Noname manuscript No.
(will be inserted by the editor)

Empirical Evidence on the Link between Object-Oriented Measures and External Quality Attributes: A Systematic Literature Review

Ronald Jabangwe · Jürgen Börstler · Darja Šmite · Claes Wohlin

Received: date / Accepted: date

Abstract There is a plethora of studies investigating object-oriented measures and their link with external quality attributes, but usefulness of the measures may differ across empirical studies. This study aims to aggregate and identify useful object-oriented measures, specifically those obtainable from the source code of object-oriented systems that have gone through such empirical evaluation. By conducting a systematic literature review, 99 primary studies were identified and traced to four external quality attributes: reliability, maintainability, effectiveness and functionality. A vote-counting approach was used to investigate the link between object-oriented measures and the attributes, and to also assess the consistency of the relation reported across empirical studies. Most of the studies investigate links between object-oriented measures and proxies for reliability attributes, followed by proxies for maintainability. The least investigated attributes were: effectiveness and functionality. Measures from the C&K measurement suite were the most popular across studies. Vote-counting results suggest that complexity, cohesion, size and coupling measures have a better link with reliability and maintainability than inheritance measures. However, inheritance measures should not be overlooked during quality assessment initiatives; their link with reliability and maintainability could be context dependent. There were too few studies traced to effectiveness and functionality attributes; thus a meaningful vote-counting analysis could not be conducted for these attributes. Thus, there is a need for diversification of quality attributes investigated in empirical studies. This would help with identifying useful measures during quality assessment initiatives, and not just for reliability and maintainability aspects.

Keywords Systematic literature review · Object-oriented system · Source code analysis · Source code measures · Software metrics · Software quality · Static analysis

Ronald Jabangwe · Jürgen Börstler · Darja Šmite · Claes Wohlin
Blekinge Institute of Technology, SE-371 79 Karlskrona, Sweden

E-mail: {ronald.jabangwe,jurgen.borstler,darja.smite,claes.wohlin}@bth.se

1 Introduction

Quality evaluation of current source code artifacts of software releases can aid with quality predictions for future releases. For example, quality evaluation can be done by investigating the link between internal source code properties, such as complexity, and external quality attributes in current releases; this would help with predicting the risk levels of source code artifacts of varying complexity in future releases. Figure 1 shows the relationship between measurable internal source code properties and external quality attributes. The relationship between internal properties and external quality attributes is also highlighted in ISO/IEC-25010 (2010).

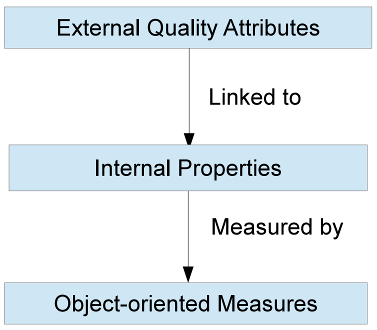


Fig. 1: Relationship of Internal and External Attributes

Quality is a multidimensional and, quite too often, a context dependent concept. ISO/IEC-25010 (2010) provides different perspectives of quality and also describes characteristics that can be used to assess both internal and external software quality attributes. Quality in our context is expressed by the collective set of external software quality attributes that we gather from studies that show an empirical link between the attributes and measures.

Quantitative analysis of internal source code properties can be done using static or dynamic measures. Static measures are used during static analysis, i.e., analysis of the systems structure, and dynamic measures are used during dynamic analysis, i.e., evaluation of system behavior during execution (ISO/IEC/IEEE-24765, 2010). In this study we conducted a systematic literature review on measures useful for static quantitative quality analysis of internal properties of source code of object-oriented systems. Dynamic measures were excluded in the review, as their performance heavily relies on the system and execution environment, hence their utility is very context dependent. Performance of static measures is less likely to be influenced by environmental factors and hence are more likely to be useful across more contexts than dynamic measures.

Software measures in general can be placed into three categories (Fenton and Pfleeger, 1998): project measures (those associated with, for example, project activities and milestones), process measures (those used for measuring certain activities associated with the software development life-cycle e.g. development process) and product measures (i.e. those that measure the resultant artifacts from process activities, e.g. source code). In the context of our systematic literature review, the product is the source code of object-oriented systems. The product measures are measures that can be derived from analysis of internal properties of source code of such systems.

A plethora of studies have been conducted on object-oriented quantitative quality analysis. The aim of this study is two-fold. First, we identify and gather measures, obtainable from the source code of object-oriented systems, that have been empirically evaluated and linked to external quality attributes, or proxies of external quality attributes, of object-oriented systems. Second, we evaluate the consistency of the link between measures and external quality attributes across studies.

The remainder of this paper is arranged as follows. Section 2 outlines related work. Section 3 provides a description of the research methodology. The steps taken to identify primary studies are described in Section 4 and validity threats are presented in Section 5. Results and analysis are provided in Section 6 and Section 7, respectively. The discussion is presented in Section 8, and conclusions in Section 9.

2 Related Work

Fundamentally, structured programming principles differ from object-oriented programming paradigms. Since the emergence of object-orientation, measures tailored specifically for the paradigm have been proposed in literature. The most well-known sets of measures that emerged from the need to evaluate unique characteristics inherent in object-oriented systems are the Chidamber and Kemerer measures (Chidamber and Kemerer, 1991; Chidamber et al., 1998), Lorenz and Kidd measures (Lorenz and Kidd, 1994), MOOD measures (Abreu and Carapuça, 1994; Abreu et al., 1995) and QMOOD measures (Bansiya and Davis, 2002). The measures are summarized in Table 1 and the grouping used in the table is adapted from the categorization used by Bansiya and Davis (2002) for the QMOOD measures. Bansiya and Davis (2002) categorizes the QMOOD measures according to the properties quantified by the measures. Grouping for the other measurement suites in Table 1 is according to the description of each measure in (Chidamber and Kemerer, 1991) and (Chidamber and Kemerer, 1994) for Chidamber and Kemerer measures, and in (Abreu and Carapuça, 1994) for MOOD measures, and in (Harrison et al., 1997) for Lorenz and Kidd measures.

2.1 Common Measurement Suites

2.1.1 Chidamber and Kemerer Measures

Chidamber and Kemerer (1991) proposed a measurement suite, commonly known as the C&K measurement suite, for measuring internal properties of object-oriented systems in 1991. The measures were developed after the empirical work conducted in collaboration with a company that was developing software using the object-oriented programming language C++. The intent for proposing the suite was to

Table 1: Measurement Suites and Internal Properties

Internal Properties	C & K measures	MOOD Measures	QMOOD Measures	L & K Measures [a]
Abstraction			Average Number on Ancestors (ANA)	
Cohesion	Lack of Cohesion in Method (LCOM)		Cohesion Among Methods in a Class (CAM)	
Complexity	Weight Methods per Class (WMC) Response For a Class (RFC)		Number of Methods (NOM)	Number of Public Methods Number of Methods (NM)
Composition			Measure of Aggregation (MOA)	
Coupling	Coupling Between Object Classes (CBO)	Coupling Factor (CF) Clustering factor	Direct Class Coupling (DCC)	Number of Friends of a Class (NF)
Encapsulation		Method Hiding Factor (MHF) Attribute Hiding Factor (AHF)	Data Access Metric (DAM)	
Inheritance	Depth of Inheritance Tree (DIT) Number of Children (NOC)	Method Inheritance Factor (MIF) Attribute Inheritance Factor (AIF) Reuse Factor	Measure of Functional Abstraction (MFA)	Number of Methods Inherited by a subclass (NMI)
Messaging			Class Interface Size (CIS)	
Polymorphism		Polymorphism Factor (PF)		
Size			Design Size in Classes (DSC) Number of Hierarchies (NOH)	Average Method Size (AMS)

[a] Measures listed in the table are just some of the many measures proposed by Lorenz and Kidd. More measures can be found in (Lorenz and Kidd, 1994; Xenos et al., 2000; Harrison et al., 1997)

provide a set of measures for quantitatively evaluating complexity and size aspects that are inherit in object-oriented systems (Chidamber and Kemerer, 1991).

2.1.2 Lorenz and Kidd Measures

Lorenz and Kidd (1994) proposed measures, often referred to as the L&K suite, for evaluating external characteristics of object-oriented systems using the internal properties complexity, size and inheritance. The number of measures proposed by

Lorenz and Kidd (1994) are vast and the main difference between the measures and the C&K measures is in the simplicity for deriving the measures. L&K measures are direct and easier to derive as most of them can be derived through a simple counting process for example "number of instance variables in a class" (Xenos et al., 2000).

2.1.3 MOOD Measures

In 1994, Abreu and Carapuça (1994) proposed the suite MOOD measures (Metrics for Object Oriented Designs). The main purpose for these measures is to enable quantitative quality evaluation of internal characteristics that are unique to object-oriented designs. The measures are obtainable during the design phases to facilitate with identifying design flaws early in the software development life cycle (Abreu and Carapuça, 1994; Olague et al., 2007). The measures are intended for a quantitative evaluation method using an *inside-out approach* (Abreu and Carapuça, 1994), i.e., linking internal properties to external quality characteristics through quantitative evaluation.

2.1.4 QMOOD Measures

Similarly to the C&K, L&K, and the MOOD measures, the QMOOD were also designed taking into consideration the unique internal characteristics of object-oriented designs. QMOOD is a model that was developed with more of a top-down approach (Bansiya and Davis, 2002). Quality attributes defined in the ISO/IEC-9126 (2001) are used to define quality and are then linked to certain internal attributes inherent in object-oriented designs. Measures for internal properties provided in the model are a combination of newly proposed measures and measures obtained through a literature survey. Bansiya and Davis (2002) also go further and propose, as well as validate, quality prediction models built using internal properties for six external product characteristics: reusability, flexibility, understandability, functionality, extendibility and effectiveness.

2.1.5 Summary

The underlying assumption for the proposal of these measurement suites and other object oriented measures is that there is a direct or indirect link between internal properties and the external characteristics of a software product; and there is an assumption that these measures can quantify the internal properties in a meaningful way. However, the relationship or the extent of the relationship between external quality attributes and measures of internal properties can vary from context to context. In addition, external quality attributes can depend on different combination of internal properties and measures; and measures can be used to measure different combination of internal properties; and can also be directly or indirectly linked to different external quality attributes. As a result, there is a plethora of studies dedicated to proposing or investigating these links.

2.2 Related Literature Reviews

Relevant reviews were identified by conducting a series of search runs that consisted of different combination of search terms in Google scholar, Compendex and Inspec databases. The search terms used were: “object-oriented”, “review”, “systematic review”, “systematic literature review”, “quality”, “evaluation”, “prediction”. After identifying the related reviews, we searched through the reference list of these papers in search of additional reviews. Both these steps were performed to identify as many reviews as possible that are related to our study.

The closest in objectives to our study is the one by Briand and Wüst (2002). The study aimed to investigate the usefulness of measures of object-oriented internal structures. The measures were investigated on their usefulness in predicting fault-proneness and effort, e.g., development effort, either individually or as a set/combination of measures. In the study the measures for internal properties unique to object-oriented systems such as coupling, size, cohesion and inheritance are the independent variables. Fault-proneness and effort are the dependent variables. Results from the study show that measures of coupling, complexity and size properties are found to be better predictors of fault-proneness than those of inheritance and cohesion properties. Size properties are identified as the most ideal predictors of effort. Fault-proneness is the most investigated proxy for external quality amongst studies that use correlation statistical analysis methods. Whereas for experimental studies the most investigated link was between the internal properties inheritance and the external characteristics maintainability and understandability. One notable difference between the study by Briand and Wüst (2002) and our study is the systematic approach we adopted for our research methodology and the gap in years since their study. The study by Briand and Wüst was conducted a decade ago and it is more of a literature survey rather than a systematic literature review. In addition our study is much broader since we consider more attributes for quality.

Genero et al. (2005) conducted a survey on object-oriented measures that can be used for measuring internal quality of UML class diagrams for external quality prediction. In the paper the authors identify and analyze existing measures on their capability to measure UML entities and attributes. The identified measures are also classified according to internal properties that they measure. Genero et al. also conducted an analysis on the validation methods used in different studies as well as tools useful for extracting and calculating the measures. Results from the survey show fault-proneness as the most investigated proxy for external quality characteristics and that definitions of measures as well as their usefulness is not consistent across studies.

Riaz et al. (2009) conducted a systematic literature review of studies reporting on measures and prediction methods for maintainability. The review identified 15 primary studies. From the primary studies, they identified 45 useful predictors of maintainability that included the well-known LOC measure and measures from the C&K set (LCOM, DIT, NOC, RFC and WMC). Out of the 15 papers 12 papers proposed prediction models, and the most used techniques were regression analysis methods. Results from the review revealed that measures useful for predicting maintainability characteristics were those that measured complexity, coupling and size properties.

Catal and Diri (2009) performed a systematic literature review on fault prediction studies that were published between 1990 and 2007. From their 74 primary studies they found that method level measures (e.g., McCabe's cyclomatic complexity (McCabe, 1976) and Halstead measures (Halstead, 1977)) are predominantly used for fault prediction. In the case of class-level measures, which are applicable to object-oriented programs, e.g., the set of measures discussed in Section 2.1, they observed that there has been a change in the percentage of papers that use them before and after 2005. In terms of techniques for building prediction models, there is a prevalent and growing interest in the use of machine learning methods for fault prediction. In comparison to before 2005, they observed a slight increase in the use of machine learning methods, and a slight decrease in studies that use statistical methods.

Xenos et al. (2000) present results from a literature survey they conducted to identify measures applicable for object-oriented systems. Included in the survey are measures that were initially introduced for systems developed using traditional methodologies, i.e. structured programming. Thus, the measures listed in the paper include traditional measures, such as LOC, and measures inherent in object-oriented systems, i.e. class, method, coupling, inheritance and system measures. Xenos et al. evaluated the measures with the goal of helping practitioners in selecting appropriate measures. The evaluation was based on seven aspects, which included effort of extracting and implementing the measures, as well as accuracy of the measures. The object-oriented measures evaluated included the MOOD, QMOOD, L&K and the C&K measures. The following measures had a positive assessment for all seven criteria: NIV, NCM, NMI, NMO and PRC (from L&K), and NPA and NPM (from the QMOOD measurement set) and CBO and RFC (from C&K). Nevertheless, Xenos et al. (2000) did not specifically focus on the predictive ability of the measures.

Saxena and Saini (2011) reviewed 14 studies on fault-proneness published between 1995 and 2010. The study results showed that regression analysis methods are the most used prediction methods and the C&K measures are the most evaluated individually as independent variables in studies for fault-proneness prediction. In their review they found that the most inconsistent of the measures were the two inheritance measures, DIT and NOC. DIT was not significant in six of the studies and NOC in five studies. However, these results were only obtained from eleven studies and hence it is a small sample to deduce a definitive evaluation of the performance or usefulness of the different measures.

Malhotra and Jain (2011) performed a literature review of empirical studies that report on the relationship between object-oriented measures and fault-proneness that were published between 1998 and 2010. Though some studies proposed new measures, they found that most of the measures used in the studies were from the C&K, MOOD, QMOOD, and L&K measurement suites (that are described in Section 2.1) as well as the coupling and cohesion measures proposed by Briand et al. Across their primary studies on fault-proneness, Malhotra and Jain made similar observations to those reported by Saxena and Saini (2011) regarding the most common measurement suit being the C&K measures and the most common statistical methods being regression methods. They did, however, observe that there has been an increase in the use of machine learning methods, such as, decision tree, support vector machine, neural networks, etc.

Whereas Malhotra and Jain (2011) limited their review specifically on object-oriented measures, Radjenović et al. (2013) performed their review on a wider scale and included studies that have empirically validated all types of measures for software fault prediction. Apart from not limiting to any specific type of measures, their search strategy covered all papers published until the date of the study in 2011. Even though the review was not limited to any specific types of measures, Radjenović et al. found that object-oriented measures were used the most for fault prediction across all of their primary studies. From the studies that used object-oriented measures, the C&K measures appeared the most frequent and they report that they perform better than the MOOD and QMOOD measures. From the C&K measures, Radjenović et al. observed that CBO, RFC and WMC performed better than LCOM, DIT and NOC in fault prediction. LCOM had a weak ability of predicting faults, and DIT and NOC were unreliable for fault prediction. They also observed that the QMOOD measures are better suited for fault prediction than the MOOD measures; and from the QMOOD measures, the CIS and NOM measures were better at predicting faults than the other measures. Radjenović et al. also found that the most used methods for building models for software fault prediction, in order of frequency across studies, were regression methods, machine-learning methods, and correlation analysis methods. However, Radjenović et al. study is limited to the linkage between measures of internal attributes and one proxy of an external attribute, i.e., fault proneness, whereas our study considers several proxies and/or external quality attributes.

With the many studies already done on software measures (Kitchenham, 2010), our study adds to the body of knowledge by providing an aggregation, using vote-counting, on the consistency and usefulness of the measures across empirical studies on quality evaluation of object-oriented systems. The necessity of this kind of review and aggregation is also noted by Kitchenham (2010). Our study also differs from other studies as our study has a wider and more diverse perspective than the other related reviews. This is mainly because we do not limit our review to measures associated to a specific external quality attribute or internal property, e.g., cohesion. The rationale is that we aim to identify the various empirical studies that have been conducted in relation to the link between measures and external quality attributes.

3 Research Methodology

A systematic literature review (SLR) facilitates in identifying and collecting key papers in a particular area of interest, and evaluating and interpreting the reported discussions and findings (Kitchenham and Charters, 2007). The purpose of such a review is to obtain an overall inclusive impact or influence of contribution of the collected studies (Kitchenham and Charters, 2007). This SLR aims at aggregating and analyzing measures that have been empirically linked with quality of object-oriented source code. The steps proposed by Kitchenham and Charters (2007) were followed for this SLR. The following are the objectives of the review:

- Identify external quality attributes that have been empirically evaluated through source code analysis of object-oriented programs;

- Identify measures obtainable from source code that have been used to evaluate external quality attributes of object-oriented programs through empirical methods (case studies or experiments);
- Aggregate relationships between external quality attributes and the associated code measures that have been established for quality analysis of object-oriented programs and evaluate the strengths of evidence behind these relationships.

3.1 Research Questions

Building from the study aims and objectives, the following main research question and four sub-questions drive this SLR:

RQ1: How are measures derived from source code of object-oriented programs used to evaluate or predict external quality attributes of object-oriented systems in empirical studies?

RQ 1.1: Which external quality attributes have been linked with object-oriented measures in empirical studies?

RQ 1.2: Which methods are used to estimate/predict the external quality attributes from the source code measures?

RQ 1.3: Which source code measures are used to evaluate external quality attributes of object-oriented programs?

RQ 1.4: What is the overall efficacy of object-oriented measures to link with external quality attributes across empirical studies?

3.2 Data Sources and Search Strategy

3.2.1 Keywords

Our search terms included key source code attributes essential for quality analysis of object-oriented programs (Chidamber and Kemerer, 1991; Chidamber et al., 1998; Lorenz and Kidd, 1994; Bansiya and Davis, 2002; Kanellopoulos et al., 2010): abstraction, cohesion, complexity, composition, coupling, encapsulation, inheritance, polymorphism, messaging, size and volume.

Search terms are placed into 6 clusters: product, analysis process, programming language paradigm, internal properties, external quality evaluation and study method. The names used for the clusters represent the different aspects, relevant to this review, covered by the search terms. Electronic databases have differing underlying models and search interfaces, and this may limit the reusability of search strings (Dybå et al., 2007; Brereton et al., 2007). Placing search strings in clusters helped in tailoring search strings to suit search functionalities offered by different electronic databases and also facilitated in identifying additional keywords, e.g., synonyms. Table 2 contains the clusters and the keywords.

Using the information provided in Table 2 the resultant search string takes the following form: Cluster1 AND Cluster2 AND Cluster3 AND Cluster4 AND Cluster5 AND Cluster6.

Table 2: Cluster and Categorization of Keywords

Cluster		Keywords
Cluster1	Product	code OR software
Cluster2	Analysis process	analysis OR check* OR evaluat* OR predict
Cluster3	Programming language paradigm	"object oriented" OR "object-oriented" OR OO OR OOP OR OOPL
Cluster4	Internal properties (for the programming language paradigm)	complex* OR abstraction OR encapsulation OR coupling OR cohesion OR volume OR messaging OR composition OR inheritance OR polymorphism OR class OR method OR function OR modul* OR attribute OR characteristic OR size
Cluster5	External quality evaluation	metric* OR measur* OR indicator
Cluster6	Study method	empirical* OR "case study" OR "case studies" OR experiment*

3.2.2 Data Sources

A search for relevant literature was conducted on the metadata, in particular, on the title, abstract and keywords. The electronic databases used to search for relevant literature were ACM, Compendex and Inspec, IEEE and Scopus. These databases were selected based on the following criteria:

- Coverage of relevant studies (Dybå et al., 2007; Brereton et al., 2007);
- Availability of a functionality to export search results.

In an experience report based on an SLR conducted in the area of agile methods, Dybå et al. (2007) found that electronic databases relevant to software engineering research, such as ScienceDirect and Wiley Science (Brereton et al., 2007), returned similar search results as ACM, Compendex and IEEE. Hence, ScienceDirect and Wiley Science databases were excluded in our review. In addition, Scopus was selected since it claims to be the largest abstract database for peer-reviewed papers. Table 17 in Appendix A shows the search strings used to find for relevant material in each database.

3.3 Study Selection

3.3.1 Inclusion and Exclusion

The criterion for inclusion and exclusion are as follow:

- Only peer-reviewed research articles are included, i.e., only articles published at workshops, conferences and journals, are included;
- An article should have empirical work illustrating the link between measures and external quality attributes, i.e., the work should involve the use of actual data;

- Measures discussed in the article should be related to external quality attributes and obtainable directly from the source code (of object-oriented programs), or if obtainable from internal properties of object-oriented designs they should be judged to be obtainable from the source code of object-oriented programs;
- Articles should have empirical work using measures (as stated in the previous bullet) to relate (including correlation), evaluate, predict or validate at least one external quality attribute or a proxy of an external quality attribute.

Papers that did not fulfill all four of the aforementioned criteria were excluded. The recently published ISO standard on quality, ISO/IEC-25010 (2010), was used to identify potential external quality attributes that the attributes described in the empirical studies could be traced to.

Exclusion of papers consisted of two steps. In step 1, the exclusion of irrelevant papers was conducted by applying the inclusion and exclusion criteria on the titles and abstracts. Two researchers reviewed each paper. The first author reviewed all papers whilst the other reviewers were assigned one third of the papers each. Before the papers were distributed to reviewers, all papers were sorted alphabetically by the articles' author lists to ensure that the researchers would review articles written by the same authors. These steps facilitated the identification of duplicate studies or studies reporting identical empirical results. Table 3 depicts the strategy for deciding whether to include or exclude an article based on reviewers assessments.

Table 3: Inclusion and Exclusion Strategy

Reviewer 1	**Reviewer 2**	**Resolution**
Relevant	Irrelevant	Conduct third review (Exclude if third assessment result is *irrelevant*, else include)
Relevant	Maybe	Include paper
Relevant	Relevant	Include paper
Maybe	Maybe	Conduct third review (Exclude if third assessment result is *irrelevant*, else include)
Maybe	Irrelevant	Exclude paper
Irrelevant	Irrelevant	Exclude paper

In step 2, papers were assessed based on their full-text and excluded based on the inclusion and exclusion criteria. Two of the authors were recognized as having substantially more experience in the area of object-oriented measures. Thus to reduce inclusion or exclusion of papers based on bias, papers were distributed in such a way that each paper was reviewed by a more experienced and lesser experienced reviewer in the area. The strategy shown in Table 3 was used for determining inclusion or exclusion or resolving assessment discrepancies between reviewers. The next activity was the quality assessment of the primary studies.

Table 4: Quality Assessment Checklist

Quality Assessment Questions		**Select Yes/No/ Somewhat**
Research Design	1.Is the research aim clearly stated?	
	2. Is the research methodology clearly described?	
Data Analysis	3. Is the programming language stated?	
	4. Is the size of the data set stated? e.g. system/class	
	5. Was an analysis conducted to check for outliers in the data? (if unclear enter No)	
	6. Is the name of the statistical test provided? (e.g. Pearson, Spearman, etc)	
Measures and Quality Attributes	7. Are the definitions for the measures provided?	
Results and Conclusion	8. Are the correlation coefficients reported?	
	9. Are the p-values reported?	
	10. Is the statistical significance provided or discussed?	
	11. Are validity threats discussed?	
	12. Do the empirical data and results support the conclusions?	

3.3.2 Study Quality Assessment

A quality assessment procedure was applied on the papers' full-text covering the following aspects: research design, data analysis, measures, results and conclusions.

The study quality assessment checklist consisting of 12 questions covering the aforementioned aspects was developed to operationalize the quality assessment activities. The checklist is based on the rigor of reporting and covers the following main perspectives: how well the reviewer is able to understand the research steps taken; details on how the link between object-oriented measures and external quality attributes (or the proxy) was established; and the traceability of the research steps and the study findings and conclusions. Grading of each question was done on a three point scale: yes (weighing two points - indicating that data for the specific question is clearly available), somewhat (weighing one point - data is vaguely available) and no (weighing zero points - indicating that data is unavailable). The questions are tabulated in Table 4.

As suggested by Kitchenham and Charters (2007), Cohen's Kappa statistics was used to measure and evaluate homogeneousness between the more and lesser experienced reviewer. Kappa statistics measure the level of agreement between observers (or in this case reviewers) i.e. the inter-observer variability (Landis and Koch, 1977; Henningsson and Wohlin, 2005). The closer the kappa coefficient is to 1 the higher the agreement level between the observers. The next step was the data extraction.

3.4 Data Extraction

To help achieve the overall aim of the study, data extraction from primary studies was conducted on two main levels: individual measures and prediction models.

3.4.1 Data for Individual Measures

To help with assessing links between individual measures and external quality attributes, the following information was extracted:

- Measures obtainable from the object-oriented source code;
- Internal properties measured by each measure;
- External quality attributes or proxies of external quality attributes;
- Method used to link measures of internal properties to external quality attributes;
- Research methodology and study context;
- Dataset(s).

A “proxy” of an external attribute is an attribute used as a surrogate or as a “stand-in” for an external quality attribute. For example fault-proneness can be used as a surrogate of the external quality attribute, reliability. For the purpose of this study we extracted proxies and/or external quality attributes as reported in the studies.

3.4.2 Data for Prediction Models

To help with understanding measures usefulness for building models for quality prediction, the following information was extracted:

- Prediction model (i.e. set of measures obtainable from source code);
- Internal properties measured by each measure in the prediction model;
- Method used for building the prediction model;
- External quality attributes or proxies of external quality attributes linked to the prediction model;
- Validation method used for the prediction model;
- Research methodology and study context;
- Dataset(s);
- Predictive ability results of the prediction model, for example, performance indicators from confusion matrix results (Jia et al., 2009) or R-squared values.

If in a particular study more than one prediction model is evaluated, or if a comparison is done between different models, the data that is extracted from the study is of the model that out-performs the other model(s). Data synthesis of individual measures was done using a vote-counting approach.

3.5 Data Synthesis: Vote-Counting

Vote-counting (method) was conducted in this study to understand the consistency of the relation that measures have with external quality attributes as reported across empirical studies. The approach helps in understanding the ability of a measure to predict certain quality attributes by combining results from individual empirical studies (Pickard et al., 1998). Vote-counting is done for studies that report significance level tests, i.e. p-values (Pickard et al., 1998). In terms of the relation between the measure and the external quality attribute or a proxy of an external quality attribute, the significance levels denote either a positive, negative or a non-significant relation (Pickard et al., 1998).

The statistical significance test results extracted and combined from the empirical studies are those done using p-values. The significance levels often used are 0.05 and 0.01. A positive or negative significant relationship indicates "success" (Pickard et al., 1998), i.e., suggesting a link between a measure and an external quality attribute. Thus a p-value less than 0.05 or 0.01 (depending on the level used in the study) denotes a success; and a p-value greater than 0.05 or 0.01 indicates failure or non-significant. The vote-counting procedure is thus done for the three possible outcomes as reported in the studies: positive, negative and non-significant outcomes. These outcomes denote the effect of the measures and proxies of external quality attributes or external quality attributes.

For each study that investigated the relationship between several measures and/or several external attributes, each measure-attribute pair is considered, i.e., counted, separately as an outcome. Different aspects of data, e.g., severity levels of defects, are also considered separately as outcomes.

A cut-off point is often used in vote-counting for rejecting a hypothesis that there is no effect between two variables (Pickard et al., 1998). We use a cut-off point of 50% of outcomes to reject the hypothesis that a measure has no effect on the external quality attribute or proxies of that external quality attribute. We believe that practitioners and researchers would benefit from (or are interested in) identifying measures that have been empirically evaluated in many studies and have been significant in more than half the studies.

4 Conducting the Review

Table 5 contains the years covered and the number of search results retrieved for each database. Evaluation of the cumulative search results (from a total of 1356 papers) revealed 677 duplicates, 20 lecture slides and 25 editorials that were removed using the Endnote and the Jabref reference manager tools. This left 634 potentially relevant studies.

The next step in the review process was the study selection process. This involved screening and removing irrelevant papers from the remaining 634 papers, based on title and abstract. After that, papers that remained were then screened and removed based on the their full-text.

4.1 Inclusion and Exclusion Process

The initial step in the exclusion of irrelevant papers was performed by applying the inclusion and exclusion criteria on the titles and abstracts of 634 papers. A total of 410 irrelevant papers and 17 duplicates were found and removed. This left 207 papers. An attempt was then made to obtain full-text of the remaining papers. After downloading from databases and emailing authors for papers we could not

Table 5: Search Process

Digital Library	**Years covered (according to the functionality)**	**Results**
Compendex & Inspec	Until 2012	Retrieved 759
IEEE	Until 2012	Retrieved 64
Scopus	Until 2012	Retrieved 467
ACM	Until 2012	Retrieved 66
	Total Papers (All Databases)	1356
	Removed	Removed: duplicates (677 papers), Lecture slides (20 papers), Editorial (25)
	Total Papers Left	634

obtain from the databases, we were unable to obtain full-text for three papers and we were unable to find the English versions of seven other papers. This left 197 papers as potential primary studies.

In the next step, the 197 papers were evaluated on their relevance based on their full-text using the inclusion and exclusion criteria. Out of 197 papers, four duplicates were found and 90 papers were found to be irrelevant. As a result, 103 papers remained. We then re-reviewed the remaining 103 papers to ensure that our list of primary studies contained studies that could be traced to at least one external quality attribute described in ISO/IEC-25010. Four papers were found to not have conducted an investigation of an external quality attribute or proxy in the context of object-oriented source code quality. Thus we were unable to map the papers to attributes defined in the new ISO standard ISO/IEC-25010. These four papers were excluded. Finally, 99 papers were selected as primary studies.

4.2 Study Quality Assessment Process

Quality assessment of the 99 studies was done concurrently with the full-text inclusion and exclusion process. As a result, the authors of this paper applied the quality assessment criteria to the full-text of the same papers that they evaluated during the inclusion and exclusion process.

Table 6: Kappa Results of Quality Assessment Process

Reviewers	**Kappa Statistics**	**Agreement Level**
ME1 and LE1	0.571	Moderate
ME1 and LE2	0.473	Moderate
ME2 and LE1	0.468	Moderate
ME2 and LE2	0.549	Moderate

Using the evaluation of Kappa statistics proposed by Landis and Koch (1977) the strength of the level of agreement between reviewers was generally moderate. The results of the kappa statistics are tabulated in Table 6. In Table 6 the more experienced reviewer is referred to as ME and the less experienced as LE.

Figure 2 summarizes steps taken to obtain primary studies.

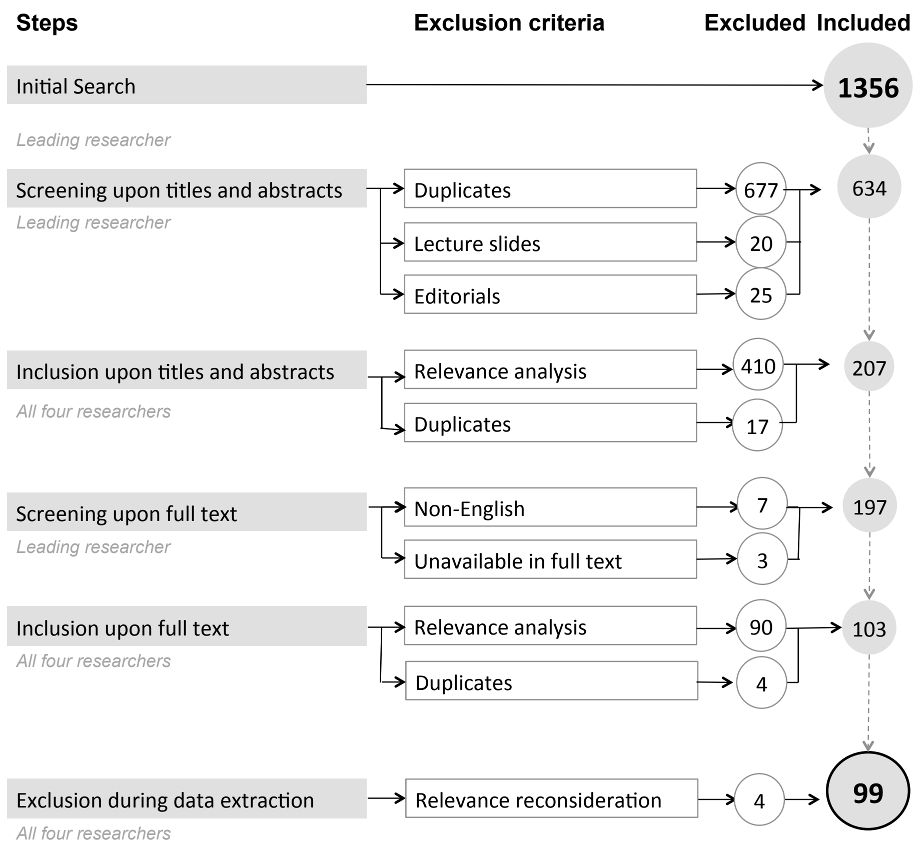


Fig. 2: Study Selection Process

5 Validity Threats

5.1 Selection Bias

It is possible to miss or exclude relevant papers during a literature review and this can significantly affect the results. During each step that involved the exclusion of papers, methods were implemented to ensure that relevant papers were not inadvertently excluded. To begin with, the identification of search terms and the formulation of search strings involved the use of a validation set to evaluate search results and to identify additional search terms. A validation set is a list of articles that should be identified by the search procedure. The papers in the validation set were chosen by identifying relevant articles in Google scholar, and by identifying relevant papers that have cited the seminal paper from Chidamber and Kemerer (1994). This procedure also helped in the formulation of suitable Boolean expressions for search strings.

Formulation of the inclusion and exclusion criteria involved two pilot runs to remove ambiguities in the criteria and to improve homogeneity between reviewers. The piloting emulated the planned inclusion and exclusion process, which was done on the papers title and abstract, so as to reduce inconsistencies between the first author and the other authors prior to the actual process. After each pilot the first author's assessment results were compared against the other authors' assessment results. A meeting was conducted to discuss contrasting interpretation of the assessment criteria and disparities in the assessment results after the first round of piloting. Amendments were duly made to the criteria to remove ambiguities before the second round of piloting. A meeting to discuss inconsistencies and to polish certain vague areas in the criteria also followed the second round of piloting. During this second meeting it was uncovered that disagreements that arose during the piloting were related to the definition of certain essential terms such as source code. Definitions were thus formally made and agreed upon, and all researchers agreed that no more piloting was needed.

Selection bias can also result in excluding relevant papers. To minimize primary study selection bias during assessment of the papers full-text division of papers was done in such a way that two authors that had lesser experience in the area of object-oriented quality assessment would not review the same papers. Thus authors that reviewed each paper consisted of both a more and a lesser experienced reviewer. In cases with a discrepancy between two reviewers, a meeting was conducted to discuss differences and to determine inclusion or exclusion of the paper. If the reviewers were unable to reach an agreement over the inclusion or inclusion of the paper, a third person would review the paper. The third reviewers' assessment would be used to determine inclusion or exclusion of the paper.

A quality assessment checklist was developed and a pilot was conducted to check for homogeneity in terms of usage and interpretation of the checklist between reviewers. After each pilot a meeting was held to discuss effectiveness and rigor of the quality criteria. During the meetings, disagreements in evaluation between the reviewers were discussed with the goal of reaching a consensus on how to approach certain vagueness or missing details found in certain papers. The meetings were also used to discuss necessary changes to the quality criteria with the goal of removing ambiguities and redundancies in the assessment criteria. The same arrangement of authors for reviewing the full-text papers used in the inclusion and exclusion process was also used for the quality assessment process.

5.2 Reviewer Bias

The rationale for measures in terms of what they intend to measure can be context dependent, and interpretation of measures and results may vary across studies (Lincke et al., 2008). Implication of results is also influenced by the dependency that measures have on the tools used for extraction. That is, results for software quality evaluation are influenced by the definition of the measures and the measurement process (Lincke et al., 2008). Measurement process, definition of the measures and interpretation of the results are subject to reviewer bias.

Furthermore, external quality attributes investigated in studies can be abstract and not explicitly stated. For example, some studies may report findings on external quality attribute proxies such as change-effort or number of bad smells. This leaves the association of the proxy discussed in the paper to a specific external quality attribute up to the interpretation and subjective views of individuals. To ensure that the studies were aligned to an appropriate external quality attribute reviewers re-reviewed the papers after the full-text assessment. During this activity, papers were ordered in an alphabetic order by author names and split amongst the reviewers such that each paper was reviewed by two reviewers that consisted of a more experienced and lesser experienced reviewer. As a result of the ordering, some reviewers reviewed some papers they had included during the full-text assessment. The reviewers extracted the external quality attributes and their proxies as stated in the studies.

In cases where the external quality attribute were not clearly stated, the reviewers used the study contexts and proxies discussed in the paper and the external quality attributes defined in ISO/IEC-25010 (2010) to derive the most appropriate external quality attribute. As a result of this re-review, three paper were identified as candidates for exclusion as they did not meet the inclusion criteria, i.e., the proxy was not clear and the external quality could not be derived. These papers were returned to the full-text assessment reviewers of the papers so they could re-review the papers and determine if they agree or disagree with the new review from the other reviewers. For two of the papers the full-text assessment reviewers agreed to exclude. For the other paper, one full text assessment reviewer agreed to exclude whilst the other wanted to include. A meeting was held between all four reviewers during which this paper was discussed and a consensus was reached to include the paper. This re-review step is indicated in Figure 2 after the full-text assessment step.

5.3 Vote-counting Bias

There are a lot of publicly available datasets, for example, open source projects, which researchers often use for their investigations, e.g., relationship between a complexity measures and a reliability proxy like post release faults. Often researchers end up using the same dataset for their investigations in different empirical studies. As a result, some measures may be investigated on the same dataset in different empirical studies. Vote-counting results for such measures can be biased and may not be indicative of the usefulness of the measure since they have only been investigated and re-investigated on the same dataset.

During the vote-counting process, if a measure was investigated on the same dataset, we took into consideration the difference in the way the dataset was used in the studies. For example, some studies may categorize the dataset by fault severity levels, and some may not, or some studies may use use only one version of the system whilst another study uses several. We also took into consideration the difference in the method used to investigate the link between the measure and the attribute under study, e.g., univariate linear/logistic regression, or artificial immune recognition system, or decision trees, etc. Thus, results from several studies that investigate the same measure using the same characteristics of a dataset, with the same statistical method, were counted as one result.

It may also occur that studies report contradicting outcomes for certain measures, i.e., positive significance in one study and negative significance in another. This may result in misleading vote-counting results. We alleviated this problem by complementing the vote-counting results with a plot of the sum of positive and negative significant results against the number of clear results or against the number of datasets (see Figure 6 and Figure 7). The resultant plot provides a visual of the overall direction of the relationship between the measure(s) and the quality attribute(s) reported by the empirical studies. This helps to compare the consistency, usefulness and strength of evidence of a measure in comparison to the other measures according to vote-counting results of reports from empirical studies.

6 Summary of Primary Studies

6.1 Primary Studies

The 99 primary studies and the citation from the reference list are tabulated in Table 7. This representation of primary studies and references is adapted from a separate study by Díaz et al. (2011). From hereon, each primary study is referred to using the ID in Table 7.

6.2 Quality Assessment Results

The quality of each paper was assessed by two reviewers. Each reviewer could give a maximum of 24 points to a paper; hence the maximum possible score for the quality assessment for each paper was 48. Scores for the primary studies range between 17 and 48, i.e., 35% and 100%. Only 9 out of 99 papers scored below half of the possible total score. Percentile rankings of the scores are highlighted in Figure 3. Studies within the 25% percentile scored very low on the data analysis and the results and conclusion aspect from our quality assessment checklist (see Table 4 in Section 3). In most cases the studies omitted discussions on certain analysis information that if not considered can potentially skew findings such as analysis of outliers during the statistical analysis. Most of the studies within this percentile also did not discuss study validity threats, which makes it difficult for the reader to understand trustworthiness of the reported findings or if other factors may have influenced the study results (Robson, 2011).

6.3 Publication Venues and Years

Figure 4 shows the distribution of primary studies by publication years and venues. As depicted in Figure 4 the list of primary studies constitute of papers published from 1996 onwards. The figure also highlights the year some of the well-known measures were proposed in relation to the publication years of the

Table 7: Studies and References

ID	Citation	ID	Citation	ID	Citation	ID	Citation
S1	Abreu and Melo (1996)	S26	Briand et al. (1997)	S51	Gyimóthy et al. (2005)	S76	Ramasubbu et al. (2012)
S2	Abubakar et al. (2006)	S27	Briand et al. (2000)	S52	Holschuh et al. (2009)	S77	Rathore and Gupta (2012a)
S3	Aggarwal et al. (2007)	S28	Briand et al. (1999)	S53	Huang and Zhu (2009)	S78	Rathore and Gupta (2012b)
S4	Aggarwal et al. (2009)	S29	Bruntink and van Deursen (2006)	S54	Janes et al. (2006)	S79	Revelle et al. (2011)
S5	Al Dallal (2012a)	S30	Cruz and Ochimizu (2010)	S55	Jia et al. (2009)	S80	Reyes and Carver (1998)
S6	Al Dallal (2011b)	S31	Cartwright and Shepperd (2000)	S56	Jin et al. (2009)	S81	Shatnawi (2010)
S7	Al Dallal (2011a)	S32	Catal et al. (2007)	S57	Kamiya et al. (1999)	S82	Shatnawi et al. (2010)
S8	Al Dallal (2012b)	S33	Ajrnal Chaumun et al. (1999)	S58	Kanmani et al. (2004)	S83	Shatnawi and Li (2008)
S9	Al Dallal and Briand (2010)	S34	Dagpinar and Jahnke (2003)	S59	Kanmani et al. (2007)	S84	Singh and Verma (2012)
S10	Al Dallal and Briand (2012)	S35	Dandashi and Rine (2002)	S60	Karus and Dumas (2012)	S85	Singh et al. (2007)
S11	Alshayeb and Li (2003)	S36	Darcy et al. (2005)	S61	Lavazza et al. (2012)	S86	Singh et al. (2009a)
S12	Aman et al. (2006)	S37	Dick and Sadia (2006)	S62	Li and Shatnawi (2007)	S87	Singh et al. (2010)
S13	Arisholm (2006)	S38	El Emam et al. (2002)	S63	Liu et al. (2009)	S88	Singh et al. (2009b)
S14	Arisholm and Sjøberg (2000)	S39	Elish and Rine (2006)	S64	Malhotra and Jain (2012)	S89	Singh et al. (2011)
S15	Babich et al. (2011)	S40	Elish (2010)	S65	Marinescu and Marinescu (2011)	S90	Singh and Saha (2012)
S16	Badri et al. (2011)	S41	Elish et al. (2011)	S66	Nguyen et al. (2011)	S91	Subramanyam and Krishnan (2003)
S17	Badri and Toure (2012)	S42	Eski and Buzluca (2011)	S67	Olague et al. (2008)	S92	Succi et al. (2003)
S18	Bakar (2011)	S43	Fioravanti and Nesi (2001)	S68	Olague et al. (2006)	S93	Szabo and Khoshgoftaar (2004)
S19	Bandi et al. (2003)	S44	Genero et al. (2001)	S69	Olague et al. (2007)	S94	Újházi et al. (2010)
S20	Bansiya and Davis (2002)	S45	Giger et al. (2012)	S70	Olbrich et al. (2009)	S95	Xu et al. (2008)
S21	Basili et al. (1996)	S46	Goel and Singh (2008)	S71	Pai and Bechta Dugan (2007)	S96	Zhou and Leung (2006)
S22	Benlarbi et al. (2000)	S47	Nair and Selvarani (2012)	S72	Yu et al. (2002)	S97	Zhou et al. (2012)
S23	Benlarbi and Melo (1999)	S48	Guo et al. (2011)	S73	Poshyvanyk et al. (2009)	S98	Zhou et al. (2010)
S24	Bocco et al. (2005)	S49	Gupta and Chhabra (2009)	S74	Pritchett and W. (2001)	S99	Zimmerman et al. (2011)
S25	Briand et al. (2002)	S50	Gupta and Chhabra (2012)	S75	Quah and Thwin (2002)		

primary studies. Journal publications constitute of 60% of all primary studies. 79% of the studies were published after 2002 (the year QMOOD measures were published). Figure 4 shows that in recent years there has been a growing number of conference and journal publications investigating the relationship between object-oriented measures and quality.

Figure 4 shows that the earliest empirical studies that we found on object-oriented measures and quality is five years after the publication of the C&K mea-

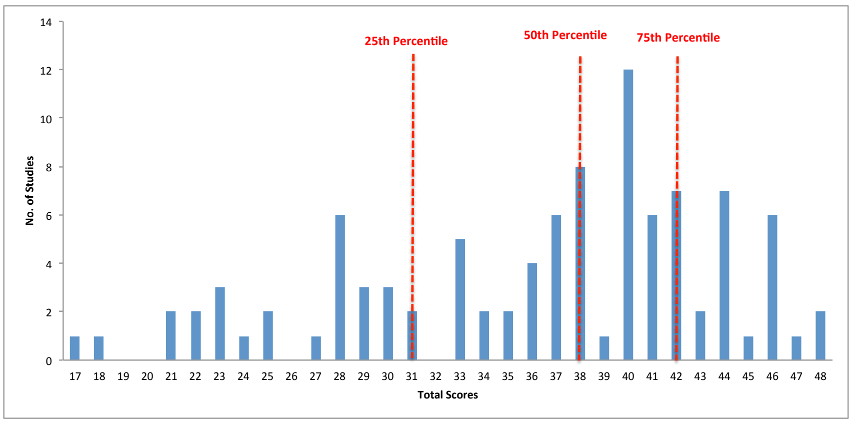


Fig. 3: Distribution of Quality Scores

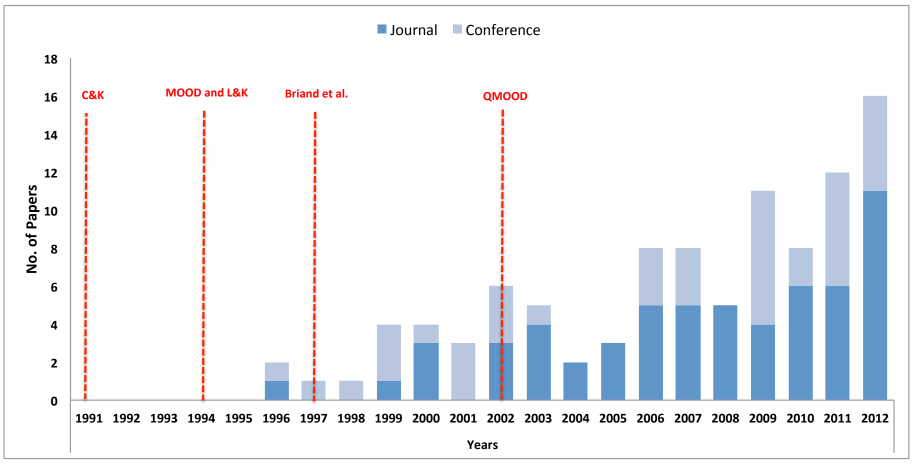


Fig. 4: Publication Venues and Years

surement suite. This led us to investigate the most commonly used measurement suites. Table 8 shows the number of primary studies, a total of 82 studies, that used at least one of the common measurement suits C&K, L&K, MOOD and QMOOD, or a combination of them. The other studies did not use these measurement suites. Despite other measures being proposed in literature, C&K measures appeared in most of our primary studies (80% of the primary studies).

There is one other set of measures proposed by Briand et al. (1997) that is not included in Table 1 that we found in some of the primary studies. These measures, however, do not measure diverse properties as the other measurement sets in Table 1. The measures were proposed to primarily measure coupling properties. Most of the primary studies used at least one of the measures from C&K, L&K, MOOD, QMOOD, and/or Briand et al. (1997) (84% of the primary studies).

Table 8: Measurement Suites from Primary Studies

	Suite Name	**Number of Studies**
Single Suite	C&K	54
	QMOOD	1
	MOOD	1
	Briand et al.	1
	L&K	0
Combination of Suites	C&K and L&K	8
	C&K, L&K and Briand et al.	6
	C&K and Briand et al.	5
	C&K and QMOOD	2
	C&K and MOOD	1
	C&K, MOOD and Briand et al.	1
	C&K, QMOOD and MOOD	1
	C&K, QMOOD, L&K and Briand et al.	1
	QMOOD and L&K	1

6.4 Research Methodology and Study Contexts

Table 9: Research Method and Contexts

	Context				
Method	Academic	Open Metrics Database	Open Source	Industry	Unclear
Experiment students	S1, S19, S24, S44, S66				
Experiment professionals	S36				S76
Survey			S61(DS22)		S35(DS1)
Case study	S3, S4, S21, S26, S27, S43, S58, S59, S85, S86			S47	S13, S57
Archival analysis	S34, S56,	S2, S32, S46, S71, S77, S78, S87, S88, S89, S95, S96	S5, S6, S7, S8, S9, S10, S11, S12, S15, S16, S17, S18, S24, S29(DS1), S30, S37, S40, S41, S42, S45, S48, S49, S50, S51, S60, S61(DS22), S62, S63, S64, S65, S68, S69, S67, S70, S73, S79, S83, S81, S82, S84, S90, S94, S98, S97, S99	S14, S20, S23, S22, S28, S25, S29(DS4),S31, S33, S38, S52, S53, S54, S55, S72, S75, S91, S92, S93	S35(DS1), S74, S80

Table 10: Most Common Datasets

System	Programming Language	No. of Studies	Studies
Eclipse	Java	13	S12, S15, S41, S45, S48, S61, S62, S65, S81, S82, S83, S98, S99
KC1 NASA dataset	C++	9	S2, S32, S46, S51, S71, S87, S88, S95, S96
Apache (ANT, Commons IO Lucene Core, and/or POI)	Java	8	S16, S17, S29, S50, S61, S64, S70, S97
Art of Illusion	Java	6	S5, S6, S7, S8, S9, S10
JabRef	Java	6	S5, S6, S7, S8, S9, S10
JfreeChart	Java	6	S16, S17, S42, S61, S90, S97
Mozilla	C++	5	S37, S51, S63, S73, S94
GanttProject	Java	5	S5, S7, S8, S9, S10
Openbravo	Java	5	S5, S7, S8, S9, S10
Rhino	Java	4	S67, S68, S69, S79
LALO	C++	3	S4, S23, S28
XGen Source Code Generator	Java	3	S40, S49, S50
12 Systems by Students at University School of Information Technology	Java	2	S3, S86
Azureus	Java	2	S12, S45

Table 9 provides a classification of the primary studies according to the study contexts and the research methods described in the studies. "Experiment students" in Table 9 refers to studies that used subjects from academia. "Experiment professionals" refers to studies conducted using subjects with multiple years of experience in their respective field. Archival analysis refers to a study conducted using historical data that was systematically collected and stored (Robson, 2011). We found some studies that use more than one dataset, of either the same or different context or research method. Such studies are denoted with *DSn* in Table 9, where *DS* refers to dataset, and *n* is the number of datasets associated with that context and research method. In this case dataset refers to the software release, the system or the project described in the studies.

With regards to the study context categorization in Table 9 we found that the studies under the context open metrics database and archival analysis, use the

KC1 NASA dataset or dataset from the PROMISE repository. Most of the studies under the open source and archival analysis uses Eclipse and Mozilla. Table 10 provides a list of datasets that was used in at least two papers from the primary studies.

Catal and Diri (2009) found that the use of publicly available datasets in fault-proneness prediction studies increased from 31% to 52% after 2005. Our study results provide a more longitudinal view and wider perspective of quality attributes across studies. We found that between 1996 and 2012, 57% of the studies used publicly available datasets, i.e., open metrics database or open source. This is well short of the 80% proposed by Catal and Diri as being the “ideal level” for helping the research community to validate each others findings.

7 Results and Analysis

7.1 External Quality Attributes (RQ1.1)

The mapping of the external quality attributes, the related proxies and the primary studies is shown in Table 11. Only 3% (3 out of 99) of the primary studies investigated more than one external quality attribute. These studies are S1 (that is traced to reliability and maintainability), S20 (that is traced to reusability, testability, flexibility, functionality, extendibility, effectiveness, understandability) and S33 (that is traced to changeability and maintainability) and they appear in Table 11 under all the external quality attributes that they investigated.

As shown in Table 11, maintainability is the second most investigated external quality attribute. However, some of the other external quality attributes traced to studies in Table 11 can be linked to maintainability particularly in the context of source code quality. For example, understandability can be linked to maintainability because in order to successfully test, change and maintain, one would need to analyze and understand the source code artifacts. In addition, ISO/IEC-25010 (2010) links reusability, testability and changeability to maintainability.

Table 11: Studies and External Quality Attributes

External Quality Attribute	**Proxy**	**No. Of Studies**	**Studies**
Reliability	Fault-proneness	47	S3, S4, S5, S6, S7, S8, S9, S10, S15, S21, S23, S22, S25-S28, S30, S32, S38, S41, S43, S46, S51, S53, S55-S57, S59, S63, S64, S69, S67, S71, S72, S77-S79, S81, S84-S89, S94, S96, S98, S99
	Defect-proneness	6	S47, S31, S52, S54, S65, S92
	Number of faults	5	S41, S58, S75, S93, S95
	Class error probability	2	S62, S82
	Number of defects	2	S2, S91

Continued on next page

Table 11 – *Continued from previous page*

External Quality Attribute	Proxy	No. Of Studies	Studies
	Defect density	2	S1, S37
	Post-release defect density	2	S18, S39
	Number of revisions	1	S74
	Post-release failure-proneness	1	S99
	Bug-proneness	1	S48
	Number of bugs	1	S48
	Perception of trustworthiness and reliability	1	S61
Maintainability	Effort	1	S19
	Maintenance effort	1	S24
	Change logs	1	S34
	Perfective maintenance effort	1	S36
	Rework	1	S1
	Maintenance time/effort	1	S44
	Maintenance effort	1	S66
	Shannon entropy and error-proneness	1	S68
	Change frequency and change size	1	S70
	Class error probability	1	S83
	Change impact	1	S33
	Code churn	1	S60
	Maintenance effort and perceived ease of maintenance	1	S76
	Change-proneness	1	S42
	Source code changes and type of source code changes	1	S45
Testability	Test class size	4	S16, S17, S90, S97
	Testing effort	1	S29
Changeability	Effort	2	S13, S14
	Change-proneness	1	S73
	Change impact	1	S33

Continued on next page

Table 11 – *Continued from previous page*

External Quality Attribute	**Proxy**	**No. Of Studies**	**Studies**
Reusability	Viewpoints of people (adaptability, completeness, maintainability, understandability)	1	S35
	Reuse	1	S80
	Perceived effort for package reuse	1	S50
	No proxies	1	S20
Understandability	Average effort to understand package	1	S40
	Ranking/Effort to understand package	1	S49
	No proxies	1	S20
Maintenance effort	Changes in classes	1	S11
Maintenance cost	Cost-proneness of classes	1	S12
Flexibility	*No proxies*	1	S20
Extendibility	*No proxies*	1	S20
Functionality	*No proxies*	1	S20
Effectiveness	*No proxies*	1	S20

For the purposes of our analysis, studies for changeability, testability and understandability are perceived as surrogates for maintainability in the context of source code quality, and are thus traced to maintainability studies. Furthermore, using definitions of reusability, flexibility and extendibility from ISO/IEC-25010, and given the context of source code quality and the descriptions in the corresponding studies traced to these attributes, we also link the studies for these attributes to maintainability aspects.

To summarize, Figure 5 provides a graphical representation of the link between maintainability and the other external quality attributes. The interpretation of Figure 5 is also supported by some of our study data. For example, there are two studies regarding the external quality attribute changeability that use the effort as a proxy. On the other hand, we have a study regarding maintenance effort as an external quality attribute.

This grouping results in all studies being traced to four main external quality attributes: reliability, maintainability, effectiveness and functionality.

7.1.1 Reliability Studies

Most of the studies, 68 out of 99 (69%), are traced to reliability. Most of the studies traced to reliability are linked to fault-proneness. Fault-proneness is the most investigated proxy across all the proxies for external quality attributes across all

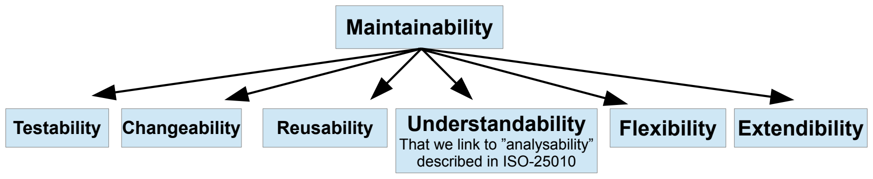


Fig. 5: Maintenance

primary studies. 47% (47 out of 99) of the primary studies were on fault-proneness. In studies traced to reliability the most measured property is coupling. Followed by size, complexity, cohesion, inheritance, stability, polymorphism, encapsulation, abstraction and messaging properties in that order. The complexity measure RFC and the inheritance measure DIT appear most frequently across studies that link measures, individually, to proxies for reliability; followed by CBO (a coupling measure), NOC (an inheritance measure), WMC (a complexity measure) and LOC (a size measure). Measures for abstraction, encapsulation, messaging, polymorphism and stability were very few or they appeared in only one or two studies.

7.1.2 Maintainability Studies

31% (31 out of 99) of the primary studies are traced to maintainability. Most of the measures we found in the studies for maintainability measured complexity properties. The properties measured, from the property most measured to the least, are: inheritance, complexity, coupling, size, encapsulation, cohesion, hierarchies and polymorphism.

7.1.3 Functionality and Effectiveness Studies

There is only one primary study, and it is the same study, that is traced to both functionality and effectiveness (i.e., 1% of the primary studies). The study uses measures from the QMOOD suite which consists of measures that quantify abstraction, cohesion, coupling, complexity, composition, encapsulation, inheritance, messaging, polymorphism and size properties.

7.2 Prediction Models (RQ1.2)

To identify useful measures for building effective prediction models, we only analyzed models that were both validated and had their predictive abilities explicitly reported in the primary studies. None of the models extracted for effectiveness and functionality met this criterion. The validation method and predictive abilities for some of the models for reliability and maintainability were reported. All models that did not meet the criterion are listed in Appendix B.

7.2.1 Reliability

The validation methods and the predictive ability were reported for 59% (29 out of 49) of the models extracted from reliability studies. More models extracted were built from regression models (30 out of 49) followed by those built using machine learning methods (19 out of 49).

Table 12 provides the list of the prediction models that had the validation method and their predictive ability clearly stated in the studies. The predictive ability shown in the table, rightmost column, is the information provided in the related study (the study ID is shown in the leftmost column). A total of 29 from 49 models extracted from reliability studies had a validation method stated and the predictive ability reported. 62% of these 29 models were built using regression methods and 38% were built from machine learning methods. Most of the models contained at least one C&K [1] measure. The coupling measure CBO, appears most frequently in the prediction models. Followed by the complexity measure RFC, the inheritance measure NOC, and then the size measure LOC.

Table 12: Reliability:Prediction Methods

Study	**Proxy**	**Best Set of Measures**	**Method**	**Validation Method**	**Predictive Ability**
S4	Fault-proneness	OMMIC, NMI, NM, RFC, PM, MPC, ICH	Multivariate Logistic Regression Analysis	9-cross validation	Accuracy (96.5%), Specificity (97.63%), Completeness (92.8%), Precision (97.2%), R-Squared (85.34%)
S9	Faults	SCC, SNHD	Multivariate Logistic Regression Analysis	ROC Analysis	Precision (69.60%) Sensitivity (69%)
S21	Fault-proneness	DIT, RFC, NOC, CBO	Multivariate Logistic Regression Analysis	Comparison with Traditional Code measures	Completeness(88%) Precision(60%)
S25	Fault-proneness	CBO', ICP_L, ICH, NIH-ICP, OCAIC, OCMIC	Multivariate Logistic Regression Analysis	10-fold cross validation	Accuracy (91%), Completeness (97.7%)
S27	Fault-proneness	RFC, NOP, RFC1_L, NMI, FMMEC, NI-HICP_L, CLD (coupling, cohesion, and inheritance measures)	Multivariate Logistic Regression Analysis	10-cross validation	R-squared (53%), Accuracy (81%), Completeness (94%)

Continued on next page

[1] From here on in, the original LCOM proposed by Chidamber and Kemerer (1991) is referred to as LCOM1

Table 12 – *Continued from previous page*

Study	**Proxy**	**Best Set of Measures**	**Method**	**Validation Method**	**Predictive Ability**
S28	Fault-proneness	OCMIC, BF5, BF6	MARS-Multivariate Adaptive Regression Splines	10-cross validation	significant at 0.01, Accuracy (68%), Completeness (73%)
S32	Fault-proneness	LOC + 6 CK measures	Artificial Immune Recognition System (AIRS)	Compare with other studies	Correctness (72.4%), Precision (74.1%) Sensitivity (71.2%)
S39	Post-release defect density	NSCGR, NSAR	Multiple Regression Analysis (Stepwise)	leave one out cross validation	R-squared (73.3%)
S58	Number of faults	CBO, CLD, Co and LCOM4	Adaptive-Neuro Fuzzy Inference System (ANFIS)	Comparison between Fuzzy and Logistic Regression models	R-Squared (82.98%)
S67	Fault-proneness	SDMC, NTM	Multivariate Bionomial Logistic Regression	Holdout cross validation	Not significant P-Value greater than 0.05
S71	Fault-proneness	CBO, WMC, RFC, SLOC, LCOM* [2]	Bayesian Binomial Logistic Regression	10-fold cross validation	Accuracy (24.8%) Specificity (85.23%) Precision (75.17%) Sensitivity (59.65%)
S72	Fault-proneness	NMC, LOC, CBOin, CBOout, RFCin, RFCout, LCOM*, DIT, NOC, Fan-in	Multivariate Analysis	Same dataset	R-squared (60.3%)
S75	Number of faults	RFC, IC, AMC, CBM, DIT, WMC, NOMA, CBO, NOC	Neural Networks (Ward Network)	compare Neural Network and Logistic regression models	R-squared (99.34%)
S85	Fault-proneness	RFC, WMC, CBO	Artificial Neural Networks	9 cross validation / AUC / ROC curves	Accuracy (83.5%), Specificity (87.5%), Sensitivity (83.78%)

Continued on next page

[2] LCOM*: variation of LCOM proposed by Rosenberg and Hyatt (1997)

Table 12 – *Continued from previous page*

Study	**Proxy**	**Best Set of Measures**	**Method**	**Validation Method**	**Predictive Ability**
S86	Fault-proneness	LOC + 6 C&K measures	Decision Tree	9-cross validation	Specificity (95%), Sensitivity (97.3%)
S87	Fault-proneness	SLOC + 6 C&K measures	Decision Tree	ROC analysis, 10-cross validation	Specificity (87.2%), Completeness (94%), Sensitivity (91.5%), Precision (88.9%)
S88	Fault-proneness	SLOC + 6 C&K measures	Support Vector Machine	10 Cross Validation	Specificity (81.39%) Completeness (85.66%) Precision (78.62%)
S96	Fault-proneness	SLOC, RFC, NOC, CBO, LCOM*	Multivariate Logistic Regression Analysis	LOO cross validation	Accuracy (63.79%), Completeness(76.68%), Precision (71.03%)
S41	Pre-release fault	NC, Ce, AHF, RFC	Multivariate Linear Regression	Mean Absolute Error, Standard Deviation of Absolute Error, and Root Mean Square Error	R-Squared (55%)
S48	Bug-proneness	SVLoD, WVLoD,CBO, WMC, RFC, LOC, DIT, LCOM1	Multivariate Logistic Regression Analysis	AUC=0.8590	Precision (99.12%), Sensitivity (67.67%)
S84	Defect prediction	unclear which CK measures	Naive Bayes	10 fold cross-validation, AUC=0.8252, F measure = 0.962	Precision (96.9%), Sensitivity (95.6%)
S47	Defect-proneness	WMC	Linear Multi-functional Regression Equations	Validated on 5 different commercial dataset	R-Squared (99.41%)
S15	Fault prediction	VAR, NEH, CUS, CLS	Multivariate Binary Logistic Regression	Validated on successive Eclipse n+1 releases (i.e., 2.0-2.1, 2.1-3.0, 3.0-3.1, 3.1-3.2, 3.2-3.3)	Average Accuracy (74.2%), Average Specificity (91%), Average Sensitivity (45.4%)

Continued on next page

Table 12 – *Continued from previous page*

Study	**Proxy**	**Best Set of Measures**	**Method**	**Validation Method**	**Predictive Ability**
S5	Fault-proneness	LCOM5, LCC, DC(D)	Multivariate Regression Analysis	k-cross validation, AUC=65.2%	Precision (88.7%), Sensitivity (87%),
S10	Fault-proneness	LSCC, CC(s), SCOM(2), LCOM2, TCC, DC(D)	Multivariate Regression Analysis	k-cross validation, ROC area = 67.2%	Precision (66%), Sensitivity (67.5%)
S64	Fault-proneness	WMC, RFC, CBO, LCOM2, SLOC	Bagging	k-cross validation (k=10), AUC= 0.876	Specificity (80.1%), Precision (81.99%), Sensitivity (82.9%)
S78	Fault-proneness	WMC, CBO, RFC, DIT, NOC, IC, CBM, CA, CE, MFA, LCOM2, LCOM3, CAM, MOA, NPM, DAM, AMC, LOC and CC.	Random Forest	Validated the model using successive releases, AUC (0.9)	Accuracy (90%), Precision (90%), Sensitivity (90%)
S89	Fault-proneness	CBO, LOC	J48	k-cross validation, size of k is unclear	Accuracy (70.24%), Precision (60%), Sensitivity (90%)
S61	Percieved reliability	average CBO, average NOM per class	Binary Logistic Regression (BLR)	Mean Magnitude Relative Error (MMRE) = 14%	R-Squared (91.9%)

7.2.2 Maintainability

Six out of 18 (33%) models extracted from maintainability studies had both the validation and predictive ability reported. Interestingly there is one instance found, S97, in which a model consists of measures that quantify the same property, i.e., cohesion. All the other measures consists of measures that quantify different properties.

Table 13: Maintainability: Prediction Models

Study	**Proxy**	**Best Set of Measures**	**Method**	**Validation Method**	**Predictive Ability**
S13	Effort	OMAEC_CP, CS_CP	Multiple Linear Regression	Cross Validation	R-squared (77.5%)

Continued on next page

Table 13 – *Continued from previous page*

Study	**Proxy**	**Best Set of Measures**	**Method**	**Validation Method**	**Predictive Ability**
S16	Testing class size	LC(D), LCOM2, LCOM5	Multivariate logistic regression analysis	AUC= 0.822	R-Squared (29.4%)
S45	Type of changes	object-oriented measures (unclear if it is all C&K measures minus NOC)	Bayesian Network	10 fold cross-validation and AUC	Precision (87%), Sensitivity (83%)
S60	Code-churn	No. of files, LOC, No. of statements, No. of lines with comments, % of lines with comments, No. of functions, No. of types (classes, interfaces, structures), Avg. No. of methods per class, Avg. No. of statements per method, Avg. No. of calls per method, % of branching statements, Max. block depth, Avg. block depth, Max. complexity, Avg. complexity	Neural Networks	7:1 split cross-validation	R-Squared (66.8 %)
S90	Testing class size	NOM, WMC, CBO, MPC, LCOM2	Multivariate regression model	70% for training and 30% for testing, and MMRE (Mean magnitude of relative error)	R-Squared (71.7%)
S97	Testing class size	LCOM1, LCOM2	Partial least square regression	LOO cross-validation	R-Squared (75.2%)

7.3 Vote-Counting (RQ1.3 and RQ1.4)

Studies traced to effectiveness and functionality were too few to conduct a meaningful analysis. Thus vote-counting was not conducted for studies traced to these attributes. Vote-counting was only done on studies traced to reliability and maintainability.

Significant levels reported in the studies are shown by denoting significant at 0.01 with either "++" (if the relationship is positive), "– –" (if the relationship is negative), and those significant at 0.05 are denoted by "+" (if the relationship is positive), "–" (if the relationship is negative). In cases where the results are not statistically significant this is denoted by "0". We also show, under the column heading "Unclear", the number of studies that do not provide any clear results due to issues in study design, execution, or reporting particularly pertaining to the statistical significance levels (e.g., missing p-values and/or statistical significance levels).

7.3.1 Reliability

Given the number of the primary studies traced to reliability, there are too many measures extracted from these studies. We analyzed measures that were investigated in the most datasets, and across the most number of papers. Thus we analyzed measured that were investigated under differing contexts depicted in Table 9. To help us perform the analysis we identified measures that had been investigated across 5 different papers and evaluated on at least 10 datasets. The vote-counting results are tabulated in Table 14 and they are sorted by the "Significant" column.

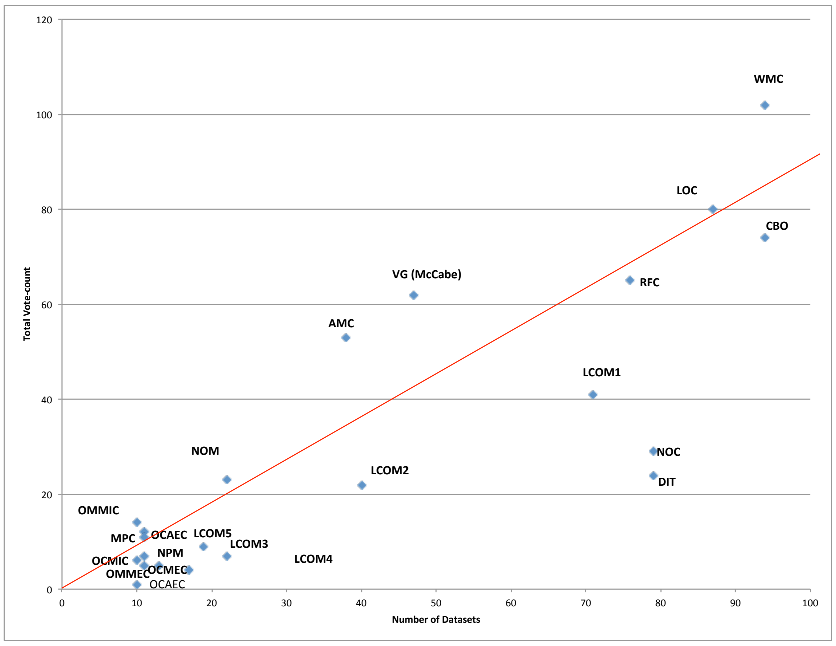
Some measures show contradicting results, i.e., both positive and negative outcomes. The complexity measure WMC, the size measure LOC, and the cohesion measure LCOM5 show a majority of positive outcomes, as well as some negative outcomes. Given that there is a large number of significant positive relationship outcomes, the negative results for WMC, LCOM5 and LOC could be considered as outliers. On the other hand, the size measure NPM, the inheritance measure NOC, the coupling measure OCAEC, and the three cohesion measures LCOM3, TCC and LCOM4, also show contradictory results in terms of the direction of the relationship with reliability and they also have a large number of instances in which they are not significant. Thus, from Table 14, it is difficult to conclude on the link as well as the direction of the relationship between the measures (NPM, NOC, OCAEC, LCOM3, TCC and LCOM4) and reliability.

The only two inheritance measures in Table 14 show poor relationship with reliability. They have been evaluated on a large number of datasets but there are more not significant outcomes than those that they are significant outcomes. Coupling and cohesion measures seem to perform better than inheritance measures. Though cohesion and coupling measures do have measures that show good link with reliability and have a large number of significant outcomes. But there are also some cohesion and coupling measures that are not significant in a large number of outcomes. In general, we can observe from Table 14 that complexity measures perform better than measures of other properties. Each complexity measure is significant in more than half of the outcomes. Using 50% as the cutoff point for the vote-counting results, there is a potential link between the following measures and reliability: OMMIC, VG (McCabe), OCAIC, WMC (or WMC-McCabe), AMC, NOM, MPC, LCO, RFC, CBO, NPM, LCOM3 and LCOM2.

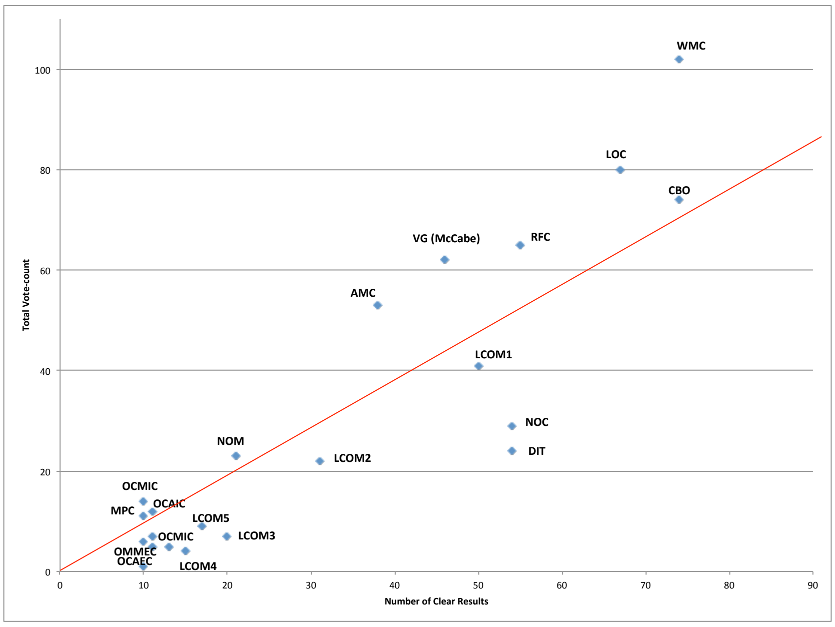
Figure 6[3] shows two figures that denote the strength of evidence for each measure using the vote counting results in Table 14. Figure 6(a) shows the distribution of all the vote-counting results found in the reliability studies. This includes unclear results as well as clear results, i.e., positive, negative and non-significant results. Figure 6(b) shows the distribution of the clear results only and shows measures that have a much stronger evidence and/or have much better consistency in terms of their relationship with reliability. The line in both figures is drawn to indicate measures that were investigated on at least 10 datasets and had at least two-thirds positive and/or one-third negative results. The purpose is to identify measures that have been investigated in a large number of datasets and that also show a better link with reliability attributes than the other measures,

[3] Total vote-counting on the y-axis is the sum of the number of "++", "– –", "+" results "–", where "++" and "– –" are both multiplied by two and each "+", "–" and "Unclear" result is one point.

i.e., measures that appear above the line. The measures that appear above the line in both figures are complexity, size and coupling measures; none of the cohesion and inheritance measures are above the line.



(a) All Vote-counting Results



(b) Vote-counting of Only Clear Results

Fig. 6: Vote-counting from Reliability Studies: Strength of Evidence

Table 14: Vote-Counting: Reliability Studies

Internal Properties	**Measure**	**Studies**	**Num. Of Datasets**	**Sig. Positive Relationship**		**No Sig.**	**Sig. Negative Relationship**		**Unclear**	**Signifi-cant**	**Not Sign.**
				"++"	"+"	**0**	"–"	"– –"			
Coupling	OMMIC	S3, S4, S26, S27, S28	10	4	6	0	0	0	0	100%	0%
Complexity	VG (Mc-Cabe)	S46, S52, S64, S67, S77, S78, S98, S99	33	18	14	0	0	0	1	91%	6%
Coupling	OCAIC	S3, S4, S25, S26, S27, S28	11	2	8	1	0	0	0	91%	9%
Cohesion	LCOM5	S2, S5, S7, S8, S17, S27, S63, S77	19	0	12	2	3	0	2	79%	11%
Complexity	WMC/ WMC-McCabe	S3, S4,S21, S22, S30, S32, S37, S41, S46, S48, S51, S52, S64, S69, S71, S77, S78, S81, S84, S87, S88, S94, S95, S96, S98,	83	30	32	1	1	0	19	78%	1%
Complexity	AMC	S64, S67, S77, S78, S98	38	18	17	3	0	0	0	75%	25%
Complexity	NOM	S4, S27, S38, S46, S54, S69, S72, S92,	22	7	9	5	0	0	1	73%	23%
Coupling	MPC	S4, S27, S28, S52, S74,	11	3	5	2	0	0	1	73%	18%
Size	LOC	S3, S4, S22, S30, S32, S38, S46, S48, S51, S52, S54, S63, S64, S67, S71, S77, S78, S86, S87, S88, S92, S94, S95, S96, S98, S99	87	23	35	8	1	0	20	68%	9%
Complexity	RFC	S3, S4, S21, S22, S28, S30, S32, S37, S41, S43, S46, S48, S51, S52, S54, S64, S69, S71, S77, S78, S81, S84, S86, S87, S88, S92, S94, S95, S96	76	15	35	5	0	0	21	66%	7%
Coupling	CBO	S3, S4, S18, S21, S22, S28, S30, S32, S41, S46, S48, S51, S52, S54, S64, S69, S71, S77, S78, S81, S86, S87, S88, S92, S94, S95,S96,	94	14	46	3	0	0	20	64%	3%
Size	NPM	S4, S27, S64, S77, S78,	13	1	5	5	2	0	0	62%	38%

Continued on next page

Table 14 – *Continued from previous page*

Internal Properties	**Measure**	**Studies**	**Num. Of Datasets**	**Sig. Positive Relationship**		**No Sig.**	**Sig. Negative Relationship**		**Unclear**	**Signifi-cant**	**Not Sign.**
				"++"	"+"	**0**	"–"	"– –"			
Cohesion	LCOM3	S2, S4, S5, S7, S8, S9, S10, S17, S27, S63, S64, S78	22	0	10	8	1	1	2	55%	36%
Cohesion	LCOM2	S2, S3, S4, S5, S7, S8, S9, S10, S27, S28, S41, S43, S48, S63, S77, S78, S84	40	1	20	10	0	0	9	53%	25%
Cohesion	LCOM1	S2, S3, S4, S5, S6, S7, S8, S9, S27, S28, S43, S63	71	8	25	17	0	0	21	46%	24%
Coupling	OCMIC	S3, S4, S25, S26, S27, S28	11	2	3	6	0	0	0	45%	55%
Coupling	OCAEC	S3, S4, S25, S27, S28	10	0	3	6	0	1	0	40%	60%
Coupling	OMMEC	S3, S4, S26, S27, S28	10	2	2	6	0	0	0	40%	60%
Cohesion	TCC	S2, S4, S5, S7, S8, S9, S27, S28, S43, S63	21	0	1	11	7	0	2	38%	52%
Coupling	OCMEC	S3, S4, S25, S26, S27, S28	11	1	3	7	0	0	0	36%	64%
Cohesion	LCOM4	S5, S7, S10, S8	17	0	5	9	1	0	2	35%	53%
Inheritance	NOC	S3, S4, S21, S22, S25, S27, S28, S32, S41, S46, S48, S51, S54, S64, S69, S71, S72, S74, S77, S78, S81, S84, S86, S87, S88, S92, S95, S96,	79	8	14	31	1	0	25	29%	39%
Inheritance	DIT	S32, S41, S48, S64, S77, S78, S84 S88, S51, S21, S37, S92, S74, S69, S81, S71, S4, S28, S25, S96, S95, S72, S87, S46, S22, S86, S54, S27, S3	79	4	16	34	0	0	25	25%	43%

7.3.2 Maintainability

The number of studies traced to maintainability were significantly less than those for reliability. Most of the measures extracted from maintainability were only evaluated in one study. The vote-counting results tabulated in Table 15 only shows results for measures that appeared in more than one primary study, and were significant in at least one dataset. Data in the table is sorted by the “Significant” column.

It can be observed from Table 15 that the inheritance measures DIT and NOC have been evaluated on many datasets, but they show the weakest link with maintainability. This is similar to the observation made for reliability for these two measures. With significant outcomes in half of the datasets and non-significant outcomes in the other half, the inheritance measure NMO also shows a weak link with maintainability.

The size measure NC, has been investigated on too few datasets to determine its strength of relationship with maintainability. However, another size measure LOC shows good link with maintainability. Complexity, coupling and cohesion measures that appear in Table 15 also seem to have a good link with maintainability.

Overall, coupling, complexity and size measures seem to have a better relationship with maintainability than inheritance measures. There is a potential link between the following measures and maintainability (i.e., they are above 50% used as the cutoff point for the vote-counting results): ICH, CAMC, NOM, LCOM5, LOC, DAC, MPC, WMC/WMC-McCabe, RFC, CBO, LCOM1, TCC and LCOM2.

Figure 7 shows the distribution of the clear results only, i.e., positive, negative and non-significant results for the measures from maintainability studies. There are insufficient numbers of studies on maintainability to draw conclusions. Nevertheless, Figure 7 shows that there is a potential link between maintainability, and measures that quantify complexity and cohesion properties.

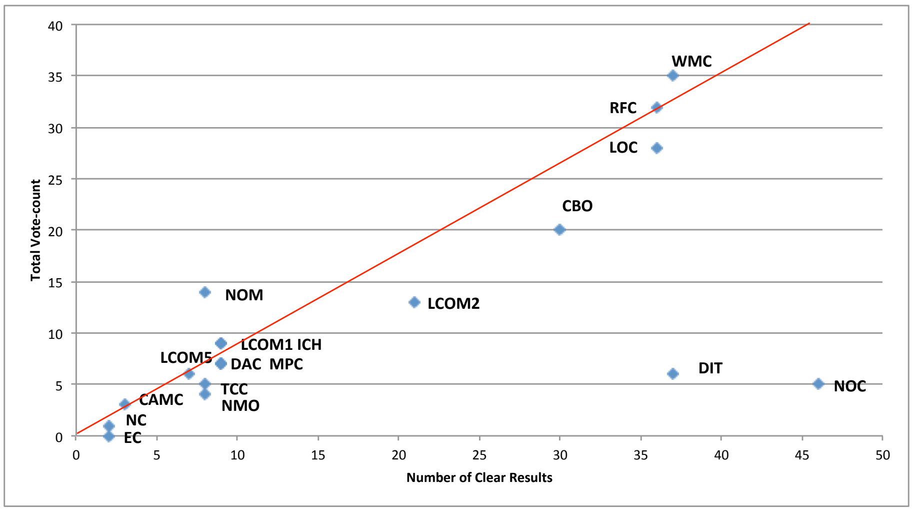


Fig. 7: Vote-counting from Maintainability Studies: Strength of Evidence

Table 15: Vote-Counting: Maintainability Studies

Internal Properties	**Measure**	**Studies**	**Num. Of Datasets**	**Sig. Positive Relationship**		**No Sig.**	**Sig. Negative Relationship**		**Unclear**	**Signifi-cant**	**Not Sign.**
				"++"	"+"	**0**	"–"	"– –"			
Cohesion	ICH	S50, S90, S97	9	0	9	0	0	0	0	100%	0%
Cohesion	CAMC	S50, S97	3	0	3	0	0	0	0	100%	0%
Complexity	NOM	S29, S44	8	7	0	0	0	0	1	88%	0%
Cohesion	LCOM5	S16, S50, S97,	7	0	6	1	0	0	0	86%	14%
Size	LOC	S17, S42, S45, S50, S76, S90, S97	36	0	28	0	0	0	8	78%	0%
Complexity	DAC	S73, S90, S97	9	0	7	1	0	0	1	78%	11%
Coupling	MPC	S73, S90, S97	9	0	7	1	0	0	0	78%	11%
Complexity	WMC/ WMC-McCabe	S29, S33, S42, S45, S90, S97, S68, S83	37	7	21	1	0	0	8	76%	3%
Complexity	RFC	S29, S42, S45, S83, S90, S97,	36	6	20	2	0	0	8	72%	6%
Coupling	CBO	S42, S45, S73, S76, S83, S90, S97	30	0	20	1	0	0	9	67%	3%
Cohesion	LCOM1	S29, S50, S97	9	3	3	3	0	0	0	67%	33%
Cohesion	TCC	S50, S90, S97	8	0	5	3	0	0	0	63%	38%
Cohesion	LCOM2	S16, S17, S42, S45, S76, S90, S97	21	0	13	0	0	0	8	62%	0%
Inheritance	NMO	S90, S97,	8	0	4	4	0	0	0	50%	50%
Size	NC	S29, S44	2	0	1	0	0	0	1	50%	0%
Inheritance	DIT	S29, S42, S45, S83, S90, S97,	37	0	6	23	0	0	8	16%	62%
Inheritance	NOC	S29, S42, S45, S50, S76, S83, S90, S97	46	0	5	33	0	0	8	11%	72%
Coupling	EC	S40, S14	2	0	0	1	0	0	1	0%	50%

7.3.3 Summary of Vote-Counting

Table 16 depicts the potential relationship between measures and external quality attributes. The table is sorted by the internal properties measured. Measures included in the table are those that show a significant relationship in one direction in over 50% of the datasets in Table 14 and Table 15. Measures linked to effectiveness and functionality are excluded from the table because vote-counting was not done for the measures. The symbol "+" denotes a potential positive relationship between the measure and the external quality attribute(s). It does not necessarily mean that measures that are not included in the table or those in the table but do not have a corresponding "+" symbol do not have a link with reliability or maintainability. We did not find enough supporting evidence to link the measure with reliability and/or maintainability.

Table 16: Relation between Measures and External Quality Attributes

Internal Properties	**Measures**	**Reliability**	**Maintainability**
Cohesion	ICH		+
Cohesion	CAMC		+
Cohesion	LCOM1		+
Cohesion	LCOM2	+	+
Cohesion	LCOM5	+	+
Cohesion	TCC		+
Complexity	AMC	+	
Complexity	DAC		+
Complexity	VG (McCabe)	+	
Complexity	NOM	+	+
Complexity	RFC	+	+
Complexity	WMC/ WMC-McCabe	+	+
Coupling	CBO	+	+
Coupling	MPC	+	+
Coupling	OCAIC	+	
Coupling	OMMIC	+	
Size	LOC	+	+

Table 16 shows that the following measures have a potential positive relationship with both reliability and maintainability: complexity measures NOM, WMC and RFC, coupling measures CBO and MPC, cohesion measures LCOM2 and LCOM5, and size measure LOC.

Inheritance measures seem to have the weakest relation with reliability and maintainability. The inheritance measures NOC and DIT have been extensively investigated but they are not significant in a much larger number of datasets than those in which they are significant for both reliability and maintainability. For example, for reliability studies NOC is significant in 29% and not significant in 39%, and DIT is significant in 25% and not significant in 43% of the datasets. The two measures also have a large number of non-significant results for maintainability. The other inheritance measure that appears in Table 15, NMO, is also not significant in 50% of the datasets. Though the measures of other properties, e.g., cohesion, complexity, coupling and size, show contradictory direction of relation on a few outcomes for reliability, their relationship with reliability and maintainability is much more consistent than for inheritance measures.

However, given that inheritance measures, as well as others that are not significant in over 50% datasets, do have instances in which they are significant for reliability and/or maintainability, this could indicate that the usefulness of some measures is context dependent.

8 Discussion

The overall goal of our SLR was to identify useful object-oriented measures for quality assessment. The results of our SLR show that an overwhelming number of empirical studies can be traced to reliability and maintainability. Most of the studies are on fault-proneness (a reliability proxy). This could be linked to that system defects, faults, and or failures are often more readily available than data for, e.g., development or maintenance effort. Apart from reliability and maintainability, there was only a single study that could be traced to other attributes (effectiveness and functionality). Thus a meaningful analysis could only be performed for maintainability and reliability studies.

The results of our SLR also shows that all or a subset of the measures can be used to build effective prediction models for reliability and maintainability. However, our vote-counting results suggest that measures for complexity, cohesion, coupling and size are more reliable when investigating reliability and maintainability than inheritance measures. Similar findings have been reported in other studies (Saxena and Saini, 2011; Riaz et al., 2009).

Results from our systematic review suggest that inheritance measures have a weak link with reliability and maintainability across studies, particularly the two inheritance measures DIT and NOC. This is consistent with findings from Briand and Wüst (2002); Saxena and Saini (2011); Radjenović et al. (2013). These two measures are evaluated extensively in reliability and maintainability studies. For maintainability a majority of the studies (62%–72%) show that there is no significant relationship with DIT and NOC. For reliability there are a many unclear results, but in outcomes showing clear results the measures are not significant in a majority of the datasets (57%–63%).

Our results further corroborate other findings in fault-proneness studies. Two decades after their initial publication, measures from the C&K measurement suite are still the most used or investigated object-oriented measures (i.e., 79% of the primary studies). Their popularity is also noted in other studies (Kitchenham, 2010; Malhotra and Jain, 2011; Saxena and Saini, 2011; Radjenović et al., 2013). The other measurement suites, L&K, MOOD, QMOOD and those from (Briand et al., 1997) appear in 29% (29 out of 99) of the primary studies. Regression and machine learning methods were the most commonly used methods for building prediction models across studies. Similar findings have been reported in reviews for fault-proneness studies and maintainability studies Riaz et al. (2009); Catal and Diri (2009); Malhotra and Jain (2011); Radjenović et al. (2013). However, the predictive ability of some of the models was not reported, thus making it difficult to assess the usefulness of the models. In particular, the evaluation and/or validation results for a majority of the models from maintainability was missing. This information would help practitioners or researchers to understand the usefulness of certain models in a particular study.

Etzkorn et al. (1997) report on how the different approaches for computing LCOMn measures can produce different results. Our SLR results corroborates the findings of Etzkorn et al. (1997). In particular, the vote-counting results show that variations of LCOMn show different strengths of relation across reliability and maintainability. Thus, it is possible that the inconsistent outcomes for some measures could be linked to the difference in methods or tools used for extracting the measures across studies (Genero et al., 2005; Kitchenham, 2010; Lincke et al., 2008). A similar argument could be made for inheritance measures. This is because there are some outcomes for inheritance measures that show significant results across studies for both reliability and maintainability, though fewer than not significant outcomes. The few significant outcomes for inheritance measures could potentially mean that inheritance measures are more context depend than measures of other properties. Confounding factors, such as, violations on the use of inheritance rules, programming style, and the programming language are some aspects that can contribute to varying results for inheritance measures across studies (Harrison and Counsell, 1998). Hence there is a need to carry out further investigations to improve understanding on the extent of the effect of confounding factors on inheritance measures.

Finally, results of our SLR suggest that during quality assessment initiatives it may be more effective to spend more time collecting measures that quantify complexity, cohesion, coupling and size properties than those for inheritance properties. However, measures for inheritance properties should not be disregarded entirely because they may be necessary in certain quality assessment contexts. Thus, an attempt should be made to understand which, and how, object-oriented constructs have been utilized in a given system. This information can be obtained from those directly involved with developing the system. Caution should also be taken when only using (object-oriented) source code measures for quality prediction, because factors that impact quality differ from system to system. For example, team structure and team strategy can vary across development settings, and studies show that such organizational characteristics have an impact on quality (Karus and Dumas, 2012; Ramasubbu et al., 2012). Therefore, practitioners should take such contextual information and confounding factors into consideration and not use object-oriented measures blindly.

9 Conclusions and Future Work

This paper reports on a systematic literature review conducted to identify measures that are obtainable from source code of object-oriented programs and to investigate their links with quality as reported in empirical studies. Primary studies are traced to five external quality attributes: reliability, maintainability, effectiveness and functionality.

Results from our systematic literature review suggest that there is an overwhelming number of studies that can be traced to reliability compared to other external quality attributes. More investigations on studies on other quality attributes would be helpful to understand the link between internal properties of object-oriented programs and various aspects of quality. 70% of the primary studies were traced to reliability and 31% of the studies were traced to maintainability; whilst studies traced to (or considered as surrogates for) effectiveness and functionality constituted of 1% of the primary studies.

According to the vote-counting results, measures for complexity, cohesion, coupling and size show better consistency on their relationship with reliability and maintainability attributes across the primary studies than inheritance. Measures that quantify inheritance properties show poor links to reliability and maintainability. Though inheritance measures are used in some of the prediction models found in reliability and maintainability studies, there is evidence that models that do not include these measures are useful as well. Thus the usefulness of inheritance measures maybe more context dependent than measures of other properties.

In summary, a meaningful analysis could only be performed for reliability and maintainability, because there were too few studies traced to effectiveness and functionality. Measures that quantify complexity, cohesion, coupling and size can be useful indicators for reliability and maintainability during quality assessment activities for object-oriented systems. Using regression and machine learning methods, a combination of all or a subset of these measures can be used to predict reliability and maintainability related concerns. Measures that quantify other properties, such as inheritance and cohesion, show poor links to reliability and maintainability.

For future work we urge researchers to diversify the types of external quality attributes that they investigate. As highlighted in the previous paragraph, there are very few studies that can be traced to external quality attributes other than reliability and maintainability. Quality is a multifaceted concept, and practitioners would be interested in understanding the link between object-oriented measures and a wider range of external quality attributes. Another possible future work would be to investigate the consistency of the relation between measures and quality attributes across certain types or sizes of datasets.

It is also important to enable other researchers to validate ones work and findings. For this reason, Catal and Diri (2009) emphasized the need to use publicly available data to enable the research community to validate and compare each others' findings. In our review, we found that less than half of the primary studies used publicly available datasets. In similar vein as Catal and Diri (2009), we would therefore like to urge researchers to use publicly available datasets in their empirical studies. We should point out that we are not proposing to neglect private datasets. But rather urging the use of publicly available datasets, which can be done together in the same study with private datasets. This would make it easier to compare findings from different settings.

Model validation results were seldom found in the primary studies. Lack of this information makes it difficult to determine the usefulness of the model or how successful the prediction model was/is in a given study. Riaz et al. (2009) reported a similar finding and concern for prediction models found in maintainability studies.

Measures such as the C&K measures have been extensively investigated. More studies using these measures may not add much to the body of knowledge. Perhaps, a meta-analysis could be performed specifically on the usefulness of the C&K measures. To enrich the body of knowledge future studies should make an effort to investigate other sets of measurement suites. We see a need to empirically investigate other measures, since there are many source code measures that have not been sufficiently investigated empirically.

There are some measures that show a significant relationship with external quality attributes in one direction in some outcomes but are not significant in a large number of outcomes. This could be an indicator that some measures might be suitable in certain contexts only, e.g., certain characteristics in a system. This may need further investigation.

Acknowledgements This work was funded by the Swedish Knowledge Foundation in Sweden under the grants 2009/0249 and 2010/0311, and Ericsson Software Research. We thank Indira Nurdiani, a PhD Student at Software Engineering Research Lab (SERL), for valuable comments on the paper.

References

Abreu, F. and Melo, W. (1996). Evaluating the impact of object-oriented design on software quality. In *Proceedings of the 3rd international software metrics symposium*, pages 90–99.

Abreu, F. B. and Carapuça, R. (1994). Object-oriented software engineering: Measuring and controlling the development process. In *Proceedings of the 4th international conference on software quality*, volume 186.

Abreu, F. B. E., Goulão, M., Esteves, R., and Abreu, O. B. E. (1995). Toward the design quality evaluation of object-oriented software systems. In *Proceedings of the 5th international conference on software quality.*, pages 44–57.

Abubakar, A., AlGhamdi, J., and Ahmed, M. (2006). Can cohesion predict fault density? In *Proceedings of the 30th IEEE international conference on computer systems and applications*, pages 890–893.

Aggarwal, K., Singh, Y., Kaur, A., and Malhotra, R. (2007). Investigating effect of design metrics on fault proneness in object-oriented systems. *The Journal of object technology*, 6(10):127–141.

Aggarwal, K., Singh, Y., Kaur, A., and Malhotra, R. (2009). Empirical analysis for investigating the effect of object-oriented metrics on fault proneness: A replicated case study. *Software process: Improvement and practice*, 14(1):39–62.

Ajrnal Chaumun, M., Kabaili, H., Keller, R., and Lustman, F. (1999). A change impact model for changeability assessment in object-oriented software systems. In *Proceedings of the Third European Conference on Software Maintenance and Reengineering, 1999*, pages 130–138.

Al Dallal, J. (2011a). Improving the applicability of object-oriented class cohesion metrics. *Information and Software Technology*, 53(9):914 – 928.

Al Dallal, J. (2011b). Transitive-based object-oriented lack-of-cohesion metric. *Procedia computer science*, 3(0):1581 – 1587.

Al Dallal, J. (2012a). Fault prediction and the discriminative powers of connectivity-based object-oriented class cohesion metrics. *Information and software technology*, 54(4):396–416.

Al Dallal, J. (2012b). The impact of accounting for special methods in the measurement of object-oriented class cohesion on refactoring and fault prediction activities. *Journal of Systems and Software*, 85(5):1042 – 1057.

Al Dallal, J. and Briand, L. (2010). An object-oriented high-level design-based class cohesion metric. *Information and software technology*, 52(12):1346–1361.

Al Dallal, J. and Briand, L. C. (2012). A precise method-method interaction-based cohesion metric for object-oriented classes. *ACM Transaction of Software Engineering Methodology*, 21(2):8:1–8:34.

Alshayeb, M. and Li, W. (2003). An empirical validation of object-oriented metrics in two different iterative software processes. *IEEE Transactions on software engineering*, 29:1043–1049.

Aman, H., Mochiduki, N., and Yamada, H. (2006). A model for detecting cost-prone classes based on mahalanobis-taguchi method. *IEICE Transactions on information and systems*, E89-D:1347–1358.

Arisholm, E. (2006). Empirical assessment of the impact of structural properties on the changeability of object-oriented software. *Information and software technology*, 48(11):1046–1055.

Arisholm, E. and Sjøberg, D. (2000). Towards a framework for empirical assessment of changeability decay. *Journal of systems and software*, 53(1):3–14.

Babich, D., Clarke, P. J., Power, J. F., and Kibria, B. M. G. (2011). Using a class abstraction technique to predict faults in OO classes: A case study through six releases of the eclipse JDT. In *Proceedings of the 2011 ACM Symposium on Applied Computing*, pages 1419–1424, New York, NY, USA. ACM.

Badri, L., Badri, M., and Toure, F. (2011). An empirical analysis of lack of cohesion metrics for predicting testability of classes. *International Journal of Software Engineering and Its Applications*, 5(2):69–86.

Badri, M. and Toure, F. (2012). Evaluating the effect of control flow on the unit testing effort of classes: an empirical analysis. *Adv. Soft. Eng.*, 2012:5:5–5:17.

Bakar, N. S. A. A. (2011). Empirical analysis of object-oriented coupling and cohesion measures in determining the quality of open source systems. In *Proceedings of the IASTED International Conference on Software Engineering and Applications, SEA 2011.*

Bandi, R. K., Vaishnavi, V. K., and Turk, D. E. (2003). Predicting maintenance performance using object-oriented design complexity metrics. *IEEE Transactions on software engineering*, 29(1):77–87.

Bansiya, J. and Davis, C. G. (2002). A hierarchical model for object-oriented design quality assessment. *IEEE Transactions on software engineering*, 28(1):4–17.

Basili, V. R., Briand, L. C., and Melo, W. L. (1996). A validation of object-oriented design metrics as quality indicators. *IEEE Transactions on software engineering*, 22(10):751–761.

Benlarbi, S., El Emam, K., Goel, N., and Rai, S. (2000). Thresholds for object-oriented measures. In *Proceedings of the 11th international symposium on software reliability engineering*, pages 24–38.

Benlarbi, S. and Melo, W. (1999). Polymorphism measures for early risk prediction. In *Proceedings of the 1999 international conference on software engineering*, pages 334–344.

Bocco, M., Moody, D., and Piattini, M. (2005). Assessing the capability of internal metrics as early indicators of maintenance effort through experimentation. *Journal of software maintenance and evolution*, 17(3):225–246.

Brereton, P., Kitchenham, B. A., Budgen, D., Turner, M., and Khalil, M. (2007). Lessons from applying the systematic literature review process within the software engineering domain. *Journal of systems and software*, 80(4):571–583.

Briand, L., Devanbu, P., and Melo, W. (1997). An investigation into coupling measures for C++. In *Proceedings of the 19th international conference on software engineering*, pages 412–421.

Briand, L., Melo, W., and Wüst, J. (2002). Assessing the applicability of fault-proneness models across object-oriented software projects. *IEEE Transactions on software engineering*, 28(7):706–720.

Briand, L. and Wüst, J. (2002). Empirical studies of quality models in object-oriented systems. *Advances in computers*, pages 97–166.

Briand, L. C., Wüst, J., Daly, J. W., and Porter, D. V. (2000). Exploring the relationship between design measures and software quality in object-oriented systems. *Journal of systems and software*, 51(3):245–273.

Briand, L. C., Wüst, J., Ikonomovski, S. V., and Lounis, H. (1999). Investigating quality factors in object-oriented designs: An industrial case study. In *Proceedings of the 21st international conference on software engineering*, pages 345–354.

Bruntink, M. and van Deursen, A. (2006). An empirical study into class testability. *Journal of systems and software*, 79(9):1219–1232.

Cartwright, M. and Shepperd, M. (2000). Empirical investigation of an object-oriented software system. *IEEE Transactions on software engineering*, 26(8):786–796.

Catal, C. and Diri, B. (2009). A systematic review of software fault prediction studies. *Expert systems with applications*, 36(4):7346–7354.

Catal, C., Diri, B., and Ozumut, B. (2007). An artificial immune system approach for fault prediction in object-oriented software. In *Proceedings of the 2nd international conference on dependability of computer systems*, pages 238–245.

Chidamber, S. and Kemerer, C. (1991). Towards a metrics suite for object oriented design. *SIGPLAN notices*, 26(11):197–211.

Chidamber, S. R., Darcy, D. P., and Kemerer, C. F. (1998). Managerial use of metrics for object-oriented software: An exploratory analysis. *IEEE Transactions on software engineering*, 24(8):629–639.

Chidamber, S. R. and Kemerer, C. F. (1994). A metrics suite for object oriented design. *IEEE Transactions on software engineering*, 20(6):476–493.

Cruz, A. E. C. and Ochimizu, K. (2010). A UML approximation of three Chidamber-Kemerer metrics and their ability to predict faulty code across software projects. *IEICE Transactions on information and systems*, 93(11):3038–3050.

Dagpinar, M. and Jahnke, J. H. (2003). Predicting maintainability with object-oriented metrics – An empirical comparison. In *Proceedings of the 10th working conference on reverse engineering*, pages 155–164.

Dandashi, F. and Rine, D. (2002). A method for assessing the reusability of object-oriented code using a validated set of automated measurements. In *Proceedings* *of the 2002 ACM symposium on applied computing*, pages 997–1003.

Darcy, D., Kemerer, C., Slaughter, S., and Tomayko, J. (2005). The structural complexity of software: An experimental test. *IEEE Transactions on software engineering*, 31(11):982–994.

Díaz, J., Pérez, J., Alarcón, P. P., and Garbajosa, J. (2011). Agile product line engineering – A systematic literature review. *Software: Practice and experience*, 41(8):921–941.

Dick, S. and Sadia, A. (2006). Fuzzy clustering of open-source software quality data: A case study of Mozilla. In *Proceedings of the international joint conference on neural networks*, pages 4089–4096.

Dybå, T., Dingsøyr, T., and Hanssen, G. (2007). Applying systematic reviews to diverse study types : An experience report. In *Proceedings of the 1st international symposium on empirical software engineering and measurement*, pages 225–234.

El Emam, K., Benlarbi, S., Goel, N., Melo, W., Lounis, H., and Rai, S. (2002). The optimal class size for object-oriented software. *IEEE Transactions on software engineering*, 28(5):494–509.

Elish, M. O. (2010). Exploring the relationships between design metrics and package understandability: A case study. In *Proeceeding of the 18th IEEE international conference on program comprehension*, pages 144–147.

Elish, M. O., Al-Yafei, A. H., and Al-Mulhem, M. (2011). Empirical comparison of three metrics suites for fault prediction in packages of object-oriented systems: A case study of eclipse. *Advances in Engineering Software*, 42(10):852 – 859.

Elish, M. O. and Rine, D. (2006). Design structural stability metrics and post-release defect density: An empirical study. In *30th annual international conference of computer software and applications*, pages 1–8.

Eski, S. and Buzluca, F. (2011). An empirical study on object-oriented metrics and software evolution in order to reduce testing costs by predicting change-prone classes. In *Proceedings of the Fourth IEEE International Conference on Software Testing, Verification and Validation Workshops, 2011*, pages 566–571.

Etzkorn, L., Davis, C., and Li, W. (1997). A statistical comparison of various definitions of the LCOM metric. Technical Report TR-UAH-CS-1997-02.

Fenton, N. E. and Pfleeger, S. L. (1998). *Software metrics: A rigorous and practical approach.* PWS Publishing, Boston, USA, 2nd edition.

Fioravanti, F. and Nesi, P. (2001). A study on fault-proneness detection of object-oriented systems. In *Proceedings of the European conference on software maintenance and reengineering*, pages 121–130.

Genero, M., Piattini, M., and Calero, C. (2005). A survey of metrics for UML class diagrams. *Journal of object technology*, 4:59–92.

Genero, M., Piattini, M., and Jiménez, L. (2001). Empirical validation of class diagram complexity metrics. In *Proceedings of the 11th internatinal conference of the Chilean computer science society*, pages 95–104.

Giger, E., Pinzger, M., and Gall, H. (2012). Can we predict types of code changes? an empirical analysis. In *Proceedings of the Ninth IEEE Working Conference on Mining Software Repositories, 2012*, pages 217–226.

Goel, B. and Singh, Y. (2008). Empirical investigation of metrics for fault prediction on object-oriented software. *Studies in computational intelligence*, 131:255–265.

Guo, Y., Wuersch, M., Giger, E., and Gall, H. (2011). An empirical validation of the benefits of adhering to the law of demeter. In *Proceeding of the 18th Working Conference on Reverse Engineering, 2011*, pages 239–243.

Gupta, V. and Chhabra, J. K. (2009). Package coupling measurement in object-oriented software. *Journal of computer science and technology*, 24(2):273–283.

Gupta, V. and Chhabra, J. K. (2012). Package level cohesion measurement in object-oriented software. *Journal of the Brazilian Computer Society*, 18(3):251–266.

Gyimóthy, T., Ferenc, R., and Siket, I. (2005). Empirical validation of object-oriented metrics on open source software for fault prediction. *IEEE Transactions on software engineering*, 31(10):897–910.

Halstead, M. H. (1977). *Elements of software science (Operating and programming systems series)*. Elsevier Science, New York, USA.

Harrison, R. and Counsell, S. (1998). The role of inheritance in the maintainability of object-oriented systems. *Proceedings of European software control and metrics conference*, pages 449–457.

Harrison, R., Counsell, S., and Nithi, R. (1997). An overview of object-oriented design metrics. In *Proceedings of the 8th IEEE international workshop on software technology and engineering practice*, pages 230–235.

Henningsson, K. and Wohlin, C. (2005). Monitoring fault classification agreement in an industrial context. In *Proceedings of the 9th conference on empirical assessment in software engineering.*

Holschuh, T., Päuser, M., Herzig, K., Zimmermann, T., Premraj, R., and Zeller, A. (2009). Predicting defects in SAP Java code: An experience report. In *Proceedings of the 31st international conference on software engineering-companion volume*, pages 172–181.

Huang, P. and Zhu, J. (2009). A multi-instance model for software quality estimation in OO systems. In *Proceedings of the 5th international conference on natural computation*, pages 436–440.

ISO/IEC-25010 (2010). Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE) – System and software quality models. International organization for standardization.

ISO/IEC-9126 (2001). Software engineering – Product quality – Part 1: Quality model. International organization for standardization.

ISO/IEC/IEEE-24765 (2010). Systems and software engineering – Vocabulary. International organization for standardization.

Janes, A., Scotto, M., Pedrycz, W., Russo, B., Stefanovic, M., and Succi, G. (2006). Identification of defect-prone classes in telecommunication software systems using design metrics. *Information sciences*, 176(24):3711–3734.

Jia, H., Shu, F., Yang, Y., and Wang, Q. (2009). Predicting fault-prone modules: A comparative study. In *Software engineering approaches for offshore and outsourced development*, volume 35, pages 45–59. Springer Berlin Heidelberg.

Jin, C., Jin, S.-W., Ye, J.-M., and Zhang, Q.-G. (2009). Quality prediction model of object-oriented software system using computational intelligence. In *Proceedings of the 2nd international conference on power electronics and intelligent transportation system*, volume 2, pages 120–123.

Kamiya, T., Kusumoto, S., and Inoue, K. (1999). Prediction of fault-proneness at early phase in object-oriented development. In *Proceedings of the IEEE 2nd international symposium on object-oriented real-time distributed computing*, pages

253–258.

Kanellopoulos, Y., Antonellis, P., Antoniou, D., Makris, C., Theodoridis, E., Tjortjis, C., and Tsirakis, N. (2010). Code quality evaluation methodology using the ISO/IEC 9126 standard. *International journal of software engineering and applications*, 1(3):17–36.

Kanmani, S., Rhymend Uthariaraj, V., Nakkeeran, R., and Inbavani, P. (2004). Object oriented software fault prediction using adaptive neuro fuzzy inference system. *WSEAS Transactions on information science and applications*, 1(5):1142–1145.

Kanmani, S., Uthariaraj, V. R., Sankaranarayanan, V., and Thambidurai, P. (2007). Object-oriented software fault prediction using neural networks. *Information and software technology*, 49(5):483 – 492.

Karus, S. and Dumas, M. (2012). Code churn estimation using organisational and code metrics: An experimental comparison. *Information and Software Technology*, 54(2):203 – 211.

Kitchenham, B. (2010). Whats up with software metrics? A preliminary mapping study. *Journal of systems and software*, 83(1):37 – 51.

Kitchenham, B. A. and Charters, S. (2007). Guidelines for performing systematic literature reviews in software engineering. Technical Report EBSE-2007-01, Keele University.

Landis, J. and Koch, G. (1977). The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159–174.

Lavazza, L., Morasca, S., Taibi, D., and Tosi, D. (2012). An empirical investigation of perceived reliability of open source java programs. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, pages 1109–1114, New York, NY, USA. ACM.

Li, W. and Shatnawi, R. (2007). An empirical study of the bad smells and class error probability in the post-release object-oriented system evolution. *Journal of systems and software*, 80(7):1120 – 1128.

Lincke, R. d., Lundberg, J., and Löwe, W. (2008). Comparing software metrics tools. In *Proceedings of the international symposium on software testing and analysis*, pages 131–142.

Liu, Y., Poshyvanyk, D., Ferenc, R., Gyimothy, T., and Chrisochoides, N. (2009). Modeling class cohesion as mixtures of latent topics. In *Proceedings of the 2009 IEEE international conference on software maintenance*, pages 233–242.

Lorenz, M. and Kidd, J. (1994). *Object-oriented software metrics: A practical guide*. Prentice-Hall, New Jersey, USA.

Malhotra, R. and Jain, A. (2011). Software fault prediction for object oriented systems: A literature review. *SIGSOFT Software engineering notes*, 36(5):1–6.

Malhotra, R. and Jain, A. (2012). Fault prediction using statistical and machine learning methods for improving software quality. *Journal of Information Processing Systems*, 8(2):241–262.

Marinescu, R. and Marinescu, C. (2011). Are the clients of flawed classes (also) defect prone? In *Proceedings of the 11th IEEE International Working Conference on Source Code Analysis and Manipulation, 2011*, pages 65–74.

McCabe, T. J. (1976). A complexity measure. *IEEE Transactions on software engineering*, (4):308–320.

Nair, T. G. and Selvarani, R. (2012). Defect proneness estimation and feedback approach for software design quality improvement. *Information and Software*

Technology, 54(3):274 – 285.

Nguyen, V., Boehm, B., and Danphitsanuphan, P. (2011). A controlled experiment in assessing and estimating software maintenance tasks. *Information and software technology*, 53(6):682 – 691.

Olague, H., Etzkorn, L., Gholston, S., and Quattlebaum, S. (2007). Empirical validation of three software metrics suites to predict fault-proneness of object-oriented classes developed using highly iterative or agile software development processes. *IEEE Transactions on software engineering.*, 33(6):402–419.

Olague, H. M., Etzkorn, L. H., and Cox, G. W. (2006). An entropy-based approach to assessing object-oriented software maintainability and degradation – A method and case study. In *Proceedings of the international conference on software engineering research and practice*, pages 442–452.

Olague, H. M., Etzkorn, L. H., Messimer, S. L., and Delugach, H. S. (2008). An empirical validation of object-oriented class complexity metrics and their ability to predict error-prone classes in highly iterative, or agile, software: A case study. *Journal of software maintenance and evolution: Research and practice*, 20(3):171–197.

Olbrich, S., Cruzes, D. S., Basili, V., and Zazworka, N. (2009). The evolution and impact of code smells: A case study of two open source systems. In *Proceedings of the 2009 3rd international symposium on empirical software engineering and measurement*, pages 390–400.

Pai, G. and Bechta Dugan, J. (2007). Empirical analysis of software fault content and fault proneness using bayesian methods. *IEEE Transactions on software engineering.*, 33(10):675–686.

Pickard, L. M., Kitchenham, B. A., and Jones, P. W. (1998). Combining empirical results in software engineering. *Information and software technology*, 40(14):811 – 821.

Poshyvanyk, D., Marcus, A., Ferenc, R., and Gyimóthy, T. (2009). Using information retrieval based coupling measures for impact analysis. *Empirical software engineering*, 14(1):5–32.

Pritchett, I. and W., W. (2001). An object-oriented metrics suite for Ada 95. In *Proceedings of the 2001 annual ACM SIGAda international conference on Ada*, pages 117–126.

Quah, J. T. and Thwin, M. M. (2002). Prediction of software readiness using neural network. In *Proceedings of 1st international conference on information technology & applications*, pages 307–312.

Radjenović, D., Heričko, M., Torkar, R., and Živkovič, A. (2013). Software fault prediction metrics: A systematic literature review. *Information and software technology*, 55:1397–1418.

Ramasubbu, N., Kemerer, C. F., and Hong, J. (2012). Structural complexity and programmer team strategy: An experimental test. *IEEE Transactions on Software Engineering*, 38(5):1054–1068.

Rathore, S. and Gupta, A. (2012a). Investigating object-oriented design metrics to predict fault-proneness of software modules. In *Proceedings of Sixth CSI International Conference on Software Engineering, 2012*, pages 1–10.

Rathore, S. and Gupta, A. (2012b). Validating the effectiveness of object-oriented metrics over multiple releases for predicting fault proneness. In *Proceedings of the 19th Asia-Pacific Software Engineering Conference (APSEC), 2012*, volume 1, pages 350–355.

Revelle, M., Gethers, M., and Poshyvanyk, D. (2011). Using structural and textual information to capture feature coupling in object-oriented software. *Empirical software engineering.*, 16(6):773–811.

Reyes, L. and Carver, D. (1998). Predicting object reuse using metrics. In *Proceedings of the 10th international conference on software engineering and knowledge engineering*, pages 156–159.

Riaz, M., Mendes, E., and Tempero, E. (2009). A systematic review of software maintainability prediction and metrics. In *Proceedings of the 3rd international symposium on empirical software engineering and measurement*, pages 367–377.

Robson, C. (2011). *Real world research.* John Wiley & Sons, West Sussex, UK, 2nd edition.

Rosenberg, L. H. and Hyatt, L. E. (1997). Software quality metrics for object-oriented environments. *Crosstalk Journal, April.*

Saxena, P. and Saini, M. (2011). Empirical studies to predict fault proneness: A review. *International journal of computer applications*, 22(8):41–45.

Shatnawi, R. (2010). A quantitative investigation of the acceptable risk levels of object-oriented metrics in open-source systems. *IEEE Transactions on software engineering.*, 36(2):216–225.

Shatnawi, R. and Li, W. (2008). The effectiveness of software metrics in identifying error-prone classes in post-release software evolution process. *Journal of systems and software*, 81(11):1868 – 1882.

Shatnawi, R., Li, W., Swain, J., and Newman, T. (2010). Finding software metrics threshold values using ROC curves. *Journal of software maintenance and evolution: Research and practice*, 22(1):1–16.

Singh, P. and Verma, S. (2012). Empirical investigation of fault prediction capability of object oriented metrics of open source software. In *Proceeding of the International Joint Conference on Computer Science and Software Engineering, 2012*, pages 323–327.

Singh, Y., Kaur, A., and Malhotra, R. (2007). Application of logistic regression and artificial neural network for predicting software quality models. In *Software engineering research and practice*, pages 664–670.

Singh, Y., Kaur, A., and Malhotra, R. (2009a). Comparative analysis of regression and machine learning methods for predicting fault proneness models. *International journal of computer applications in technology*, 35(2):183–193.

Singh, Y., Kaur, A., and Malhotra, R. (2009b). Software fault proneness prediction using support vector machines. In *Proceedings of the world congress on engineering*, volume 1, pages 1–3.

Singh, Y., Kaur, A., and Malhotra, R. (2010). Empirical validation of object-oriented metrics for predicting fault proneness models. *Software quality journal*, 18(1):3–35.

Singh, Y., Kaur, A., and Malhotra, R. (2011). Comparative analysis of J48 with statistical and machine learning methods in predicting fault-prone classes using object-oriented systems. *Journal of statistics and management systems*, 14(3):595–616.

Singh, Y. and Saha, A. (2012). Prediction of testability using the design metrics for object-oriented software. *International journal of computer applications in technology*, 44(1):12–22.

Subramanyam, R. and Krishnan, M. (2003). Empirical analysis of CK metrics for object-oriented design complexity: Implications for software defects. *IEEE*

Transactions on software engineering, 29(4):297–310.

Succi, G., Pedrycz, W., Stefanovic, M., and Miller, J. (2003). Practical assessment of the models for identification of defect-prone classes in object-oriented commercial systems using design metrics. *Journal of systems and software*, 65(1):1–12.

Szabo, R. M. and Khoshgoftaar, T. M. (2004). Classifying software modules into three risk groups. *International journal of reliability, quality and safety engineering*, 11(1):59–80.

Újházi, B., Ferenc, R., D., P., and Gyimóthy, T. (2010). New conceptual coupling and cohesion metrics for object-oriented systems. In *Proceedings of the IEEE working conference on source code analysis and manipulation*, pages 33–42.

Xenos, M., Stavrinoudis, D., Zikouli, K., and Christodoulakis, D. (2000). Object-oriented metrics – A survey. In *Proceedings of the European software measurement conference*, pages 1–10.

Xu, J., Ho, D., and Capretz, L. F. (2008). An empirical validation of object-oriented design metrics for fault prediction. *Journal of computer science*, 4(7).

Yu, P., Systa, T., and Muller, H. (2002). Predicting fault-proneness using OO metrics: An industrial case study. In *Proceedings of the 6th European conference on software maintenance and reengineering*, pages 99–107.

Zhou, Y. and Leung, H. (2006). Empirical analysis of object-oriented design metrics for predicting high and low severity faults. *IEEE Transactions on software engineering*, 32(10):771–789.

Zhou, Y., Leung, H., Song, Q., Zhao, J., Lu, H., Chen, L., and Xu, B. (2012). An in-depth investigation into the relationships between structural metrics and unit testability in object-oriented systems. *Science china information sciences*, 55(12):2800–2815.

Zhou, Y., Xu, B., and Leung, H. (2010). On the ability of complexity metrics to predict fault-prone classes in object-oriented systems. *Journal of systems and software*, 83(4):660–674.

Zimmerman, T., Nagappan, N., Herzig, K., Premraj, R., and Williams, L. (2011). An empirical study on the relation between dependency neighborhoods and failures. In *Proceedings of the IEEE Fourth International Conference on Software Testing, Verification and Validation, 2011*, pages 347–356.

10 Appendix

Appendix A

Table 17: Search Strings and Results

Digital Library	**Search String**	**Years Covered**	**Results**
Compendex & Inspec	(((((code Or software) WN KY) AND (analysis OR check* OR evaluat* OR predict*)WN KY)) AND ((”Object Oriented” OR “Object-Oriented” OR “OO” OR “OOP” OR “OOPL”) WN KY) AND ((Complex* OR abstraction OR encapsulation OR coupling OR cohesion OR volume OR messaging OR composition OR inheritance OR polymorphism OR Class OR Method OR function OR Modul* OR attribute OR characteristic OR Size) WN KY) AND ((metric* OR measur* OR indicator) WN KY) AND ((Empirical* OR ”Case Study” OR ”Case Studies” OR Experiment*) WN KY))	Until 2012	Retrieved 759
IEEE	((code Or software) AND (analysis OR check OR evaluat* OR predict) AND (”Object Oriented” OR “Object-Oriented” OR “OO” OR “OOP” OR “OOPL”) AND (Complex OR abstraction OR encapsulation OR coupling OR cohesion OR volume OR messaging OR composition OR inheritance OR polymorphism OR Class OR Method OR function OR Modul* OR attribute OR characteristic OR Size) AND (metric OR measure OR measurement OR indicator) AND (Empirical* OR “Case Study” OR “Case Studies” OR Experiment*))	Until 2012	Retrieved 64
Scopus	(TITLE-ABS-KEY(code OR software) AND TITLE-ABS-KEY(analysis OR check* OR evaluat* OR predict*) AND TITLE-ABS-KEY(”Object Oriented” OR “Object-Oriented” OR “OO” OR “OOP” OR “OOPL”) AND TITLE-ABS-KEY(complex* OR abstraction OR encapsulation OR coupling OR cohesion OR VOLUME OR messaging OR composition OR inheritance OR polymorphism OR class OR method OR function OR modul* OR attribute OR characteristic OR size) AND TITLE-ABS-KEY(metric* OR measur* OR indicator) AND TITLE-ABS-KEY(empirical* OR “Case Study” OR “Case Studies” OR experiment*))	Until 2012	Retrieved 467
ACM	(Abstract:((code Or software) AND (analysis OR check OR evaluat* OR predict) AND (“Object Oriented” OR “Object-Oriented” OR “OO” OR “OOP” OR “OOPL”) AND (Complex OR abstraction OR encapsulation OR coupling OR cohesion OR volume OR messaging OR composition OR inheritance OR polymorphism OR Class OR Method OR function OR Modul* OR attribute OR characteristic OR Size) AND (metric OR measure OR measurement OR indicator) AND (Empirical* OR “Case Study” OR “Case Studies” OR Experiment*)))	Until 2012	Retrieved 66
		Total Papers (All Databases)	**1356**

Appendix B

Table 18: Reliability: No Validation or Predictive Capability

Study	**Proxy**	**Best Set of Measures**	**Method**	**Validation Method**	**Predictive Ability**
S1	Defect Density (DD)	MHF AHF MIF AIF POF COF	Multiple Regression Analysis		Defect density - R-Squared (99.818%) Failure Density - R-squared (96.68%) Normalized Rework (NR) R Square (99.97%)
S2	No. of Defects	PPD, CBO, DEPTH, LCOM, NOC, FAN_IN, RFC, WMPC	Stepwise Regression		
S3	Fault-proneness	OMMIC, WMC, RFC	Multivariate Logistic Regression Analysis		R-squared (70%), Accuracy (88.24%), Specificity (93.75%), Precision (90.91%), Sensitivity (81.89%)
S23	Fault-proneness	SPA, SP, NIP, OCAIC, OCMIC	Multivariate Logistic Regression Analysis		significant at 0.05
S26	Fault-proneness	DIT, RFC, OCMEC, FMMEC	Multivariate Logistic Regression Analysis		
S31	Defect Proneness	EVNT, INHRTS	Multivariate Linear Regression		R-Squared 89.7%
S37	Defect Density	CSA, CSAO, CSI, CSO, DIT, LCOM, NAAC, NAIC, NOAC, NOCC, NOIC, NOOC, NPavgC, Osavg, PA, PPPC, RFC, WMC	Fuzzy c-means (FCM)	Compare with Gath-Geva clusting and ranking method	
S43	Fault-proneness	CO, ICPL, LCOM1, LCOM2, N F R , RFC, TCC, NAI,NAML, N M I m p, NDSTT,STMTS	Multivariate Logistic Regression Analysis		Accuracy (84.9%) Specificity (79%) Completeness (92.4%), R-Squared 43.2%
S46	Fault-proneness	DIT, CC, NMC	Multivariate Linear Regression		
S51	Fault-proneness	CBO, DIT, WMC, LOC	Multivariate Logistic Regression Analysis		Accuracy (69.61%), Precision (72.57%)

Continued on next page

Table 18 – *Continued from previous page*

Study	**Proxy**	**Best Set of Measures**	**Method**	**Validation Method**	**Predictive Ability**
S53	Fault-proneness		MI learning method (Set-kernel)	5-fold cross validation and comparison with Bayesian-kNN and Citation-kNN, Decision Tree, Back propagation Neural Network, Support Vector Machine	
S56	Fault-proneness	CBM, CBO, DIT, IC, LCOM, NOC, RFC, WMC, and NOMA	Radial Basis Function Neural Network (RBFNN)		Accuracy (90%), Precision (89.66%), Sensitivity (86.60%)
S57	Fault-proneness	CP4: DIT, SLOC	multivariate logistic regression analysis		Accuracy (86%), Completeness (70%)
S59	Fault-proneness	DIT, CLD, NOC, CBO, RFC, AMMIC, OMMIC, DMMEC, OMMEC, LCOM1, LCOM4, CO, LCOM5, Coh, TCC, NA, NM	Probabilistic Neural Network (PNN)		Accuracy (98.74%), Completeness (99.92%)
S62	Bad Smells	God Method, God Class and Shotgun Surgery	Multivariate Logistic Regression Analysis or Multinomial Multivariate Logistic Regression (MMLR)		Each signficant at 0.05
S63	Fault-proneness	MWE, LOC	Multivariate Logistic Regression Analysis		R squared (16.60%)
S67	Fault-proneness	CBO, DIT, LCOM, NOC, wmcM (is WMC-McCabe)	Multivariate Binary Logistic Regression		Accuracy (85.9%)
S81	Error-proneness	CTA,CTM, NOA	Multinomial Multivariate Logistic Regression (MMLR)	ROC Curve	
S33	No. of Defects	LOC, WMC, CBO, DIT	Multivariate Linear Regression		R-square (23.7%)
S94	Faults	Not Clear: the actual measures from the 34 stated that were actually used in the models	Discriminant modelling		

Table 19: Maintainability: No Validation or Predictive Capability

Study	**Proxy**	**Best Set of Measures**	**Method**	**Validation Method**	**Predictive Ability**
S11	Changes in classes	WMC, DIT, LCOM, NLM, CTA, and CTM	Multiple Linear Regression		
S12	Cost-proneness of classes	DlT, NAI, NCM, NMQ	Mahalanobis-Taguchi method	use three different number sets of measures on three different data sets and compare p-values	
S24	Maintenance effort	Number of Methods, Number of associations	Multivariate regression analysis (stepwise)		R-Squared (27.7%)
S34	Change logs	iic, niic, icmic, nicmic, immic, nimmic (coupling measures)	Stepwise multiple regression analysis		R-Squared across releases (82.40% - 85.00%)
S36	Perfective maintenance effort	Coupling and Cohesion	ANOVA		
S40	Average effort to understand package	NC, Ca	Multivariate Linear Regression		MMRE (0.464)
S42	Change-proneness	LOC, CBO, RFC, NOM	Pearson	R-Squared (81%)	
S68	Maintenance effort	E = 43.9 + (2.8 * Add + 5.3 * Mod + 1.3 * Del) * EAF	Least Squares Regression		R-squared (75%), MMRE (20%)
S71	Change frequency and change size	God Class (CS): WMC, TCC, ATFD	T-Test (two sample)		
S76	Maintenance Effort	CBO, LCOM, team-strategy	MANOVA		F-statistic (3.08)
S81	Class error probability	CTA,CTM, NOA	Multinomial Multivariate Logistic Regression (MMLR)	ROC Curve	
S80	Reuse	NIMC, NCMC, NDSuB, CC, NIV, NPubM, N sup, NP, CyCC	Multiple Linear Regression Analysis (Stepwise)		R-squared (85%)