detail of the information given in the abstract, keywords and conclusion parts are very important criteria at the paper selection activity.

In order to avoid subjectivity in the selection of the papers, it was decided that the researcher and the supervisor would vote the papers, and then results would be compared. The given score “0” indicates strong opinion for the exclusion of the articles and conference papers and “3” points suggests strong opinion for the inclusion of articles and conference papers. If the authors give the score very differently, they would discuss the reasons for this conflict and different perspectives, until a joint decision was formed. If this SLR study was repeated, selected set of primary papers might have deviated in small amount, but we believe that the big picture of the result given at Figure 4.34 would not change much.

5.2. Construct Validity

Construct validity for this study is defined as the deviation rate of the expected and actual results. In this study, the final pool of papers and systematic mapping criteria are consistent. The SM-SLR study protocol given in Figure 3.4 and flow in the CADIMA tool [43], greatly facilitated taking the right steps respectively. Paper selection took place firstly based on the title and abstract part of the papers. In order to determine whether to include a paper or not, full text of the paper was read. After the final paper pool was constructed, data was systematically mapped in accordance with the research questions which are given in the Section 3.2 of this thesis.

While the author of this thesis systematically mapped the data, the supervisor carried out the peer review at the same time. When a need for sharing of the comments emerged at the end of the peer review, supervisor and author hold face-to-face meetings or communicated over e-mail. Also, after the papers were read, some of the important parts and evidences for mapped categorization have been extracted through highlighting the papers or writing the summary of data into the online data repository. With the help of the peer review, meetings and evidence based data production; reliability and validity of this study took its final form.

5.3 Conclusion Validity

Conclusion validity is maintained by providing comprehensive information in the final pool of papers that meets the criteria through which our research questions can

be effectively handled and satisfyingly answered. As presented in Section 4; graphs, trends and analysis have been generated directly from the raw and synthesized data to ensure the conclusion validity. To ensure the accuracy and consistency of the data, the entire raw data of SM and SLR study is presented in the form of excel spreadsheet, which is available to everyone on Google Drive. By using this raw data, this SM and SLR study can be replicated by following the same steps defined in the thesis.

5.4 External Validity

At the end of the SM and SLR study, it was found out that these results are valid for software engineering domain related projects, products and academia. In the other sections of this thesis, some results of this study are compared with the results of other SM and SLR studies, articles and conference papers.

CHAPTER 6

CONCLUSION AND FUTURE WORK

This study will provide input and hopefully function as a starting point for other studies related to software quality metrics like thesis, works, articles, conference papers and various other researchers. Also, results of this study can be used by industry through applying the most often used quality metrics to measure quality attributes, and using threshold values to compare and analyze the results. By comparing their results in accordance with thresholds, corrective or preventive actions can be taken to eliminate possible threats in the projects.

In this thesis, an overview of software quality metrics as reflected in the literature in last ten years, was evaluated with the help of systematic mapping and systematic literature review.

Initial search consisted of 1039 papers. 70 were included in this study, after a careful elimination process. Software as a service (SaaS), Quality of Service (QoS) and Quality of Experience (QoE) are out of the scope of this SLR study. We selected only the quality metrics that are directly related to pure software. Further SM or SLR studies might focus on this subject, in order to analyze and synthesize the metrics related to the measuring of quality of network performance.

In view of this, suggestions for future works are listed in below:

1. The majority of the papers mention the low understandability level and lack of examples of quality models. Quality models like ISO 25010, ISO 9126 and others define the high-level quality attributes, which are commonly used to characterize the quality of software. However, these quality models are too abstract to be operationalized for the quality assessment of a software system. To increase the usability of the quality models to generate more quality metrics, additional documents can be prepared to give more explanations and examples to clarify the obscure research areas.

2. There is a need for integration of multiple quality models in one model to decrease inconsistencies. Besides, more information is needed to be able to practice the quality models. Practically implementation of quality attributes can be the new area for researchers [S39].
3. Most used data quality characteristics are Understandability and Accuracy. There is no metric related to other data quality models like: Credibility, Currentness, Accessibility, Compliance Confidentiality, Efficiency, Availability, Portability, Recoverability.
4. There is no quality attribute at the standards or quality models for the metrics which are related to the culture, attribute, attitude and point of view of the employers, developers, testers, managers to quality attributes and quality standards, process models [S37].
5. Authors of paper [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. Usage and contribution of this new quality characteristic can be analyzed and then included in the new version of the ISO 25010.
6. The market's requirements and the standards in software industry are moving towards the agile trend and the race is accelerating. For instance, CMMI v2.0 and 6th edition of PMBOK [9] mostly emphasized the adoption of the processes in accordance with Agile. Unfortunately, only two papers have metrics related to Agile. Therefore, more metric are required for measuring the quality of agile projects to improve both the performance and the quality of projects. Moreover, metrics may help boost the processes of Agile methods towards a continuous improvement through taking the balance of efforts and costs of the improvements into account.
7. There is a need for extraction of coupling and cohesion metrics from UML representations instead of only by source code to predict bugs at the more previous phases.

8. Majority of the metrics proposed in the literature measure the quality of Object-Oriented software for medium sized systems. Judging from this, it could be stated that 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.
9. There is a need for the new quality metrics for new technology trends; Artificial Intelligence, Internet of things (IoT), Cloud Computing (CoT), Machine Learning and Robotics [S50].
10. There is also a need to analyze which metrics are non-applicable to companies of which sizes and why they cannot be implemented. Metrics should be tailored in accordance with the company size structure (small, medium and big).
11. There is a need for one-shot magic formula metric so as to extract software quality situation of the company at a glance and thus derive an effective project metric that from the vision, mission and strategic plan of the company. Also, project metrics will be derived from that one metric. In this way, traceability between organization, process and project metrics are likely to increase; so tracking the success and consistency of metrics' data will be visible. In example; "flight time of airplane in air without any problems occurred." can be metric for avionics related software companies.
12. There is a need for more reliable attack surface metrics/measurements to quantify different aspects of security and safety (unintended threats) for specific goals.
13. There is a difference between the metric ontology between quality models and standards. This prevents the common usage of the standards in industry and papers. Difference in ontology is one of the biggest obstacles for continuous improvement of standards. Also, there is a difference between the suggested, example set of metrics by standards, not used effectively by the industry and academic studies. Conferences and workshops might be productive grounds where the discrepancy observed between the standards and academic articles might be prevailed.

14. Calculation of metrics at different tool platforms and then bringing them together to calculate another metric is a hard and time-consuming activity. So, 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. In this way, instead of spending time on metric calculation; 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 any time they want to analyze the current situation of the project.
15. There is a need for tool automatization to integrate different sets of software metrics by using third-party statistical and machine learning tools to obtain predictions from software archives without much human intervention.
16. 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 experience, so we cannot measure defect prediction by using the personal data of developers. There is no chance to calculate the defect prevention metrics on a personal base. 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 the effectiveness of the developer/tester on the metrics like number of developers, culture, attributes, point of views to the processes and standards, etc.
17. In the last years, number of papers related to the quality metrics decreased. There are very few open source measurement database programs and open source projects. To encourage the researches in this area, there is a need for more open source project databases to test and validate the quality metrics.
18. 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 requirement related to security can affect the working of critical software of hospitals. Therefore, there is a need for more quality metrics for non-functional requirements to predict and eliminate the bugs in proactive manner.

19. EFQM defined Benchmarking as a systematic comparison of approaches with other relevant organizations which gain insights that will help the organization to take action to improve its performance [70]. EFQM suggest that benchmarking increases the improvement opportunities. Therefore, usage of the similar metric formula needs to be encouraged among the software companies which are in the same job sector. In this way, companies can take the opportunities to improve their processes. Moreover, international conferences and workshops can be a beneficial tool for dissemination of the same metrics to be used by the companies in the same sector.
20. There is a need for cross-fertilization between the international workshop and conferences on software metrics communities related to metrics, software analytics, and mining software repositories to get more benefits about quality [S9]

REFERENCES

[1] CMMI Institute, “CMMI Development”, v2.0, 2018.

[2] S. Ian, “Software Engineering”,10th Ed., H.Marcia, Pearson, 2016.

[3] ISTQB, “International Software Testing Qualifications Board (ISTQB) Glossary”, Internet: https://glossary.istqb.org/en/search/, [August 1, 2019].

[4] IEEE Computer Society, IEEE STD 1061-1992, “IEEE Standard for a Software Quality Metrics Methodology”,1992.

[5] ISO, ISO/IEC 25000, “Software Engineering - Software Product Quality Requirements and Evaluation (SQuaRE)-Guide to SQuaRE”, 2005.

[6] ISO, “ISO/IEC 9126-1: Information Technology - Software Product Quality - Part 1: Quality Model”, 2001.

[7] ISO, ISO/IEC 25010:2011, “Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models”, Internet: https://iso25000.com/index.php/en/iso-25000-standards, [August 1, 2019].

[8] ISO, ISO/IEC 25012:2008, “Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Data quality model”, Internet: https://iso25000.com/index.php/en/iso-25000-standards/iso-25012, [August 1, 2019].

[9] Cap Gemini, S. M., “World Quality Report-2018-2019”, Micro Focus, Internet:https://community.microfocus.com/t5/Application-Delivery-Management/The-World-Quality-Report-2018-19-is-now-available/ba-p/1666131/page/2, Sept., 2018 [August 1, 2019].

[10] Project Management Institute (PMI), “A Guide to the Project Management Body of Knowledge, PMBOK GUIDE”, 6th Ed., 2017.

[11] B. Kitchenham and S. Charters, “Guidelines for performing Systematic Literature Reviews in Software Engineering,” Tech. Rep., vol. 2, no. 3, pp. 1–57, 2007.

[12] The Cochrane Collaboration, 2005, “Glossary terms in the Cochrane Collaboration, Internet: www.cochrane.org, [August 1, 2019].

[13] S. Parthasarathy and N. Anbazhagan, “Analyzing the Software Quality metrics for Object Oriented Technology,” *Information Technology Journal*, vol. 5, no. 6. pp. 1053–1057, 2006.

[14] 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. 01–07, 2012.

[15] D. Santos, A. Resende, P. A. Junior, and H. Costa, “Attributes and Metrics of Internal Quality that Impact the External Quality of Object-Oriented Software: A Systematic Literature Review,” *Proc. 2016 42nd Lat. Am. Comput. Conf. CLEI 2016*, 2017.

[16] E. Ronchieri and M. Canaparo, “A preliminary mapping study of software metrics thresholds,” *ICSOFT 2016 - Proc. 11th Int. Jt. Conf. Softw. Technol.*, vol. 1, no. Icsoft, pp. 232–240, 2016.

[17] C. Calero, M. F. Bertoa, and M. Á. Moraga, “A systematic literature review for software sustainability measures,” 2013 2nd Int. Work. Green Sustain. Software, GREENS 2013 - Proc., pp. 46–53, 2013.

[18] T. Tahir, G. Rasool, and C. Gencel, “A systematic literature review on software measurement programs,” *Inf. Softw. Technol.*, vol. 73, no. August 2018, pp. 101–121, 2016.

[19] 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.

[20] M. Riaz, E. Mendes, and E. Tempero, “A Systematic Review of Software Maintainability Prediction and Metrics,” *2009 3rd Int. Symp. Empir. Softw. Eng. Meas. ESEM 2009*, pp. 367–377, 2009.

[21] V. Soares Fonseca Monalessa Perini Barcellos Ricardo de Almeida Falbo, “A Tools Integration for Supporting Software Measurement: A Systematic Literature

Review," iSys – Rev. Bras. Sist. Informação, Rio Janeiro, CESI/SBC, vol. 8, no. 4, pp. 80–108, 2015.

[22] T. Beranič and M. Heričko, "Approaches for software metrics threshold derivation: A preliminary review," *CEUR Workshop Proc.*, vol. 1938, pp. 11–13, 2017.

[23] J. Saraiva *et al.*, "Aspect-oriented software maintenance metrics: A systematic mapping study," *IET Semin. Dig.*, vol. 2012, no. 1, pp. 253–262, 2012.

[24] Vanitha N, ThirumalaiSelvi R, ThirumalaiSelvi R Vanitha N, Vanitha N, and ThirumalaiSelvi R, "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.

[25] K. Fan, Y. Ren, and Z. Yan, "Construction of a Software Measurement Tool Using Systematic Literature Review," *IEEE Confs Internet Things, Green Comput. Commun. Cyber, Phys. Soc. Comput. Smart Data, Blockchain, Comput. Inf. Technol. Congr. Cybermatics*, 2018.

[26] N. Nagappan, L. Williams, M. Vouk, and J. Osborne, "Design Quality Measurement for Service Oriented Software on Service Computing System: a Systematic Literature Review," *Int. Conf. Inf. Technol. Syst. Innov.*, pp. 375–380, 2018.

[27] M. Unterkalmsteiner, T. Gorschek, A. K. M. M. Islam, C. K. Cheng, R. B. Permadi, and R. Feldt, "Evaluation and measurement of software process improvement-A systematic literature review," *IEEE Trans. Softw. Eng.*, vol. 38, no. 2, pp. 398–424, 2012.

[28] H. Danilo Santos, Antonio Resende, Paulo Afonso Junior, Costa, "External Quality Metrics for Object-Oriented Software: A Systematic Literature Review," *CLEI Electron. J.*, vol. 20, no. 3, p. 2017, 2019.

[29] A. Meidan, J. A. García-García, I. Ramos, and M. J. Escalona, "Measuring Software Process: A Systematic Mapping Study," *ACM Comput. Surv.*, vol. 51, no. 3, pp. 58:1--58:32, 2018.

[30] 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," *Proc. - Int. Conf. Chil. Comput. Sci. Soc. SCCC*, pp. 1–11, 2017.

[31] S. El-Sharkawy, N. Yamagishi-Eichler, and K. Schmid, “Metrics for analyzing variability and its implementation in software product lines: A systematic literature review,” *Inf. Softw. Technol.*, vol. 106, no. August 2018, pp. 1–30, 2019.

[32] E. Ronchieri and M. Canaparo, “Metrics for Software Reliability: A systematic mapping study,” *J. Integr. Des. Process Sci.*, vol. 22, no. 2, pp. 5–25, 2019.

[33] R. Malhotra and A. Bansal, “Predicting change using software metrics: A review,” *2015 4th Int. Conf. Reliab. Infocom Technol. Optim. Trends Futur. Dir. ICRITO 2015*, 2015.

[34] S. Lehrig, H. Eikerling, and S. Becker, “Scalability , elasticity , and efficiency in cloud computing : a systematic literature review of definitions and metrics,” *Proc. 11th Int. ACM SIGSOFT Conf. Qual. Softw. Archit.*, no. 1, pp. 83–92, 2015.

[35] D. Radjenović, M. Heričko, R. Torkar, and A. Živkovič, “Software fault prediction metrics: A systematic literature review,” *Inf. Softw. Technol.*, vol. 55, no. 8, pp. 1397–1418, 2013.

[36] S. Stevanetic and U. Zdun, “Software Metrics for Measuring the Understandability of Architectural Structures -A Systematic Mapping Study,” *Int. Conf. Eval. Assess. Softw. Eng.*, pp. 1–14, 2015.

[37] Suali, A.J. & Fauzi, S.S.M. & Nasir, M.H.N.M. & Sobri, W.A.W.M. & Raharjana, Indra. (2019). Software quality measurement in software engineering project: A systematic literature review. Journal of Theoretical and Applied Information Technology., vol.97. pp. 918-929, 2019.

[38] Isong, Bassey & Ekabua, Obeten, “State-of-the-Art In Empirical Validation of Software Metrics for Fault Proneness Prediction: Systematic Review”, International Journal of Computer Science & Engineering Survey., vol.6, no.6, pp.1-18, 2016.

[39] N. Marsyahariani, N. Daud, and W. M. N. W. Kadir, “Systematic Mapping Study of Quality Attributes Measurement in Service Oriented Architecture,” *Int. Conf. Inf. Sci. Digit. Content Technol.*, pp. 626–631, 2012.

[40] E. Kupiainen, M. V. Mäntylä, and J. Itkonen, “Using metrics in Agile and Lean software development - A systematic literature review of industrial studies,” *Inf. Softw. Technol.*, vol. 62, no. 1, pp. 143–163, 2015.

[41] School of Information Technology and Engineering, University of Ottawa, Canada, "PROMISE Software Engineering Database", Internet: http://promise.site.uottawa.ca/SERepository/, [August 1, 2019].

[42] B. Kitchenham, S. C., "Guidelines for Performing Systematic Literature Reviews in Software engineering. Evidence-Based Software Engineering", 2007.

[43] C. Kohl et al., "Online Tools Supporting the Conduct and Reporting of Systematic Reviews and Systematic Maps: A case study on CADIMA and review of existing tools," Environ. Evid., vol. 7, no. 1, pp. 1–17, 2018.

[44] Mind Mapping, "Brainstorming and Project Planning Software", Internet: https://imindmap.com/, v10, March, 2019 [August 1, 2019].

[45] K. Petersen, R. Feldt, S. Mujtaba, M. Mattsson, "Systematic Mapping Studies in Software Engineering", 12th International Conference on Evaluation and Assessment in Software Engineering (EASE), 2008, pp. 71–80.

[46] Halevi, G., Moed, H. and Bar-Ilan, J, "Suitability of Google Scholar as a source of scientific information and as a source of data for scientific evaluation-Review of the Literature" Journal of Informetrics, 11(3), pp.823-834, 2017, https://doi.org/10.1016/j.joi.2017.06.005.

[47] ELSEVIER, "SCOPUS: Content Coverage Guide", Internet: https://www.elsevier.com/__data/assets/pdf_file/0007/69451/0597-Scopus-Content-Coverage-Guide-US-LETTER-v4-HI-singles-no-ticks.pdf, 2017, August [August 1, 2019].

[48] SocialMediaToday, "Why Google Scholar Adding Elsevier "ScienceDirect" Data is Significant?", Internet:https://www.socialmediatoday.com/content/why-google-scholar-adding-elsevier-sciencedirect-data-significant, 2007, July 9 [August 1, 2019].

[49] OpenAIRE, "What is OpenAIRE?", Internet: https://www.openaire.eu/faqs, [August 1, 2019].

[50] Murdoch University, "Systematic Reviews-Research Guide", Internet: https://libguides.murdoch.edu.au/systematic/PICO, July 27, 2019 [August 1, 2019].

[51] D. S. Cruzes and T. Dybå, "Synthesizing evidence in software engineering research," ESEM 2010 - Proc. 2010 ACM-IEEE Int. Symp. Empir. Softw. Eng. Meas., no. 7491, 2010.

[52] D. S. Cruzes and T. Dybå, “Recommended Steps for Thematic Synthesis in Software Engineering,” Int. Symp. Empir. Softw. Eng. Meas., no. 7491, pp. 275–284, 2011.

[53] CMMI Institue, “CMMI Adoption Graphs in December 2018”, https://cmmiinstitute.com/getattachment/9613f059-0b66-44a8-8804-fb0564cf6935/attachment.aspx, Dec., 2018 [August 1, 2019].

[54] Question Pro, “Empirical Research: Definition, Methods, Types and Examples”, Internet: https://www.questionpro.com/blog/empirical-research/, [August 1, 2019].

[55] Research Gate, “Analytic and Empirical Methods.”, Internet: https://www.researchgate.net/, Jan., 2018 [August 1, 2019].

[56] Everitt, B. S, Skrondal,S., “The CAMBRIDGE Dictionary of Statistics”, Fourth Edition, Cambridge University Press, 2010.

[57] Figueiredo. E, “Goal-Question-Metric” Internet: https://homepages.dcc.ufmg.br/~figueiredo/disciplinas/lectures/goal-question-metric_v01.pdf, 2014 [August 1, 2019].

[58] *D. N. Card and C. L. Jones, "Status report: Practical Software Measurement,"* Third International Conference on Quality Software, 2003.

[59] “The Analytic Hierarchy Process”, Internet: http://www.dii.unisi.it/~mocenni/Note_AHP.pdf, [August 1, 2019].

[60] MathWorks, “Genetic Algorithm”, Internet: https://www.mathworks.com/help/gads/genetic-algorithm.html?searchHighlight=genetic%20algorithm&s_tid=doc_srchtitle, [August 1, 2019].

[61] MathWorks, “Swarm Optimization”, Internet: https://www.mathworks.com/help/gads/particleswarm.html?searchHighlight=swarm%20optimization&s_tid=doc_srchtitle, [August 1, 2019].

[62] Everitt, B. S., “An Introduction to Optimization Methods and Their Application in Statistics”, Chapman and Hall/CRC Press, London, 1987.

[63] Cramer, D., Howitt, D.L, “The SAGE Dictionary of Statistics: A Practical Resource for Students in Social Sciences”, first edition, May 2004.

[64] ISACA, “ISACA DriveTransparent and Measurable Value With COBIT 5 Process Metrics”, Internet: http://www.isaca.org/COBIT/FOCUS/Pages/drive-

transparent-and-measurable-value-with-cobit-5-process-metrics.aspx, 2017, December 18 [August 1, 2019].

[65] M.Aydın, "GUI Testing Of Android Applications: A Systematic Mapping", M.S. Thesis, METU, Turkey, December 2014.

[66] B. Keser, T. Iyidogan, and B. Ozkan, "ASSIST: An Integrated Measurement Tool," *Proc. - Jt. Conf. 23rd Int. Work. Softw. Meas. 8th Int. Conf. Softw. Process Prod. Meas. IWSM-MENSURA 2013*, pp. 237–242, 2013.

[67] QSM, "Industry Database (Quantitative Software Management Tool)", Internet: https://www.qsm.com/luminosity/QSM_Demo/, [August 1, 2019].

[68] Buzan, Tony. Mind Map Mastery: The Complete Guide to Learning and Using the Most Powerful Thinking Tool in the Universe, March 13, 2018, Watkins Media Ltd.

[69] E. J. Braude and M. E. Bernstein, "Software Engineering: Modern Approaches", 2nd Ed., United States: John Wiley and Sons, Inc., 2011.

[70] EFQM, "EFQM User Guide: EFQM Benchmarking Guidelines", 2013

APPENDICES

APPENDIX A

SYSTEMATIC LITERATURE REVIEW REPOSITORY

Throughout the completion of this study, thesis data, which includes the systematic literature review analysis data in excel file and links of PDF files downloaded from search engines, was stored and managed on Google Drive. Google Drive provides a huge secure storage area as well as offering easy access from any device, and hence it provides the opportunity to easily share the link of Google Drive between the researcher and the thesis supervisor. In this way, progress of this thesis could be monitored on a daily basis, and the supervisor could give swift feedback. With all these features, Google drive certainly increased the collaboration between the researcher and the supervisor in thesis work.

I gladly share the Google drive link of thesis data for public access.

(https://drive.google.com/drive/folders/1EWQhj8ONiI1CWLdfPPtvNB21bHDIBcMV?usp=sharing)

APPENDIX B

SLR STUDY REFERENCES

[S1] P. Rotella and S. Chulani, "Implementing Quality Metrics and Goals at the Corporate Level," p. 113, 2011.

[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," pp. 119–128, 2014.

[S4] J. H. Y. and A. D., "Measuring the Unmeasurable Characteristics of Software Product Quality," *Int. J. Adv. Comput. Technol.*, vol. 2, no. 4, pp. 95–106, 2011.

[S5] T. R. G. Nair and S. V, "Estimation of Characteristics of a Software Team for Implementing Effective Inspection Process through Inspection Performance Metric," 2011.

[S6] D. I. K. Sjoberg, B. Anda, and A. Mockus, "Questioning Software Maintenance Metrics: A Comparative Case Study," *Proc. ACM-IEEE Int. Symp. Empir. Softw. Eng. Meas.*, no. Mi, pp. 107–110, 2012.

[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. H. 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, 2014.

[S10] K. Jinzenji, T. Hoshino, L. Williams, and K. Takahashi, “Metric-based Quality Evaluations for Iterative Software Development Approaches like Agile,” *Proc, 23rd IEEE Int. Symp. Softw. Reliab. Eng. Work. ISSREW 2012*, pp. 54–63, 2012.

[S11] K. Lochmann and L. Heinemann, “Integrating Quality Models and Static Analysis for Comprehensive Quality Assessment,” p. 5, 2011.

[S12] C. E. Otero, E. Dell, A. Qureshi, and L. D. Otero, “A quality-based Requirement Prioritization Framework Using Binary Inputs,” *AMS2010 Asia Model. Symp. 2010 - 4th Int. Conf. Math. Model. Comput. Simul.*, pp. 187–192, 2010.

[S13] L. Ping, “A Quantitative Approach to Software Maintainability Prediction,” *Proc. - 2010 Int. Forum Inf. Technol. Appl. IFITA 2010*, vol. 1, pp. 105–108, 2010.

[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] A. Vinobha, V. S. Senthil, and C. Babu, “Evaluation of Reusability in Aspect Oriented Software using Inheritance Metrics,” *Proc. 2014 IEEE Int. Conf. Adv. Commun. Control Comput. Technol. ICACCCT 2014*, no. 978, pp. 1715–1722, 2015.

[S16] K. Z. Sultana, B. J. Williams, and A. Bosu, “A Comparison of Nano-Patterns vs. Software Metrics in Vulnerability Prediction,” *Proc. - Asia-Pacific Softw. Eng. Conf. APSEC*, vol. 2018-December, pp. 355–364, 2019.

[S17] S. Husein and A. Oxley, “A Coupling and Cohesion Metrics Suite for Object-Oriented Software,” *ICCTD 2009 - 2009 Int. Conf. Comput. Technol. Dev.*, vol. 1, pp. 421–425, 2009.

[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, 2013.

[S19] P. K. Manadhata and J. M. Wing, “An Attack Surface Metric,” *IEEE Trans. Softw. Eng.*, vol. 37, no. 3, pp. 371–386, 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,” *Lect. Notes Eng. Comput. Sci.*, vol. 1, pp. 61–66, 2013.

[S21] B. L. Sousa, M. A. S. Bigonha, and K. A. M. Ferreira, “An Exploratory Study on Co-occurrence of Design Patterns and Bad Smells using Software Metrics,” *Softw. - Pract. Exp.*, vol. 49, no. 7, pp. 1079–1113, 2019.

[S22] J. Dong, “An Improved Fuzzy Synthesis Evaluation Algorithm for Software Quality,” *2009 Int. Conf. Inf. Manag. Innov. Manag. Ind. Eng. ICIII 2009*, vol. 2, pp. 565–569, 2009.

[S23] A. Tahir and R. Ahmad, “An AOP-based Approach for Collecting Software Maintainability Dynamic Metrics,” *2nd Int. Conf. Comput. Res. Dev. ICCRD 2010*, pp. 168–172, 2010.

[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,” *Proc. - 4th IEEE Int. Conf. Softw. Testing, Verif. Valid. Work. ICSTW 2011*, pp. 566–571, 2011.

[S25] R. Malhotra and M. Khanna, “A New Metric for Predicting Software Change using Gene Expression Programming,” *5th Int. Work. Emerg. Trends Softw. Metrics, WETSoM 2014 - Proc.*, pp. 8–14, 2014.

[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,” *Proc. - 2014 13th Int. Conf. Mach. Learn. Appl. ICMLA 2014*, pp. 356–361, 2014.

[S27] 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. SBES 2010*, pp. 11–20, 2010.

[S28] D. Westermann, J. Happe, R. Krebs, and R. Farahbod, “Automated Inference of Goal-Oriented Performance Prediction Functions,” *2012 27th IEEE/ACM Int. Conf. Autom. Softw. Eng. ASE 2012 - Proc.*, pp. 190–199, 2012.

[S29] H. W. and N. S. Kehan Gao, Taghi M. Khoshgoftaar, “Choosing Software Metrics for Defect Prediction: an investigation on feature selection techniques,” *Softw. - Pract. Exp.*, vol. 41, pp. 579–606, 2011.

[S30] M. Bansal and C. P. Agrawal, “Critical Analysis of Object Oriented Metrics in Software Development,” *Int. Conf. Adv. Comput. Commun. Technol. ACCT*, pp. 197–201, 2014.

[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, 2016.

[S32] L. B. L. De Souza and M. De Almeida Maia, “Do Software Categories Impact Coupling Metrics?,” *IEEE Int. Work. Conf. Min. Softw. Repos.*, pp. 217–220, 2013.

[S33] A. Kaur, P. S. Sandhu, and A. S. Brar, “Early Software Fault Prediction using Real Time Defect Data,” *2009 2nd Int. Conf. Mach. Vision, ICMV 2009*, pp. 242–245, 2009.

[S34] U. Erdemir, U. Tekin, and F. Buzluca, “E-quality: A graph based Object Oriented Software Quality Visualization Tool,” *Proc. Viss. 2011 - 6th IEEE Int. Work. Vis. Softw. Underst. Anal.*, 2011.

[S35] M. P. Barcellos, R. De Almeida Falbo, and A. R. Rocha, “Establishing a well-founded Conceptualization about Software Measurement in High Maturity Levels,” *Proc. - 7th Int. Conf. Qual. Inf. Commun. Technol. QUATIC 2010*, pp. 467–472, 2010.

[S36] J. Feigenspan, S. Apel, J. Liebig, and C. Kästner, “Exploring Software Measures to Assess Program Comprehension,” *Int. Symp. Empir. Softw. Eng. Meas.*, pp. 127–136, 2011.

[S37] M. Umarji and C. Seaman, “Gauging Acceptance of Software Metrics: Comparing Perspectives of Managers and Developers,” *2009 3rd Int. Symp. Empir. Softw. Eng. Meas. ESEM 2009*, pp. 236–247, 2009.

[S38] A. Tripathi and K. Sharma, “Improving Software Quality based on Relationship Among the Change Proneness and Object Oriented Metrics,” *2015 Int. Conf. Comput. Sustain. Glob. Dev. INDIACom 2015*, pp. 1633–1636, 2015.

[S39] S. Srivastava and R. Kumar, “Indirect Method to Measure Software Quality using CK-OO Suite,” *2013 Int. Conf. Intell. Syst. Signal Process. ISSP 2013*, pp. 47–51, 2013.

[S40] A. J. Molnar, A. Neamţu, and S. Motogna, “Longitudinal Evaluation of Software Quality Metrics in Open-Source Applications,” *ENASE 2019 - Proc. 14th Int. Conf. Eval. Nov. Approaches to Softw. Eng.*, no. Enase, pp. 80–91, 2019.

[S41] T. Ruhroth, H. Voigt, and H. Wehrheim, “Measure, Diagnose, Refactor: A Formal Quality Cycle for Software Models,” *Conf. Proc. EUROMICRO*, pp. 360–367, 2009.

[S42] P. K. Kapur, G. Singh, N. Sachdeva, and A. Tickoo, “Measuring Software Testing Efficiency using Two-way Assessment Technique,” *Proc. - 2014 3rd Int. Conf. Reliab. Infocom Technol. Optim. Trends Futur. Dir. ICRITO 2014*, 2015.

[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, 2016.

[S44] Y. Shi, M. Li, S. Arndt, and C. Smidts, “Metric-based Software Reliability Prediction Approach and its Application,” *Empir. Softw. Eng.*, vol. 22, no. 4, pp. 1579–1633, 2017.

[S45] C. Santos, T. Novais, M. Ferreira, C. Albuquerque, I. H. De Farias, and A. P. C. Furtado, “Metrics Focused on Usability ISO 9126 based,” *Iber. Conf. Inf. Syst. Technol. Cist.*, vol. 2016-July, 2016.

[S46] S. U. Farooq, S. M. K. Quadri, and N. Ahmad, “Metrics, Models and Measurements in Software Reliability,” *IEEE 10th Jubil. Int. Symp. Appl. Mach. Intell. Informatics, SAMI 2012 - Proc.*, pp. 441–449, 2012.

[S47] S. R. Ragab and H. H. Ammar, “Object Oriented Design Metrics and Tools a Survey,” *INFOS2010 - 2010 7th Int. Conf. Informatics Syst.*, 2010.

[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, 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,” *Proc. 2011 Int. Conf. Process Autom. Control Comput. PACC 2011*, 2011.

[S52] P. Oliveira, F. P. Lima, M. T. Valente, and A. Serebrenik, “RTTool: A Tool for Extracting Relative Thresholds for Source Code Metrics,” *Proc. - 30th Int. Conf. Softw. Maint. Evol. ICSME 2014*, pp. 629–632, 2014.

[S53] M. Aniche, C. Treude, A. Zaidman, A. Van Deursen, and M. A. Gerosa, “SATT: Tailoring Code Metric Thresholds for Different Software Architectures,” *Proc. - 2016 IEEE 16th Int. Work. Conf. Source Code Anal. Manip. SCAM 2016*, pp. 41–50, 2016.

[S54] J. A. Wang, H. Wang, M. Guo, and M. Xia, “Security Metrics for Software Systems,” *Proc. 47th Annu. Southeast Reg. Conf. ACM-SE 47*, 2009.

[S55] A. Tiwari and P. S. Chakraborty, “Software Component Quality Characteristics Model for Component based Software Engineering,” *Proc. - 2015 IEEE Int. Conf. Comput. Intell. Commun. Technol. CICT 2015*, pp. 47–51, 2015.

[S56] G. Lajios, “Software Metrics Suites for Project Landscapes,” *Proc. Eur. Conf. Softw. Maint. Reengineering, CSMR*, pp. 317–318, 2009.

[S57] M. F. S. Oliveira, R. M. Redin, L. Carro, L. D. C. Lamb, and F. R. Wagner, “Software Quality Metrics and Their Impact on Embedded Software,” *Proc. 5th Int. Work. Model. Methodol. Pervasive Embed. Software, MOMPES 2009*, no. Mompes, pp. 68–77, 2009.

[S58] K. Punitha and S. Chitra, “Software Defect Prediction using Software Metrics,” *2013 Int. Conf. Inf. Commun. Embed. Syst. ICICES 2013*, pp. 555–558, 2013.

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

[S60] G. D. S. 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,” *ITNG2010 - 7th Int. Conf. Inf. Technol. New Gener.*, no. Ci, pp. 1177–1182, 2010.

[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, 2010.

[S62] M. K. Debbarma, N. Kar, and A. Saha, “Static and Dynamic Software Metrics Complexity Analysis in Regression Testing,” *2012 Int. Conf. Comput. Commun. Informatics, ICCCI 2012*, pp. 1–6, 2012.

[S63] A. Janus, R. Dumke, A. Schmietendorf, and J. Jäger, “The 3C Approach for Agile Quality Assurance,” *2012 3rd Int. Work. Emerg. Trends Softw. Metrics, WETSoM 2012 - Proc.*, pp. 9–13, 2012.

[S64] G. Rakic, Z. Budimac, and K. Bothe, “Towards a ‘Universal’ Software Metrics Tool: Motivation, process and a prototype,” *ICSOFT 2010 - Proc. 5th Int. Conf. Softw. Data Technol.*, vol. 2, pp. 263–266, 2010.

[S65] G. Yilmaz, S. Tunalilar, and O. Demirors, “Towards the Development of a Defect Detection Tool for COSMIC Functional Size Measurement,” *Proc. - Jt. Conf. 23rd Int. Work. Softw. Meas. 8th Int. Conf. Softw. Process Prod. Meas. IWSM-MENSURA 2013*, pp. 9–16, 2013.

[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, 2011.

[S67] U. L. Kulkarni, Y. R. Kalshetty, and V. G. Arde, “Validation of CK Metrics for Object Oriented Design Measurement”, *Proc. - 3rd Int. Conf. Emerg. Trends Eng. Technol. ICETET 2010*, pp. 646–651, 2010.

[S68] M. Rudolph and R. Schwarz, “A Critical Survey of Security Indicator Approaches”, *Proc. - 2012 7th Int. Conf. Availability, Reliab. Secur. ARES 2012*, pp. 291–300, 2012.

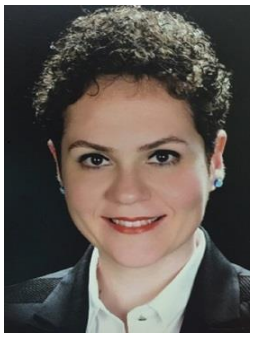
[S69] P. S. Sandhu, A. S. Brar, R. Goel, J. Kaur, and S. Anand, “A Model for Early Prediction of Faults in Software Systems,” *2010 2nd Int. Conf. Comput. Autom. Eng. ICCAE 2010*, vol. 4, pp. 281–285, 2010.

[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.

APPENDIX C

Fatıma Nur ÇOLAKOĞLU

Quality Assurance Specialist-HAVELSAN
M.S, Software Engineering, Atılım University
Contact Info:
colakoglu.fatmanur@student.atilim.edu.tr,
fcolakoglu@gmail.com.tr



Educational Background

Name of the University	Department	Degree
ATILIM University	Software Engineering, 2019	M.S
BILKENT University	Computer Technology and Information Systems (CTIS), 2011	B.S
BILKENT University	Computer Technology and Programming (CTP), 2009	A.S

Work Experiences

Position	Company Name	Company Sector
Quality Specialist	HAVELSAN (Air Defense and Trade)	Defence Industry and Information Systems
Quality Assurance Responsible	TÜBİTAK İLTAREN (Scientific and Technological Research Council of Turkey- Advanced Technologies Research Institute)	Defence Industry-Government Agency
Institute Committee Membership	TÜBİTAK İLTAREN	Defence Industry Government Agency
Software Quality Assurance Responsible	AYESAŞ (Aydın Software and Electronic Industry)	Defence Industry
Avionic Software Tester	AYESAŞ	Defence Industry
Avionic Software Tester-Intern	AYESAŞ	Defence Industry

Awards&Achievements

- %80 Merit-Based Scholarship: Bilkent University, CTIS
- %100 Merit-Based Scholarship: Bilkent University, CTP
- Graduation with 1st Rank from CTP and CTIS department, Bilkent University
- Award of Excellent Academic Success, Bilkent University, CTIS, 2011
- Earned Honor and High Honor Student Certificates during the education period of university

Publications

- Process Tailoring Approach Example, UYMS (Turkish National Software Engineering Symposium) 2016.
- Project-level audits as part of an Effective Quality Assurance Process: Applied Practices and Relevant Lessons Learned, UYMS 2017
- Usage of Mind Map Technique in Education and Training, International Eurasian Educational Research Congress, Ankara University, 2019

Certificates

Subject of Certification	**Company**
KAİZEN Continuous Improvement	KALDER (Quality Society, TURKEY)
DO178C and DO254 Software/Hardware Considerations in Airborne Systems and Equipment Certification	AFuzion
CMMI V2.0 EXAM	CMMI Institute (validity period: 3 years)
Foundations of Capability, CMMI v2.0	CMMI Institute
Building Development Excellence, CMMI v2.0	CMMI Institute
System Analysis and Design	PSConsultech
Risk-based Process Management	TSE (Turkish Standardization Institute)
ISO 27001 Information Security Management System	TÜBİTAK Cyber Security Institute
ISO 9001:2015 Lead Auditor	BSI
ISO 9001:2008 Internal Auditor	KALDER
CMMI v1.3	Carnegie Mellon Software Engineering Institute
AS9100 Rev C, Quality Management System standard for the Aviation, Space and Defense Industry	BSI

Interests

Participating Running Race Organizations, Painting, Amateur photography.