International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 3 Issue12, December-2014

A Survey on Modelling of Software Metrics for Ranking Code Reusability in Object Oriented Design Stage

Manoj H. M
Research Scholar
Jain University, Bangalore.
Karnataka, India

Dr. Nandakumar A. N
Principal
R.L Jalappa Institute of Technology
Bangalore, Karnataka, India

***Abstract* — In software engineering discipline caliber of software during object oriented (OO) design phase has been a briny objective among researchers. The design complexity plays a significant function in object oriented environments for predicting the reusability. And it is measured by using the metrics, where metrics have long term impact on the software development. It enables the software developers to achieve quality cost effective software. Only the object oriented model for software development differs from traditional procedure so that traditional metrics cannot be employed for object oriented software. Software metrics are measures of the attributes of software products. Because of knowing the importance of quality of the design stage, in that respect is the need of metrics that measures the goodness of design and it supplies the designer with improved insight that contributes to a higher degree of tone. Even today, there is a lack of availability of metrics that measures all aspects of OO systems. Hence, additional metrics are needed to measure currently unexplored or partially explored dimensions of OO systems. Various prior researchers have empirically discussed that higher up-front investment in design helps in holding costs as easily as in improving character. The aim of this paper is to contemplate in details about software metrics consideration which play a critical part in the quality of a software product and an extensive literature survey on ranking code reusability in software engineering and also events in code reusability at the conception level.**

***Keywords:-* object oriented (OO), Metrics, and Reusability.**

I. INTRODUCTION

Now a day's many software metrics are available for the estimation of quality of the objects oriented software. Most of these metrics can be employed only when the product is completed or almost finished. Initially they make use of the software code to fetch the software metrics used in determining its quality. This makes it difficult to restructure the design for its improvement which is possible only at the early stage of software development process. Detection of a faulty design of the later stages of software development bears a heavy price in terms of effort and cost. Thus, there is a need for a metrics / model which could make an evaluation of the software quality during the design phase of software development [1].

The Object Oriented model for the software development differs from the traditional procedural counterpart, and then the traditional metrics can't be applied on OO software [1]. Opponents of the use of traditional metrics within the OO paradigm argue that such metrics were originally designed to go along procedural methodologies and languages, and therefore fail to capture concepts as inheritance and polymorphism which are unique to the OO pattern. Moreover Traditional model requires more effort during the coding and maintenance phases as compared to OO paradigm, which put more emphasis on the earlier stages of software development, especially on design phase [1]. Various prior researchers [3] [4] have empirically discussed that higher up-front investment in design helps in controlling costs as well as in improving quality. Software metrics are measures of the attributes of software products and processes. Because of the importance of knowing the quality of the design phase, there is the need of metrics that measures the goodness of design and it provides the designer with improved insight that leads to a higher level of quality. Chidamber and Kemerer metric & MOOD metric are considered the best OO design metrics. But The Chidamber and Kemerer (CK) [5] metrics suite is the most criticized, perhaps due to its popularity.

Even today, there is a lack of availability of metrics that measures all aspects of OO systems. Hence, additional metrics are required to measure currently unexplored or partially explored dimensions of OO systems. Research on metrics for OO software development is limited and empirical evidence linking the OO methodology and project outcomes is scarce. Recent work in the field has also addressed the need for research to better understand the determinants of software quality.

The rest of this paper is divided as follows. Section II presents about software reusability and measurement. An overview of factors distressing reusability is described in section III. Then, in section IV overview of object oriented metrics and structure. In Section V, illustrated Issues in Code-Reusability, Section VI. Demonstrated existing work, Research Gap has demonstrated in Section VII and Finally conclusion presented in the section VIII.

II. SOFTWARE REUSABILITY & ITS MEASUREMENT

Reusability is the process of utilizing present software apparatus .the purpose of Reuse is to decrease the instance tired in creating way out through avoiding duplicate work .reuse is more Beneficial for reporting the flakes etc reusable software typically used in to get advanced efficiency. It is also used to gain good quality, for software developed by means of presented apparatus can be additional consistent

than those developed in new because the reused device are usually examine the data and enclose it in numerous developments. Though, the reusable works have to survive previous can also be reused. Reusing offering components are in a regular use in software production and human dilemma. On the other hand, reuse in software development is additional effectual when it is carry out properly. Proper reusability is employed only when the designers have the knowledge of their existence and the functionality of the present system General objective of reuse is division of an organization is to provide software structure procedure developers tasked to find out and use existing artifacts. Developers have to be in guarantee that the ultimate produce can also be reused in point of view for next development [6].

- *Uniqueness of Reusability:*-Reusability resources are dissimilar in some different contexts. But, they have present some Uniqueness so that frequently used for reusability of resources still a lot of characteristics are Appling to the assets Usability is the quantity in which an use of simple asset in a logical way of effort in the software engineering. since Usability is an independent of the system design are function .the fig explains the Sub characteristics of usability[6]
- *Assets in Reusability*: - A group of objects has measured an advantage in reusable Requirements such as Architectures, execution, Design; Data Program code etc in the utilization of product in software for example operating systems or database system is usually not measured as reuse. If any module is not well thought out as division of the system design, is not to be taken as reuse.

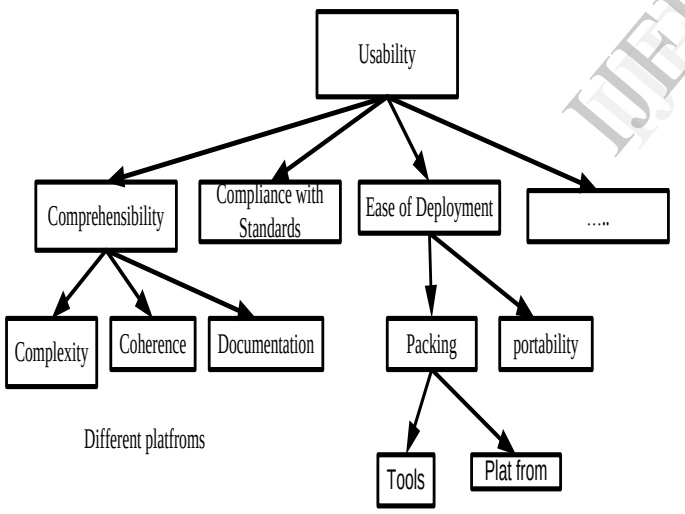

Figure 1: Uniqueness of usability

III. FACTORS DISTRESSING REUSABILITY

The figure 2 shows the diagram which describes the disturbing factors in reusability. And also this figure gives the idea of dependability factors in reuse such as effectiveness, expenditure of software application and memory time complexity and space.

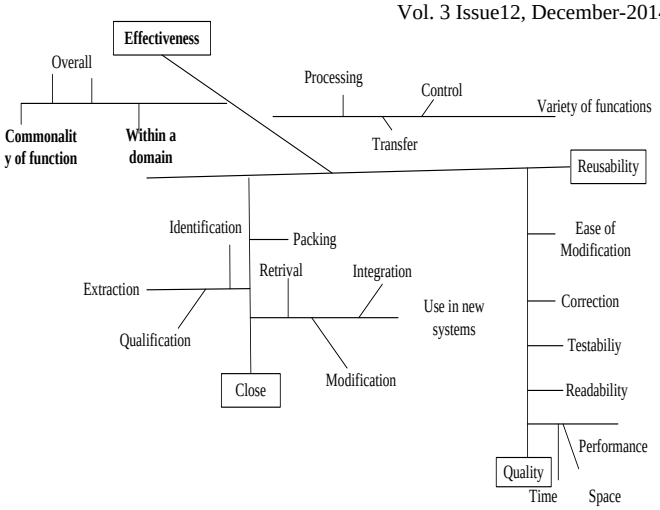

Figure 2: Factors distressing reusability

- *Effectiveness*: - When the functionality of the equipment is useful in the implementation of the new system than it can say reuse. it is very tuff and difficult to say a robotic method part are use full in a new system are not. This view fact of the result is mainly considered from domain knowledge and Necessities of the new system. But still for making use of reusability of the existing component from the origin of new system there was huge development of an oblique automatable measuring technique it have high assumptions for using the good content during the implementation of the new system . in addition to our assumption there was some limitations in to it. It is likely to keep out of those individuals area of apparatus that are not frequently used in the presented system. The significant role in this building .Expert domain knowledge is very important in the effectiveness**.**
- *Expenditure of software application:* - Expenditure involves cost for finding a module from the obtainable system, and compositing them into a original system. The Measures of amount and difficulty of a component give a incomplete sign of complexity in preparing it to reuse in a new system. The cost to reuse the module is partial by the reliability of its code, uniqueness that can be more incompletely in estimating the extent and difficulty procedures. Specifically minute and easy code wastes are more often than not easier to read and get used to in larger and composite garbage.
- *Memory time complexity and space:* - The quality of the element plays a vital role to achieve in reuse-driven growth. Numerous individuality are essential for constituent reuse of suitability, readability, testability, effortlessness of change, and presentation, however the majority of them are not openly computable. Measures of amount and difficulty of a constituent on the other hand offer a partial suggestion of the occurrence of these behaviors in it.

IV. OBJECT ORIENTED METRICS AND STRUCTURES

- *Metrics:*- In the software engineering matrices become significant task in reusability mainly in quality assurance and gaining the components .Metrics gives an idea to adapt obtainable module by looking cost effective ,it tells reuse is acceptable .metrics is mainly

for investments on module base are desirable in use .In software evaluation, matrix is the primary parameter of quantitative products to collects or computes numbers by using precise intention connected to sympathetic, scheming or humanizing the software with its creation. It is classified into product metrics to make possible for determine the properties of the software products and process metrics to facilitate properties in the obtained the products .Product metrics are sub divided into two categories namely external product metrics has enveloped properties observable product of the users. Internal product metrics cover up properties noticeable only just wards development team

The external metric products consist of:-

1. Product non-reliability metrics is used to assessing the numeral of outstanding defects.
2. To measure the product useful functionality, functionality metrics are used.
3. To measure the available resource (space occupancy, computation speed) usage of the product Performance metrics is developed.
4. To access the easiness in learning and usage of product Usability metrics is used.
5. Expense of the product is measured by the Cost metrics.

The internal Product Metrics contains mainly of

1. Size metrics used for measures of how large a produce is internally.
2. Complexity metrics used for assessing complexity of the product
3. Style metrics, used for assessing observance to inscription procedure for product components etc

Process Metrics consists of Cost metrics, used for the measuring the cost of a project, and Effort metrics using for the estimating the human element of the expenditure and characteristically considered in person to days or person to months. And also it includes Advancement metrics, intended for estimating the amount of close of a produce beneath the structure, the number of defects uncovered are assigned using Process non-reliability metrics. Growth benefits from previous development are assigned Reuse metrics [7].

The Figure 3 gives pictorial representation of Object oriented Structures in new Object oriented methods have their own style to reproduce the novel structural concepts. And table 1 brief explanation of Object Oriented Metrics. From the fig 3an object-oriented system can be defined from the class it includes attributes and operations which is related and close to it. Hierarchical trees are formed using the classes which are the basis for objects. . Attributes and operations from its parent class inherit to an object along with their own attributes and operations. An object can also turn out to happen to class for additional objects. While an object is useful in contains information or data in sequence it forms instantiation of the object. Objects are used to cooperate or converse by passing messages, in between any two objects message is passed, the object are seed to be coupled. In a class the degrees of methods are related to together, X is an object is coupled to only when X communicates to Y. Inheritance is a connection between classes, and methods in which one class shares the arrangement or methods in one or more other class The process of introducing an example of object and binding or adding the specific data in a Message known Instantiation,, a demand has made from other object makes of another object to carry out an operation [6].

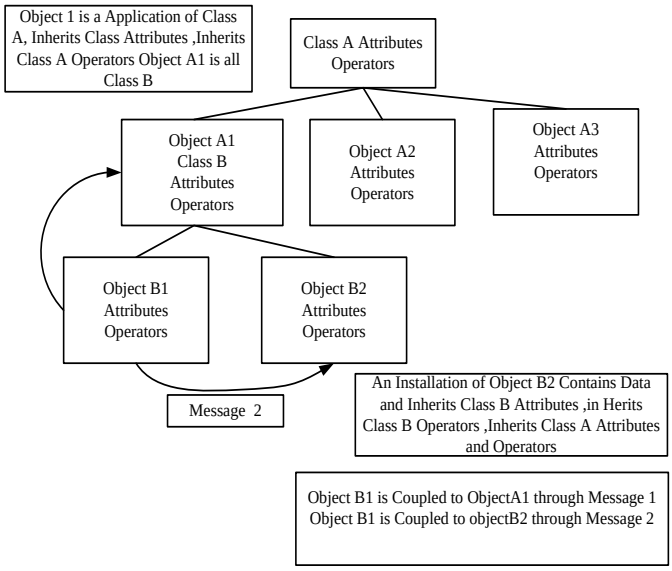

Figure 3- Object oriented Structures

In an OO design C&K metrics is set of matrices which is a collection of six design metrics i.e., Depth of Inheritance Tree (DIT), Weighted Methods Per Class (WMC),Number of Children (NOC), Response For a Class (RFC), Lack of Cohesion in Methods (LCOM)[2] and Coupling Between Object Classes (CBO) . In between these six metrics, LCOM, , CBO,DIT and WMC are used in object oriented classes to measure the reusability . But still in the black box components CBO and LCOM cannot be used .CBO used to compute the relations between classes and source codes of the classes of analyzation. And LCOM is for investigation of the fields and methods in the classes via analyzing the source codes .DIT and WMC without analyzing the source codes it measure the individuality of a single piece of a class. For that reason, black-box components are use to it. A quantity of object oriented metrics is available for measuring the design attributes such as, message passing, complexity inheritance, polymorphism, defeat issue, is that combination, organization, reusability etc,., A metrics plan specifically referring to the aims of an association resolve assist converse, determine development towards, and finally achieve those aims and objectives . Community will give motivation and an effort in the direction of complete what they consider to designate considerable. Elegant metrics through recognized the important goals will be able to assist an association get in order of requirements in the direction of continue to get better in software processes, products and services even as to maintain a focal point on top of what is essential. A reasonable, systematic for start-to-end process of designing selecting, and implementing software metrics is a precious support. This metrics is used in large number of projects to estimate and measure up to the concert of the code by means of an object oriented paradigm.

Table.1: Object Oriented Metric

Object Oriented Characteristic Metric	Model Design	Technique Use for Re- usability	Explanation
Cyclomatic Complexity Method	Difficulty	paths tested Algorithmic	Low decision overdue from end to end significance passing , less complex
SIZE Lines of Code Method	Difficulty	Physical lines statements ,comments	Must be Small
COM Comment Percentage Method	Usability Reusability	Apparatus separated by the total line countless blank	It gives 20 to35%
Weighted Methods Per Class	Complication Usability Reusability	1)Methods implemented within a class 2)Sum are difficulty of methods	Greater Complexity ,and Decreased Understandability ,Test and Debug More Complicated
Lack of Cohesion of Methods /Class	Design Reusability	Similarity of Methods within a Class by Attributes	High Class Subdivision , Increased Complexity Subdivide is Low
Coupling between Objects	Design Reusability	Distinct and Non-Inherited ,Related Classes are Inherited	High in poor Design , and Difficult to Understand , Decreased Reuse , Increased Maintenance
Depth of Inheritance Tree	Reusability Understandability Testability	Maximum Time taken from Class Node to Root	Complexity is high , more in reuse
Number of Children / Inheritance	Design	Instantaneous Subclass	Higher in More Reuse; Poor Design Increasing Testing

V. ISSUES CODE-REUSUBILITY

- Software metrics is required for the researcher software developer to get better application which is efficient in both code complexity and time complexity and error free in nature
- Software Reuse promises significant improvements in software productivity and quality.
- Most Prior research studies of metrics is only limited to requirement, design and implementation phase.
- The object oriented(OO) method for the software development differs from traditional procedural counterpart so the traditional metrics used to evalviate procedural programming software reusability can't be applied on OO software
- Prior research on the software matrices to encourage the software development process exposed the following barriers like lacking in the theoretical basis , show the insufficiency in generalization , implementation ,measurement properties deficiency and being too labor-intensive to collect.
- In Object oriented environment the design complexity plays an important role in predicting the reusability.
- The use of traditional metrics which are specially designed for procedural methodologies and languages. So, they fail to capture concepts of inheritance and polymorphism which are unique to the OO paradigm.
- Even today, there is lack of availability of metrics that measures all aspects of OO systems
- Reusable modules and classes reduce implementation time increase the likelihood that prior testing and use has eliminated bugs and localizes code modifications when a change in implementation is required.
- The intend of Metrics is to expect the quality of the software products. A variety of attributes, which establish the quality of the software, include maintainability, defect density, fault proneness, normalized rework, understandability, reusability however the issue of how to recognize good reusable components from existing systems has remained relatively unknown.
- Non-linear and complex engineering neural networks is an efficient technique for where requires controlling ,inference and analyzing according to [8]
- According to [8] k-nearest neighbor's algorithm (k-NN) is a technique for categorizes objects based on closest training examples in the feature space. K-NN is a type of instance-based learning, or lazy learning where the purpose is only approximated nearby and all calculation is deferred awaiting classification.
- In software matric the most obvious extension of the work is to analyze the degree up to which the proposed

metrics correlate with managerial performance indicators such as testing, maintenance effort and quality.

- CK metric is most preferred metrics in object Oriented (OO) design but in [9] they conducted inspection in the CK metrics, tried to reduce the inconsistencies and restructured metrics framework to get the structural analysis of OO depends on software components.
- A Neural Network approach could serve as an economical, automatic tool to generate reusability ranking of software by formulating the relationship based on its training.
- In [9] while designing software metric technology with Neural Networks alone, the network is a black box that needs to be defined; this is a highly compute-intensive process. One must develop a good sense, after extensive experimentation and practice, of the complexity of the network and the learning algorithm to be used.
- Neural nets and fuzzy systems, although very different, have close relationship: they work with impression in a space that is not defined by crisp, deterministic boundaries. Neural network can be used to define fuzzy rules for the fuzzy inference system. A neural network is good at discovering relationships and pattern in the data, so neural network can be used to pre-process data in the fuzzy system. Furthermore,
  To develop the fuzzy adaptive system, neural network which learn new relationships with new input data is used to refine fuzzy rules. With the objective of taking advantage of the features of the both, they used Neuro-Fuzzy approach to economically determine reusability of OO-based software components in existing systems as well as the reusable components that are in the design phase.
- Recent research study has recognized two types of factors called module design factor and module implementation factor which are going to use to characterize successful reuse-based software development. The module reuse is characterized by the module design factor by excluding the revisions in the code. In the same type characterizations of the module reuses is done by the module implementation factors by excluding revision of the small size in source lines and have many assignment statements. Faults per source line and fault effort correction are reduced in the modules reused without revision. In opposite that the modules reused with major revision had the highest fault isolation effort , highest fault correction effort ,highest change correction effort and more modification per source line.
- In McCabe Cyclomatic Complexity (CC) metric Cyclomatic complexity is a measure of a module control flow complexity based on graph theory. A high Cyclomatic complexity indicates that the code may be of low quality and difficult to test and maintain.
- In Source Lines of Code (SLOC) metric measures the number of physical lines of active code, that is, no blank or commented lines code. Counting the SLOC is one of the best methods to measuring complexity. If the number of SLOC is more in the module it conveya that module is less understandable and not maintained properly
- SLOC metric is most criticized method.
- Comment Percentage (CP) metric is defined as the number of commented lines of code divided by the number of non-blank lines of code. High cp convya good maintainability of the code.
- In ck metrics code Reusability increases with increase in DIT(*Depth of Inheritance Tree of a Class*) and NOC(*Number of Children*) factors but it decreases with increase in CBO(*Coupling between Objects*)
- Exception handling is the one of the desirable feature in the software environment which gives robust design of the application. So, handling of the exception is important in software development in [10] they discussed the below issue because of the failure in the exception management. Exception failures take place when a program is prevented by unexpected circumstances from given that it's particular service. Exemption failures can account for up to two-thirds of system crash therefore, are commendable of severe attention. On 4th June 1996, maiden flight 501 of the European Space Agency's new Ariane 5 heavy-lift rocket, developed at a cost of $7000 M over a 10 year era, ended in failure, after 39 seconds of its initiate. The problem was recognized as a software exemption caused through execution of a data exchange from 64-bit floating point to 16-bit format; the number was too big, so that a spill over error resulted after 36.7 seconds.
- For analysis of software evaluation, refactoring is the one of the principal techniques to restructure a software system's design and decrease its complexity and it also give information about the following topic like error detection, capturing intent of change, capturing and replaying changes, and relating refactoring and software metrics so, it requires to research about this topic in detail for the betterment of the software development .
- In one of the published book on refactoring they defines refactoring as "a change to the internal structure of software to make it easier to understand and cheaper to modify without changing its observable behavior".
- Conventional metric those require the source code is not applicable in many situations because the availability of the source code for many of the components is not possible.
- In [11] they proposed the metrics for the measurement of the reusability for the components without the source code.
- The metrics indicate the actual reuse rates of the reused component in a component library and in a software product. However, the metrics cannot because in a situation where sufficient time has not passed since the target component was developed.

VI. EXISTING SYSTEM

The paper was conducted by Churcher et al. [12] a semantic model which contains the general semantics of OO languages along with the to and from language-specific constructs mappings. They briefed how to evaluate the metrics which are benefited to evaluate the parameterized representation and heuristics evaluation by the common

semantic model. Along with the heuristic evaluation it also enables sets of language-independent heuristics to be managed and integrated with other software engineering tools.

The review was presented by Rohde et al. [13] a new vector-space technique for deriving word-meanings from huge corpora that was stimulated by the HAL and LSA models; however that attains enhanced and extra dependable results in predicting human similarity judgments. They elucidate the novel model, known as COALS, and how it relates to previous techniques, and after that appraise the different models on a variety of tasks, counting a novel place of semantic similarity ratings connecting both semantically and morphologically related terms.

The paper was demonstrated by Biffl et al. [14] a linking OSS data sources using semantic web technologies as base for providing integrated needles project status analysis. They introduce the design idea of a project monitor cockpit, Pro-MonCo, and evaluate the possibility and effectiveness with a pro-to-type for calculating statement metrics in a real-world background, the Apache Tomcat project. Key result was which Pro-MonCo competently supports frequent project monitoring by calculating statement metrics based on semantically included data originating from heterogeneous OSS project data sources.

The review was illustrated by Kang et al. [15] OWL 2 reasons, four state of the art, real world ontologies and Machine erudition techniques is reviewed to encounter the problems and the contribution is classified in to two categories like learning the classifiers that are predict the accurate time of classification for ontology based on the metric values and to identify the number of metrics that required to predict the reasoning performance.

This paper study was conducted by Gui and Scott [16] Coupling and cohesion metrics are designed to measure the reusability of java components retrieved from the internet and the above metrics is different from the mass established metrics. They measure following aspects like degree up to which modules are coupled and similar each other. They account the cohesion relationship and coupling and they represent the functional complexity of classes and methods.

The review was proposed by Vezhnevets et al. [17] a semantic-segmentation of weekly supervised method. An exercise images are tags only by the lessons they surround, not by their position in the image. An analysis images instead, the technique predicts a class label for every pixel. Their major invention multi-image model (MIM) – to obtain the training images pixel labels. The study was discussed by Heitlager et al. [18] Discuses the problems arise due to MI, and identifying the number of requirements that are provided by maintainability modal to which are used in practice. they designed the new maintainability model to encounter the problems and using IT management consultancy activities they discuss their experiences. The paper was proposed by Washizaki et al. [19] a metrics to measure the reusability of black-box components by making use of limited information which is gathered from the outside components without the source code. To measure the component's understandability, flexibility, portability, with assurance periods they defined the five metrics by statistical analysis a number of JavaBeans components.

The review was illustrated by Saini et al. [20] of the major issues in the software reusability and metrics established to quantify the concept. The concept of reuse in the software development environment involves decisions related to reuse or not in the development environment. Reusability brings financial benefits as well as it reduces development time, development cost, and brings direct and indirect benefits which are hard to quantify.

The paper was replicated by Biegel et al. [21] a well-known experiment from Weißgerber and Diehl, bunging in three dissimilar similarity events (text-based, AST-based, token-based). They look at the overlap of the results obtained by the different metrics, and they compare the results using recall and the computation time. They conclude that the different result sets have a large overlap and that the three metrics perform with a comparable quality. The review was conducted by Chatzigeorgiou et al. [22] on the observation that the role of each class in a system depends not only on the number of incoming and outgoing messages, however as well on the significance of the courses to which it is associated. The proposed method extends a link analysis algorithm currently employed for information retrieval from the Web. Authority and hub weights are obtained for each class, capturing the combined effect of communicating with other classes for servicing or issuing requests.

The review was discussed by Munassar et al. [23] on the comparison between Traditional approaches and Object-Oriented advances. Conventional advance has a batch of models that deal with different types of projects such as waterfall, spiral, iterative and v-shaped, however all of them and other need plasticity to deal with other kinds of projects like Object-Oriented. The paper was proposed by Aggarwal et al. [24] a related to various object-oriented builds like class, blending, cohesion, inheritance, information defeating and polymorphism. The applicability of metrics demonstrated by earlier researchers is typically partial to requirement, design and implementation phase. Omission handling is a attractive feature of software which pilots to robust intend and must be measured. This research concentrates to this need and presents a new point of design metrics for object-oriented code.

VII. RESEARCH GAP

In the present progressing era software development technology is the widely employed technology in every field in the world to get the better service and progress. So, to achieve good software developer in object oriented environment to get better application which is efficient in both code complexity and time complexity and error free in nature the need of software metrics to raking the code reusability is more. The traditional software metric and the existing metrics to rank the code reusability face the following constrains like procedural method of software metrics which is fail to capture the concepts object oriented concepts , inconsistency ,non predictable behavior because of inappropriate mathematical properties and non-availability of the metrics which measure all the all aspects of OO systems. The above said information convey that there is much need in research to development the software metrics in simplified approach which ranks the code reusability in the OO system. The research is to overcome the following the aspects like procedural based,

inconsistency and dependency on source code. The major research is to develop the metrics situations where the source codes are unavailable and where the components were newly developed and Simplifying the object-oriented approach through its steps to use the smallest projects that deal with simple programming.

VIII. CONCLUSION

This paper talks about the reusability concepts of component based systems, several existing metrics to measure reusability directly or indirectly are given and also issues in software metrics for ranking code reusability in software engineering are presented. An extensive literature study of software metrics for ranking code reusability in software technology is summarized and research gap is given based on literature review.

IX. REFERENCES

[1] J.Bansiya and C.G.Devis, "A Hierarchical Model for Object Oriented Design Quality Assessment", IEEE transactions on Software Engineering, Vol. 28, No. 1, 2002

[2] S.Sandhu, H.Singh, H.Singh, "A Critical Suggestive Evaluation of CK Metric", PACIS, pp.189-192, 2005

[3] R.K.Keller, R.Schauer, "Design Components: Towards Software Composition at the Design Level", Proceedings of the 1998 International Conference, pp. 302–311, 1998

[4] M.K.Zand, M.H.Samadzadeh, "Software Reuse-Issues and Perspective", Potentials, IEEE, Vol. 13, No-3, pp. 15-19, 1994.

[5] S.R.Chidamber, C.F.Kemerer, "A Metric Suite for Object Oriented Design", IEEE Transaction on Software Engineering, Vol. 22, No-6, pp. 476-493, 1994

[6] P.K.Suri and N.Garg, "Software Reuse Metrics: Measuring Component Independence and its applicability in Software Reuse", International Journal of Computer Science and Network Security, Vol. 9, No. 5, pp. 237-248, 2009

[7] S.Pasupathy, R.Bhavani, "Object Oriented Metrics Estimation and Performance Matching", International Journal of Advanced Research in Computer Science and Software Engineering, Vol. 4, Iss.1, 2014

[8] M. Kaur, M. Mahajan and P. S. Sandhu, "A k-NN based approach for Reusability Evaluation of Object Oriented Based Software Components" Retrived 5th December, 2014

[9] P.S.Sandhu and H.Singh, "A Reusability Evaluation Model for OO-Based Software Components", World Academy of Science, Engineering and Technology, Vol.2, 2008

[10]K.K.Aggarwal, Y. Singh, A. Kaur and R. Malhotra, "Software Design Metrics for Object-Oriented Software", Journal of Object Technology, 2006

[11]H.Washizaki,H.Yamamoto, and Y. Fukazawa, "A metrics suite for measuring reusability of software components", In Software Metrics Symposium,Proceedings. Ninth International, pp. 211-223, 2003

[12] N.Churcher, S. Frater, C.P.Huynh, and W.Irwin, "Supporting OO design heuristics", In Software Engineering Conference, ASWEC, 18th Australian, pp. 101-110, 2007

[13] D.Rohde, L.T, Laura M. Gonnerman, and D.C. Plaut, "An improved model of semantic similarity based on lexical co-occurrence", Communications of the ACM, Vol. 8, pp.627-633, 2006

[14]S.Biffl, W. D. Sunindyo, and T.Moser, "A Project Monitoring Cockpit Based On Integrating Data Sources in Open Source Software Development", In SEKE, pp. 620-627, 2010.

[15]Y-B.Kang, Y-F.Li, and S. Krishnaswamy, "Predicting reasoning performance using ontology metrics", In The Semantic Web–ISWC,Springer Berlin Heidelberg, pp. 198-214, 2012.

[16] G.Gui and P. D. Scott, "Measuring software component reusability by coupling and cohesion metrics", Journal of computers, Vol. 4, No. 9, pp. 797-805, 2009.

[17] A.Vezhnevets, V.Ferrari, and J.M. Buhmann, "Weakly supervised semantic segmentation with a multi-image model", In Computer Vision (ICCV), IEEE International Conference, pp. 643-650, 2011.

[18] I.Heitlager, T. Kuipers, and J. Visser, "A practical model for measuring maintainability", In Quality of Information and Communications Technology, QUATIC,6th International Conference, pp. 30-39, 2007.

[19]H.Washizaki, H.Yamamoto, and Y. Fukazawa, "A metrics suite for measuring reusability of software components", In Software Metrics Symposium, Proceedings. Ninth International, pp. 211-223, 2003

[20] D.K. Saini, S.A. Maskari, and L. Hadimani, "Mathematical Modeling of Software Reusability", In 3rd IEEE International Conference on Machine Learning and Computing (ICMLC) Singapore, 2011

[21] B.Biegel, Q.D. Soetens, W. Hornig, S. Diehl, and S. Demeyer, "Comparison of similarity metrics for refactoring detection", In Proceedings of the 8th Working Conference on Mining Software Repositories, pp. 53-62, 2011

[22]A. Chatzigeorgiou, S. Xanthos, and G. Stephanides, "Evaluating object-oriented designs with link analysis", In Proceedings of the 26th International Conference on Software Engineering, pp. 656-665, 2004

[23] N.M.A.Munassar and A.Govardhan,"Comparison between traditional approach and object-oriented approach in software engineering development", International Journal of Advanced Computer Science and Applications, Vol. 2, No. 6, 2011

[24] K.K.Aggarwal, Y.Singh, A.Kaur, and R.Malhotra, "Software Design Metrics for Object-Oriented Software", Journal of Object Technology, Vol. 6, No. 1, pp. 121-138, 2007