Indian Journal of Science and Technology, Vol 10(3), DOI: 10.17485/ijst/2017/v10i3/107289, January 2017
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Utility of an Object Oriented Reusability Metrics and Estimation Complexity

Neelamadhab Padhy[1]*, Suresh Satapathy[2] and R. P. Singh[1]

[1]SSSUTM, School of Computer Science and Engineering (SSSUTM), Bhopal – 466001, Madhya Pradesh, India; neela.mbamtech@gmail.com
[2]ANITS, Vishakhapatnam - 531162, Andhra Pradesh, India; sureshsatapathy@ieee.org

Abstract

Background/Objectives: In this 21st century, Reusability imparts powerful tools in the software industry. More and less 80% code is reused in the new project. Evaluation of metrics form the software code is now a challenging task as well as how much percentage of code is used from the existing one. This can be achieved by using CK (Chidambaram and Kemmerer Metrics). **Methods/Statistical Analysis:** There are numerous metrics are defined which distinguish the actual object. The proposed new metrics which is the combinations of one of the CK metrics suite and which calculates the reusable codes in the object oriented programme. **Findings:** In the inheritance if we take maximum depth of class in the hierarchy then found more chance for reusability of the inherited metrics. So DIT (Depth of Inheritance) has positive sign on the reusability of the class. If reasonable value for number of children then more scope of reuse in the class. If we have more number of methods in the class then more impact will be more on the children class and restrictive the possible of reuse. **Conclusion:** The OOS (Object Oriented System) using the parameterized constructor in C++ programs is more reusable up to some extent. When we will get the larger ethics (values) of proposed Metrics-2 and - 3 then definitely it gives the negative collision on the reusability. So the constructor having parameters (parameterized Constructor) gives the negative impact on the reusability of the classes.

Keywords: CK Metrics, Object Oriented Metrics, Reusability Factor

1. Introduction

In software development lifecycle the maintenance plays the crucial role, it is nearly about 65-75 %. The most important factor is the customer satisfaction. The top stockholders of the product are the customer. To become the successful project, time, cost and effort are the key attributes. There are many researcher has been conducted over the years to study the object oriented metrics[1–5]. Apart from the above attributes, other three are dominant and demand the quality products these are fault proneness, reusability and maintainability. The product will be success, if and only if that the developers must think about the existing code and utilization these code in the proper way. During the testing stage the product will be examined and verified meticulously. The objective is very clear that bug must be identified and prevented as early as possible in the SDLC (Software Development Life Cycle). The developers intension is very clear and concise that to provide the qualitative product. During SDLC phases, a rigorous testing demands the reliability

code which enhance the quality of the product, consequently it lead customers satisfaction. In this era maximum projects developed using short span of time by using the reusable components. Automatically the codes are minimized which reduce the time complexity as well as staffing. Software measurement can be done in one of the approach, it may be traditional or object oriented. The latest approach in the industry is an object oriented metrics which helps to create the reusable components through the inheritance. Within short span of time the developer develops the quality product. Another benefits the customer well get that is quality product[6]. At the end, if we compare among the newly developed product and

*Author for correspondence

the component which follows existing code, then definitely the reused product provides more quality because it has been tested several times under the supervision of the sophisticated tester. Another aspect is to get the less number of errors. Exactly reusability means new component will be created by using the existing one. It may be classes, methods, libraries or modules. Sometimes may be existing documents. When OO (Object Oriented) approach came to the market, it evolves conquered the software industry. These approaches dominate over last decades in the market. The prime reason is maintainability of the code. This can achieves if and only if design must be clearly understood and not frequently modified, Because once the architectural plan out for the product is fixed then further modification is one of the most expansive task. After the design there are different software metrics are available to review the excellence of the plan. The A huge amount of researchers studied about the OOM (Object-Oriented Metrics) which enhance the quality of the product. In the middle of the researcher one of them popularly known as CK (Chidamber and Kemerer)[8]. He proposed the 6 matrices. These matrices are DIT (Depth of Inheritance), NOC (Number of Children, CBO (Coupling Between Objects), RFC (Response for a Class), WMC (Weighted Method per Class), LCOM (Lack of Cohesion in Methods). In this paper we have designed a new metrics which will estimate the reusability. This is the matter of fact that software component reusability has enabled cost and time efficient software design, however the results of the excessive reutilization have not been studied so far. Most of the existing researches are primarily focused on component reutilization, where software metrics such as inheritance, Depth of Inheritance Tree (DIT), cohesion and coupling etc. based reusability have been examined. On the other hand, the fact that introducing excessive reusability might result into complexity too, has not been addressed so far. In recent years, the impact of code complexity on the software aging probability has become an attractive research domain. The two inter-dependent issues, higher reusability and complexity caused aging can be a novel research domain for future, especially with the goal to enhance the components based software design.

2. Review of Work already done in this field

The eminence of Object Oriented product is needy on a choice of different attributes like quality concepts like density, usability, consistency, testability, understandability etc. All these attributes are maticulsoly verified in the software product. If the product is not verified properly then it frequently demands the necessary changes, instead of repeatedly changing, we must enhance its functionality or correct defects in the software project[7]. There is variety of literature survey done by the different researchers about the metrics complexity from different angles. Amongst them Chidamber and Kemerer which is popularly known as CK metrics suite[2]. For developing the Object Oriented design measurement, MOOD metrics developed by[8], similarly for accessing and testing the measuring of the code in the s/w (software), Brinder, 1994 used the metrics[9]. Measurement can be done either in process or product i.e. called as process metrics and product metrics. How product metrics design occur that concept derived by[10] (Purno and Vaishnavi 2003; Vaishnavi et al 2007), Lorentz and Kidd used the other metrics which is related to product and process (Lorenz and Kidd 1994)[11]. Above these developed metrics are able to evaluate in traditional way then after some of the researcher redeveloped the metrics suite for object oriented approach which closely support all the OO features (Inheritacne, Data Abstraction, Polymorphism, cohesion, coupling etc.). Amongst them, Dubey and Rana, 2010) one of them[12]. Consequently, the worth of OO package can be evaluated in terms of OO design perspective like using OO metrics. It is also calculate approximately the size of the metrics i.e. size estimation in OO project[13], Metrics measurement is possible in the Extensive Markup Language (XML) in the web engineering[14]. Some of the measurements are very closely associated with other researchers' works which has described in[15–18]. Each and every metrics it has own its merits and demerits. There are different metrics are available which able to quantify the percentage of use of existing code into the new class. Several benefits found like cost, time and trustworthiness, efficiency, quality. The important thing is computation methods, evaluation different parameters, proposed model, recognize the dominant attribute in the source code. Inferiority of the metrics can be developed during the SDLC (Software Development Life Cycle) model, particularly in the design phase only. To achieve the above benefits researcher should proposed some framework which is capable to estimate the reusability.[19] they proposed to compute the OORPI (Object Oriented Reusability Parameter Index). Similarly Parvinder S. survey the object oriented metrics for reusability[20]; K. K.

Agrawal the same work he has done with a little modification[21]. Investigating existing literatures or researches and assessing their respective strengths as well as weaknesses play significant role in formulating or proposing certain effective approach to alleviate existing limitations. With this objective in this section, some of the literatures discussing quality oriented software reusability assessment are discussed in this section. In[23] studied and examined various factors influencing testability of OOP based software systems. To examine software testability based on OO metrics, they applied Analytical Hierarchical Process (AHP). In[24] applied CK metrics and investigated it to provide accurate object oriented software design parameter and proposed a new set of metrics to examine the reusability. In[25] explored different software quality estimation approaches to form software quality model using ANN, Case-Base Rule, Regression Tree, Rule Based System, Multiple Linear Regression and Fuzzy Systems. They found that ANN can give better result for quality prediction. In[26] employed key concepts such as reusability, scalability and flexibility for web applications design while targeting on reducing development timeframe. In[27] proposed various components based metrics and suggested for its application in OOP based software design. In[28] used Fuzzy logic algorithm to estimate reusability of aspect oriented software. In[29] derived four independent metrics Number of Template Children (NTC), Depth of Template Tree (DTT) Method Template Inheritance Factor (MTIF) and Attribute Template Inheritance Factor (ATIF), to estimate the reusability in OOP based software systems. In[30] estimated coupling and cohesion in between objects in software. In addition, they estimated the relationship between classes, direct and indirect dependencies, input-output dependencies. In[31] empirically validated the object oriented metrics for predicting the fault proneness from the models. They have exhibited the performance of machine learning methods are better than the logistic regression method with respect to all the severities of faults. Ruchika Malhotra also investigates the relationship between the object-oriented metrics and change proneness[32]. Researchers in[33] explored machine learning schemes for anomalies detection in component-based applications. They applied historical data and processed for correlation analysis with the collected historical data elements. Later they introduced synthetic aging conditions such as CPU contention and memory leaks in the application to assess software quality. In[34] Ruchika defined how to find the defect by using the Multi-nominal Multi vitiate Logistic Regression as well as the two machine learning methods like MLP (Multi Layer Perception) and DT (Decisions Tree). Again[35] used the machine learning methods to improve the software quality as well as find the fault and predict.

3. Some Popular Object Oriented Metrics

There are numerous researcher pointed out about the OOM (Object Oriented Measurement) in the literature. These metrics suites are given below in the tabular form. Chidambaram SR. Kemerer CF (1994) they proposed the six metrics which able to predict and identify

Table 1. CK metrics suite

Name of the Metrics	Description of the Metrics	Purpose	Reference
CBO	It is the calculation of sum of all the classes to whom it is coupled as well as vice-versa	Measuring for coupling	[2,5]
WMC	This metrics measures the complexities in terms of weights in the class	Estimation the Size of the class	[2,5]
RFC	The number of functions or procedures that can be potentially be executed in a class	Measuring coupling	[2,5]
LCOM	This can be potentially measure each data field in a class and the percentage of methods in the class Using the data field.	Only for cohesion	[2,5]
NOC	The number of children can be measure as the number of immediate subclasses of a class	Used for Inheritance only	[2,5]
DIT	It is represented as the level of the class in the Inheritance hierarchy, with the root class being Zero. DIT= maximum inheritance path from the class to the root class	Used for Inheritance only	[2,5]

the quality attribute in the software product. Reusability measurement can be done automatically provided that the researcher must developed such type of application where object oriented algorithms must be executed. A proposed model is developed for evaluation of reusability from the code which is described in the Section 6.1.1. This model accomplished 4 different phases where the developed develop the algorithms for OOM. Not only the researcher focuses the algorithmic part as well as to develop the product in a structured approach to make the code us reused, efficient and developed the high quality product. It is identifying as the key factor for measurement and testability of the object oriented code.

This paper is arranged in the different sections. Section 1 and 2 focuses the literature survey and importance of the object oriented metrics and its reusability. Section 3 importance is about the popular metrics which is described by the CK.

4. Most Reusable Components

During literature survey we found some of the reusable factors which impact more on the software development and some of them are listed in the tabular form

When we want to achieve the reusability feature in the software code then it should have planned on purpose and must be an organized one, provided that it is to give large payoffs. Reusability enhances the productivity and quality. A metric is a quantitative indicator of an attribute. A model specifies relationships between metrics.

5. Problem Specification

How to estimate the reusability factor in the object oriented code?

6. Research Methodology

The proposed metrics will be developed which will measures the reusable codes in the multi paradigm languages like (C++, Java Script, Ruby, Php, Perl, C# etc). Specially for the validation purpose the Java Script, C# and Python are used as Multi Paradigm in the research. The proposed metrics is the combination of one of the six metrics developed by Chidambaram SR. Kemerer CK (1994)[2]. These metrics are DIT, NOC, WMC, RFC and LCOM. In this proposed synopsis we have studied and measure the depth of CK metrics. The reusability is the prime attribute of

Table 2. Reusable factors

Sl No	Name of the reusable component	Meaning of the reusable components	Action/Result
1	Data Hiding	Packing of data and function together.	The more number of capsulation that we use, then more chances of reusability occurs.
2	Complexity	How best the metrics values are calculated.	If the metrics are having less complex then chances of reusable component is more
3	Members of the Class	the variables which are used in the class	It has been observed that if we take less number of class variables then we get the higher reusability.
4	Cohesion	It means the degree to which independent components are required to carry out the similar job.	If we will use more cohesion then we will enhance reusability.
5	NOC	It means the number of children's in the class	If more number of children's are available then more chances of reusability
6	Coupling	It means how one object interact with other objects	If less number times interaction occurs in between the objects then we will get more reusability
7`	Size of the program	It is the sum of Local + Global variables + all the modules	If the program size is more then there is less chance of reusability
8	Genericity	The level to which the module has been ready to perform the generic. This is one of the key attribute in the process of reusability.	It rivets the different domains of the particular class used and provides better level of the programmer.
9	Portability	How the classes are easily reused in another classes	Significantly used in the project

the software development environment. We found that when the reusability increases the reciprocally it increases of depth of inheritance as well as number of children. In other hands if the reusability is decreases then simultaneously it increase the couple between object as well as lack of cohesion.

6.1 Proposed Metrics

There are 3 different proposed metrics are defined which the combination of CK metrics.

Proposed Metrics -1 (WMPRC)

Weighted Method Per Class + Response for Class
Estimate and compare the complexity of the metrics

Figure 1. WMPRC.

Proposed Metrics -2 (DITNC)

Coupling Between the Objects + Lack of Cohesion Methods
Estimate the complexity of the metrics

Figure 2. DTNC.

Proposed Metrics -3(CBLCM

Depth of Inheritance (DIT) + Number of Children (NOC)
Estimate the complexity of the metrics

Figure 3. For CBLCM

6.1.1 Proposed Model for Object Oriented Metrics Evaluation

This proposed model is helpful for the researchers who will implement the Object Oriented Metrics. Some of the steps are available these are as below:

- The first step is the fundamentals of requirement.
- Evaluation by using different parameters like theoretical or scale measurement.
- Validation by using the empirical study.
- Identifying the Threshold.

6.2 Proposed Metrics-01 (WMPRC) Weighted Method per Class + Response for Class

Case I: If we have more number of methods in the class then more impact will be on the children class and restrictive the possible of reuse. From the above we conclude that WMC gives the not a good for reusability of a class.

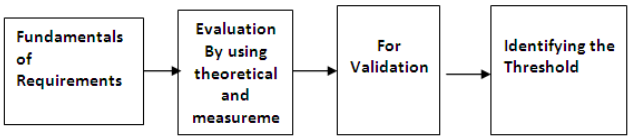

Figure 4. For Proposed model for Object Oriented Metrics evaluation.

Case II: If the bigger no. of the functions that can be accessed from the class through the message then found the more complexity of the class. Like WMC, RFC also have not a positive sign regarding the reusability.

6.3 Proposed Metrics 02 (DITNC) Depth of Inheritance (DIT) + Number of Children (NOC)

In the inheritance if we make maximum depth of class hierarchy then found more chance for reusability of the inherited metrics. It proves that if more number of reusability of a class increases subsequently depth of inheritance increases. So DIT has positive sign on the reusability of the class. If reasonable value for number of children then more scope of reuse in the class. We conclude that if more reusability found in the class then increase the both depth of inheritance as well as number of children. Both have a good impact on the reusability of the classes.

6.4 Proposed Metrics 03 (CBLCM) Coupling Between the Objects + Lack of Cohesion Methods

Case I: If we will use extremely coupling between the objects in the class then there will be a data hiding problem so that again it suffer the reusability problem. It indicates that CBO has not a positive sign in the context of reusability in the class.

Case II: If there will be a more number of lacks of cohesion in the class then increases the more complexity. So LCOM signifies that non positive result in the reusability of the class

7. Implementation

In this part of this article we have taken as C++ program as the key language where we have applied the proposed

metrics in the concept of Constructor and Destructor. Our objective of taking Constructor is that to calculate the how much reusability in concern classes. Both the cases we conclude that not in the favor of reusability in the class.

8. Motivational Example of Source Code

```
#include<iostream.h>
#include<conio.h>
Class GIET
        {
                int a;
                public:
         GIET(int x)
        {
                 a=x;
        }
         int Accepta()
        {
                return a;
        }
};
class GITA: public GIET
        {
                int b;
         public:
 GITA(int x, int y):GIET(y)
        {
                b=x;
        }
          int Acceptb( )
        {
                return b;}        };
           class GEC : public GITA
                {
        int c;
        public:
GEC(int x, int y, int z):GITA(y,z)
                {
                        c=x;
        }
        void put( )
                {
        cout << Accepta( ) <<" " << Acceptb( ) << " " ;
                cout << c << "\n";
                }
                };
int main( )
        {
        GEC ob(1,2,3);
        ob.put( );
        cout << ob.Accepta( ) <<" " <<
        ob.Acceptb( ) << "\n";
        getch( );
        return(0);
}
```

9. Execution Screen

In the above source code we get the output as well as we have estimated the metrics values of the proposed metrics .There are 3 proposed metrics are defined and collected the metrics values and formed the data set. From the table (), the dataset have been prepared and plot the graph.

10. Metrics Values Generated through the Proposed Metrics

In the below Table 4 the values are derived from the above source code and created the data set. The graph is plotted according to the data set. This Table contains the values of the proposed metrics which exhibits the relationship between the CK metrics and conclude that the first proposed metric that is WMPRC is negligible as compared to proposed metrics-2 and 3.

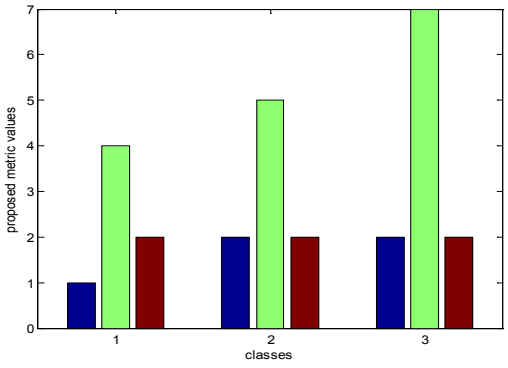

Figure 6. Represents the analysis of proposed graph.

11. Graph for Proposed Metrics

The graph is plotted as per the data sets calculated in the above Table 4. The values are derived in the source code which is written in the C++ .

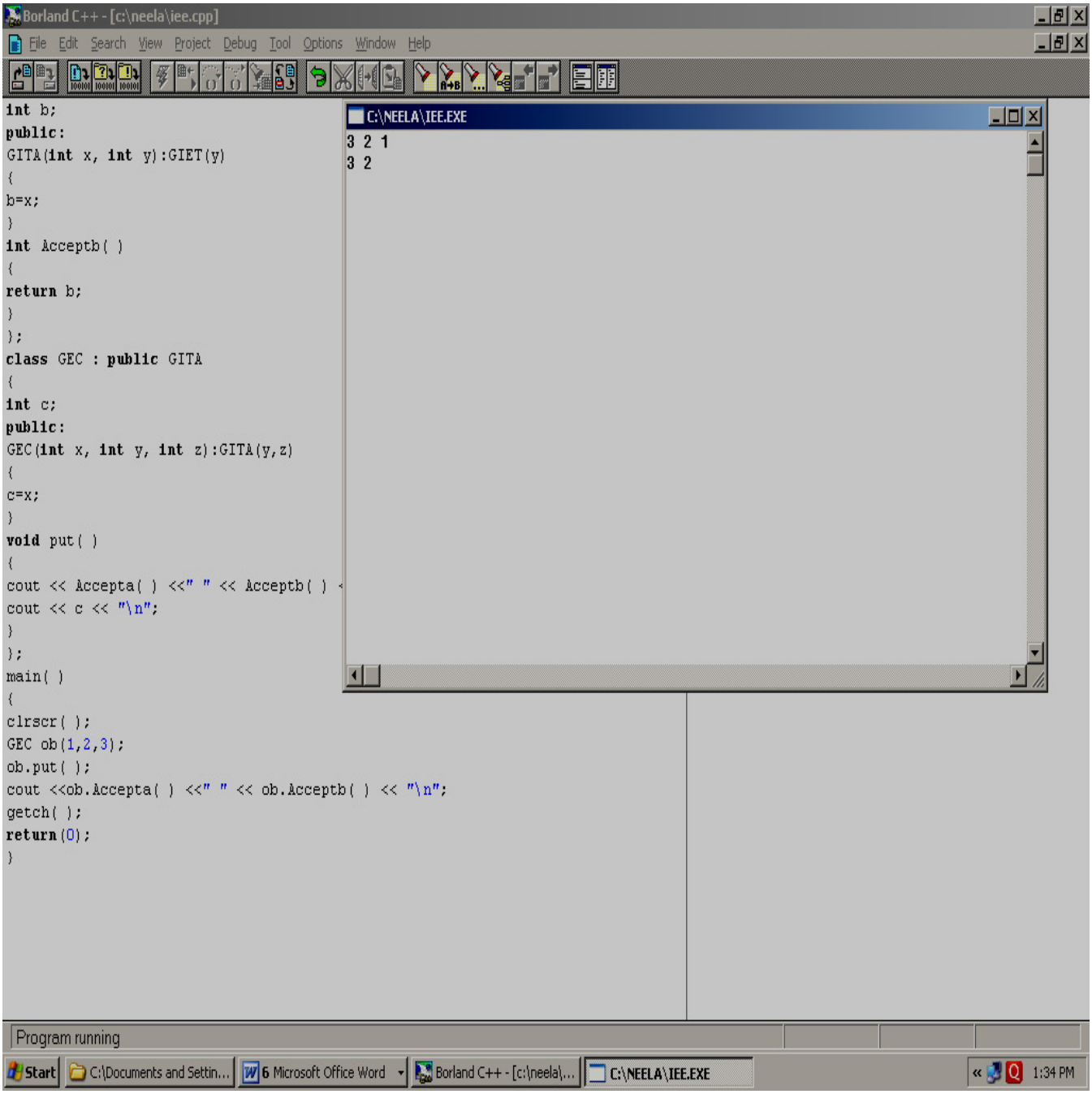

Figure 5. Source code output screen.

11.1 Analysis of the Proposed Metrics and Graph

The graph is implemented on the MATLAB 11.0 version and a data set has been generated through the program which is represented in the Table 3. In the above the graph and the table we found that the proposed metrics-1 i.e. (Depth of Inheritance and Number of children) are smaller amount as comparison to proposed metrics-2 and proposed metrics-3. From the above facts we conclude that the OOS (Object Oriented System) using the parameterized Constructor in C++ programs is more reusable up to some extent. When we will get the larger ethics (values) of proposed Metrics-2 and 3-them definitely it gives the negative collision on the reusability. So the constructor having parameters (parameterized Constructor) gives the negative impact on the reusability of the class.

Table 4. Data set created from the above source code

Ino Serial Number Sl No	Name of the Metrics Used in the class for Reuse	Depth of Inheritance Metrics (DIT)	Number of Children Metrics (NOC)	Proposed Metrics-1 (DIT+NOC)	Weighted Method per class (WMC)	Response for Class (RFC)	Proposed Metrics:2 WMC+RFC)	Coupling Between the Objects (CBO)	Lack of Cohesion Methods (LCOM)	Proposed Metrics: 3: (CBO+LCOM)
1	2	3	4	5	6	7	8	**9**	**10**	**11**
1	GIET(Parent Class	Zero	One	One	Two	Two	Four	One	One	Two
2	GITA (Intermeditory Base class	One	One	Two	Two	Three	Five	One	One	Two
3	GEC(Child Class)	Two	Zero	two	Two	Five	seven	One	One	Two

12. Conclusion and Future Scope

In the above literature review we found the several profits and drawbacks with the object oriented metrics using the software reusability. The reusability and OOM (Object Oriented Metrics) are closely related to each other. In this paper, we have applied the reusability concept as part of the CK metrics. The reusability is the prime attribute of the software development environment. We found that when the reusability increases the reciprocally it increases of depth of inheritance as well as number of children. In other hands if the reusability is decreases then simultaneously it increase the couple between object as well as lack of cohesion. The concept of reusability has several benefits like economical, qualitative and time management. The researcher should developed in such type of frame work where the reusable components are being used in the life cycle process models so that the model will provide qualitative reusable components in the phase of SDLC. The challenging task of the researcher is to get benefits of reusability by using the Weyuker's Properties and how these properties will be verifies it is another new challenging task.

13. References

1. Li W, Henry S, Kafura D, Schulman R. Measuring object oriented design. Journal of Object Oriented Program. 1995; 8(4):48–55.
2. Chidambaram SR, Kemerer CF. A metrics suite for object oriented design. IEEE Transaction. Software Engineering.1994; 20(x):476–93.
3. Lorentz M, Kidd J. Object-Oriented software metrics. Prentice Hall object-oriented series. Englewood Cliffs: Pretice Hall; 1994.
4. Henderson-sellers B. Object-oriented metrics. Measures of complexity. New Jersey: Prentice Hall; 1996.
5. Aggarawal KK, Singh Y, Kaur A, Malhotra R. Empirical study of Object-Oriented metrics. Journal of System Software. 2006; 23:111–22.
6. Taylor D. Object-Oriented Technology: A managers Guide. Second Printing. Reading U.S: Addison-Wesley; 1990.
7. Aggarawal KK, Singh Y. Software engineering. 2nd ed. Delhi: New Age International Publishers; 2007.
8. Harrison R, Counsell SJ, Nithi RV. An evaluation of the MOOD set of Object Oriented Software Metrics. IEEE Transactions on Software Eng.1998; 24(6): 491–6.
9. Binder RV. Object-oriented software testing. Communications of the ACM. 1994; 37(9):28–9.
10. Purao S, Vaishnavi VK. Product metrics for Object Oriented Systems. ACM Computing Surveys. 2003; 35(2):191–221.
11. Lorenz M, Kidd J. Object-oriented software metrics. Englewood Cliffs, New Jersey: Prentice Hall; 1994.
12. Dubey SK, Rana A. A comprehensive assessment of Object-Oriented Software Systems using metrics approach. IJCSE. 2010; 2(8):2726–30.
13. Costagliola G, Ferrucci F, Tortora G, Vitiello G. Class points: An approach for the size estimation of object-oriented systems. IEEE Transactions on Software Eng. 2005; 31(1):52–74.

14. Basily VR, Briand LC, Melo WL. A validation of object oriented design metrics as quality indicators. IEEE Transactions on Software Eng. 1996; 22(1): 751–61.
15. Babsiya J, Davis CG. A hierarchical model for object oriented design quality assessment. IEEE Transactions on Software Eng. 2002; 28:4–17.
16. Briand LC, Wust J. Modelling development effort in object oriented system using design properties. IEEE Transactions on Software Eng. 2001; 27(11):963–86.
17. Kim K, Shin Y, Wu C. Complexity measures for object-oriented program based on the entropy. Proc Asia Pacific Software Eng; 1995. p. 127–36.
18. Olague HM, Etzkorn LH, Gholston S, Quattlebaum S. Empirical validation of three software metrics suites to predict fault-proneness of object-oriented classes developed using highly iterative or agile software development processes. IEEE Transactions on Software Eng. 2007; 33(6):402–19.
19. Nair TRG, Selvarani R. Estimation of software reuability: An Engineering Approach, Association for Computing Machinery (ACM)-SIGSOFT, USA. 2010; 35(1).
20. PS, Aashima, Kakkar P, Sharma S. A survey on software reusability.IEEE; 2010.
21. Aggarwal KK, Singh Y, Kaur A, Malhotra R. Software reuse metrics for object-oriented systems. IEEE; 2005.
22. Conte SD, Dunsmore HE, Shen VY. Software engineering metrics and models. Benjamin/Cusmmings; 1986. p. 62–70.
23. Singhani H, Suri PR. Testability assessment model for object oriented software based
24. on internal and external quality factors. Global Journal of Computer Science and Technology: C Software and Data Engineering. 2015; 15(5).
25. Goel BM, Bhatia PK. Analysis of reusability of Object-Oriented System using CK metrics. International Journal of Computer Applications. 2012 Dec; 60(10):32–6. 0975 – 8887.
26. Gupta D, Goyal VK, Mittal H. Comparative study of soft computing techniques for software quality model. International Journal of Software Engineering Research and Practices. 2011; 1(1).
27. Nuruzzaman M, Hussain A, Tahir HM. Towards increasing web application development productivity through object-oriented framework. International Journal of Future Computer and Communication. 2013 Jun; 2(3).
28. Kumar M, Aloysius A. A review on component based software metrics. International Journal of Fuzzy Mathematical. 2015; 7(2):185–94.
29. Sing PK, Om SP, Sing AP. A framework for assessing the software reusability using fuzzy logic approach for aspect oriented software. I. J. Information Technology and Computer Science. 2015; 02:12–20.
30. Gandhi P, Bhatia PK. Reusability metrics for object-oriented system: An alternative approach. IJSE. 2010; 1(4):63–72.
31. Patidar K, Gupta RK, Chandel GS. Coupling and cohesion measures in object oriented programming. International Journal of Advanced Research in Computer Science and Software Engineering. 2013 Mar; 3(3).
32. Malhotra R, Singh Y, Kaur A. Empirical validation of object-oriented metrics for predicting fault proneness models. Int Journal of Software Quality. 2010. p. 3–33.
33. Malhotra E, Khanana M. Investigation of relationship between object-oriented metrics and change proneness. Int Journal of Machine Learning and Cyber. 2013. p. 273–86.
34. Maggo S, Gupta C. A machine learning based efficient software reusability prediction model for java based object oriented software. I J Information Technology and Computer Science. 2014; 02:1–13.
35. Malhotra R, Jindal R, Jain A. Prediction of defect severity by mining software project reports. Int Journal of System Assurance Engineering and Management. Springer; 2016.
36. Malhotra R, Jain A. Fault prediction using statistical and machine learning methods for improving software quality. Int Journal of Information Processing Systems. Korean Journal. 2016; 8(2).