DOI: http://dx.doi.org/10.26483/ijarcs.v9i2.5836

ISSN No. 0976-5697

Volume 9, No. 2, March-April 2018

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

A COMPREHENSIVE INTERPRETATION OF OBJECT ORIENTED METRICS FOR QUALITY REFINEMENT IN SOFTWARE DEVELOPMENT

Mohit Kumar Sharma
Research Scholar
Computer Science
GNA University, Phagwara, India

Dr. Shailesh Kumar
Associate Professor
Faculty of Computational Science
GNA University, Phagwara, India

Dr. Amardeep Gupta
Principal
J.C. D.A.V. College
Dasuya, Punjab, India

Abstract: In Digital era, Software in electronic devices has become an indispensable to our daily life. Software Metric is a significant software engineering field that plays a quality role in software measurement. Better measurement and metrics are stepping stone to software growth with distinction. Moving from measurement to metrics is like moving from observation to understanding. Metrics are conceived by the user and designed to reveal a chosen characteristic in a reliable and meaning manner. Object Oriented Software is based on approach that works around the real-world entities and their characteristics. Object Oriented Software measurement is procedure in which calculations are done on real world entities to describe them according to clearly defined rules. Object Oriented Metric plays a vital role to find the efficiency of the software and improvement for future. The measurement of object oriented software seems to be a powerful tool for product effectiveness. This paper will analyze different Object Oriented Metrics and helpful for ensuring quality design with high cohesion and low coupling for advancement in Object Oriented Software Development.

Keywords: Software Measurement; Object Oriented Metrics; Analysis; Design; Quality Factors

I. INTRODUCTION

Software Development is a salient phase in software life cycle that is to be created as per user specification requirements and it can be assessed for well-engineered quality product against predefined criteria [1]. Software Design is a backbone of four major areas of concern - data, architecture, interfaces and components [2]. Software design and development process is very necessary step of software development life cycle. The emphasis in design is on quality and this phase provides depiction of software that can be assessed for well-engineered quality [3].

Software Quality is a mechanism that evaluates, assesses, and improves the accomplishment of software. Software quality is elucidating as the degree to which software meets requirements for reliability, maintainability, portability as contrasted with functional, performance, and interface requirements that are satisfied as a result of software engineering [4,10]. In the past decade, many IT companies have started to deploy object oriented technology in their software development efforts. Object Oriented Software Development is concept of the real-world entities and their features creation instead of functions involved in the software. Objects have their own internal data structure which defines their data and functions. Object Oriented Design restrained all the properties and worth of software that is allied to any large or small project.

Software Metric is a measurement term of a degree to which a software system holds some characteristics. These metrics are based on actual project experiences; these are not law of nature. These are guidelines that give indication of the progress that a project has made and the quality of the design [5]. Moving from measurement to metrics is like moving from observation to understanding. Metrics are conceived by the user and designed to reveal a chosen characteristic in a reliable and meaningful manner [6]. Object Oriented Metrics concentrate on measurement that can be applied to the class and the design characteristics as Localization, Encapsulation, Inheritance, Information hiding, Polymorphism, Messaging and Object abstraction [7]. There is exigency for Object Oriented Metrics due to visibility, planning and control, quality, productivity [8]. Software quality and reliability describes as fault-free software operation for a specified period of time [9].

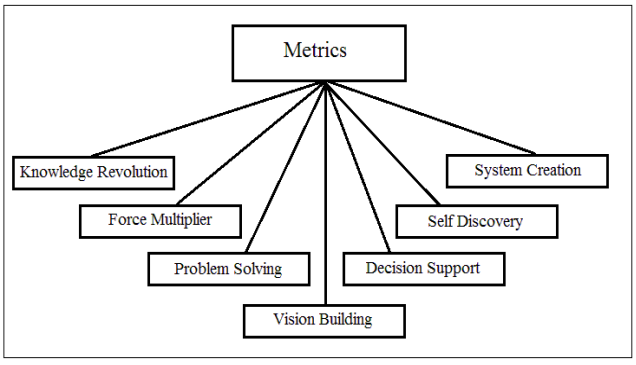

Fig. 1 – Significance of Metrics [6]

Software Metric recommends for project managers to find proficiency of the software. This is done by collecting quality, productivity and effectiveness of data and then analyzing and comparing these data with past averages in order to find whether quality improvements have occurred [10]. Different Object Oriented metrics have been proposed by various researchers and Metric Quality Design suggested in this work based on object oriented metric interpretation to deliver quality product.

II. LITERATURE REVIEW

Lorenz, M., et al. explored metrics based on static snap-shots of the system at a point in time. None of them are currently based on the runtime execution of the system. Any explicit or implied use of the word dynamic refers to the changes in the state of project measurements over multiple static snapshots. These metrics should be used to support the desired motivation [5].

Chidamber, S.R., et al. proposed six object oriented software metrics and these are the intellectual research for object oriented software development. These metrics are utilized with a theoretical idea in measurement of objects, and which incorporate the experiences for software managers and evaluation done by these metrics against criteria for credibility and existing empirical data from commercial software to explain the features of metrics on real world applications and recommends the process in which these metrics may be utilized [11].

Xenos, M., et al. explained the outcome derived from survey on software metrics used in object oriented environments and covers a small set of the most known and commonly used traditional software metrics which could be applied to object oriented programming and collection of metrics. These metrics interpreted with present meta metrics based on the interpreter's point of view, and used implementation in three languages as Object Pascal, C++ and Java [12].

Aggarwal, K.K., et al. conducted an analysis of 22 metrics given by different researchers and utilized these metrics on software for empirical study. The metrics are first defined and then explained using practical applications. They have applied on standard projects on the basis of which descriptive statistics, principal component analysis and correlation analysis was done [13].

Rao, N.S. in thesis, a set of object oriented reuse metrics have been proposed at class level namely Method Reuse Factor, Attribute Reuse Factor, Method Reuse Factor with Inheritance and Attribute Reuse Factor with Inheritance. This research work explored on process and product to improve the process level measurement, Object Oriented DFP has been chosen and the attribute reuse factor with inheritance has been applied on the OODFP and it has been proved by incorporating [14].

Kayarvizhy, N., et al. explored on the design of an automated object oriented software metric tool which has a generic framework for computing the metrics automatically. This software tool converts the source code developed using a particular object oriented programming to a language independent XML format which is then used for computing the needy software metrics [15].

Dubey, S.K., et al. interpreted object oriented software metrics ensuring to minimize maintenance cost effort to estimate software failures and errors. They reviewed object oriented software metrics and analysis table is explored by which evaluation of the comparison between all various software metrics done efficiently [16].

Gulia, P. in thesis, contributes to a better understanding of object oriented system, its measurement and testing. Firstly, the proposed work provides a theoretical framework to the measurement and testing of object oriented system. This framework is helpful for designing appropriate research studies to know the objectives and make decisions on the basis of these objectives [17].

Huang, R., et al. research seeks to describe concepts and techniques to improve the quality of the object-oriented System. Many metrics relating to product quality have proved their value for system maintenance and modification. The object-oriented metrics criteria selected are used to evaluate the system attributes [18].

Kajla, P. in thesis, recommended a new metric as Nested Class Complexity Metric (NCCM) to find the complexity of nested classes and outcome is compared with present available metrics. Nested classes are the general requirement in the programming like Java. These programming languages facilitate Nested Methods into the method of the same class [19].

Nicolaescu A., et al. Coupling is one of the most important properties that affect the quality of the design and implementation of a software system. In this work, review and critically analyze the developments in this domain by considering the most influential research addressing object oriented software coupling [20].

Li, W. et al., software metrics have been analyzed in the procedural paradigm as a quantitative means of assessing the software development process as well as the software quality of products. Many studies have validated that various software metrics are important indicators of maintenance effort in the procedural paradigm [21].

Kumar, L., et al. empirically found the association of present class level metrics with a software quality factor as maintainability. Various object oriented metrics has been utilized to give required input to design the models for forecasting maintainability quality factor using Neuro-Genetic algorithm [22].

III. PROBLEM STATEMENT

Failure rates and software maintenance cost have been extended due to different problems of inefficiency and non-reliability. Complexity has been constituted in object oriented software's due to high-coupled and low cohesive design. Further effective metric tools, techniques, design and theory are mandatory to utilize through actual use of software metrics to attain quality intention with perfection and reliability in software growth.

IV. OBJECTIVES OF THE STUDY

A. To interpret and scrutinize various Object Oriented Metrics to clear ideation of computation opinions and to explore the base for software quality framework.

B. To propose Metric Quality Design for productive software measurement tool to ensure standard and exposition of factors influencing quality of Object Oriented Software Development.

V. RESEARCH METHODOLOGY

Analytical research utilized for literature interpretation of existing object oriented metrics proposed by different researchers, and make an evaluation of the metric quality design for software growth. In analytical research, facts and information already available and analyze them to

interpret results for future work. Conceptual and Experimental research is also used for generation of design framework and to reinterpret existing ones with results. Data sources for research are reputed research papers, reference books, review papers, thesis, object oriented software's testing tools and internet.

VI. INTERPRETATION OF OBJECT ORIENTED METRICS FOR SOFTWARE COMPUTATION

Kenneth Morris research in thesis as "Metrics for Object Oriented Software Development Environments (1989)", due to restricted literature on object oriented software design and coding at that time his research was centered around productivity software metrics instead of software complexity metrics. The outcome of the literature review called "productivity impact variables". These are Maintainability, Reusability, Extensibility, Testability, Comprehensibility, Reliability and Author ability.

Object Oriented Metrics by Shyam R. Chidamber and Chris F. Kemerer is the intellectual research in object oriented design analysis in 1994. These software metrics are also called as "C-K metrics" [11].

Weyuker's 2nd property and 8th property is instinctive utilized for different object oriented software metrics. They have explained six object oriented metrics for software measurement and computation [11,23].

Metric :1 Weighted Methods per Class (WMC)

WMC is computed as a total of the complexities of all different class functions or methods. Let a class M1 with class functions or methods N1, N2, . . ., Nn and c1 . . . cn be the static complexity of class function or method.
Then

$$\text{Weighted Method per Class} = \sum_{i=1}^{n} ci$$

All the functions of WMC are deemed to be uniform, then WMC = n, then no of functions are weighted methods per class.

Metric :2 Depth of Inheritance Tree (DIT)

DIT is calculated as the largest length from the node to the root of the tree in class.

In project's, when Depth of Inheritance is maximum, then probability of failure and errors occurrence will be maximum.

DIT metric meets the Weyuker's properties 1,2,3,4,5 and property 6 is not fulfilled by DIT metric.

Metric :3 Number of Children (NOC)

NOC is computed as total of immediate inherited classes subordinated to a class. NOC metric is a computation of no of derived classes that are going to inherit the methods of the super class. The reusability factor maximum with increase of NOC metric.

Metric :4 Coupling between object classes (CBO)

Coupling explores communication between these object modules of classes. When different class methods declared in one class use methods or instance variables of the other class are said to be coupled. High level of CBO metric computation shows weak outcomes and more rework effort is required to do.

Metric :5 Lack of Cohesion in Methods (LCOM)

Cohesion describes that object modules should be independent to process. Lack of Cohesion (LCOM) compute the dissimilarity of functions or methods in a class by instance attributes. High cohesion indicates better class subdivision. Lack of Cohesion metric is measure the no of method pairs whose similarity is 0 minus the method pairs whose similarity is not zero.

Metric :6 Response for a Class (RFC)

It is explained as number of class methods in the set of all class methods that can be called in result or response to a message sent to an object of a class. Response of a class includes methods invoked from outside the class. It is a computation of association between class and other classes. If value of RFC metric is maximum in project's, then testing of class become much complicated.

Table 1 – C-K Metrics Analysis

Sr. No.	C-K Metric	Object Oriented Attribute
1	Weighted Methods per Class (WMC)	Class
2	Depth of Inheritance Tree (DIT)	Inheritance
3	Number of Children (NOC)	Inheritance
4	Coupling between object classes (CBO)	Coupling
5	Lack of Cohesion in Methods (LCOM)	Cohesion
6	Response for a Class (RFC)	Class

Metrics for Object Oriented Design (MOOD) by F. Brito e Abreu and explored a basic structural procedure of the object oriented attributes as encapsulation (MHF and AHF), inheritance (MIF and AIF), polymorphisms (PF), message passing (CF) are presented as important quotients. MOOD includes the six different metrics used for software and the metrics operate at system level, providing an overall performance of a software [23,24].

Metric :1 Method Hiding Factor (MHF)

Method Hiding Factor describes as ratio of the addition of the invisibilities of all functions or methods declared in all classes to the total number of functions or methods declared in the software under consideration.

Metric :2 Attribute Hiding Factor (AHF)

Attribute Hiding Factor explored as the ratio of total of the invisibilities of all class attributes to the total number of class attributes declared in the software under consideration [25].

Metric :3 Method Inheritance Factor (MIF)

Method Inheritance Factor is the ratio of the total of the inherited functions or methods in all classes of the software under consideration to the total number of available all classes' functions.

Metric :4 Attribute Inheritance Factor (AIF)

Attribute Inheritance Factor is the ratio of the total addition of all class inherited attributes of the software under consideration to the total numbers of available all classes attributes.

Metric :5 Polymorphism Factor (PF)

Polymorphism Factor states as the ratio of the actual number of different possible polymorphic situation for class Mi to the maximum number of possible distinct polymorphic situations for class Mi.

Metric :6 Coupling Factor (CF)

Coupling Factor describes as the ratio of the maximum possible number of couplings in the software to the actual number of couplings not imputable to inheritance.

Table 2 – MOOD Analysis

Sr. No.	Metrics for Object Oriented Design	Object Oriented Attribute
1	Method Hiding Factor (MHF)	Information Hiding
2	Attribute Hiding Factor (AHF)	Encapsulation
3	Method Inheritance Factor (MIF)	Inheritance
4	Attribute Inheritance Factor (AIF)	Inheritance
5	Polymorphism Factor (PF)	Polymorphism
6	Coupling Factor (CF)	Coupling

Lorenz and Kidd described six object oriented metrics to find assessment of object oriented software quality. These object oriented metrics are classified into three types as size, inheritance, internal and external based measurement.

Size based metrics for class points on addition of class attributes and class operations. Inheritance based metrics points on the class function or method in which operations are reused through class hierarchy. Internal based metrics follows cohesion and code oriented issue of a class. External metrics focus on coupling and reuse of class [5,23].

Metric :1 Attribute Count (AC)

Attribute Count Software metric is computation of total number of class attributes including both inherited class attributes and the attributes defined in the class are computed.

Metric :2 Method Count (MC)

Method Count Software metric is computation of total number of class methods including both inherited class methods and the methods defined in the class are computed.

Metric :3 Number of Operation Overridden by Subclass (NOO)

Number of Operation Overridden by subclass software metric is computation of total number of subclass operations overridden. Maximum values of this software metric states that there is a fault in software design and outcomes in unique new method names.

Metric :4 Specialization Index (SI)

Specialization Index software metric is based on indication of degree of specialization of every subclass in software. This can be computed by adding and deleting operations by overriding.

Metric :5 Number of Operations Added be a Subclass (NOA)

This software metric is computed by total number of class specialized functions or methods and addition of subclass attributes.

Metric :6 Average Number of Parameters per method (ANP)

This software metrics is mean of the number of values passed to each of class methods and mean number of values per operation should be kept minimum.

Table 3 – L & K Metrics Analysis

Sr. No.	L & K Metrics	Object Oriented Attribute
1	Attribute Count (AC)	Class
2	Method Count (MC)	Class
3	Number of Operation Overridden by Subclass (NOO)	Inheritance
4	Specialization Index (SI)	Inheritance
5	Number of Operations Added be a Subclass (NOA)	Inheritance
6	Average Number of Parameters per method (ANP)	Class

VII. METRIC QUALITY DESIGN FOR REFINEMENT IN OBJECT ORIENTED SOFTWARE DEVELOPMENT

Better measurements and better metrics are quality stone to software growth with distinction [26]. Object Oriented Metrics are significant for ensuring software quality and used by software development companies [27,32]. Different object oriented metrics proposed by various researchers to measure the software productiveness and efficiency. Metric Quality Design proposed based on object oriented metrics comprehensive interpretation to deliver standard software. Different metric consequences are discussed in this and considered for development of object oriented software and results explored. This design is suggested for measurement tools and software testing team, to consider these consequences while developing software. High cohesive and low coupling design is important for object oriented software development [28,33]. Cohesion describes that object modules should be independent to process [29]. Coupling explores communication between these object modules [34]. Metric testing tool can be developed based on this framework for object oriented software measurement and quality results.

Table 4 – Metrics Results for Object Oriented Software Development

Sr. No.	Metrics Consequences	Results
1	Decision making	Metrics are helpful to software engineers to take correct decision for software development and maintenance.
2	Significant Improvement of software	Metrics makes important changes in software to update new version based on user requirements and needs.
3	Reliability as failure free software	Metrics ensures reliability as error free software to customer.
4	Estimating the effort involved for maintenance	Metrics are helpful in estimating cost and feasibility of software product as well as for maintenance.
5	Quality of the design	Metrics are used for making good quality software based on correct measurements.
6	Measure complexity of the code	Metrics ensures to measure complexity level of software to make it high cohesive and low coupled design.
7	Prediction of faults	Metrics helps to predict failures and errors to make accurate software code.
8	Process efficiency	Metrics makes software development process efficient as per correct user requirements.
9	Product Effectiveness	Metrics ensures quality product by removing ambiguities and flaws in design.

In fig 2. Metric Quality Design framework suggested with different components for metrics tool generation and metric consequences interpreted by software testing team for quality of object oriented design. Team will comprehensively analyze the factors of quality and consider for design of software. Object and module analysis done on different real world entities on source code and software module analysis done on whole code, after this object interface analysis done for measuring coupling between objects. All inputs to object oriented metric analyzer for comprehensive evaluation of quality of software with high cohesion and low coupling.

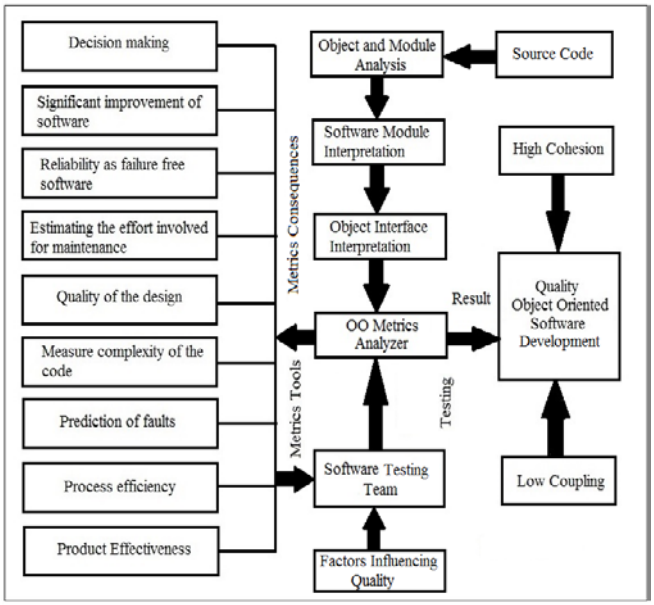

Fig. 2 – Metric Quality Design Framework

Object Oriented Software Quality is a procedure that evaluates, assesses, and improves the performance of real world entity applications [30,31].

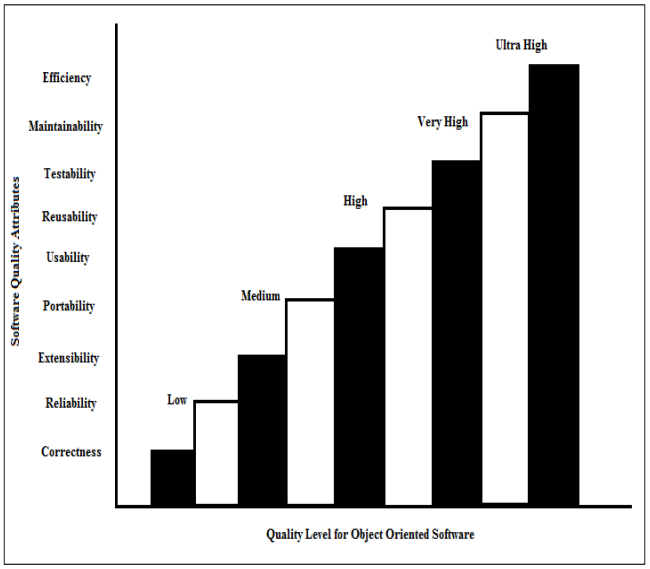

Fig. 3 – Software Quality Levels

It is defined as the degree to which software meets requirements for reliability, maintainability, portability as contrasted with functional, performance, and interface requirements that are satisfied as a result of software engineering in context of real world entities i.e. objects [35,37]. The McCall et al., factor model provides a software quality factors, practical, up-to-date method for classifying software requirements. In fig. 3 different software quality

attributes explored with quality levels for object oriented software development. Low and medium quality level ensures low quality software due to limited features. High and very high quality level ensures good software quality, but ultra-high quality level ensures excellent software quality including all features related to software in future use. It involves user requirements and performance requirements, documentation and all well engineered qualities required for object oriented software [36,38]. Object Oriented Testing describes degree to which a user requirement is explained in terms that permit establishment of test criteria and the performance of tests to determine whether those criteria met [39,40].

Table 5 – Analysis of Quality factors influencing Software Development

Sr. No.	Factors	Results
1	Correctness	Correctness deals to which a program satisfies its user requirement specifications. If software does not work as required, it has no worth.
2	Reliability	Reliability defines how well the software meets its requirements as failure free software. If software is not error free, it is un-reliable.
3	Extensibility	Extensibility is the ability of the software to be upgraded beyond the functionality of the original software. If software coding is not able to update by adding new features, it is wasteful.
4	Portability	Portability describes requirement to transfer the software from one configuration machine to another machine. If software is not able to install any other machine, it is has no worth.
5	Usability	Usability or the effort required locating and fixing failures in programs. If software is not user friendly, then it is difficult for user to handle.
6	Reusability	Reusability is the extent to which parts of the software can be reused in other related software. If software is not reusable, it is waste.
7	Testability	Testability is required to test to ensure that the software performs its required function. If testing is not able to do, it makes software failures.
8	Maintainability	Maintainability is the effort required to update the software. If software is not able to add new features, it is wasteful.
9	Efficiency	Efficiency is concerned with all issues in the execution of software; it includes response time, memory requirement, and throughput of system fast. If software is not making computer fast, it is wasteful.

VIII. CONCLUSION AND FUTURE WORK

Object Oriented Metrics performed a vital role in software measurement and in helping software managers to understand design and development of object oriented software. Object Oriented Software Development helps in designing and creating all real world entities and use of these metrics to evaluate quality improvements. Failure rates and Maintenance cost have been increased due to complexity and unreliability. There is need of high cohesion and low coupling in object oriented software to ensure reliability and effectiveness. This work has analyzed different object oriented metrics given by different researchers for improving further quality of design. Further Metric Quality Design framework suggested based on metrics interpretation to deliver object oriented quality software. In this design, different metric consequences discussed and considered for quality growth of object oriented software and results explored. In future research work can be to develop productive software metric tool or analyzer based on this framework for improvement of time schedule availability, reduce maintenance cost, enhance reliability and standard quality growth.

REFERENCES

[1] Roger S. Pressman, *Software Engineering*, Tata-McGraw Hill Publishing House, 2009

[2] Ali Behforooz, *Software Engineering Fundamentals*, Oxford University Press. Frederick J.H, 1996

[3] E.M. Awad, *Systems Analysis and Design*, Galgotia Publications Ltd, 1993

[4] Anirban Basu, *Software Quality Assurance and Testing*, Prentice Hall of India, 2015

[5] Mark Lorenz and Jeff Kidd, *Object-Oriented Software Metrics*, Pearson Education, 1994

[6] C. Ranindranath Pandian., *Software Metrics*, Auerbach Publications CRC, 2011

[7] Stephen H. Kan, *Metrics and Models in Software Quality Engineering*, Pearson Education, 2nd ed. 2016

[8] Norman Fenton and James Bieman, *Software Metrics in Software Engg.*, CRC Press, Chapman Hall Book, 2014

[9] Glenford J. Myers, *Software Reliability Principles*, John Wiley & Sons, 2002

[10] Nasib S. Gill, *Software Engineering-Software Reliability and Testing*, K.B. Publishing, 2016

[11] Shyam R. Chidamber and Chris F. Kemerer, "A Metrics Suite for Object Oriented Design", *IEEE Transactions for Software Engineering*, Vol. 20 No. 6, pp. 476-493, June 1994

[12] M. Xenos, D. Stavrinoudis, K. Zikouli and D. Chistodoulakis, "Object Oriented Metrics – A Survey", *Proceedings of the FESMA 2000*, Fedration of European Software Measurement Associations, Madrid, Spain, 2000

[13] K.K.Aggarwal, Yogesh Singh, Arvinder Kaur and Ruchika Malhotra, "Empirical Study of Object Oriented Metrics", *Journal of Object Technology, ETH Zurich*, Vol. 5, No. 8, pp. 149-173, November-December 2006

[14] N. Sambasiva Rao, "Software Reuse Metrics for Object Oriented Systems", *Ph.D. Thesis*, Anna University, Chennai, July 2007

[15] N. Kayarvizhy and S. Kanmani, "An Automated Tool for Computing Object Oriented Metrics Using XML", *Springer-Verlag Berlin Heidelberg, A. Abraham et al. (Eds.):* ACC 2011, Part II, CCIS 191, pp. 69–79, 2011

[16] Sanjay Kumar Dubey, Amit Sharma and Dr. Ajay Rana, "Comparison Study and Review on Object- Oriented Metrics", *Global Journal of Computer Science and Technology*, Volume 12 Issue 7 Version 1.0, Global Journals Inc. (USA) ISSN: 0975-4350, April 2012

[17] Preeti Gulia, "Analysis and Design of Object Oriented Complexity Metrics and Test Cases", *Ph.D. Thesis*, Maharshi Dayanand University Rohtak, 2012

[18] Rui Huang, Mingyu Li, and Zhang Li, "Research of Improving the Quality of the Object-Oriented System", *International Journal of Information and Education Technology*, Vol. 3, No. 4, August 2013

[19] Parveen Kajla, "Study and Design of Object-Oriented and Component-Based Metrics", *Ph.D. Thesis*, Maharshi Dayanand University, Rohtak, Haryana, February, 2014

[20] Ana Nicolaescu, Horst Lichter and Yi Xu, "Evolution of Object Oriented Coupling Metrics: A Sampling of 25 Years of Research", *Software Architecture and Metrics (SAM), 2015 IEEE/ACM 2nd International Workshop on*, pp.48-54, 16-16 May 2015
[21] Wei Li and Sallie Henry, "Object-Oriented Metrics that Predict Maintainability", *Journal of Systems Software, Elsevier Science Publishing Co., Inc. 655 Avenue of the Americas, New York*, pp. 111-121, 1993; 23:111-122
[22] Lov Kumar, Debendra Kumar Naik, Santanu Ku. Rath, "Validating the Effectiveness of Object-Oriented Metrics for Predicting Maintainability", *Third International Conference on Recent Trends in Computing (ICRTC' 2015), Elsevier Procedia Computer Science* 57 (2015) 798 – 806
[23] K.P. Srinivasan and T. Devi, "A Comprehensive Review and Analysis on Object-Oriented Software Metrics in Software Measurement", *International Journal on Computer Science and Engineering*, ISSN: 0975-3397 Vol. 6 No.07 Jul 2014
[24] Basili, V.R., Briand, L.C., and Melo, W.L., "A Validation of Object-Oriented Design Metrics as Quality Indicators", *IEEE Transactions on Software Engineering*, Vol. 22, No. 10, October 1996, pp. 751-761
[25] Churcher, N.I., and Sheppard, M.J., "Comments on a Metrics Suite for Object-Oriented Design", *IEEE Transactions on Software Engineering*, Vol.21, No.3, March 1995, pp. 263-265.
[26] M. Thirugnanam, and J.N. Swathi, "Quality Metrics Tool for Object Oriented Programming", *International Journal of Computer Theory and Engineering*, Vol. 2, No. 5, October 2010, pp. 712-717
[27] Pushpa R. Suri and Harsha Singhani, "Object Oriented Software Testability (OOST) Metrics Analysis", *International Journal of Computer Applications Technology and Research*, Volume 4– Issue 5, 359 - 367, 2015, ISSN: - 2319–8656
[28] M. Badri, "Empirical Analysis of Object-Oriented Design Metrics for Predicting Unit Testing Effort of Classes," *Journal of Software Engineering Appl.*, vol. 05, no. July, pp. 513–526, 2012.
[29] Amit Kumar, Sanjiv Kumar and Rohit Sharma, "Evolution of Object Oriented Analysis & Design in Software Engineering", *International Journal of Advanced Research in Computer Science*, 3 (3), May –June, 2012,456-463
[30] Vipin Saxena and Santosh Kumar, "Performance Computation Metrics for Object-Oriented Software Systems", *International Journal of Advanced Research in Computer Science*, 3 (6), Nov, 2012, (Special Issue), 7-10, ISSN No. 0976-5697
[31] Manik Sharma, Gurdev Singh, Anish Arora and Parneet Kaur, "A Comparative Study of Static Object Oriented Metrics" *International Journal of Advancements in Technology*, ISSN 0976-4860
[32] S. Pasupathy and R. Bhavani, "Object Oriented Metrics Evaluation", *International Journal of Computer Applications (0975 – 8887)* Volume78– No.1, September 2013
[33] R.V. Krishnaiah and Banda Shiva Prasad, "Analysis of Object Oriented Metrics", *International Journal of Computational Engineering Research (ijceronline.com)* Vol. 2 Issue. 5
[34] Brij Mohan Goel and Satinder Bal Gupta, "Dynamic Coupling Based Performance Analysis of Object Oriented Systems", *International Journal of Advanced Research in Computer Science, 8 (5)*, May-June 2017, 112-115, ISSN No. 0976-5697
[35] Mehmet Aksit and Lodewijk Bergmans, "Obstacles in Object-Oriented Software Development", *TRESE project University of Twenty Faculty of Computer Science Enschede*, The Netherland
[36] Jacobson, I., Christerson, M., Jonsson, P., and Overgaard G.: *Object-Oriented Software Engineering: A Use-Case Driven Approach*, Addison-Wesley, 1992
[37] Wirfs-Brock, R., Wilkerson, B., Wiener, L., *Designing Object Oriented Software*, Prentice-Hall, Englewood Cliffs, N.J., 1990
[38] Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F., Lorensen, W., *Object Oriented Modelling and Design*, Prentice-Hall, Englewood Cliffs, N.J., 1991
[39] Coad, P. and Yourdon, E. 1991. *Object-Oriented Analysis*, Prentice Hall, Englewood Cliffs, N.J. ASDS
[40] B. Henderson and Sellers, *Object-Oriented Metric*, New Jersey: Prentice Hall, 1996

ABOUT AUTHORS

Mohit Kumar Sharma is a Research Scholar of Computer Science with Faculty of Computational Science, GNA University, Phagwara, India. He is working as Sr. Assistant Professor (HOD), Post Graduate Department of Computer Science, J.C. D.A.V. College, Dasuya, Punjab, India. He received his M.Phil. (Computer Sc.) degree from Madurai Kamaraj University, India. He has 10 years' teaching and 2 years' research experience. He has 5 research paper publications in international journals, conferences and 3 book publications. His research interest is in Software Engineering and Object Oriented Design and Analysis.

Dr. Shailesh Kumar is an Associate Professor of Computer Science with the Faculty of Computational Science, GNA University, Phagwara, India. He received his Ph.D. (Computer Sc.) degree from Banasthali Vidyapith, India. He has 16 years' teaching and 6 years' research experience. He has more than 25 research paper publications in international journals and conferences. His research interest is in Database Management, Data warehouse and Data Mining, Knowledge Management Systems, Software Quality Engineering and Object Oriented Design.

Dr. Amardeep Gupta is the Principal of J.C. D.A.V. College, Dasuya, India. He received his Ph.D. (Computer Sc.) degree from I. K. Gujral Punjab Technical University, India. He was former Head of the Department, Post Graduate Department of Computer Science, DAV College, Amritsar, Punjab. He has 20 years' teaching and 6 years' research experience. He has more than 32 research paper publications in international journals, conferences and 12 book publications. His research interest is in Parallel Processing and Object Oriented Programming.