An overview of Object Oriented Metrics A complete Survey

Gomathi. S
Department of Information and Computer Technologies
Sri Krishna Arts and Science College
Coimbatore, India
mailtogomathisrinivasan@gmail.com

Edith Linda. P
School of IT & Science
Dr. G. R. Damodaran College of Science
Coimbatore, India
p.lindavinod@gmail.com

Abstract— Object oriented metrics have become more important in software engineering field. They are used to measure software quality and to estimate the cost, to enhance the reliability, maintainability and effort of software projects. Object oriented metrics evaluate the complexity of OO program. Object oriented concepts are dominating the software industry and engineers need proper measuring parameters in order to make software more efficient and reused. Object Oriented Design metrics is an essential part of software engineering. This paper collects many object oriented metrics proposed by various researchers and a final conclusion is to overcome the drawbacks of existing metrics and to deliver efficient metrics which is effectively used to measure the programs.

Keywords-Software architecture; Coupling; cohesion; reusability; maintainability

I. INTRODUCTION

Object-oriented measurements are used to evaluate and predict the quality of software [7]. The validation of these metrics demonstrates that the metric measures what it purports to measure and the metric is associated with an important external metric, such as reliability, maintainability and fault-proneness [12]. Object oriented design is concerned with developing an object-oriented module of a software system to apply the identified requirement s. Designer will use OOD because it is a faster development process, module based architecture, contains high reusable features, increases design quality and so on.

II. LITERATURE REVIEW

Amit Sharma, Sanjay Kumar Dubey [2] highlighted the the classification of metrics like software quality metrics and the object-oriented metrics or all the software quality metrics like the HALSTEAD metrics, size merics, quality metrics and all the object oriented metrics which are proposed from 90's like CK metrics, Moose Metrics, QMOOD Metrics, GQM, MOOSE, EMOOSE and maintain the comparison table through which we can easily analyze the difference between all the object-oriented metrics

Hitz and Montazeri [10] argue that coupling between two classes should be multi-faceted rather than being a singular relation. In other words, there should be many aspects taken into account when measuring the coupling relationship between classes within a system. Briand et al. [4] identify eighteen distinct aspects of coupling with each focusing on a different type of relationship. These relationships are finer-grained than previous approaches where they tend to only pay attention to method-method, class-method, class-attribute, etc.

Li and Henry [15] propose two additions to the existing CK suite of metrics. Message Passing Coupling (MPC) is the number of messages (method invocations) a class sends to other classes. Paul Goodman [14] conducted a comparative analysis of the efficiency of the process and the product metrics for defect prediction and showed that process metrics perform definitely better. They also reported that pre-release defects can be successfully used in predicting post-release defects.

III. PROBLEM SPECIFICATION

Many researchers proposed various level of object oriented metrics which used to measure the object oriented systems based on their usage. The important concepts of OO are inheritance (reusability), coupling (message passing, complexity), Cohesion (Maintainability, complexity). Many parameters are based on a particular measurements like coupling between objects, number of children, data abstraction coupling etc., While searching for object oriented metrics we find a particular metrics parameter and many are scattered. This paper mainly aims at collecting all those important parameters, combining it and presenting in a single paper.

IV. OO METRICS

Object-oriented technologies emerge to support major applications [8]. Object oriented software development need a different approach from more traditional functional decomposition and data flow development methods.

A. MOOSE metric suite

Chidamber and Kemerer (CK) et al. [6] proposed some metrics that have generated a significant amount of interest and are currently the most well known object-oriented suite of measurements for Object-Oriented software. The CK metrics suite consists of six metrics used to assess different characteristics of the object-oriented design are tabulated in table I

TABLE I. MOOSE PARAMETERS [6]

S.No	Parameter	Acronym	Description
1	Weighted Method per class	WMC	sum of complexity of the methods in a class
2	Depth of Inheritance Tree	DIT	To find the length of the maximum path from the root node to the end node of the tree.
3	Number of children	NOC	Immediate sub class coordinated by the class in the form of class hierarchy
4	Coupling between Objects	CBO	To count the number of the class to which the specific class is coupled.
5	Response for class	RFC	Defined as set of methods that can be executed in response and messages received a message by the object of that class.
6	Lack of Cohesion in Methods	LCOM	Count the number of disjoints methods pairs minus the number of similar method pairs used.

B. EMOOSE Metrics

W.Li et al. [11,15] proposed this metrics of the EMoose model. They may be described in the table II.

TABLE II. EMOOSE PARAMETERS [11,15]

S.No	Parameter	Acronym	Description
1	Message Pass Coupling	MPC	The number of message that can be sent by the class operations.
2	Data Abstraction Coupling	DAC	Count the number of classes which an aggregated to current class and also defined the data abstraction coupling.
3	Number of Methods	NOM	To count the number of operations that are local to the class

C. MOOD Metrics

B.F. Abreu et al. [1] defined MOOD (Metrics for Object-Oriented Design) metrics. MOOD refers a structural model of the object oriented paradigm like encapsulation as (MHF, AHF), inheritance (MIF, AIF), polymorphism (POF), and message passing (COF) which are depicted in table III.

TABLE III. MOOD PARAMETERS [1]

S.No	Parameter	Acronym	Description
1	Method Hiding Factor	MHF	ratio of the sum of the invisibilities of all methods defined in all classes to the total number of methods defined in the system under consideration.
2	Attribute Hiding Factor	AHF	ratio of the sum of the invisibilities of all attributes defined in all classes to the total number of attributes defined in the system under consideration.
3	Method Inheritance Factor	MIF	ratio of the sum of the inherited methods in all classes of the system under consideration to the total number of available methods for all classes.
4	Attribute Inheritance Factor	AIF	ratio of the sum of inherited attributes in all classes of the system under consideration to the total number of available attributes for all classes.
5	Polymorphism Factor	POF	Ratio of the actual number of possible different polymorphic situation for class Ci to the maximum number of possible distinct polymorphic situations for class Ci.
6	Coupling Factor	COF	ratio of the maximum possible number of couplings in the system to the actual number of couplings not imputable to inheritance.

D. QMOOD Metrics

The whole description for QMOOD can be get from the Bansiya's [3] thesis through which, The QMOOD metrics can further classified into two measures are shown in the Table IV

TABLE IV. QMOOD PARAMETERS [3]

Measure	Acronym	Description
System Measures	DSC	Design Size in Metrics
	NOH	Number of Hierarchies
	NIC	Number of Independent classes
	NSI	Number of Single Inheritance
	NMI	Number of multiple Inheritance
	NNC	Number of Internal Classes
	NAC	Number of Abstract Classes
	NLC	Number of Leaf Classes
	ADI	Average Depth of Inheritance
	AWI	Average Width of Classes
	ANA	Average Number of Ancestors
Class Measures	MFM	Measure of Functional Modularity
	MFA	Measure of Functional Abstraction
	MAA	Measure of Attribute Abstraction
	MAT	Measure of Abstraction
	MOA	Measure of Aggregation
	MOS	Measure of Association
	MRM	Modeled Relationship Measure
	DAM	Data Access Metrics
	OAM	Operation Access Metrics
	MAM	Member Access Metrics

E. Chen Metrics

Chen [5] proposed metrics with 8 unique measurements. They are tabulated in table V. Metrics 1 through 3 are subjective in nature; metrics 4 through 7 involve counts of features; and metric 8 is a boolean (0 or 1) indicator metric.

TABLE V. CHEN PARAMETERS [5]

S.No	Parameter	Acronym
1	Class Coupling Metric	CCM
2	Operating Complexity Metric	OXM
3	Operating Argument Complexity Metric	OACM
4	Attribute Complexity Metric	ACM
5	Operating Coupling Metric	OCM
6	Cohesion Metric	CM
7	Class Hierarchy of Method	CHM
8	Reuse Metric	RM

V. CONCLUSION AND FUTURE WORK

This contribution of this survey is to understand the various parameters of object oriented metrics. Tables are provided for comparing all the object oriented software metrics which define all the methods, attributes are used in software engineering environment. The increase in software development means the measurement was also so high. The increasing significance being placed software measurement which has to lead and increase amount of research on developing the new software measures. In this paper, we have presented some of the software metrics for object oriented development. They provided a basis for measuring all of the characteristics like size, complexity, performance and quality. In rely of some notions the quality may be increased by added some features like abstraction, polymorphism and inheritance which are inherent in object orientation. This paper

provides some help for researchers and practitioners for better understanding and selection of object oriented metrics for their purposes.

REFERENCES

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

[2] Amit Sharma, Sanjay Kumar Dubey, **"**Comparison of Software Quality Metrics for Object-Oriented System and Analysis" , ISSTA'04, 86–96, 2004.

[3] 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.

[4] Briand, L.C., J.W. Daly, and J.K. Wust, "A Unified Framework for Coupling in Object-Oriented Systems", IEEE Transactions on Software Engineering, v.25(1): p. 91-121, 1999.

[5] Chen, J-Y., Lum, J-F.: "A New Metrics for Object-Oriented Design.", Information of Software Technology 35,4(April 1993) :232-240

[6] Chidamber, Shyam , Kemerer, Chris F. "A Metric s Suite for Object-Oriented Design.", M.I.T. Sloan School of Management E53-315, 1993

[7] El Emam, K., "A Methodology for Validating Software Product Metrics", National Research Council of Canada, Ottawa, Ontario, Canada NCR/ERC-1076, June 2000 June 2000.

[8] H.Olague, L.Etzkorn, S.Gholston and S.Quattlebaum, "Empirical in source code files with decision tree learners," in the 2006 incidence using software change history", IEEE Trans. Softw. Eng., International Conference on Dependability of Computer Systems International Workshop on Mining Software Repositories, 2006. Iterative or Agile Software Development Processes," IEEE

[9] Harrison, R., Counsell, S. J., and Nithi, R. V., "An Evaluation of the MOOD Set of Object-Oriented Software Metrics," IEEE Transactions on Software Engineering, vol. 24, pp. 491-496, June 1998.

[10] Hitz. M and Montazeri. B, "Measuring coupling and cohesion in objectoriented systems", In Proceedings of the International Symposium on Applied Corporate Computing, Monterrey, Mexico, 1995.

[11] Li. W. and Henry. S, "Object-oriented metrics that predict maintainability", Journal of Systems and Software , 1993, 23:111-122

[12] M. Alshayeb and Li.W.,"An empirical validation of object-oriented metrics in two different iteration software processes", IEEE transactionod Software Engineering, Vol-29, no-.11, Nov 2003.

[13] M.Subramanyam and R.Krishnan: "Emphirical Analysis of CK metrics for OOD complexity: Implication for software defect", IEEE

[14] Paul Goodman, "Practical Implementation of software Metrics", McGraw Hill Book Company, UK, 1993. Proneness of Object-Oriented Classes Developed Using Highly

[15] W. Li, Sallie, Henry "Metrics for Object-Oriented system", Transactions on Software Engineering, 1995.