ISSN: 2321-7782 (Online)
Volume 2, Issue 5, May 2014
International Journal of Advance Research in Computer Science and Management Studies
Research Article / Survey Paper / Case Study
Available online at: www.ijarcsms.com

A Tool for Quality Measurement of Software based on Object Oriented Design

Manoj Kumar[1]
Research Scholar
Dept. of CSE
LR Institute of Engineering & Technology
Solan (H.P.) – India

Ravinder Thakur[2]
Assistant Profess
Dept. of CSE
LR Institute of Engineering & Technology
Solan (H.P.) – India

Manish Mann[3]
Assistant Profess
Dept. of CSE
LR Institute of Engineering & Technology
Solan (H.P.) – India

Abstract: ***The role of software measurement is increasing, leading to the development of new measurement techniques. Many metrics have been developed related to the various object-oriented (OO) constructs like class, coupling, cohesion, polymorphism inheritance, information hiding. The objective of this metrics is to show relationships between the existing design metrics and chances of fault detection in classes. The main objective is to provide empirical evidence to draw the strong conclusions across studies. We used the data collected from Java applications for constructing a prediction model. With the implementation of CK and MOOD metrics suites the programmer as well as students will get benefit to test their software against the quality and also to learn the concept of quality. Results of this study show that many metrics capture the same dimensions in the metric set, hence are based on comparable ideas and provides duplicate information. It is shown that by using a subset of metrics prediction models can be built to identify faulty classes.***

Keywords: Software measurement; Prediction model; CK & MOOD Metrics; Quality.

I. INTRODUCTION

It is widely accepted that object oriented development requires a different way of thinking than traditional structured development and software projects are shifting to object oriented design. The main advantage of object oriented design is its modularity and reusability. Object oriented metrics are used to measure properties of object oriented designs. The basic goal of software engineering is to produce quality software. Quality can be defined as those product features which met the needs of the customers and thereby provide product satisfaction. The three aspects of software quality are functional quality, structural quality, and process quality [7].

(i) Functional quality means that the software correctly performs the tasks it's intended to do for its users.

(ii) Structural quality means that the code itself is well structured.

(iii) Process quality is the effectiveness of the development process.

1.1 Object-Oriented Design

Object-oriented design is a programming paradigm that began in the late 60's as software programs became more and more complex. The idea behind the approach was to build software systems by modeling them based on the real-world objects that they were trying to represent [11]. Object-oriented design is the discipline of defining the objects and their interactions to solve a problem that was identified and documented during object-oriented analysis.

Designers can perform a good Object Oriented design by following the OOD principles. If designers know the reasons of a bad design then it is helpful for them to avoid the bad design. There are some reasons for bad design, as for example: changing technology, domain complexity, lack of design skills and design practices and so on. Martin [13] proposes four primary symptoms which tell whether designs are bad. They are not orthogonal, but are related to each other in ways that will become obvious. They are: rigidity, fragility, immobility, and viscosity.

1.2 Object-Oriented Quality Metrics Suites

In parallel with the rise to prominence of the Object Oriented paradigm has come the acceptance that conventional software metrics are not adequate to measure object-oriented systems. This has inspired a number of software practitioners and academics to develop new metrics that are suited to the object oriented paradigm[14].A significant no. of metrics has been developed, from which CK and MOOD metrics suites are the most popular ones[12]. The next section will describe these metrics suites.

II. OVERVIEW OF MOOD AND CK METRICS SUITES

A significant number of object oriented metrics have been developed in literature. But the Metrics defined by Chidamber, Kemerer i.e. CK suite and metrics defined by Abreu i.e. MOOD metric suite become popular. The next section gives a detailed description of the MOOD Metric suite and Ck metric suite.

2.1 MOOD Metrics Suites

MOOD metrics set was first proposed in 1994 by the MOOD project team, headed F. B. Abreu. MOOD refers to a basic structural mechanism of the object-oriented paradigm as encapsulation (MHF, AHF), inheritance (MIF, AIF), polymorphism (POF), and message passing (COF) [10].

MHF (Method hiding factor): The MHF metric states the sum of the invisibilities of all methods in all classes.

AHF (Attribute hiding factor): The AHF metric shows the sum of the invisibilities of all attributes in all classes.

MIF (Method Inheritance factor): The MIF metric states the sum of inherited methods in all classes of the system under consideration.

AIF (Attribute inheritance factor): AIF is defined as the ratio of the sum of inherited attributes in all classes of the system.

POF (Polymorphism factor): The POF represents the actual number of possible different polymorphic situation.

COF (Coupling factor): The COF is defined as the ratio of the maximum possible number of couplings in the system to the actual number of coupling is not imputable to inheritance.

2.2 CK Metrics Suites

CK metrics suite was defined by the Chidamber and Kemerer in 1994[12]. Chidamber and Kemerer proposed six metrics; the following discussion shows their metrics.

WMC (weighted method per class): the count of methods implemented within a class (unweighted) or the sum of complexities of its methods (weighted).

DIT (Depth of Inheritance tree): DIT metric is the length of the maximum path from the node to the root of the tree.

NOC (Number of children): This metric measures how many subclasses are going to inherit the methods of the parent class.

CBO (Coupling between objects): The idea of this metrics is that an object is coupled to another object if two object act upon each other.

RFC (Response for class): RFC is the number of methods that can be invoked in response to a message in a class.

LCOM (Lack of cohesion in method): LCOM measures the amount of cohesiveness present, how well a system has been designed and how complex a class is.

2.3 Need & Scope of Study

Object oriented design is becoming more popular in software development environment and object oriented design metrics is an essential part of software environment. The researches of Abreu, Chidamber and Kemerer shows that any novice to experienced programmer can produce reliable and robust softwares using these object oriented metrics suites. As such many tools are available in market, which claims to analyse software on the basis of MOOD metrics, CK metrics or any other metrics, but either a simple user have to pay heavily to acquire them or either they are complex to use and some of them are not even producing the correct results and just pretending to produce the correct results. So a software tool is required that can be in direct reach of users ranging from a simple user to an experienced user and all programmers can be benefitted, and can produce reliable products.

So we are proposing to develop an object oriented software metrics tool based on CK and MOOD metrics suite. Due to time constraints our prime focus here is only CK and MOOD metrics suite, but we are keeping this tool as open source so that tool can further be extended to any other metrics suite or individual metric, so that tool can be more versatile.

III. ANALYSIS

After the Extensive literature survey we can summarize that object oriented Metrics helps to evaluate the development and testing efforts needed, the understandability, maintainability and reusability. The following table summarizes all these metrics and software quality parameters:

Metric / SQ Parameter	LCOM	RFC	CBO	NOC	DIT	WMC	CF	PF	MIF & AIF	MHF & AHF
Testing Effort	*High*	*Low*	*Low*	*High*	*High*	*High*	*High*	*High*	*High*	*High*
Reusability	*Low*	*High*	*High*	*High*	*High*	*Low*	*Low*	*High*	*High*	*Low*
Understandability	*Increases*	*Decreases*	*Decreases*	*Decreases*	*Decreases*	*Decreases*	*Decreases*	*Decreases*	*Decreases*	*Increases*
Maintainability	*Difficult*	*Difficult*	*Difficult*	*Difficult*	*Difficult*	*Difficult*	*Difficult*	*Difficult*	*Difficult*	*Difficult*
Development Effort	*Low*	*High*	*High*	*Low*	*Low*	*Low*	*High*	*Low*	*Low*	*Low*
Objective	*Decrease*	*Decrease*	*Decrease*	*Balanced*	*Balanced*	*Balanced*	*Decrease*	*Balanced*	*Balanced*	*High*

Table 3.1 *Parameters influenced by Object Oriented Metrics*

The first row of the table shows all the metrics from CK and MOOD suite and first column shows all the software quality parameters affected by these metrics in development process. The second row describes with the increase in corresponding metric testing efforts will be either low or high. Third row describes the factor of reusability will be either increase or decrease with the increase in the percentage of metric. Fourth and fifth row describes that with the increase in metric understandability will either increase or decrease and maintainability of the software will become either difficult or easier. Sixth row of the table describes that development efforts for the software are also influenced by these metrics; development efforts will either be high or low. The seventh row describes overall objective, whether we want to increase the percentage of metric in our software or whether we want to limit any metric for further increase.

Proposal for Object-Oriented Software Metrics Tool: Here we are proposing a framework for the development of an Object Oriented software metrics tool. The tool will simply use the attributes of the software to measure the quality of the software design. The framework is divided into three layers: analysis layer, processing layer and display layer.

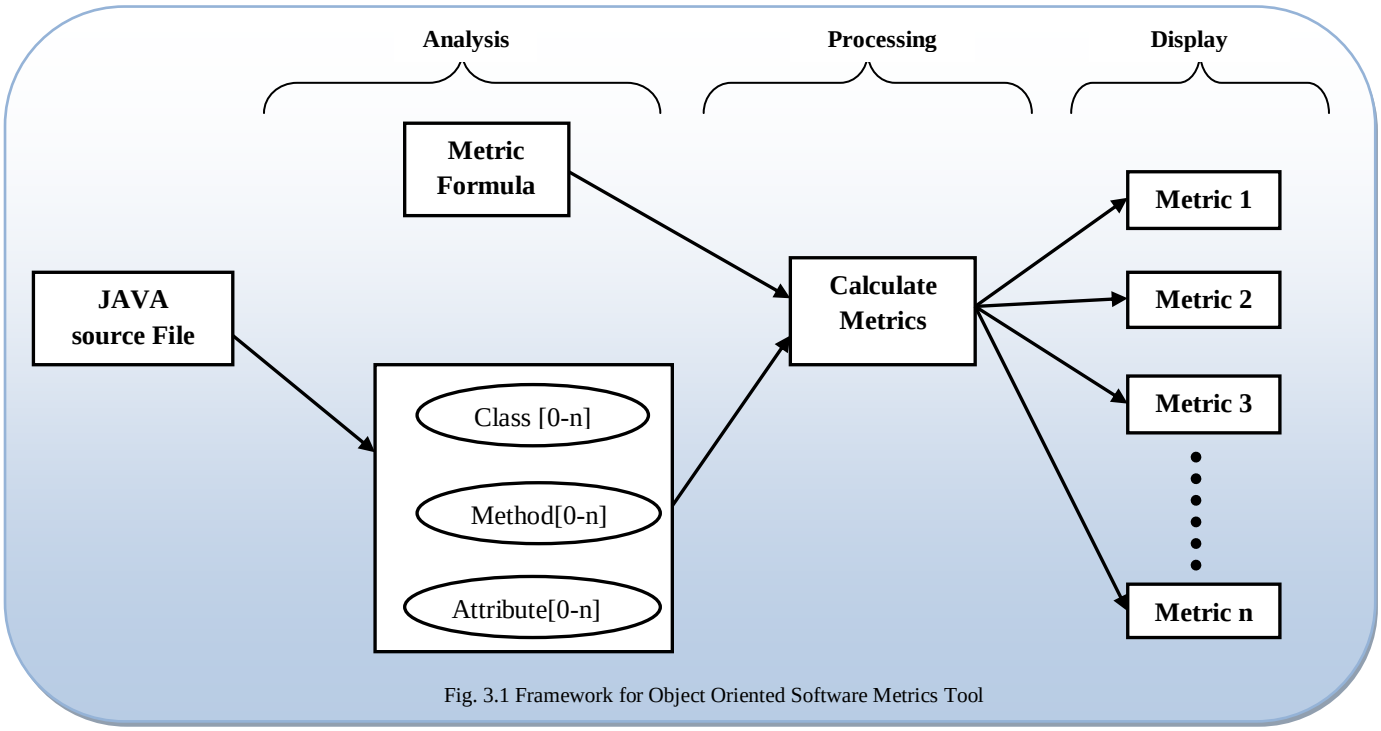

Fig. 3.1 Framework for Object Oriented Software Metrics Tool

Analysis Layer: The tool will take java source file as its input. The tool will use Java as the programming language. In the analysis layer the java source file is scanned for necessary attributes like number of classes, number of methods and number of attributes, their names and also their visibility. We are using the input streams and Stringbuffer classes of java to handle the inputted file. For scanning the file for methods, classes and attributes we are using the Scanner class, which will return tokens depending upon certain criteria from a stream. The output of the Analysis layer will be names, visibility and quantity of all the classes, methods and attributes.

Processing Layer: The processing layer of the Object Oriented Software Metrics tool will take the output of the analysis layer as the input. This layer will make use of these attributes and the formula's proposed by Abreu et. al for MOOD metrics and Formula's proposed by Chidamber and Kemerer for CK Metrics as the basis for calculating the corresponding metric.

Display Layer: The output of the processing layer will be the input to the display layer. In this layer the result of the analysis and calculation performed in the previous layers is presented to the user. The result will be displayed in terms of statistics as well as in the form of graphs. The user will be provided with the options of selecting specific metric or group of metrics on which software will be analyzed in the analysis layer. In the analysis layer the user will be provided with a well defined interface where user can input the java source file, check the corresponding metric or group of metrics to be calculated.

IV. Conclusion

This paper mainly focus on the study of CK and MOOD metrics suite for quality measurement of software based on object oriented design and secondly to propose the outline for development of a software metrics tool based on CK and MOOD metrics suite. With the implementation of CK and MOOD metrics suites the programmer as well as students will get benefit to test their software against the quality and also to learn the concept of quality. This tool can further be extended to support more software metrics apart from CK and MOOD Metrics. This tool will be kept as open source so that anybody can use it, learn the implementation of MOOD and CK metrics suite and can extend its versatility and at last produce the quality software.

References

1. Aman Kumar Sharma, Arvind Kalia, Hardeep Singh, "Metrics identification for measuring object oriented software quality", International journal of soft computing and engineering (IJSCE) ISSN: 2231-2307, Volume-2, Issue-5,2012.
2. Amandeep kaur et al., "Empirical analysis of CK & MOOD metric suit", International journal of innovation, management and technology, Vol. 1, No. 5, ISSN: 2010-0248,2010.

3. B. Meyer, "Object-oriented Software Construction", Second Edition, Santa barbara,Prentice-Hall, 1988.
4. C. Neelamegam and M. Punithavalli, "A survey-object oriented quality metrics", Department of information technology sri Nehru maha vidyalaya college arts and science, proceeding of Coimbatore global journal of computer science and technology, pp. 183-186,2009.
5. S. M. Jamali, "Object oriented metrics", Department of computer engineering, Sharif university of technology, Tehran, Iran,2006.
6. D.Jackson and M. Usher, "Grading student programs using ASSYST", Proceeding 28 the ACM SIGCSE tech. symposium on computer science education, San Jose, California, USA, pp. 335-339,1997.
7. David chappell, "The three aspects of software quality: functional, structural, and process", principal Chappell & Associates, San Francisco, California.
8. E. Chandra, P. Edith Linda, "Class break point determination using CK metrics thresholds",Global journal of computer science and technology,Vol.10 Issue 14 (Ver.1.0),2010.
9. F. B. Abreu and R.Carapua, "Candidate metric for OOS within taxonomy framework", Journal of system & software Vol. 26, No.1, 1994.
10. F. B. Abreu, "Design metrics for object-oriented software systems" ECOOP'95 quantitative methods workshop, Aarhus, 1995.
11. Linda H. Rosenberg, "Applying and Interpreting object oriented Metrics"||, Track 7 Measures/ Metrics in software technology conference, 1998.
12. Muktamyeesarker, "An overview of object oriented design metrics", master's thesis, department of computer science, Umea university, Sweden, 2005.
13. Robert C. Martin, "Design principles and design patterns", online www.objectmentor.com, Apr 1994.
14. Tobias mayer & Tracy hall, "Measuring OO systems: a critical analysis of the MOOD metrics", centre for systems and software engineering (CSSE) south bank university, London, UK, 1999.
15. Sonia Chawla, "Review of MOOD and QMOOD metric sets", International journal of advanced research in computer science and software engineering 3(3), pp. 448-451,2013.
16. Roger s. pressman, "Software engineering a practitioner's approach", Seventh edition, Newyork, Mc Graw Hill, 2010.
17. Meryem Lamrani, Younes El Amrani, Aziz Ettouhami, "A formal definition of metrics for object oriented design: MOOD metrics", University Mohammed V Agdal, Computer science department PB 1014 Rabat, Morocco journal of theoretical and applied information technology, Vol. 49 No.1,2013.

