Volume 3, Issue 3, March 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Review of MOOD and QMOOD metric sets

Sonia Chawla
Department of Computer Science and Applications
Kurukshetra University,Kurukshetra
Haryana,India

Abstract— Measurement is required to access quality and improve performance of the product. Software metrics help in decision making and identifying areas that need improvement. Software quality nowadays has become an essential element. Some metrics can be applied in the early stages of product development that helps in eliminating the complexity at later stages. This paper reviews the two widely used object oriented metrics- MOOD and QMOOD set of metrics.

Keywords— Software quality, object-oriented metrics, quality attributes, complexity .

I. **Introduction**

Tom DeMarco's [9] statement "You can't control what you can't measure" played an important role in influencing software engineers in the field of software quality. Metrics are required for estimating size, measure complexity, quality in terms of quality attributes like reliability, usability etc. A good software should fulfill all the essential quality factors.
IEEE [12] defines metric as "a quantitive measure of the degree to which an item possesses a given quality attribute". With the help of metrics, risk areas can be identified. Object-oriented metrics focus on concepts like class, inheritance, abstraction etc. So far, many object-oriented metrics have been propsed but all of them doesn't satisfy many attributes. A metric should be simple, consistent, objective and computable early in the life cycle. As there are lot of metrics, so the metrics that provide relevant information must be used. It enables an engineer to assess software early in the process, making changes that reduces complexity and improve long term viability of the end product [17].
Rest of the paper is organized as follows: Next section gives an overview of the existing studies in object oriented metrics. Section III and Section IV describes MOOD (Metrics for object-oriented design) and QMOOD (Quality model for object-oriented design) set of metrics respectively. Section V concludes the paper.

II. **Related Work**

Among all the metric suites, Chidamber and kemerer [8] metric suite is the most referenced one. They defined six metrics-Weighted Methods per Class (WMC), Depth of Inheritance (DIT), Coupling Between Objects (CBO), Response For a Class (RFC), Lack of Cohesion in Methods (LCOM), Number of Children (NOC). Various studies have being done on their validation by many researchers. Li et al. [19] validated CK metrics using statistical analysis on two commercial systems. Five of the six metrics (except CBO) helped predict maintenance effort and proposed many metrics to evaluate maintainability. Lorenz and Kidd [13] divided design metrics into four categories: size, internals, externals, inheritance.
MOOD (Metrics for Object Oriented Design) metric set was proposed by Abreu et al. [3]. These metrics measure the object oriented mechanisms such as inheritance (Method Inheritance Factor, Attribute Inheritance Factor), encapsulation (Method Hiding Factor, Attribute Hiding Factor), polymorphism (Polymorphism Factor), message passing (Coupling Factor). Rosenberg [18] proposed nine metrics to evaluate attributes like efficiency, complexity, reusability, testability, understandability. Three of them are the traditional metrics- LOC, Comment percentage, Cyclomatic Complexity and the rest six were same as those of CK metrics.
Basili, Briand and Melo [6] investigated CK metrics and presented that out five out of six metrics appear to be useful to predict class fault proneness during the early phases of the life cycle. Cartwright and Shepperd [7] studied telecommunication system and concluded that DIT and NOC were found to influence defect density. Bansiya J. et al. [5] defined Quality Model for Object Oriented design (QMOOD) metrics. They defined relation between quality attributes and design properties with the help of equations.

III. **THE MOOD METRICS SET**

F.B.Abreu et al. [3] defined MOOD (Metrics for Object Oriented Design) metrics. They evaluated that how OO design mechanisms like inheritance, polymorphism, information hiding and coupling can make an influence on quality characteristics like defect density (a reliability measure) and rework (a maintainability measure). They also derived certain criterias like metrics should be formally defined, dimensionless, obtainable early, down-scaleable, easily computable. They should be language and size independent. MOOD metrics refers to a basic structural mechanism of the object-oriented paradigm like encapsulation (MHF and AHF), inheritance (MIF and AIF), polymorphism (PF) and message passing (CF). MOOD metrics are based on set theory and includes simple mathematics. These are applicable as soon as a preliminary design is available so the flaws can be detected in the early phase. Subjectivity is avoided as these are formally defined. MOOD metric suite [1] can be summarized as follows :

A. Method Hiding Factor (MHF) : The MHF is the ratio of sum of the invisibilities of all methods defined in all classes to the total number of methods defined in the system under consideration. The invisibility of a method is the percentage of the total classes from which this method is not visible.
If all methods are private, MHF=100% . If all methods are public, MHF=0% .

B. Attribute Hiding Factor (AHF): The AHF is the ratio of sum of the invisibilities of all attributes defined in all classes to the total number of attributes defined in the system under consideration. The invisibility of an attribute is the percentage of the total classes from which this attribute is not visible.
If all attributes are private, AHF=100% . If all methods or attributes are public, AHF=0.

C. Method Inheritance Factor (MIF): The MIF is the ratio of sum of the inherited methods in all classes of the system under consideration to the total number of available methods (locally defined plus inherited) for all classes.
If there is no reusability,then MIF=0.

D. Attribute Inheritance Factor (AIF): The AIF is the ratio of sum of the inherited attributes in all classes of the system under consideration to the total number of available attributes(locally defined plus inherited) for all classes.
If there is no reusability,then AIF=0.

E. Polymorphism factor (PF): It equals the number of actual method overrides divided by the maximum number of possible method overrides.
If all methods are overridden in all derived classes, then PF=100%.

F. Coupling Factor (CF): Coupling Factor is the actual couplings among classes in relation to the maximum number of possible couplings.
Maximum possible couplings happens when all classes are coupled to and from all other classes. If no classes are coupled, CF = 0%.

MOOD metrics are expressed as percentages, ranging from 0% (no use) to 100% (maximum use) and are dimensionless [10].

IV**. THE QMOOD METRICS SET**

QMOOD (Quality Model for Object Oriented Design) was proposed by Bansiya and Davis [5]. It is the comprehensive model that assess quality attributes like reusability, functionality, effectiveness, understandability, extendibility, flexibility. There are four levels (L1 through L4) and three mappings to connect these levels in QMOOD.
The four levels are:

A. Design Quality Attributes.
B. Object oriented design Properties.
C. Object oriented design Metrics.
D. Object oriented design Components

A. Design Quality Attributes: QMOOD design quality attributes are functionality, effectiveness, understandibilty, extendibility, reusability and flexibility.

B. Object oriented design Properties: Design properties included in this set of metrics are inheritance, encapsulation, polymorphism, abstraction, coupling, cohesion, messaging, hierarchies, composition, design size, and complexity.

C. Object oriented design Metrics: Metrics in QMOOD are DSC, NOH, ANA, NOP, CIS, NOM, DCC, CAM, MOA, MFA, DAM. Design metrics used to assess design properties are shown in table I.

D. Object oriented design Components: Design components includes attributes, methods, objects (classes), relationships and class hierarchies.

The complete set of metrics in QMOOD are as follows:
1) Design Size in classes (DSC): This metric is a count of the total number of classes in the design.

2) Number of Hierarchies (NOH): This metric is a count of the number of class hierarchies in the design.

3) Average Number of Ancestors (ANA): This metric value signifies the average number of classes from which a class inherits information.

4) Data Access Metric (DAM): It is the ratio of the number of private attributes to the total number of attributes declared in the class.

5) Direct Class Coupling (DCC): It is a count of different number of classes that a class is directly related to.

Table I
Design Metrics for Design Properties[5]

Design Property	Derived Design Metric
Design Size	Design Size in Classes (DSC)
Hierarchies	Number of Hierarchies (NOH)
Abstraction	Average Number of Ancestors (ANA)
Encapsulation	Data Access Metric (DAM)
Coupling	Direct Class Coupling (DCC)
Cohesion	Cohesion Among Methods in Class (CAM)
Composition	Measure of Aggregation (MOA)
Inheritance	Measure of Functional Abstraction (MFA)
Polymorphism	Number of Polymorphic Methods (NOP)
Messaging	Class Interface Size (CIS)
Complexity	Number of Methods (NOM)

6) Class interface Size (CIS): It is a count of the number of public methods in a class.

7) Measure of aggregation (MOA): It measures the extent of part-whole relationship, realized by using attributes

8) Cohesion Among Methods of Class (CAM): It computes the relatedness among methods of a class based upon the parameter list of methods.

9) Measure of Functional Abstraction (MFA): It is the ratio of the number of methods inherited by a class to the total number of methods accessible by member methods of the class.

10) Number of Polymorphic methods (NOP): This metric is a count of the methods that can exhibit polymorphic behavior.

11) Number of methods (NOM): It is a count of all the methods defined in a class .

QMOOD includes equations that defines relationship between quality attribute and properties. These are shown in table II.

Table II
Computation formulas for quality attributes [5]

Quality Attribute	Index Computation Equation
Reusability	-0.25 * Coupling + 0.25 * Cohesion + 0.5 * Messaging + 0.5 * Design Size
Flexibility	0.25 * Encapsulation - 0.25 * Coupling + 0.5 * Composition + 0.5 * Polymorphism
Understandability	-0.33 * Abstraction + 0.33 * Encapsulation - 0.33 * Coupling + 0.33 * Cohesion - 0.33 * Polymorphism - 0.33 * Complexity - 0.33 * Design Size
Functionality	0.12 * Cohesion + 0.22 * Polymorphism + 0.22 Messaging + 0.22 * Design Size + 0.22 * Hierarchies
Extendibility	0.5 * Abstraction - 0.5 * Coupling + 0.5 * Inheritance + 0.5 * Polymorphism
Effectiveness	0.2 * Abstraction + 0.2 * Encapsulation + 0.2 * Composition + 0.2 * Inheritance + 0.2 * Polymorphism

V. CONCLUSION

MOOD and QMOOD metric sets have been reviewed in this paper and shows that these metrics are the good indicators of quality of the software and are computable early in the design phase that helps in reducing complexity at the later stages. Both the sets include simple mathematics.
MOOD metrics avoids subjectivity as these are formally defined. These are expressed as percentages and are based on set theory. QMOOD is the hierarchical model that defines relation between quality attributes and design properties with the

help of equations. It is the inclusive model and can be refined. Design properties like compostion, design size have also been included in it.

REFERENCES

[1] Abreu, B.F. and W.L. Melo (1996): "Evaluating the impact of Object-Oriented Design on Software Quality", Proceedings of METRICS '96, IEEE, 1996.pp. 90-99

[2] Abreu F.B.(1995):. ECOOP'95 Quantitive Methods Workshop"Design Metrics for Object-Oriented Software Systems"1995

[3] Abreu F.B. and R.Carapuca."Object-Oriented Software Engineering: "Measuring and Controlling the Development Process". Proceedings of the 4th International Conference on Software Quality, McLean,Virginia, USA,October ,1994.

[4] Automated Metrics and Object-Oriented Development : http://collaboration.cmc.ec.gc.ca/science/rpn/biblio/ddj/Website/articles/DDJ/1997/9712/9712d/9712d.htm

[5] Bansiya J. and C. G. Davis (2002): A Hierarchical Model for Object-Oriented Design Quality Assessment, IEEE Transactions on Software Engineering, pp. 4-17, 2002.

[6] Basili, V.R., L. C. Briand and W. L Melo. (1996): "A Validation of Object-Oriented Design Metrics as Quality Indicators", IEEE Transactions on Software Engineering, 22(10), 1996, pp. 751-761

[7] Cartwright, M. and M. Shepperd (2000): "An Empirical Investigation of an Object-Oriented Software System", IEEE Transactions Software Eng., 26(7), 2000, pp. 786-796.

[8] Chidamber, S. R. and C. F. Kemerer (1994):"A Metrics Suite for Object Oriented Design", IEEE Transactions on Software Engineering, 20(6), 1994, pp. 476–493.

[9] DeMarco,T., "Controlling Software Projects: Management,Measurement and Estimation" ,Yourdan Press,New York,1982

[10] Design Quality Metric for Object Oriented Software Systems: http://www.ercim.eu/publication/Ercim_News/enw23/abreu.html

[11] K.K. Aggarwal, Yogesh Singh, Arvinder Kaur, Ruchika Malhotra 2006. "Empirical Study of Object-Oriented Metrics." Journal of Object Technology Vol. 5. No. 8, pp. 149-173. http://www.jot.fm/issues/issue_2006_11/article5

[12] IEEE Std 610.12 (1990),"IEEE Standard Glosary of Software Engineering Terminology", The Institute of Electrical and Electronics Engineers, NY,USA,1990 .

[13] Lorenz, M. and J. Kidd (1994): "Object-Oriented Software Metrics", Prentice Hall, 1994.

[14] Dubey S.K.and Ajay Rana (2010). "A Comprehensive Assessment of Object- Oriented Software Systems Using Metrics Approach." International Journal on Computer Science and Engineering, Vol. 02, No. 08, pp. 2726-2730.

[15] Dubey S. K. and Ajay Rana (2011): "Assessment of Maintainability Metrics for Object Oriented Software System", ACM SIGSOFT Software Engineering Notes, 2011.

[16] Mood and Mood2 metrics: http://www.aivosto.com/project/help/pm-oo.html

[17] Pressman,R.S., "Software Engineering-A Practitioner's Approach", The McGraw-Hill ,Fifth Edition, 2001.

[18] Rosenberg L. H. and L. Hyatt (1995): "Software Quality Metrics for Object-Oriented Environments", SATC, NASA Technical Report SATC-TR-95-1001, 1995.

[19] W.Li and S.Henry, "Object Oriented Metrics that Predict Maintainability,"J.Systems and Software, vol.23, pp.111-122,1993.