International Journal of Engineering, Science and Mathematics
Vol. 7Issue 4, April 2018,
ISSN: 2320-0294 Impact Factor: 6.765
Journal Homepage: http://www.ijesm.co.in, Email: ijesmj@gmail.com
Double-Blind Peer Reviewed Refereed Open Access International Journal - Included in the International Serial Directories Indexed & Listed at: Ulrich's Periodicals Directory ©, U.S.A., Open J-Gage as well as in Cabell's Directories of Publishing Opportunities, U.S.A

EVALUATION AND APPLICABILITY OF WEYUKER'S PROPERTIES ON SOFTWARE METRICS FOR COMPONENT BASED SOFTWARE SYSTEM

Pooja Rana*
Rajender Singh**

Abstract

Background/Objectives:Component-based software engineering (CBSE) is a process of reusing pre- built software components to built a new software. CBSE is based on good software engineering design principles. CBSE is based on black box technique, in which the implementation of components are hidden in nature and the communication between the components is through well-defined interfaces. Component platforms are shared and help in reducing the development costs. To determine the complexity of a software different software metrics are used. It is predetermined that for fineness in software complexity the cohesion should be high and coupling should be low. In our approach we are determining the reusable components of a software system and enhancing the accuracy of the methods for determining them. Method : an attempt has been made to present an analytical evaluation on cohesion metrics which was proposed by Rana and Chillar [1] against Weyuker's[2] properties and an attempt has also been made to present the results of empirical evaluation based on the case study. Java Beans has been used for validating the Metrics. Findings: The Result of the present study is quite satisfactory and may further help in estimation of the complexity of components. In our approach we have found the reusable components of a software system by not only finding the dependencies between the different elements of a single package but also the finding dependencies between the elements of different packages of an already existing project that makes our approach different from existing study.

Keywords:

Component based software engineering(CBSE), Cohesion, complexity, Weyuker properties.

Author correspondence:

Correspondence Author
*Pooja Rana
Research Scholar
Department of Computer Science and Applications
Maharashi Dayanand University, Rohtak

**Rajender Singh
Professor
Department of Computer science and Applications
Maharashi Dayanand University, Rohtak

International Journal of Engineering, Science and Mathematics
Vol. 7Issue 4, April 2018,
ISSN: 2320-0294 Impact Factor: 6.765
Journal Homepage: http://www.ijesm.co.in, Email: ijesmj@gmail.com
Double-Blind Peer Reviewed Refereed Open Access International Journal - Included in the International Serial Directories Indexed & Listed at: Ulrich's Periodicals Directory ©, U.S.A., Open J-Gage as well as in Cabell's Directories of Publishing Opportunities, U.S.A

1. Introduction

Software components are pre-fabricated blocks are designed to perform specific tasks and are capable to communicate with each other using industry standard messaging interfaces. The components are larger modules that represent a higher level of task or functionality, different from software objects.. A component has an external specification which is independent of its internal mechanisms and can be deployed as a black box. Component based software engineering (CBSE) denotes the process of building software by using pre-built or pre-existing software components based on the meaning of software components. Metrics and their Measurements are an important element for controlling the process of software engineering. Software metrics are quantifiable measures that are used to measure different characteristics and features of a software development process or the software system itself. Software metrics plays a vital role in assessing and predicting the various attributes of software such as maintainability, reusability, testability complexity, etc. Among all these attributes, the complexity factor affects all other attributes of the software e..l Gill and Balkishan[9]. Software metrics are essential to predict plan, execute, monitor, control and evaluate the processes and products .

The primary aim of the software metrics is to reduce the costs, Improve quality, Control and Monitor the time schedule, reduce the testing efforts, and help in effective use of reusable blocks or fragments. The paper is organized in various sections; section 2 takes literature review of some basic cohesion metrics. Section 3 deals with the theoretical evaluation of metrics using Weyuker properties, section 4 describes the empirical evaluation of the metrics and at the last paper concludes with a discussion of the impact and implications of the research..

2. Traditional Cohesion Metrics

Cohesion can be defined as the measure of strength of the association of elements and objects within a module. In other words, it is the extent to which all elements and instructions within a module relate to a single given function.

LCOM (Lack of Cohesion in Methods) is one of the metric from the CK Suit e.l. Chidamber[3]. It was later modified to LCOM2 e.l Chidamber[13]. In the empirical study LCOM2 is not used because it cannot differentiate two software by providing them cohesion value as zero. LCOM and LCOM2 do not consider the method of invocation. Li proposed RLCOM[10] in 2000. It is an extension of LCOM in which the number of non-similar method pairs is divided by the total number of method pairs.
In 1995, Hitz and Montazeri [11] proposed a cohesion metrics (LCOM3). Its an improved version of LCOM. It shows the relationship between methods of a class by an undirected graph. Methods of a class are the nodes. There should be an edge if at least one attribute is common in two methods It should be noted that LCOM, LCOM3, and RLCOM are in fact measures of lack of cohesion.

TCC(Tight class cohesion), it measures cohesion rather than its absence. TCC (Tight class cohesion) was proposed by Bieman and Kang in 1995 [12]. These measure consider common attributes is to be used by methods, these measures also consider invocation between methods. If a method *m* invokes another method *n*, all attributes used in method *n* would be used by method *m* as well. Two methods are called connected if they use (by referencing or invoking) common attributes These cohesion metrics consider similarity of method as an intransitive relationship. LCOM3 and TCC incorporate indirect relationships between the methods. LCOM3 and TCC treat indirect and direct cohesion in a similar manner e.l Gandhi and Gui[7,8].

International Journal of Engineering, Science and Mathematics
Vol. 7Issue 4, April 2018,
ISSN: 2320-0294 Impact Factor: 6.765
Journal Homepage: http://www.ijesm.co.in, Email: ijesmj@gmail.com
Double-Blind Peer Reviewed Refereed Open Access International Journal - Included in the International Serial Directories Indexed & Listed at: Ulrich's Periodicals Directory ©, U.S.A., Open J-Gage as well as in Cabell's Directories of Publishing Opportunities, U.S.A

3. Theoretical Evaluation Of Cohesion Metrics Using Weyuker's Properties 's

Weyuker, Gandhi, Kumar [2,7] proposed several properties for evaluating complexity of the software. Rana and Chillar[1] proposed some cohesion complexity metrics, here those metrics are evaluated against Weyuker properties for compatibility. The properties are:

Let M be Metric of component A and B

Property 1:

Given a component A another component B can always be found such that, M(A) ≠M(B). This implies that not every component can have the same value for a metric[2,7].

Property 2:

Let C be a nonnegative number. Then there are finitely many components A for which M(A)= C.[2,7]

Property 3:

There are distinct Components A and B such that, M(A) = M(B).[2,7]

Property 4:

There exist component A and B having same functionality but their complexities could be different.[2,7]

$$(\exists A)(\exists B)(A \equiv B)\ \&\ (M(A) \neq M(B))$$

Property 5:

This property is the property of monotonic. When two components are concatenated, their metric value should be greater than the metrics of each of the individuals.[2,7]

$$(\forall A)(\forall B)(M(A) \leq (M(A+B))\&M(B) \leq M(A+B))$$

Property 6:

This property suggests non-equivalence of interaction. If there are two program bodies of equal metric value which, when separately concatenated to a same third program, yield program of different metric value. For components A, B & C [2,7]

$$(\exists A)(\exists B)(\exists C)(\ M(A) = M(B)\&M(A+C) \neq M(B+C))$$

Property 7:

Two components A and B where the body of B component is formed by permuting the order of statements of A and B[2,7]

Property 8:

It specifies that "if A is a renaming of B, then M(A) = M(B).[2,7]

Property 9:

This property states that the sum of the metric values of a component could be less than the metric value of the concatenated component.[2,7]

International Journal of Engineering, Science and Mathematics
Vol. 7Issue 4, April 2018,
ISSN: 2320-0294 Impact Factor: 6.765
Journal Homepage: http://www.ijesm.co.in, Email: ijesmj@gmail.com
Double-Blind Peer Reviewed Refereed Open Access International Journal - Included in the International Serial Directories Indexed & Listed at: Ulrich's Periodicals Directory ©, U.S.A., Open J-Gage as well as in Cabell's Directories of Publishing Opportunities, U.S.A

$$(\exists A)(\exists B)(M(A)+M(B)<M(A+.$$

The above Weyuker's properties are evaluated for cohesion metrics Cohesion of variables with in component (COVC) and Cohesion of methods within component(COMC) metrics proposed by Rana and Chillar [1] as described below.

1. Two different components can have different complexities. This means that not every component will have the same value of the metrics. Hence this will satisfy the first Weyuker property
2. As each component will have at least one method with some instance variables. Therefore the value of complexity metrics will be some positive value. Thus validating the second property
3. Two different components with different functionality can have same complexity metric values. This implies that the components may have same number of variables and methods with same frequency but with different functionality and hence the value of complexity metrics calculated could be same. So it satisfies property 3
4. When two components functionalities are same then complexities of both the components may be different as there may be possibility that these components may be using different programming approach and frequency of instance variables and functions may also differ at the same time. Hence It validates property number 4.
5. Concatenation of two components will give integrated component. In integrated component the total number of instance variables and the frequency of instance variables will be on the higher side. Calculations for the complexities show that this fifth property is partially satisfied i.e. it is possible $(\forall A)(\forall B)(M(A) \leq M(A+B)\&M(B) \geq M(A+B)$ or vice versa.
6. Two components with the same complexity means both will have same frequency of instance variables and methods. However, they may be developed by using different programming methodologies and therefore when integrating in the system, both may have different integration code and implementation thus resulting in different complexities of the system in both the cases. This validates property number

6. If the ordering of statements or methods in a component is changed, then it will not change the complexity of the new modified component. So, this property is not satisfied by given Cohesion metrics

7. Renaming of a variable/method of a component will not affect the complexity of that component, thus satisfying the property no 8.
8. In given cohesion metrics the sum of metric value of components will not be less than metric value of integrated component .Thus this property is not satisfied

4. Empirical Evaluation of Cohesion Metrics

A. Validation on Component Based Software (Java Beans)

To validate proposed complexity metrics, an experiment is conducted on the component based software which is implemented in Java using Java Beans. This software has many java bean components having different number of instance variables and methods. have same frequency of the variables and some have different frequency. According to these frequencies the value of COVC [1]is calculated.

International Journal of Engineering, Science and Mathematics
Vol. 7Issue 4, April 2018,
ISSN: 2320-0294 Impact Factor: 6.765
Journal Homepage: http://www.ijesm.co.in, Email: ijesmj@gmail.com
Double-Blind Peer Reviewed Refereed Open Access International Journal - Included in the International Serial Directories Indexed & Listed at: Ulrich's Periodicals Directory ©, U.S.A., Open J-Gage as well as in Cabell's Directories of Publishing Opportunities, U.S.A

1) Cohesion of Variables within a Component (COVC)[1]:

Cohesion of variables in a component represents the frequency of different type of variables binds or strengthens the component. In the example there are thirteen components C1 to C13. In each component there are some instance variables and methods. The table1 shows the frequency of different type of variables (simple, moderate and complex). Some components

TABLE I.

TABLE 1 SHOWS THE FREQUENCY OF DIFFERENT TYPE OF VARIABLES OF DIFFERENT COMPONENTS

Component	***Fvsi***	***Fvmi***	***Fvci***	***Tv***	***FIV***	***COVC***
C1	2	2	2	3	1.2	0.40
C2	2	2	2	3	1.2	0.40
C3	4	2	3	4	1.7	0.43
C4	4	2	3	4	1.7	0.43
C5	2	28	15	16	10.3	0.64
C6	2	4	3	4	1.9	0.48
C7	0	3	2	2	1.2	0.60
C8	4	2	3	4	1.7	0.43
C9	2	2	2	3	1.2	0.40
C10	2	4	3	4	1.9	0.48
C11	0	3	2	2	1.2	0.60
C12	2	2	2	3	1.2	0.40
C13	2	2	2	3	1.2	0.40

$$COVC=\sum_{i=0}^{n}\frac{FIV}{TV}$$

$$FIV=\sum_{i=0}^{n}\{[f(vsi)*Ws]+[f(vmi)*Wm]+[f(vci)*Wc]\}$$

Here

FIV = frequency of the instance variables within a component

TV= total no of Instance Variables in a component

F(vsi)= Frequency of occurrence of standard variables

F(vmi)= Frequency of occurrence of moderate variables

F(vci)= Frequency of occurrence of critical variables

Ws, Wm, Wc are the weight factors of the standard, moderate and critical type of variables respectively

International Journal of Engineering, Science and Mathematics
Vol. 7Issue 4, April 2018,
ISSN: 2320-0294 Impact Factor: 6.765
Journal Homepage: http://www.ijesm.co.in, Email: ijesmj@gmail.com
Double-Blind Peer Reviewed Refereed Open Access International Journal - Included in the International Serial Directories Indexed & Listed at: Ulrich's Periodicals Directory ©, U.S.A., Open J-Gage as well as in Cabell's Directories of Publishing Opportunities, U.S.A

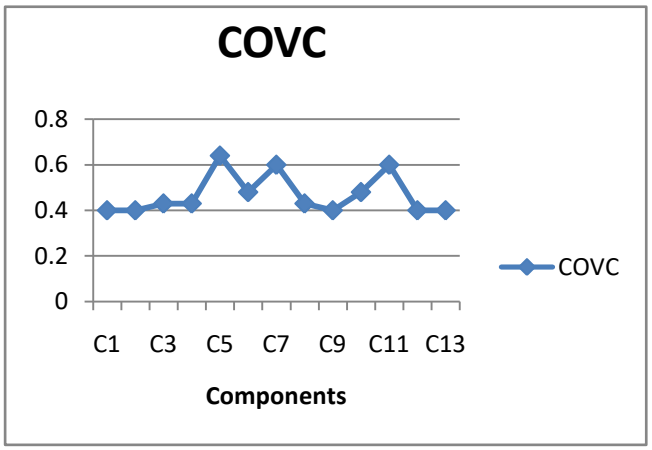


Figure 1 Graph represents the line graph of values of COVC of different components.

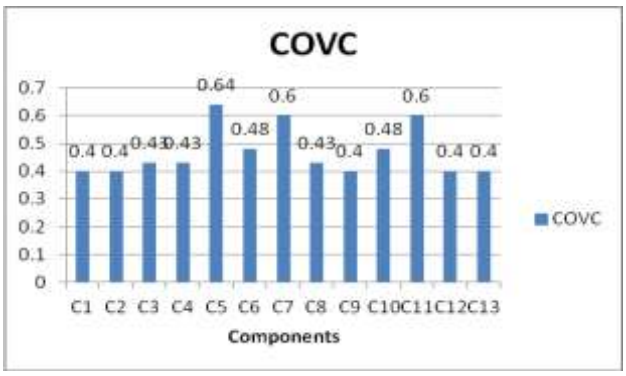


Figure 2 shows the Bar graph of COVC

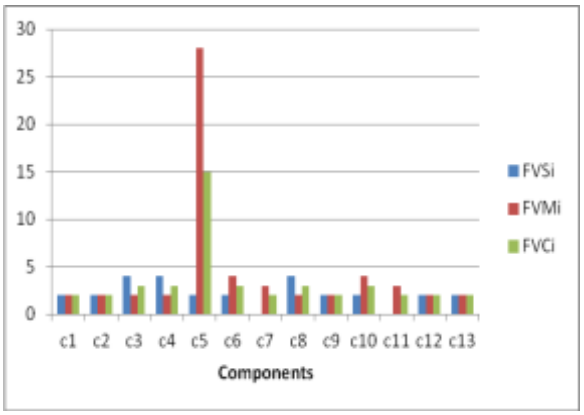


Figure 3 shows the frequency of different components.

C5,C6,C7,C10 & C11 components show that the more is the usage of Moderate instance variables within a component, the more is the Cohesion value (COVC) and hence the reusability factor for developing a totally new application becomes high.

International Journal of Engineering, Science and Mathematics
Vol. 7Issue 4, April 2018,
ISSN: 2320-0294 Impact Factor: 6.765
Journal Homepage: http://www.ijesm.co.in, Email: ijesmj@gmail.com
Double-Blind Peer Reviewed Refereed Open Access International Journal - Included in the International Serial Directories Indexed & Listed at: Ulrich's Periodicals Directory ©, U.S.A., Open J-Gage as well as in Cabell's Directories of Publishing Opportunities, U.S.A

C1, C2, C3,C4,C8,C9,C12 & C13 components have more no. of simple instance variables, followed by Complex instance variables and have minimum Moderate instance variables. Here the Cohesion values are on the lower side indicating the low strength of the overall component.

Conclusion is that the use of Moderate instance variables within a component should be high for having the best results for the strengthening of the component, which results reusability of the component for developing a new application.

2) Cohesion of Methods within Component (COMC)[1]

Cohesion of Methods in a component refers to the relatedness of methods and instance variables of a component [1]. This metrics considers the interaction between the methods with in a component [1]. In the example there are thirteen components C1 to C13. Each component has some instance variables and methods. Table2 shows the number of methods which are using simple type, moderate type and complex type variables. According to these frequencies the value of COMC is calculated.

$$COMC = \sum_{i=0}^{n} \frac{COM}{TM} \qquad (2)$$

$$COM = \sum_{i=0}^{n} \{(Msi * Ws) + (Mmi * Wm) + (MCi * Wc)\}$$

COM = count of methods that use same type of variables

TM= total no of methods

Msi= sum of methods that use Standard type of variables.

Mmi= sum of methods that use Moderate type of variables.

Mci= sum of methods that use critical type of variables.Ws, Wm, Wc are weight factor for standard, moderate and critical type of variables[1]

TABLE 2 HOWS THE FREQUENCY OF METHODS USING DIFFERENT TYPE OF VARIABLES OF DIFFERENT COMPONENT

Component	***Msi***	***Mmi***	***Mci***	***Tm***	***COM***	***COMC***
C1	2	2	2	4	1.2	0.30
C2	2	2	2	4	1.2	0.30
C3	4	2	3	6	1.7	0.28
C4	4	2	3	6	1.7	0.28
C5	2	28	15	30	10.3	0.34
C6	2	4	3	6	1.9	0.32
C7	0	2	2	2	1.0	0.50
C8	4	2	3	4	1.7	0.43
C9	2	2	2	4	1.2	0.30
C10	2	4	3	6	1.9	0.32
C11	0	2	2	2	1.0	0.50
C12	2	2	2	4	1.2	0.30
C13	2	2	2	4	1.2	0.30

International Journal of Engineering, Science and Mathematics
Vol. 7Issue 4, April 2018,
ISSN: 2320-0294 Impact Factor: 6.765
Journal Homepage: http://www.ijesm.co.in, Email: ijesmj@gmail.com
Double-Blind Peer Reviewed Refereed Open Access International Journal - Included in the International Serial Directories Indexed & Listed at: Ulrich's Periodicals Directory ©, U.S.A., Open J-Gage as well as in Cabell's Directories of Publishing Opportunities, U.S.A

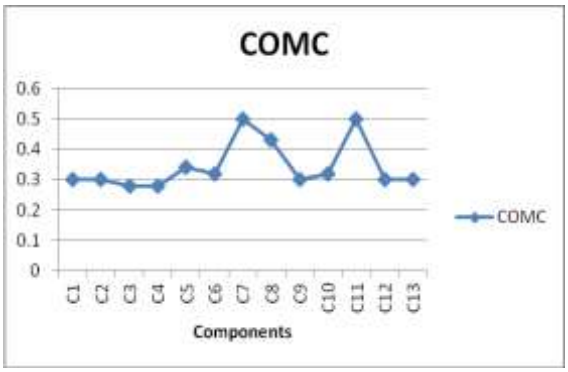


Figure 4 shows the line graph of COMC

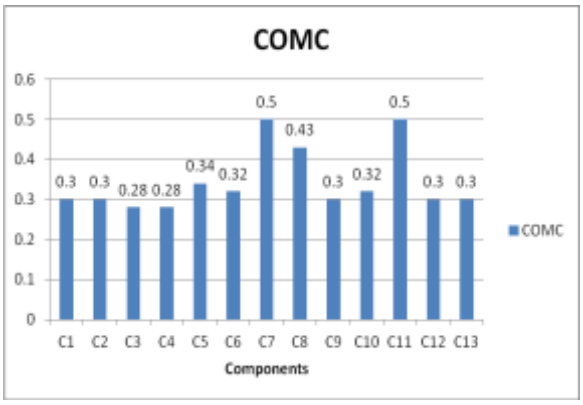


Figure 5 shows the bar graph of COMC

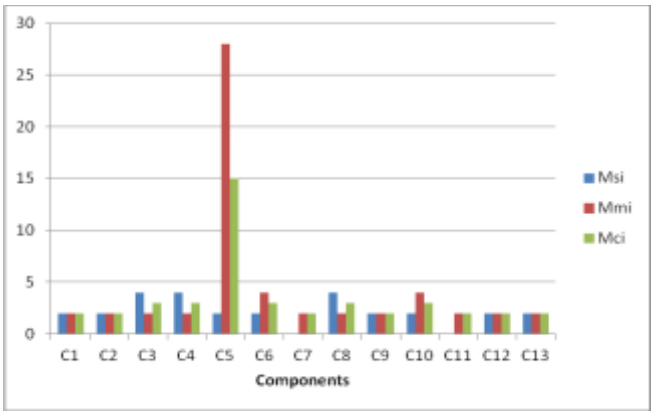


Figure 6 Shows the bar graph of frequency of methods used different type of variables

C5,C6,C7,C10 & C11 components show that the more is the usage of Methods using Moderate Variables within a component, the more is the Cohesion value (COMC) and hence the reusability factor for developing a totally new application becomes high.

C1, C2, C3,C4,C8,C9,C12 & C13 components have more no. of methods using simple instance variables, followed by Methods using complex instance variables and have minimum methods

using moderate instance variables. Here the Cohesion values are on the lower side indicating the low strength of the overall component.

Conclusion is that the usage of methods using moderate instance variables within a component should be high for the intensification of the component, which in turn supports the reusability of the component for developing a new application.

3)Total Cohesion Complexity of a Component (TCCC)[1]

Total cohesion complexity metrics calculated by addition of cohesion of variables (COVC) and cohesion of Methods (COMC) metrics. In the table there are thirteen components and TCCC is calculated by adding the values of COVC and COMC.

TCCC= COVC + COMC (3)

COVC= cohesion of variables within a component metric

COMC= cohesion of methods within a component metric[1]

Component	*COVC*	*COMC*	*TCCC*
C1	0.40	0.30	0.7
C2	0.40	0.30	0.7
C3	0.43	0.28	0.71
C4	0.43	0.28	0.71
C5	0.64	0.34	0.98
C6	0.48	0.32	0.8
C7	0.60	0.50	1.1
C8	0.43	0.43	0.86
C9	0.40	0.30	0.7
C10	0.48	0.32	0.8
C11	0.60	0.50	1.1
C12	0.40	0.30	0.7
C13	0.40	0.30	0.7

TABLE 3 SHOWS THE VALUES OF COVC, COMC AND TCCC OF DIFFERENT COMPONENT

International Journal of Engineering, Science and Mathematics
Vol. 7Issue 4, April 2018,
ISSN: 2320-0294 Impact Factor: 6.765
Journal Homepage: http://www.ijesm.co.in, Email: ijesmj@gmail.com
Double-Blind Peer Reviewed Refereed Open Access International Journal - Included in the International Serial Directories Indexed & Listed at: Ulrich's Periodicals Directory ©, U.S.A., Open J-Gage as well as in Cabell's Directories of Publishing Opportunities, U.S.A

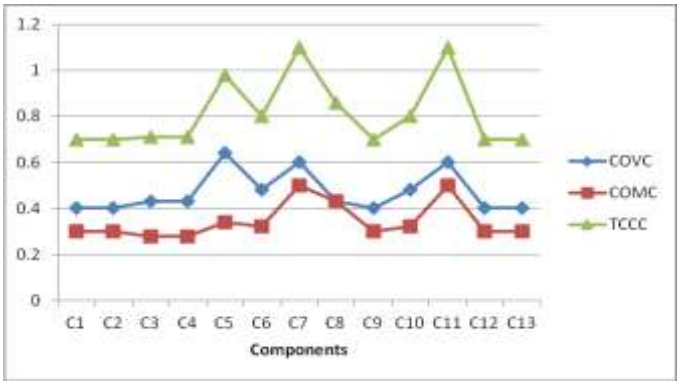
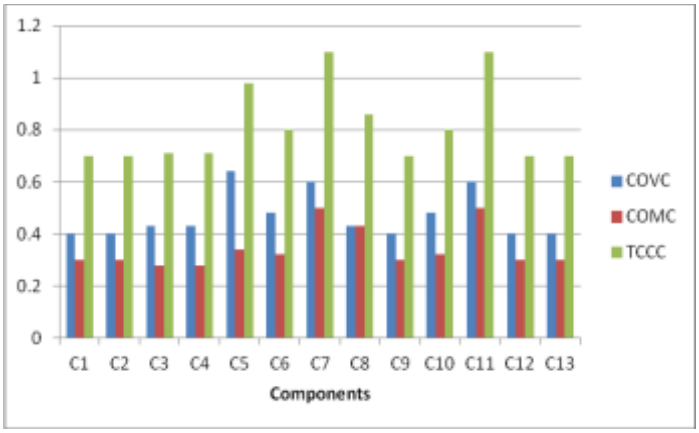


Figure 7 shows the comparison of COVC, COMC and TCCC (Using Line Graph)



`Figure 8 shows the comparison of COVC, COMC and TCCC (Using bar chart)

Graphs show the pictorial way to represent the result. From these graphs conclusion can be drawn, that C5, C6, C7, C8, C10, C11 components are highest strengthen component. Here the frequency of moderate type instance variables is high in proportion to the total number of variables and methods used.

5. CONCLUSION

In examples there are various components, every component have a class and having some member functions and some instance variables. The conclusion can be drawn after comparing the results .The deviation of the result depends on the frequency of instance variables. In standard rule Cohesion should be high and coupling should be low. Cohesion represents the strengthening s of variables and methods of the component. In the case study after implementing metrics on Java Beans software the values of COVC, COMC and TCCC are higher for C5, C6, C7, C8, C10, C11 components. After comparing their results these components are having high frequency of moderate type instance variables. Findings from this case study are the complexity (cohesion) of the component depends on the frequency of the variables and the type of variables. The result shows that these parameters affect the complexity of the component. The given cohesion complexity appears to be logical and fits the intuitive understanding but is not the only criteria for deciding the overall complexity of a CBSE.

Finally conclusion can be drawn that the usage of moderate instance variables within a component and methods using moderate instance variables within a component should be on the higher side for having the best results for the strengthening of the component (High Cohesion), which in turn supports the reusability of the component for developing a new application. For optimizing the result of proposed metrics genetic algorithm and MATLAB can also be one of the future works.

References

[1] PoojaRana, Rajender Singh (2016), "A Design of Cohesion and Coupling Metrics for Component based Software Systems", International Journal of Computer Applications, (0975 – 8887) Volume 146 – No.4, July 2016, PP- 23-27

[2] E. J. Weyuker. Evaluating software complexity measures. IEEE Trans. Software Engineering, Vol. 14, no. 9, 1988, pp. 1357–1365

[3] Chidamber, S.R. and Kemerer, C.K. towards a Metrics Suite for Object Oriented Design. Proceedings of 6th ACM Conference on Object Oriented Programming, Systems, Languages and Applications (OOPSLA'91), (Phoenix, Arizona,1991), 197-211.

[4] Hitz, M. and Montazeri, B. Measuring coupling and cohesion in object oriented systems. Proceedings of International Symposium on Applied Corporate Computing. (Monterrey,Mexico, 1995).

[5] Li, X, Liu, Z. Pan, B. and Xing, B.(2001) A Measurement Tool for Object Oriented Software and Measurement Experiments with IT. In Proc. IWSM 2000. (Lecture Notes in Computer Science 2006, Springer- Verlag, Berlin, Heidelberg, 2001),44-54

[6] Biemen, J. M. and Kang, B-Y. Cohesion and Reuse in an Object-Oriented System. In Proc. ACM Symposium on Software Reusability (SSR'95). (April 1995) 259-262.

[7] Gandhi parul and kumar bhatia pradeep (2012) Analytical Analysis of Generic Reusability Weyuker's Properties in International Journal of Computer Science Issues (IJCSI)

[8] Gui, Scott,(2008) New Coupling and Cohesion Metrics for Evaluation of Software Component Reusability, 9th International Conference For Young Computer Scientists, IEEE 2008. DOI 10.1109/ICYCS.2008.270

[9] Gill, N.S, Balkishan (2008): Dependency and Interaction Oriented Complexity Metrics of Component-Based Systems, ACM SIGSOFT Software Engineering Notes, 33 (2), pp. 1-5.

[10] Li, W. and Henry, S. Object-Oriented metrics that predictmaintainability. Journal of Systems and Software. 23(2) 1993 111-122

[11] Hitz, M. and Montazeri, B. Measuring coupling andcohesion in object-oriented systems. *Proceedings of* International Symposium on Applied Corporate Computing. (Monterrey, Mexico, 1995).

[12] Bieman, J. M. and Kang, B-Y. Cohesion and Reuse in anObject-Oriented System. In *Proc. ACM Symposium on Software Reusability (SSR'95)*. (April 1995) 259-262.

[13] Chidamber, S. R. and Kemerer, C. K. A Metrics Suite for Object Oriented Design. IEEE Transactions on Software *Engineering*, Vol. 20 (June 1994), pp.476-493.