International Journal of Applied Engineering Research ISSN 0973-4562 Volume 14, Number 22 (2019) pp. 4161-4172

Object Oriented Dynamic Metrics in Software Development: A Literature Review

Ramesh Ponnala[1],

Research Scholar in Osmania University-Hyderabad and Asst. Professor, Department of MCA, CBIT (A) – Hyderabad, Telangana, India-500075.

Dr.C.R.K.Reddy[2],

Professor and Head, Department of CSE, MGIT–Hyderabad, Telangana, India-500075.

Abstract:

Software development is the process of analyzing, designing, programming, documenting, testing, and bug fixing involved in creating and maintaining applications as per the client requirement. In software application development, Object Oriented Programming (OOP) Approach plays a vital role. Object Oriented (OO) metrics are used to measure properties of object oriented software applications. Many software metrics have been developed for OO paradigms such as abstraction, class, object, inheritance etc. to compute various attributes like software quality, coupling, cohesion etc. OO metrics are classified into static and dynamic. The aim of this paper is to analyze the significance of these OO metrics in software development. The limitations of static and dynamic metrics are discussed further. The importance of dynamic metrics and future research scope for the researchers by combining these two metrics: hybrid metrics is also discussed.

Keywords: Software Engineering, Static Metrics, Dynamic Metrics, Object Oriented Metrics, Hybrid Metrics

I. INTRODUCTION

Software metrics are measurements that are used to measure and characterize the software engineering products. Software metrics are one of the most important tools used in Software Engineering in order to enhance the quality of software applications. Software metric is a simple quantitative measure derivable from any attribute of the software life cycle.

The role of software metrics is to locate significant estimates for software products and direct us in intriguing managerial and technical decisions. Software metrics have grown to be an important section of software development and are utilized during every phase of the software development life cycle. The name software metric is connected with varied measurements of computer software and its development. Research in the region of software metrics tends to concentrate predominantly on static metrics which can be obtained by static analysis of the program artifact.

Structured programming applications concentrate on operations rather than data. However data is closely related to their operations in real world applications. Therefore emphasizing on only operations leads to data insecurity. To overcome this Object Oriented Programming (OOP) paradigm came into picture. OOP concepts are most popular in today's software application development. OOP is used to develop software applications to fulfill the requirements of clients.

In this paper, a detailed study of various object oriented metrics has been carried out. Summary of the major contributions of different authors on OO static as well as dynamic metrics have been presented. Also gaps for the future research work to be carried on OO static and dynamic metrics as hybrid metrics were identified.

II. OBJECT ORIENTED PARADIGM

Booch et al. [4] defines object oriented design to be the process of identifying objects and their attributes, identifying operations required on each object and establishing interfaces between objects. Design of classes involves three steps: first, definition of objects, second data members of objects and third, communication between objects. Class design is therefore at a higher level of abstraction than the conventional data or procedural approach. It is the task of class design that makes OOD different from traditional procedural design.

Some of the basic terms commonly used in object oriented metrics are as follows:

1. *Object*: Object is an entity able to save a state and offers a number of operations to either examine or affect this state.
2. *Message*: It is a request to do an operation by an object on the other object.
3. *Class*: A set of objects that share a common structure and behavior manifested by a set of methods. It serves as a template from which object can be instantiated.
4. *Method*: An operation upon an object, available to all instances of class, need not be unique.

5. *Instantiation*: The process of creating an instance of the object and binding or adding the specific data.
6. *Inheritance*: A relationship among classes, wherein an object in a class acquires characteristics from one or more other classes.
7. *Cohesion*: The degree to which the methods within a class are related to one another.
8. *Coupling*: Object A is coupled to Object B, if and only if A sends a message to B.

Software quality metrics are a part of software metrics that concentrate on the quality areas of the product, process, and project. Generally, software quality metrics are far more closely related to process and product metrics than to project metrics. Nonetheless, the project parameters such as number of developers, their skill levels, schedule, size, and the corporation structure certainly affect the caliber of the product.

A. General Uses of Software Metrics

Software metrics are made to get objective reproducible measurements, which will be useful for quality assurance of the software application. Metrics are used to evaluate performance of software application. They are also used in debugging, management, and estimating cost of the project.

Software metrics must be computed continuously throughout the development process to allow the perfect tracking. Moreover, software metrics must be definable by development teams not only to cover general factors, but also to measure company goals. They are used to full fil project or team specific goals. Team specific goals are like finding defects in code, predicting defective code, predicting project success and predicting project risk etc in software application development.

Software metrics are also used in scheduling software project, size/complexity of software development involved, cost of project, and quality of software etc.

III. OBJECT ORIENTED METRICS

The OOP approach depends on Object Oriented Metrics, which will be useful in the design and implementation phases of Software Applications.

Metrics can be classified into different categories like metrics for analysis model, metrics for design model, metrics for source code, metrics for testing, metrics for quality assurances etc., The major classification of OO metrics are OO Static, Dynamic metrics.

Static metrics are collected from the static artifacts of the software such as specification documents, design diagram and code listings. For example Lines of Code (LOC), Weighted Methods per Class (WMC), Coupling Between Objects (CBO) etc. [Chidamber et al [3]]

Dynamic metrics are evaluated using data collected from the runtime behavior of software. For example Dynamic Coupling, Dynamic Lack of Cohesion and Dynamic Coupling Between Objects etc. [Mitchel et al [15]]

Object oriented static metrics are design metrics that are evaluated from static analysis data of an object oriented software systems in order to find the quality of the software system. Various object oriented metrics are suggested by different researchers.

Chidamber and Kemerer (CK Metrics) [3] are mostly referred metrics; they have defined six metrics WMC, RFC, LCOM, CBO, DIT and NOC. These metrics are for evaluating the complexity in relation to the usability, maintainability, functionality, reliability quality factors. All these metrics aim to evaluate the design of object oriented application, rather than implementation of the system.

Lorenz et al. [5] have given metrics for static aspects of software design, which depend on Class Size, Class Inheritance, Class Internal and emphasis on counts of attributes, operations, reusability of operations in inheritance hierarchy, cohesion oriented in class internal etc.,

MOOD metrics proposed by Abreu et al. [9], are defined to evaluate the design of object oriented design methods like Method Inheritance Factor (MIF), Attribute Inheritance Factor (AIF), Method Hiding Factor (MHF), Attribute Hiding Factor (AHF), Polymorphism Factor (POF) and Coupling Factor (COF).

J. Bansia et al. [13] defined QMOOD metrics. Quality Model for Object Oriented Metrics given as Average Number of Ancestors (ANA), Cohesion Among Methods of Class (CAM), Data Access Metric (DAM), Measure of Aggregation (MOA), Class Interface Size (CIS), Direct Class Coupling (DCC), Measure of Functional Abstraction (MFA), Number of Methods (NOM), Number of Polymorphic Methods (NOP), Number of Hierarchies (NOH), and Design Size of Class (DSC).

All these metrics are specially defined to compute design metrics in the early of design of phase of Software System. The following TABLE I [Hemalatha Sharma et al. [36]] describes the list of Static Metrics, TABLE II [Hemalatha Sharma et al. [36]] describes list of Dynamic Metrics.

TABLE - I: LIST OF STATIC METRICS [HEMALATHA SHARMA ET AL. [36]]

Metric	Description
Weighted Methods per Class (WMC)	The sum of McCabe's Cyclomatic Complexities of all local methods in a class
Depth of Inheritance (DIT)	The metric measures class level in the inheritance tree, root class is considered as zero.
Number of Children (NOC)	It counts number of immediate sub classes of a class in a hierarchy.
Coupling Between Objects (CBO)	It represents the number of classes to which the given class is coupled.
Response For a Class (RFC)	The number of local methods plus the number of non local methods called by local methods.
Lack of Cohesion of Methods (LCOM)	The number of disjoint sets of local methods. Each method in a disjoint set shares at least one instance variable with at least one member of the same set.
Lines of Code (LOC)	The number of lines of code excluding comments.
Average Method Complexity (AMC)	This metric measures the average method size for each class. Change The total number of lines Added, Deleted and Modified in a class.
Afferent Coupling (Ca)	A class's afferent coupling is a measure of how many other classes use the specific class.
Efferent Coupling (Ce)	A class's efferent coupling is a measure of how many other classes is used by the specific class.
Number of Public Methods (NPM)	All methods which have public access specifiers are counted by NPM.
Data Access Metrics (DAM)	The metric computes the ratio of attributes declared as private or protected to the total number of attributes declared in the class.
Measure of Aggregation (MOA)	It measures the HAS-A relationship between attributes at run time.
Measure of Functional Abstraction (MFA)	The metrics computes the count of the number of inherited methods of a class divide by the total number of methods which are accessible by member methods of the class.
Cohesion Among Methods of Class (CAM)	The relevance between the class methods based on the list of specifications of the methods is computed by this metric
Inheritance Coupling (IC)	This metric produces the total number of super classes to which a given class is coupled.
Coupling Between Methods (CBM)	The metric measure the total count of new or redefined methods to which all the inherited methods are coupled.
Average Method Complexity (AMC)	The average method size for each class is measured by AMC, where the size of a method is equivalent to the number of java binary codes in the method.

TABLE- II: LIST OF DYNAIC METRICS [HEMALATHA SHARMA ET AL. [36]]

Metric	Description
Loose Class Coupling (LCC)	This metric calculates the low dependency between object-structure at run-time.
Lack Of Cohesion in Methods (LCOM)	It is used to measure the lack of cohesion between methods at run time. This group of metrics aims to detect problem classes. A high LCOM value means low cohesion. This varies from 0 to 4.
Number of Assertions per KLOC (NAK)	Counts the number of declarations per KLOC at run time.
Number of Children (NOC)	The metric find out the total number of direct sub classes of a class at run time.
Number Of Fields (NOF)	This metric counts the number of data declarations used at run time.
Number Of Methods (NOM)	Total number of objects interacted at run time
Number of Static Fields(NOSF)	It measures the total number of static attributes in the selected scope.
Number of Static Methods (NOSM)	This metric is the sum of methods that are static at run time.
Number of Test Methods (NTM)	Total sum of test methods that are encountered at run time is calculated by this metric.
Tight Class Coupling (TCC)	It measures the tight class coupling is the high dependency between object-structure at run time.
Weighted Method Count (WMC)	It counts run time method complexity. A high WMC reduces the reuse probability for the class.
Lines Of Code (LOC)	This metric find out the total lines of code encountered at run time.
Lines Of Comments (LOCm)	Total number of lines of comments used at runtime.
Change	The total number of lines Added, Deleted and Modified in a class.

IV. OVERVIEW OF VARIOUS OOP METRICS

Chidamber and Kemerer (CK) [1] proposed a Metric Suite containing the most influential coupling metric Coupling between Object Classes (CBO) as well as Response For a Class (RFC). The CBO value of a class is the number of classes that it is using and that it is used by. Hence, a class A is coupled to another class B, if A uses B's methods or attributes or vice versa. CBO doesn't take inheritance between classes into account. The publication of the CK Metric Suite was a landmark with in the development of OO coupling metrics.

Yacoub et al. [11] defined a set of object-level dynamic coupling metrics such as Export Object Coupling (EOC), Import Object Coupling, Object Request for Service (OQFS), Object Response for Service (OPFS), Operational Complexity Metric (OCPX) are designed to evaluate the change-proneness of a design. The metrics were applied at an initial development phase to identify design quality. They have used runtime object-oriented design models to model the application to be tested. The metrics were computed for a number of different execution scenarios at runtime, and extended the scenarios to have an application scope.

Subramanian et al. [12] centered on analyzing certain software metrics in an object-oriented (OO) environment. The metrics collected and analyzed include size, quantity of message (NOM) sends, reuse, inherited methods, and hierarchical nesting level. The website used could be the factory systems department of a big sizable manufacturing company. This department uses Smalltalk whilst because the OO programming language to implement the OO design paradigm. Using automation tools developed in Smalltalk, these metrics were collected from three domain applications comprising 600 classes. Four propositions are empirically tested and the outcomes are provided in this study.

Arisholm et al. [16] proposed two dynamic coupling metrics; Import Coupling (IC) and Export Coupling (EC) measures quantifying message communication among objects during execution time. These measures are further classified at object-level and class-level with each metric depending upon either number of messages sent/received or methods invoked or classes accessed. IC_CD, IC_CM, IC_CC, IC_OD, IC_OC, IC_OM are the import coupling metrics and EC_CD, EC_CM, EC_CC, EC_OD, EC_OC, EC_OM are the export coupling metrics. They studied the relationship between these metrics with the change proneness of a system. They found that the dynamic coupling measurement did capture additional properties to the static coupling metrics and were good predictors of the change proneness of a class.

Mitchell and Power [14 & 15] conducted a number of studies on the quantification of many runtime class-level coupling metrics like Dynamic or Runtime Coupling Between Objects for a Class (DCBO), Degree of Dynamic coupling between Classes, Runtime Simple Lack of Cohesion Metric R_{LCOM} and Runtime-Call Weighted Lack of Cohesion Metric (RW_{LCOM}) for object-oriented (Java) programs. They have also statistically analyzed the differences in the underlying dimensions of coupling captured by static versus dynamic coupling metrics using CK metrics [Chidamber and Kemerer [4]] like CBO and LCOM. The results indicated that the run-time metrics did capture different properties than the static metrics alone.

Hassoun et al. [17 & 18] studied object-level coupling as it computes during runtime and proposed a dynamic coupling measure that takes object interactions into consideration. They also proposed a Dynamic Coupling Metric for an Object P and Dynamic Coupling Metric for a Complete Class. These metrics can be used to compare systems built on meta level architectures with systems having no reflective features yet, at the same time, exhibiting the same interface.

Varun Gupta, PhD Thesis [21] in their study new static and dynamic metrics have been proposed, evaluated and validated for measurement of cohesion, coupling and complexity for object-oriented software. Major contributions of the work reported in this thesis are: New metrics for the measurement of package cohesion and package coupling have been proposed and validated, Aspect-oriented approach of collection of run-time data has been found to be better than other approaches for this purpose, New dynamic cohesion and coupling metrics have been proposed and validated, A dynamic analyzer tool has been developed using aspect-oriented programming (Aspectj) to perform dynamic analysis of Java applications, New spatial complexity measures for the estimation of comprehensibility of Java programs have been defined, and New object-oriented cognitive-spatial complexity metrics have been proposed and evaluated.

Jitender Kumar et al [22] did a survey on various existing and proposed dynamic metrics and found that dynamic metrics play a vital role in performance evaluation using coupling, cohesion and complexity metrics. An extreme study of the dynamic metrics, proposed till date on various static and dynamic metrics, pseudo-dynamic metrics and hybrid approach of static as well as dynamic metrics.

P.Singh et al. [23] proposed four class-level dynamic coupling metrics such as Dynamic Afferent Coupling (DCa), Dynamic Key Server Class (DKSC), Dynamic Key Client Class (DKCC), Dynamic Key Class (DKC), and Percentage Active Classes (PAC) to assess the quality of object oriented software systems. They focused on finding the key coupled classes, i.e. the most active classes at runtime, with the help of both import and export coupling measures.

Gupta et al. [24] proposed three dynamic coupling metrics for object-oriented software systems such as Dynamic Object Coupling between Objects (DOC), Total Dynamic Object Coupling (TDOC) and Class Level Dynamic Coupling (CDC) which plays vital role in all major types of relations possible between objects of classes at run-time such as aggregation, inheritance, references and invocations.

Varun Gupta et al. [47], focused on the previous study and discussion about dynamic metrics provides the dynamic behavior. Again defined a component of a unit cohesion metrics are SFC (Strong Functional Cohesion), WFC (Weak Functional Cohesion). The expectation of the dynamic metrics shows the run time performance on the dynamic slicing. They are used dynamic slices of outputs to measured unit of cohesion. According to author define SFC dynamic metric is

module cohesion obtained from common definition-use pairs of each type common to the various dynamic slices of the output variables and WFC metric cohesion obtained from definition-use pairs of each type find in dynamic slices of multiple object values.

CRK Reddy et al. [26], assessed the fault proneness of object oriented software , mediated class relation and method calls as confounding factor on coupling and cohesion, is evaluated and proposed Mediated Cohesion (MH), Mediated Coupling Between Object (MCBO) metrics, which measures numeric values rather than in binary quantity. These measures are shown to be superior at the measure of the fault proneness.

Singh et al. [27] emphasized on a new framework to gain access to the Aspect Oriented Software's (AOS) using software metrics. Software metrics for the qualitative and quantitative assessment may be the combination of static and dynamic metrics for software's. It is located from the literature survey that till date most of the framework only considered the static metrics based assessment for aspect oriented software's. Within their work they've mainly considered the set of static metrics alongside dynamic software metrics specific to AspectJ. This framework may give a new research direction while predicting the software attributes because earlier dynamic metrics were neglected while evaluating the standard attributes like maintainability, reliability, understandability for AO software's. Centered on basic fundamentals of software engineering dynamic metrics are equally important as well as static metrics for software analysis. An identical concept is borrowed to use on aspect oriented software development by the addition of dynamic software metrics. Presently they've only proposed a construction and model using the static and dynamic metrics for the assessment of aspect oriented system but nonetheless the proposed approaches have to be validated.

Rani Geetika et al [31] gave a thorough survey of dynamic coupling metrics for OO systems. An attempt has been made to bring forth and characterize metrics accuracy to some major characteristics like Dynamic Analysis Techniques (DAT), metrics validation, real world applicability etc. Dynamic coupling metrics need to be evaluated for a wide range of large scale real world applications for both metric validation and effective utilization for software quality assessment, working with different programming languages not only Java based applications.

Nelamadhab et al. [30] identified several profits as well as loop holes with object oriented metrics and software reuse. In this review they have presented a set of six well established OO metrics like WMC, DIT, NOC, CBO, RFC, MPC and compared all the OO software metrics which focuses on all the attributes, methods used in the software environment. Reusability is the prime attribute and found that when reusability increases then the Depth of Inheritance increases. If the reusability decreases conversely the Coupling Between Object and Lack of Cohesion will be decreased.

Hemalatha Sharma et al. [36] analyzed the effectiveness of dynamic metrics for maintainability prediction. To assess the capability of dynamic metrics in capturing the OO software characters, HoduKu, open source software was used. In developing the prediction model they have used 4 Machine Learning algorithms: Linear Regression, Multi Layer Perception, Gausion Process and SMOreg. Compared the results of prediction using static and dynamic metric deployed with all Machine Learning algorithms suing prediction accuracy measures such as Mean Absolute Error (MAE) and Root Mean Square Error (RMSE).

Ana Nicolacscu et al. [35], in software development, coupling metrics and their impact on quality attributes have been investigated for the past 25 years and considered 26 of the most influenced research practices and software analysis tool. The direction of current research should be moved towards atomizing and evaluating existing results rather than exploring new application domains and determining new metric suits.

N.Raj Kumar et al. [32] focus on object oriented quality metric parameters. To extract the information for cohesion and coupling measurement Java Reflection API can be used in a manner similar to measuring these metrics. The low coupling and high cohesion improve the product quality and reusability. The result helps to predict the fault of an object oriented system after the coding phase is completed.

Saleh Almugrin et al. [33] proposed a practical enhancement approach that manages packaging in object oriented software development by analyzing all kinds of dependency relations of packages. The results produced by these metrics are more accurate than those provided by direct metrics. Two major frameworks, Property-based measurement framework and Distance Framework were used to evaluate the newly developed metrics to demonstrate that they are valid. They have also designed a plan to enhance the newly developed distance metric in terms of its ability to reveal thresholds that present in maintainability levels and to study the relationship between the newly developed metric values and to change the history of a system.

MykolaTkachuk et al. [37] presented an approach to effectiveness estimation of modern Post Object Oriented Technologies (POOT) in software maintenance, which aims to utilize domain specific knowledge for this purpose. To process these data quantitative metrics and expert oriented estimation algorithms were elaborated, which are formalized and combined logically in the proposed algorithmic model. Fuzzy Logic method and the appropriate CASE Tool were used to define POOT effectiveness assessment, to find final complex estimation values.

Ashu Jain et al. [46] proposed the use of dynamic metrics for the software maintainability prediction (SMP). Six machine learning algorithms are used to build the prediction models for both static and dynamic metrics. The performance of all models is compared using prevalent accuracy measures. Results show that dynamic metrics perform better than static metrics, and can be combined as a sound alternative for software maintainability prediction.

Chengying Mao et al. [39] aimed to measure the dynamic complexity of a service system based on the metric of entropy. Two models, service unit vector and service pair metric is presented to represent the execution traces of the system. They have also adapted distance entropy as well as Shannon

entropy, are used to measure dynamic coupling of a service system based on the metric of entropy. The proposed metrics need to be further validated on some large scale and more complex service systems.

Johannes Brauer et al. [40] shows the results of an online survey aimed at identifying the importance of 49 design best practices on design quality. In total 214 people participated in the survey, resulting in an average of 138 opinions for each practice. Based on these opinions, five very important, 21 important, 12 moderately important and 11 unimportant design best practices could be derived. This has 49 measurable design best practices for Java and is a valuable support for practitioners in Software Engineering.

Neelamadhab et al. [38] reviewed the object oriented metrics and analyze the difference between all the object oriented metrics through the comparison table. They have taken a case study that how to evaluate the reusability by using Machine Learning Regression algorithms and proved that standard instance based learning with no distance weighting is the best regression algorithm using WEKA Software for compare and plotting graph. And proved "Instance Based (IB) k with No Distance Weighting algorithm" is better than other regression algorithms in evaluation of reusability of object oriented metrics.

Aswini S et al. [41] said that software complexity as well as the cost of the projects increases iff LOC increases. The error will be very high whenever the COCOMO, SLIM and SLOC metric is used for calculating the metrics. They have included the tools to find LOC of Routing Algorithm and compared LOC of Routing Algorithm between the tools.

Amit Kumar Dogra et al. [42] presented a live streaming based approach using PaaS to overcome hurdles. Experiments were carried out on five sample applications with varying size ranging from 100 to 1250 classes. The major objective of this research is to design and implement an approach that process huge amount of trace data in parallel on cluster of cloud gears for live dynamic metric evaluation (DMA Tool).

Praveen Kumar et al. [43] proposed a metric Component Interface Metric (CIM), which is defined by a number of components called for and executed during the rendering process. It first validates properties of components by passing some parameters, calculates its dependency factors and interface metric.

Henning Schnoor et al. [45] compared static coupling metrics with proposed dynamic metric, Number of Interactions (NOI) or Method Calls, during the run time of the system. Results suggested that there is significant difference between static coupling metrics and dynamic coupling metric. There is a scope to investigate how these metrics can be used to evaluate the quality of software systems.

Neelamadhab et al. [44] focus on the assets and found 11 most reusable attributes and found that most of the researchers observed that reusable assets are validated strongly in academics but poorly in production sector. In comparison software academic and industry, the industry oriented survey is less than academic. Reusability plays a vital role in the industry at the same time the new kinds of challenge is aging. The following table [TABLE III] describes the summary of different research scholars' work on object oriented metrics.

TABLE – III: Summary of Metrics Proposed By Different Researchers in the Decade

Published By	Year of Publication	Metric Type	Proposed Work	Remarks
Jitender Kumar et al. [22]	2010	Survey on Dynamic Metrics	Did survey on various existing and proposed dynamic metrics and found that dynamic metrics play vital role in performance evolution using coupling, cohesion, and complexity metrics	Lacking in pseudo dynamic metrics and hybrid dynamic metrics
Gupta et al. [24]	2011	Dynamic Coupling Metrics	They proposed three dynamic coupling metrics for object-oriented software systems such as Dynamic Object Coupling between Objects (DOC), Total Dynamic Object Coupling (TDOC) and Class Level Dynamic Coupling (CDC).	These metrics places vital role in all major types of relations possible between objects of classes at run-time such as aggregation, inheritance, references and invocations.
Dr. CRK Reddy et al. [26]	2012	Static Coupling and Cohesion Metrics	They have assessed the fault proneness of object oriented software , mediated class relation and method calls as confounding factor on coupling and cohesion, is evaluated and proposed Mediated Cohesion (MH), Mediated Coupling Between Object (MCBO) metrics, which measures numeric values rather than in binary quantity.	These metrics are superior at the measure of the fault proneness.

Published By	Year of Publication	Metric Type	Proposed Work	Remarks
Singh et al. [27]	2013	Aspect Oriented Software Metrics using static and dynamic metrics	They have emphasized on a new framework to get access to the Aspect Oriented Software's (AOS) using software metrics.	They have only proposed a construction and model using both types of metrics for the assessment of AOS but nonetheless the proposed approaches have to be validated.
Rani Geethika et al. [31]	March 1, 2014	Survey on Dynamic Coupling metrics	They have provided a thorough survey of dynamic coupling metrics for object oriented systems. An attempt has been made to bring forth and characterize metrics according to some major characteristics like dynamic analysis techniques, metrics validation, real world applicability etc.	Dynamic coupling metrics need to be evaluated for a wide range of large scale real world applications for both metric validation and effective utilization for software quality assessment. Working with different programming languages not only java based applications.
Hemalatha Sharma et al. [36]	2015	Dynamic Metrics for maintainability prediction	They analyzed the effectiveness of dynamic metrics for maintainability prediction using HoduKu, open source software. In developing the prediction model they have used 4 Machine Learning algorithms: Linear Regression, Multi Layer Perception, Gausion Process and SMOreg.	Compared the results of prediction using both metrics deployed with all Machine Learning algorithms using prediction accuracy measures such as MAE and RMSE.
Ana Nicolacscu et al. [35]	2015	Survey on static coupling metrics	In software development, coupling metrics and their impact on quality attributes have been investigated for the past 25 years and considered 26 of the most influenced research practices and software analysis tool.	The direction of current research should be moved towards atomizing and evaluating existing results rather than exploring new application domains and determining new metric suits.
N.Raj Kumar et al. [32]	April, 2015	Static coupling, cohesion metrics	They focus on object oriented quality metric parameters. To extract the information for cohesion and coupling measurement Java Reflection API can be used in a manner similar to measuring these metrics. The low coupling and high cohesion improve the product quality and reusability.	The result helps to predict the fault of an object oriented system after the coding phase is completed.
Saleh Almugrin et al. [33]	2015	Package level static metrics	They have proposed a practical enhancement approach that manages packaging in object oriented software development by analyzing all kinds of dependency relations of packages. The results produced by these metrics are more accurate than those provided by direct metrics. Two major frameworks, Property-based measurement framework and Distance Framework were used to evaluate the newly developed metrics to demonstrate that they are valid.	They have also plan to enhance the newly developed distance metric in terms of its ability to reveal thresholds that present in maintainability levels add to study the relationship between the newly developed metric values and the change history of a system.

Published By	Year of Publication	Metric Type	Proposed Work	Remarks
MykolaTkachuk et al. [37]	June 8, 2016	Dynamic metrics on Software maintenance	They have given the intelligent approach to effectiveness estimation of modern post object-oriented technologies (POOT) in the software maintenance, which aims to utilize domain-specific knowledge for this purpose. This knowledge is complex and interconnected data resources organized in a form of the multi-dimensional information space, where the following characteristics can be defined: (1) the structural complexity of legacy software; (2) the dynamic behavior of user's requirements; (3) architectural-centered implementation efforts of different POOTs. To process these data quantitative metrics and expert-oriented estimation algorithms were elaborated, which are formalized and combined logically in a form of the proposed algorithmic model. Final complex estimation values of POOT effectiveness assessment are defined using the fuzzy logic method and the appropriate CASE-tool, which were successfully tested on real-life legacy software applications.	In future planned to extend the collection of metrics for POOT-features assessment, and to apply some of alternative (to fuzzy logic method) approaches to final decision making support.
Ashu Jain et al. [46]	December 5, 2016	ML algorithms are used to predict Static and dynamic metrics	Six machine learning algorithms are used to build the prediction models for both the static and dynamic metrics. The performance of all models is compared using prevalent accuracy measures.	Results show that dynamic metrics perform better than static metrics, and can be used as a sound alternative for SMP.
Aswini S et al. [41]	June 14, 2017	Static metrics on lines of code	Software complexity as well as the costs of the projects in software increases if and only if LOC increases. The error level is high whenever the LOC increases. The performance, the efficiency and the effort increases whenever the LOC increases. The LOC is given as the input in the models called COCOMO and SLIM. For calculating the metrics SLOC metrics is used. There are some controls or constraints for the SLOC metric which makes unprotected while depending on it. We can easily measure the LOC by using the tools in software metrics. Size will be taken as the input in many cases and also be counted in LOC, FP, and class point. In these we have implemented Routing algorithm in different tools it shows different lines of code, even though it has a similar program coding.	The lines of code vary from one another and also a well-known programmer may produce the logic in less LOC whereas the other gives more SLOC even they did in similar language.

Published By	Year of Publication	Metric Type	Proposed Work	Remarks
Chengying Mao et al. [39]	August 10, 2017	Dynamic metric on Complexity	Based on the modeling representation of execution traces, the basic Shannon entropy and the adapted distance entropy are used to measure the dynamic complexity of a service system.	The proposed metrics need to be further validated on some large scale and more complex service systems. There are still some directions could be explored in ongoing research. In addition, the determination of the set size of execution traces is also worthy of in-depth exploration.
Johannes Bräuer et al. [40]	August 10, 2017	Survey on Design Quality	This paper shows the result of an online survey aimed at identifying the importance of 49 design best practices on design quality. In total, 214 people participated in the survey, resulting in an average of 138 opinions for each practice. Based on these opinions, five very important, 21 important, 12 moderately important and 11 unimportant design best practices could be derived. This information about importance helps managing design improvements in a focused way.	Having a set of 49 measurable design best practices for Java is a valuable support for practitioners in software engineering. Nevertheless, the completeness of design best practice collection is not known. In general, this investigation should reveal gaps in measuring the design principles and object-oriented design.
Parveen Kumar et al. [43]	October 17, 2017	Dynamic metrics-Component Integration	They have focused on dynamic metrics of component integration at run time. The dependency function measures the Component-Usage Metric and Component Interface Metric results the number of components called and executed during the rendering process.	The aim of the paper is to enhance the performance and predict the quality of CBS in real time dynamically.
Amit Kumar Dogra et al. [42]	October 18, 2017	Dynamic Coupling metrics	It presents a live execution trace streaming based approach using platform as a service (PaaS) to overcome this hurdle. The proposed approach is the first of its type having the ability to provide the users with live dynamic coupling metric trends. In this work only six dynamic coupling metrics were captured, so the tool lacks support for a complete dynamic metric suite.	Proposed approach can be further enhanced to include all the dynamic coupling and cohesion metrics in future. Secondly, considering the continuous drift of software industry from desktop applications to mobile and web applications, the proposed approach should be further developed to analyze such applications.
Neelamadhab Padhy et al. [38]	December 20, 2017	Case study on reusability metrics using WEKA Software	This paper reviews the object oriented metrics and analyzed the difference between all the object oriented metrics through the comparison table. They have been taken a case study that how to evaluate the reusability by using machine learning regression algorithms and proved that Standard instance-based learning with no distance weighting is the best regression algorithms among others .Finally they	"IBk with No Distance Weighting" algorithms can be used frequently for evaluation of reusability of an Object Oriented Metrics.

Published By	Year of Publication	Metric Type	Proposed Work	Remarks
			have compared the novel regression algorithms and mentioned with a tabular form then have been used WEKA software for compare and plotting the graph.	
Schnoor et al. [45]	June 23, 2018	Static and Dynamic metrics compared on coupling-	Introduced new dynamic metric called Number of Interactions (NOI) & used the data collected from an experiment to compute NOI metric and compared the results to a static coupling analysis. We observed an unexpected level of correlation and significant differences between class- and package-level analyses.	Despite the large (and expected) difference, there is significant correlation. Further study of dynamic quantitative measurements is a promising line of research.

V. CONCLUSION

Object Oriented Dynamic Metrics (OODM) is very important to analyze the runtime behavior of Software Application. As the dynamic metric can be evaluated during the runtime of the application, it is difficult to compare to the static metrics evaluation. Even though dynamic metrics have an advantage over the static metrics, they are difficult to evaluate. So the static metrics can be used during the evaluation of dynamic metrics. For example Coupling Between Objects (CBO) is static metric, which can be used in evaluating the Dynamic Coupling Between Objects (DCBO). Combination of static and dynamic metrics will result more insights to the developer, which is called working with Hybrid Metrics. This hybrid approach can combine static as well as dynamic metrics to make the measurement process more effective. Thus, future research work can be done on Hybrid Metrics approach where the dynamic analysis results can be augmented by static information for collection of metrics information.

ACKNOWLEDGEMENTS

We are thankful to the Management and Principal, CBIT (A)-Hyderabad for providing e-resources for referring various journals and conferences. We are grateful to Dr.Ch.Suvarna Ragini, Faculty of English, CBIT (A), Hyderabad for her extensive support in this article writing in providing language help.

REFERENCES

[1] S. R. Chidamber and C. F. Kemerer, “Towards a metrics suite for object oriented design,” in Conference Proceedings on Object-oriented Programming Systems, Languages, and Applications, ser. OOPSLA ’91.New York, NY, USA: ACM, 1991, pp. 197–211.

[2] Somerville, Ian, “Software Engineering”, Addison-Wesley Publishing Company, 1992.

[3] S. R. Chidamber and C. F. Kemerer, “A Metrics Suite for Object Oriented Design”, IEEE Transactions on Software Engineering, Vol. 20, No.6, pp. 476-493, 1994.

[4] Booch, Grady, “Object Oriented Analysis and Design with Applications”, The Benjamin/Cummings Publishing 1. Company, Inc., 1994.

[5] Lorenz, Mark and Kidd, Jeff, “Object Oriented Software Metrics”, Prentice Hall Publishing, 1994.

[6] McCabe & Associates, McCabe Object Oriented Tool User’s Instructions, 1994.

[7] Chidamber, Shyam and Kemerer, Chris, "A Metrics Suite for Object Oriented Design", IEEE Transactions on Software Engineering, June, 1994, pp. 476-492.

[8] M.Lorenz and J. Kidd, “Object Oriented Software Metrics”, Prentice Hall, 1994. [

[9] F.B. Abreu, “The MOOD Metrics Set”, In Proc. ECOOP’95, Workshop on Metrics, 1995.

[10] Rosenberg, Linda H., “Metrics for Object Oriented Environments”, EFAITP/AIE Third Annual Software Metrics Conference, December, 1997.

[11] S.M. Yacoub, H.H. Ammar and T. Robinson, “Dynamic Metrics for Object Oriented Designs”, Software Metrics Symposium, pp 50-61, Boca Raton, Forida, USA, 1999.

[12] Subramanian, Girish, and William Corbin. "An empirical study of certain object-oriented software metrics." Journal of Systems and Software 59, no. 1 (2001): 57-63.

[13] J Bansiya and C.G. Davis, “A Hierarchical Model for Object Oriented Design Quality Assessment”, IEEE Transactions on Software Engineering, Vol. 28, No. 1, 2002.

[14] A. Mitchell, J.F. Power, “Runtime Coupling Metrics for the Analysis of Java Programs”, Preliminary results

from SPEC and Grand Suites, Technical Report NUIMCS-TR2003-07, Department of Computer Science, National University of Ireland, Maynooth Co. Kildare, Ireland, 2003.

[15] A. Mitchel, J.F. Power, "An Empirical Investigation into the Dimensions of Runtime Coupling in Java Programs", Third Conference on the Principles and Practice of Programming in Java, pp 9-14, Las Vegas, Nevada, USA, 2004.

[16] E. Arisholm, L.C Briand, A. Foyen, "Dynamic Metric Coupling Measures for Object Oriented Software", IEEE Transactions of Software Engineering, 30(8):491-506, 2004.

[17] Y.Hassoun, R. Johnson, S. Counsell, "A Dynamic Runtime Coupling Metric for Meta Level Architectures", in Proceedings of Eighth Euromicro Working Conference on Software Maintenance and Reengineering (CSMR '04), pp. 339, 2004.

[18] Y. Hassoun, R. Johnson, S. Counsell, Dynamic coupling metric: proof of concept, in IEE Proceedings -Software, 152: 273-279, 2005.

[19] Dr. Waralak V. Siricharoen, "Ontologies and Object models in Object Oriented Software Engineering", IAENG International Journal of Computer Science, 33:1, IJCS_33_1_4, February 2007.

[20] K.K.Agarwal, Yogesh Singh, "Software Engineering", 3rd Edition, New Age International Publishers, 2008, ISBN(13):978-81-224-2360-0

[21] Varun Gupta, PhD Thesis, "Object-Oriented Static and Dynamic Software Metrics for Design and Complexity," (2009), National Institute of Kurukshetra, June 2010.

[22] Jitender Kumar Chhabra, Varun Gupta V, "A Survey of Dynamic Software Metric", International Journal of Computer Science and Technology, Vol. 25, No. 5, 1016-1029, Sep-2010.

[23] P. Singh, H. Singh, "Class-level Dynamic Coupling Metrics for Static and Dynamic Analysis of Object-Oriented Systems", International Journal of Information and Telecommunication Technology, 1(1): 16-28, 2010.

[24] V. Gupta, "Validation of Dynamic Coupling Metrics for Object-Oriented Software", ACM SIGSOFT Software Engineering Notes, 36(5), 2011.

[25] Dr. Rakesh Kumar, Gurvinder Kaur, "Comparing Complexity in Accordance with Object Oriented Metrics", International Journal of Computer Applications, Volume 15, Issue 8, February 2011,PP42-45

[26] Dr. CRK Reddy, Amjan Shaik, Dr. Damodaran, "Impact of mediated relations as Confounding Factor on Cohesion And Coupling Metrics For Measuring Fault Proneness in Object Oriented Software Quality Assessment ", Global Journal of Computer Science and Technology Software &Data Engineering, Vol.12, Issue 13, 2012.

[27] Singh, Pradeep Kumar, and Om Prakash Sangwan. "Aspect Oriented Software Metrics Based Maintainability Assessment: Framework and Model." (2013): 1-07.

[28] Shweta Sharma, Dr. S. Srinivasan, "A review of Coupling and Cohesion metrics in Object Oriented Environment", International Journal of Computer Science & Engineering Technology (IJCSET), ISSN: 2229-3345, Vol. 4 No. 08, Aug 2013.

[29] H. Hlomani, and Deborah Stacey, "Approaches, methods, metrics, measures, and subjectivity in ontology evaluation-A survey", IOS Press, 2014.

[30] Neelamadhab Padhy, Rashmitha Panigrahi, Sarada Baboo, "A Systematic Literature Review of an Object Oriented Metrics: Reusability", International Conference on Computational Intelligence and Networks, IEEE, 2014.

[31] Rani Geethika, Paramvir Singh, "Dynamic Coupling Metrics for Object Oriented Software Systems-A survey", ACM SIGSOFT Software Engineering Notes, Vol.39, No.2, March 2014.

[32] N.Raj Kumar, C.Viji and S.Duraisamy, "Measuring Cohesion and Coupling in Object Oriented System using Java Reflection", ARPN Journal of Engineering and Applied Sciences, Vol. 10, No. 7, April 2015.

[33] Saleh Almugrin, Austin Milton, "Indirect Package Coupling Based on Responsibility in an Agile, Object Oriented Environment", 2nd International Conference on Trustworthy Systems, and Their Applications, IEEE 2015.

[34] Ankush Vesra, Rahul "A Study of Various Static and Dynamic Metrics for Open Source Software", International Journal of Computer Applications (0975 – 8887) Volume 122 – No.10, July 2015.

[35] Ana Nicolaescu, Horst Lichter, 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, doi: 10.1109/SAM 2015.14, May 2015.

[36] Hemalath Sharma, Anuradha Chuf, "Dynamic Metrics are Superior than Static Metrics in Maintainability Prededictiоn: An empirical case Study", IEEE, 2015.

[37] MykolaTkachuk, Konstiantyn Nagornyi, and Rustam Gamzayer, "Models, Methods and Tools for Effectiveness Estimation of Post Object Oriented Technologies in Software Maintenance", ICTERI 2015, CCIS 594, pp. 20-37, Springer International Publisher Switzerland, 2016.

[38] Neelamadhab Padhy, Suresh Satapathy, and R.P.Singh, J.Sethlani, "A Systematic Literature Review of an Object Oriented Metrics Components: Case Study for Evaluation of Reusability Criteria",

International Conference on Advanced Studies in Engineering and Sciences, December 2017.

[39] Chengying Mao, Changfu Xu, "Entropy Based Dynamic Complexity Metrics for Service Oriented Systems", 24th Asia-Pacific Software Engineering Conference Workshops, IEEE, 2017.

[40] Johannes Braver, Reinhold Plosch, Matthias Saft and Christian Korner, "A Survey on the Importance of Object-Oriented Design Best Practices", 43trd Euromicro Conference on Software Engineering and Advanced Applications, IEEE, 2017.

[41] Aswini S, Yazhini M, "An Assessment Framework of Routing Complexities using LOC Metrics", International Conference on Innovations in Power and Advanced Computing Technologies, IEEE, 2017.

[42] Amit Kumar Dogra, Harkomalm Singh, Paramvir Singh, "Execution Trace Streaming based Real Time Collection of Dynamic Metrics using PaaS", ACM/IEEE, 8th Workshop on Emerging Trends in Software Metrics", IEEE, 2017.

[43] Praveen Kumar, Pradeep Tomar, "Design of Dynamic Metrics to Measure Component Based Software", International Conference on Computing, Communication and Automation, IEEE, 2017.

[44] Neelamadhab Padhy, Suresh Satapathy, and R.P.Singh, "State-of-The-Art Object Oriented Metrics and its Reusability: A Decade Review", Smart Computing and Informatics, Smart Innovation Systems and Technology, 2018.

[45] Henning Schnoor, Wilhelm Hasselbring, "Toward Measuring Software Coupling via Weighted Dynamic Metrics", 40th International Conference on Software Engineering: Common Proceedings, ACM/IEEE, 2018.

[46] Ashu Jain, S.Tarwani and A.Chug, "An empirical Investigation of Evolutionary Algorithm for Software Maintainability Prediction", Electrical, Electronics and Computer Science, 2016 IEEE Students' Conference, July 2016.

[47] Varun Gupta, Jitender Kumar Chhabra, "Dynamic cohesion measures for object-oriented software", Journal of Systems Architecture, pp: 452-462, Volume 57 Issue 4, April, 2011.

AUTHOR PROFILE

Ramesh Ponnala is a research scholar, Department of Computer Science and Engineering, University College of Engineering, Osmania University, Hyderabad. Presently he is working as Assistant Professor in Department of MCA, CBIT (A)-Hyderabad. He received his MCA from Kakatiya Unviersity, Warangal, M.Tech. (CSE) from JNTU Hyderabad. He has published 4 International Journals. His main research interests are Software Metrics, Software Fault Prediction, Software Engineering, Software Quality and Object Oriented Metrics.

Dr. C.R.K. Reddy is working as a Professor in CSE, Department of CSE, Mahatma Gandhi Institute of Technology, Hyderabad, India. He received M.Tech. (CSE) from JNTU Hyderabad and Ph.D. in Computer Science and Engineering from Central University of Hyderabad, Telangana. He has been published and presented wide range of Research and Technical Papers in National and International Journals and Conferences. His main research interests are Program Testing, Software Engineering, Software Metrics, Software Architectures.