Advances in Computer Science and Information Technology (ACSIT)
Print ISSN: 2393-9907; Online ISSN: 2393-9915; Volume 2, Number 4; April-June, 2015 pp. 338-341

http://www.krishisanskriti.org/acsit.html

Evaluation of Object Oriented Software Metric for Quality Improvements

Manoj Wadhwa[1], Aarti Sharma[2] and Nidhi Sharma[3]

[1,2,3]Department of Computer Science Echelon Institute of Technology Faridabad, India
E-mail: [1]manojkw@gmail.com, [2]sharmaaarti05@gmail.com, [3]nim.nidhi23@gmail.com

Abstract—*Today object oriented design metric play an important role in software development environment. To produce a high quality object oriented software application we emphasis on design phase of software. Object oriented metrics are used to evaluate and predict the quality of software. The need of object oriented and high quality software increase day by day. Traditional metrics could not enhance the quality of software as they based on structural programming paradigm where design and data structures are measured independently. But an object oriented metric can measure data structure and design structure in combined way.*

In this paper we have evaluated all six CK metrics suits which are weighted method per class(WMC) ,depth of inheritance (DIT), number of children(NOC), coupling between objects(CBO) ,response for a class(RFC), lack of cohesion (LCOM) .We also evaluated some MOOD suits like MHF, AHF, MIF, AIF and PF. we have done an empirical study on CK metrics suits and tried to find out the quality improvements with the help of these metrics.

Keywords: *Object Oriented Metric ,CK Metric Suits ,MOOD metrics suit.*

1. INTRODUCTION

For software modularization we need object oriented decomposition which is an approach for improving the organization and quality of a software source code. Software metric provide an important role for improving the quality of source code as well as software. Software metrics helped for analyzing, designing, coding, documentation and testing. Object oriented metrics help in evaluation of quality improvements. Object oriented metrics based on classes and objects. Classes and object relationship can be represented by software engineers. These metrics are used for analysis and design of a system. Classes use polymorphism, encapsulation, inheritance and abstractions for OO metrics. In classes there should be high cohesion and less coupling among them. These classes have some properties comprehension, testing, reusability, and maintainability etc. In this paper we introduced cohesion and coupling of classes. In OO software systems, cohesion is measured at class level and coupling defines the relationship between classes and methods. There are many OO metrics available to predict the quality of software like CK Metrics, MOOD and Lorenz and Kidd.

In this paper there are 9 sections. In first section, there is an introduction about object oriented software metrics suits. This section tells about the quality factors of software and its characteristics. In second section, there is a review of object oriented metrics which are classified into 3 categories such as CK suite, MOOD and Lorenz and Kidd. But in this paper there is description of only two suits which are further classified on system level and class level. In system level the entire MOOD metrics like MHF, AHF, AIF, MIF and PF are describes for quality metrics. In class level all the CK metrics are describe like WMC, RFC, DIT, NOC and LCOM metrics. In third section there is a description of proposed model which shows the working of metrics tool. The fourth section describes the characteristics of object oriented design quality metrics. The fifth section describes the advantages of software metric and sixth section describes the limitations of software metrics. The seventh section describes the conclusion and future scope. The eighth section describes the acknowledgement and last ninth section is references.

2. REVIEW OF METRICS

There are several metrics for improving quality of software which are based on object oriented metrics these are: CK Metric, MOOD, Lorenz and Kidd metric suits.

2.1 System level metrics

These metrics can be derived with the help of class metrics with statistics, as relative measure, identifying system. The MOOD (Metrics for object oriented design) suits are used for defining characteristics of a system. MOOD metrics suits were proposed by Fernando Brito and Rogerio Carpuca in 1994for identification of quality, abstraction and quantitative measurement of object oriented software. It include six metrics which measure the presence of OOD (object oriented design) attribute. These metrics values lie between 0 and 1. The MOOD metrics are:

(i)Method Hiding Factor (MHF)

It defines the ratio of sum of the invisibilities of all the methods in all classes to the total number of methods defined

in a system . the invisibilities of method can be the percentage of all the classes in a system from which this method is not visible. If methods are private then MHF=100%.

(ii)Attribute Hiding Factor (AHF)

It defines the ratio of sum of the invisibilities of all the attributes in all classes to the total number of attributes defined in a system. The invisibilities of attributes can be the percentage of all the classes in a system from which this attribute is not visible. If methods are private then MHF=100%.

(iii)Method Inheritance Factor (MIF)

It is the ratio of sum of inherited methods to the total number of methods in all classes for the system. If no re- usability of methods then MIF=0.

(iv)Attribute Inheritance Factor (AIF)

It is the ratio of sum of inherited attributes to the total number of attributes in all classes for the system. If no reusability of attributes then AIF=0.

(v)Polymorphism Factor (MIF)

It is the ratio of actual no. of methods override to the maximum number of methods override in all classes for the system. If all the methods are overridden in all derived classes then PF=100%.

(vi)Coupling Factor (CF)

It is the ratio of actual coupling among classes to maximum number of coupling possible in all the classes. If all the classes are coupled then, CF=100%.

2.2 Class Metric Level

CK suits are the class level metric used for enhancing the quality of a software. Shyam R.Chidamber and Chris F. Kemerer (CK) developed a metric suit which is based on OOD. CK metric suit provide 6 metrics which give different characteristics of a software. These metrics are used for identify characteristics of a class, told about different aspects of class abstraction and the remedial actions may be taken for that class. Some metrics are:

(i)Weighted Method per Class (WMC)

WMC is defined as the count of methods implemented within a class or can be defined by sum of the complexities of the methods in a class. WMC measure the reusability, understandability, complexity and maintainability of a class or system. Number of methods inherited in all the classes increases the complexity and decrease the reusability and understandability of a class.

(ii)Response for a Class (RFC)

RFC is defined by total number of methods within a set that can be invoked in response to message sent to an object to perform an operation on that class. It count the accessible methods in a class. It also measure complexity and maintainability of a class. If complexity is high due to large number of invoked methods then maintainability of a class decrease and quality of software is also decrease.

(iii)Lack of Cohesion Method (LCOM)

It defines the similarities and dissimilarities of methods in classes. Similarity of 2 methods can be defined with number of attributes used in common. It measures the cohesiveness of a class by common attributes. If LCOM is high then cohesion is less .Methods in a class is highly desirable cohesive, we cannot divide the classes. Due to less cohesion encapsulation and complexity of a class is increase. So number of errors increase in development process. If the value of LCOM is vary near 0 then high cohesion occur and less faults present in software.

(iv)Depth of Inheritance (DIT)

It defines how deep a class hierarchy is. It measure maintainability and reusability of a class. A class with small DIT has much potential for reuse then deeper classes. If DIT is high then effort required to monitor the functionality of a system is high.

(v)Number of Children (NOC)

It measure the number of classes associated with a given class using inheritance relationship. In a class, the number of children should be less. We can enhance the maintainability and complexity by inheriting less number of children in a class.

(vi)Coupling between Object Classes (CBO)

It is count of number of other classes to which it is coupled. Coupling is occurring between two classes when one class uses methods of another class. If coupling is more between classes then reusability will be less. Classes with more coupling makes the software maintenance, complexity and testing difficult. If coupling is strong then system will become more complicate and classes are harder to understand, change or modified by itself, if it is interrelated with other classes. So coupling should be less between classes to avoid complexity in a system.

3. MODEL OF PROPOSED SYSTEM

In proposed system, the system is based on object oriented software metrics. It use all the proposed metric of OOD like CK and MOOD metric for enhancing the quality of a system .It uses C++ and JAVA classes which are working on this system. OOD measure the complexity maintainability and reusability by using this system. In fig. 1 shows block diagram of the system which consist of some input values or source

code, cohesion, coupling, encapsulation and inheritance. After applying these functions it display result and the important gathering information (structured or unstructured form) about the software.

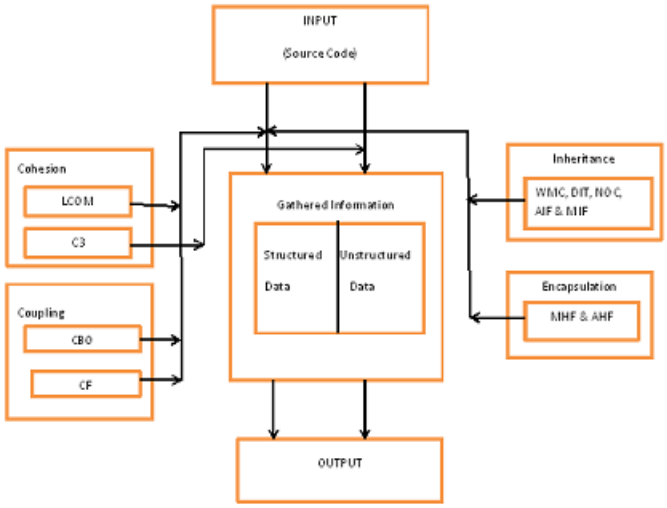

Fig. 1: Block Diagram of Proposed System

i.)Steps for solving the problem

1. Give the source code of java or C++ project file as input.
2. Gather information in structured or in unstructured way.
3. Apply the cohesion or coupling or encapsulation or inheritance methods (Algorithm).
4. Get the appropriate result or output of system.

ii.) Steps for Cohesion

1. input any java or C++ project file .
2. Gather information in structured or unstructured way.
3. Apply LCOM procedures on structured data and C3 procedures on unstructured data.
4. Get the appropriate output of cohesion.

iii.) Steps for Coupling

1. Input any java project file.
2. Gather information in structured or unstructured data
3. Apply the CBO and CF procedure on structured data.
4. Get the appropriate output of cohesion

iv.) Steps for Inheritance

1. Input any java or Cpp project file as input.
2. Gather information in structured or unstructured way.
3. Compile the file using WMC, AIF, MIF,NOC & DIT methods.
4. Display the fields and methods which are used in project.
5. Find the metrics.
6. Get the appropriate result or output of inheritance.

v.)Steps for Encapsulation

1. Input any java or Cpp project file.
2. Compile the file by using MHF and AHF methods.
3. Display the fields and methods which are used in project.
4. Find the metrics.
5. Get the appropriate output of encapsulation.

We can also perform polymorphism methods on this system for getting more results related to project or measure some advance features also.

4. CHARACTERISTICS OF IMPROVING OOD QUALITY METRICS

Based on the review of existing software metrics suite, a list of parameters needed for accepting or discarding software metric. There are some characteristics on which quality of a software depend are:

- Functionality
- Complexity
- Usability
- Efficiency
- Maintainability

In software functionality plays an important role. It acts as interoperability of a system, which tells about the ability of a software component to interact with other components or systems. Complexity and maintainability of a software depend on the higher values of metrics used in software. Efficiency tells about the time and resources behavior required for software. Usability is the understandability and learnability of a system.

Table I - Table II illustrates, in general, whether a high or low value is desired for the metric for better code quality.

Table 1: Showing the quality factors of MOOD Metrics

MOOD METRICS (High Values)	Complexity	Usability	Efficiency	Maintainability	Desired Value Of Metrics
MHF	Less	Less	High	Less	High
AHF	Less	Less	High	Less	High
MIF	High	High	High	High	Low
AIF	High	High	High	High	Low
PF	High	High	High	High	Low
CF	High	High	High	Less	Low

Table. I show MOOD metrics values for each characteristic of quality metrics and it give the desired result after applying these characteristic on the software.

Table 2: Showing the quality factors of CK Metrics

CK METRICS (High Values)	Complexity	Usability	Efficiency	Maintainability	Desired Value Of Metrics
WMC	High	High	High	High	Low
DIT	High	High	High	High	Low
NOC	High	High	High	High	Low
CBO	High	High	High	High	Low
RFC	High	High	High	High	Low
LCOM	Less	Less	Less	Less	High

Table II shows the object oriented CK metrics which shows the metrics value for each characteristics of quality metrics. Here WMC, DIT, NOC, CBO, AND RFC shows high complexity, usability, efficiency and maintainability of a software and the desired result shows low metric value . But LCOM shows high metric value for cohesion.

5. ADVANTAGES OF OBJECT ORIENTED SOFTWARE METRICS

- It is useful in comparative study of various design methodology of software systems.
- Software metrics analyze comparison and critical study of various programming languages.
- They are used for comparing and evaluating capabilities and productivity of people involved in software development.
- It is also used for enhancing the quality of software.
- It provides guidance to resource manager for proper utilization.

6. LIMITATION OF OBJECT ORIENTED SOFTWARE METRICS

- The cost of implementing software metrics application is very high.
- The verification and justification of software metrics is depend on historical data which is difficult to verify.
- Software metrics are not used for evaluating performance of the technical staff.
- Most of the software development models are probabilistic and empirical.
- They don't know the exact estimates of certain variables used in software

7. 7. CONCLUSION & FUTURE SCOPE

Software metrics are used for communication, measuring progress of software and achieving the goals. In this paper, there is a description of two object oriented software metrics suits which help in evaluation of quality metrics. These metrics are CK suite and MOOD suite. Lorenz and Kidd are also object oriented metric suit but they are not able to play any role in quality of software. Lorenz and Kidd are statistical measures for software in terms of counting, count only number of methods and variables under various categories. From among the suits analyzed in the study are useful in evaluation of software quality. These metrics are CK suits like WMC, RFC, DIT, NOC, CBO, LCOM and MOOD suits are MHF, AHF, AIF, MIF, and PF. These ten metrics affect the characteristics of object oriented system i.e. inheritance encapsulation, abstraction and polymorphism. A model of software metric gives the description about working of a metric tool which helps in measuring the components in software. The tool can be enhanced to asses object oriented languages such as C++, JAVA, C Sharp etc., The tool will be helpful in assessing the quality in advance to develop awareness for quality issues such as reliability, testability and maintainability. These tools are also used for reducing complexity and reusability of a software. The information gathering from the system used to measure the cohesion of software to extract the information for cohesion measurement.

Further research can be extended on the empirical validation of object oriented design metrics for improving the quality of software. Lorenz and Kidd metrics is also used for increasing the performance of software.

8. ACKNOWLEDGEMENTS

We would to like to convey my thanks message Dr.(Prof.) Manoj Wadhwa . Their advice and support make us able to complete our work and also thankful to Echelon Institute of Technology for providing us all the facilities and services during our studies.

REFERENCES

[1] Aman Kumar Sharma, Arvind Kalia, and Hardeep Singh, "An Analysis of Optimum Software Quality Factors", IOSR Journal of Engineering, vol. 2 issue 4, 2012.

[2] Basli VR, Briand LC, Melo WL. "A Validation of object oriented design metric as quality indicator", Technical Report, University of Maryland, Department of Computer Science,1-24,1995,

[3] Gurdev Singh, Dilbag Singh, and Vikram Singh, "A Study of Software Metrics", International Journal of Computational Engineering and Management (IJCEM), vol. 11, 2011.

[4] M. Subramanyam, and R. Krishnan, "Empirical Analysis of CK Metrics for OOD Complexity: Implication for Software defect", IEEE transactions on software engineering, 2003.

[5] R. Harrison, S. Counsell, and R. Nithi, "An Overview of Object-Oriented Design Metrics", Proceedings of the 8th International Workshop on Software

[6] Roger S. Pressman, "Software Engineering: A Practitioner's Approach", 6th ed., McGraw Hill International, 2005.

[7] Shyam R. Chidamber, and Chris F. Kemerer, "A Metrics Suite for Object Oriented Design", IEEE Transactions on Software Engineering, vol. 20, no. 6, 1994.