Journal of Theoretical and Applied Information Technology
31st July 2018. Vol.96. No 14
© 2005 – ongoing JATIT & LLS

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

COMPARATIVE ANALYSIS OF COHESION METRICS FOR COMPONENT BASED SOFTWARE SYSTEM

[1]POOJA RANA, [2]RAJENDER SINGH
[1]Research scholar, Department of Computer Science and Applications, M.D. University, Rohtak, India
2 Professor, Department of Computer science and Applications, M.D. University, Rohtak, India
[1]poojakaul24@gmail.com, [2]chhillar02@gmail.com

ABSTRACT

Background/Objectives: Component-based software engineering (CBSE) is a process of reusing pre- built software components to build 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. Proposed: Two cohesion metrics are proposed Cohv(Cohesion of variables) and Cohm(cohesion of methods). Method : an attempt has been made to present an analytical and empirical evaluation of cohesion metrics proposed in this paper and comparison is drawn between different cohesion metrics which were proposed by Rana and Rajender Singh [11] and Yadav and Tomar[23]. 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 and SPSS tool is used to find out the correlation between different variables and metrics and T test is applied to find out the significance of the metrics. Findings: The Result of the present study is quite satisfactory and may further help in estimation of the complexity of components. The comparative analysis performed between proposed metrics and different cohesion metrics and find that the cohesiveness of proposed metrics is more than existing metrics and the possibility of reusability for developing new applications become high.

Keywords: *Cbse, Testing, Black Box, Metrics, Cohision*

1. INTRODUCTION

Software components are pre-fabricated blocks 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[5]. 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 Problem Description Section 4 Proposed Work Section 5 Comparative Analysis of different Cohesion Metrics at the last paper concludes with a discussion of the impact and implications of the research..

2. LITERATURE REVIEW

A) 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[4]. 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[11] 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 [9] 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 [1]. 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].

ICM(Interface Complexity Metric) is proposed by sharma[17]. It considers parameter and return values of its interface methods to measure the proposed metrics. It models the external behavior of the component as aggregation components methods and properties complexity factors. After validation the author concludes that the complex component takes more time to execute and hard to maintain and reuse.

BICM(Bounded Interface Complexity Metrics) is proposed by Tobias, Mwangi and Michael[19], this metrics is an extension of ICM. Its bounded that it may not necessarily grow with the size. The analysis of this metrics concludes that it is independent of interface size. BICM can be applicable in evaluating components self completeness, independence and portability.

Kartika Yadav and Tomer(2014)[25] proposed two metrics Cohesion in Class(CIC) and Cohesion between Component(CBM) for component based software system. These metrics are helpful for the improvement of CBSS design quality. These metrics are used to identify poorly designed classes and components in CBS.

Rana and Rajender Singh(2016)[13] proposed two metrics Cohesion of Variables within a Component(COVC) and Cohesion of Methods within a component(COMC). These metrics shows the relationship of variables used in different methods. The authors find that the complexity of the component depends on the frequency of the variables and the type of variables.

B) Limitations

i) In above discussion most of the static metrics consider direct coupling and cohesion between classes and direct similarity between methods. One of the cohesion metrics LCOM3 has suggested extension to incorporate indirect relationships between methods. It treats indirect and direct cohesion in the same way and cannot numerically specify the indirect and direct cohesion

ii) Another limitation is ICM grows with the size of the component interface i.e. the complexity of a component will increase with its size. This means that due to increased complexity the new improved component will be rated low, while that component is much more self contained. The analysis of BICM represent that it is independent of interface size but there is a need to analyze it on overall system not on one component.

3. PROBLEM DESCRIPTION

The aim of software engineering is to develop high quality software that can be maintained with very low cost. The quality of software can assess at different levels of software development. It can also be assessed at design level. In component based development system, the design of component has two perspectives internal and external. Component developer has more focus on internal design. If the internal design of the component is not good then the cost of the component automatically increases. To make component reusable the line of code is to be increased and efforts to update component would be more. Good design leads to the high component reusability and low dependency among component. 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[6]. 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. In this paper cohesion metrics are proposed to check the strength of the component. Our main focus is on internal attributes of a component like methods, variables, parameters etc.

4. PROPOSED WORK

Cohesion is the measure of strength of the association of elements within a component. In a truly cohesive component, all of the instructions in the component pertain to performing a single unified task. The cohesive component only needs to take the data it is passed, act on them, and pass its output on to its super-ordinate component. Cohesion specifies the similarity of methods in a component. It is a measure of the extent to which the various functions performed by a component are related to one another.

4.1 Cohesion Metrics

Cohesion shows the relationship of different attributes among a component. It shows strength of the component. Highly cohesive component can be reused because it is an independent component. The reusability factor is increased if the component is highly cohesive.

CohV(Cohesion of variables)

Cohesion of variables means frequency of variables in a component. If the association of variables declared in component is focused on accomplishing a single task then component is cohesive. Cohesion of variables(CohV) refers to the frequency of variables usage in component by total number of variables.

$$CohV = \frac{\sum_{i=1}^{n} f(Ai)}{Tm}$$

Here F(Vi) = frequency of each attribute that are used in component.

Tv= Total no of variables in component

CohM(Cohesion of Methods)

Cohesion of methods refers to relatedness of methods with variables used in methods. This metric considers the interaction of methods with in component to find the strength of the component. This measure is used to find out the cohesion of methods in a component by counting the methods that use same type of variables and dividing by the total number of methods

$$CohM = \frac{\sum_{i=1}^{n} f(Mi)}{m^2 - m}$$

F(Mi)=count of methods that use same type of variables.

Tm= Total methods in a component.

5. COMPARATIVE ANALYSIS OF DIFFERENT COHESION METRICS

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.

5.1 CohV(Cohesion of variables)

Cohesion of variables represents the frequency of variables used in the component by total number of variables. 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 variables. Some components have same frequency of the variables and some have different frequency. According to these frequencies the value of CohV is calculated.

Table 1 shows the frequency of variables and CohV value

Components	F(Vi)	Tv	CohV
C1	6	3	2
C2	6	3	2
C3	9	4	2.25
C4	9	4	2.25
C5	45	16	2.8125
C6	9	4	2.25
C7	5	2	2.5
C8	9	4	2.25
C9	6	3	2
C10	9	4	2.25
C11	5	2	2.5
C12	6	3	2
C13	6	3	2

5.2 Cohm(Cohesion Of Methods)

Cohesion of Methods in a component refers to the relatedness of methods and instance variables of a component. This metrics considers the interaction between the methods with in a component. 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 same type of variables.

Table 2 shows values of CohM

Components	F(Mi)	Tm^2-Tm	CohM
C1	6	12	0.5
C2	6	12	0.5
C3	9	30	0.3
C4	9	30	0.3
C5	45	870	0.05
C6	9	30	0.3
C7	4	2	2.0
C8	9	12	0.75
C9	6	12	0.5
C10	9	30	0.3
C11	4	2	2.0
C12	6	12	0.5
C13	6	12	0.5

Yadav and Tomar[25] proposed two cohesion metrics CIC(Cohesion in Class) and CBM(Cohesion between Methods)[25]. Cohesion in class refer to the frequency of attributes(variables) usage by the methods of the class in a component[25]. Cohesion between methods refers to the relatedness of class members [25].

5.3 CIC(Cohesion In Class)[25]

$$CIC=\sum_{i=0}^{n} f(Ai)/TM$$

N= Total No of attribute in class

F(Ai)= frequency of each attribute that are used by methods in the class

TM= total no of methods in class

Table 3 shows frequency of attributes and CIC value.

Components	F(Ai)	Tm	CIC
C1	6	4	1.5
C2	6	4	1.5
C3	9	6	1.5
C4	9	6	1.5
C5	45	30	1.5
C6	9	6	1.5
C7	5	2	2.5
C8	9	4	2.25
C9	6	4	1.5
C10	9	6	1.5
C11	5	2	2.5
C12	6	4	1.5
C13	6	4	1.5

5.4 CBM(Cohesion Between Method)[25]

$$CBM=\frac{\sum_{i=0}^{a} Mi(Ai)}{am(m-1)}$$

Mi(Ai)= sum of the method that are used same type of attribute

m= no of method in class

a= no of variables

Table 4 shows CBM value

Components	Mi(Ai)	M	m-1	A	am(m-1)	CBM

C1	6	4	3	3	36	0.166
C2	6	4	3	3	36	0.166
C3	9	6	5	4	120	0.075
C4	9	6	5	4	120	0.075
C5	45	30	29	16	13920	0.003
C6	9	6	5	4	120	0.075
C7	4	2	1	2	4	1.000
C8	9	4	3	4	48	0.1875
C9	6	4	3	3	36	0.166
C10	9	6	5	4	120	0.075
C11	4	2	1	2	4	1.000
C12	6	4	3	3	36	0.166
C13	6	4	3	3	36	0.166

To validate these metrics an empirical analysis based on JavaBeans components is to be performed. Same java beans project is to be taken. For each of java beans component CIC and CBM is to be calculated. Table 3 shows frequency of attributes and CIC values of each component. Table 4 shows sum of methods that are used same type of attributes and CBM value of each component.

To find out the significance of the results of CohV , CIC and CohM and CBM statistical tool is to be applied. T test is to applied on these metrics. T test is inferential statistics. It is used to determine whether there is a significant difference between the means of two groups.

Table 5 Mean and Std. Deviation of CohV and CIC

	Mean	N	Std. Deviation	Std. Error Mean
CohV	2.2354	13	.24972	.06926
CIC	1.7115	13	.40628	.11268

Table 6 paired sample t-test

	Paired Differences	T	DF	Sig.(2-tailed)
	95% Confidence Interval of the Difference			
	Upper			
Pair1 Cohv-CIC	.74784	5.096	12	.000

Paired sample t-test (Result in Table 6) are applied on the data and found that the cohesiveness of proposed metrics CohV is more than CIC[Yadav and Tomer][25]. The mean value is reflecting that cohesion value of proposed metrics is higher than the value of CIC which is proposed by Yadav and Tomer[25]. The T-test value is 5.096 and the same is significant at 99% level of confidence. It means that the cohesion value is higher and significant for CohV.

Table 7 Mean and Std. Deviation of CohM and CBM

	Mean	N	Std. Deviation	Std. Error Mean
CohM	.6538	13	.62030	.17204
CBM	.25538	13	.335085	.092936

Table 8 Paired Sample T test

	T	Df	Sig. (2-tailed)
Pair 1 CohM-CBM	4.939	12	.000

Paired sample t-test (Result in Table 6) are applied on the data and found that the cohesiveness of proposed metrics CohM (Cohesion of Methods) is more than CBM(Cohesion between methods)[Yadav and Tomer][25]. The mean value is reflecting that cohesion value of proposed metrics(CohM) is higher than the value of CBM which is proposed by Yadav and Tomer[25]. The T-test value is 4.939 and the same is significant at 99% level of confidence. It means that the value of CohM (cohesion of methods) is higher and significant in proposed metrics(CohM).This comparison concludes that the proposed metrics (CohV & CohM) are significant in comparison with CIC and CBM [Yadav and Tomer][25].

The proposed metrics CohV and CohM and Yadav and Tomar[25] proposed CIC and CBM , these metrics are based on the variables which are used by different methods of the components. The improved version of CohV and CohM are COVC(Cohesion of Variables within a component) and COMC(Cohesion of Methods within a component) proposed by Rana and Rajender Singh[13]. These two metrics are also based on variables and methods but authors categorize variables according to the hardness like standard, moderate and critical and they consider weights to normalize. Cohesion of variables in a component (COVC) represents the frequency of different type of variables binds or strengthens the component. Cohesion of Methods in a component refers to the relatedness of methods and instance variables of a component [13]. This metrics considers the interaction between the methods with in a component [13]

5.5 COVC (Cohesion Of Variables In A Component)[13]

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

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

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

Java beans project of thirteen components is to be taken for empirical evaluation. Table 9 shows the frequency of different type of variables and the value of COVC.

To validate these metrics an empirical analysis based on java beans components is to be performed. For each of the JavaBeans component

Table 9 Shows the frequency of different type of variables of different components and covc values of all 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

Table 10 shows Pearson correlation among cohesion measures against frequency of different type of variables

*Significant at 5% level

Type of Variables	Pearson Correlations (r)	Significance	n
Standard	-0.562	0.046*	13
Moderate	0.637	0.019*	13
Critical	0.577	0.039*	13

Conclusion is drawn that the usage of Moderate instance variables within a component is high and hence the reusability factor for developing a totally new application becomes high. A correlation analysis is applied on Rana and Rajender Singh[13] metrics to find out the relationship between cohesion and frequencies of different type of variables used in the component.

To study the relation between the cohesion measure and the frequency of different type variables(standard, moderate, critical) Pearson correlation method is applied. The value of Pearson correlation (Table No- 10) is -.562 for standard type variables. It means that if we increase the standard type variables in a component the value of COVC decreased by .562 per unit. The correlation value of moderate type variable is .637. It means that if we increase the moderate type variables in a component the value of COVC increased by .637 per unit. Similarly the Pearson correlation is .577 for critical type variables. It means if the frequency of critical type variables is increased in a component the value of COVC increased by .577 per unit. So it is suggested to the researchers to reduce the usage of simple type variables and increase the usage of moderate and critical type variables. As we know that to make the component independent the value of cohesion should be high and coupling should be low. 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.

5.6. COMC (Cohesion Of Methods In A Component)[13]:

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

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

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.

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

Table 11 shows the frequency of Methods using different type of variables of different component

Mci= sum of methods that use critical type of variables.

Ws, Wm, Wc are weight factor for standard, moderate and critical type of variables[13]

Java beans project of thirteen components is to be taken for empirical evaluation. Table 11 shows the frequency of methods using different type of variables of different components and the value of COMC.

To validate these metrics an empirical analysis based on java beans components is to be performed. For each of the JavaBeans component conclusion is drawn that 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. A correlation analysis is applied on Rana and Rajender Singh[13] metrics to find out the relationship between cohesion and frequencies of methods using different type of variables of different component.

Table 12 shows Pearson correlation among cohesion measures against frequency of methods using different type of variables of different components.

Type of Variables	Pearson Correlations(r)	Significance	n
Standard	-.582	.037*	13
Moderate	-.029	.926*	13
Critical	-.042	.893*	13

*Significant at 5% level

Pearson correlation method is applied to study the relation between the cohesion measure and the frequency of methods using different type of variables (standard, moderate, critical). The value of Pearson correlation (Table No- 12) is -.582 for frequency of methods using standard type variables. It means that if we increase the number of methods using standard type variables in a component the value of COMC decreased by .582 per unit. The correlation value of frequency of methods using moderate type variable is -.029. It means that if we increase the number of methods using moderate type variables in a component the value of COMC decreased by .029 per unit. Similarly the Pearson correlation is -.042 for critical type variables. It means if the number of methods using critical type variables is increased in a component the value of COMC decreased by .042 per unit. So it is suggested to the researchers to reduce the usage of methods which uses simple type variables and increase the usage of methods that uses moderate and critical type variables. As we know that to make the component independent the value of cohesion should be high and coupling should be low. Conclusion is 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.

6. CONCLUSION

Component based system is known for on time delivering of projects at reasonable cost. Metrics are developed to evaluate the complexity of the projects. Here an empirical evaluation of the existing and proposed metrics has been performed on Java Beans projects. In depth analysis for cohesion metrics have been carried out.

The comparative analysis performed between proposed metrics and different cohesion metrics. The proposed metrics CohV and CohM is to be compared with CIC and CBM using statistical tool (SPSS). t-test is applied on the data and finding that the cohesiveness of proposed metrics CohV and CohM are more than CIC and CBM.

An improved version of CohV and CohM is COVC and COMC, the Pearson Correlation method is applied to show the relation between the cohesion measure and the frequency of different type variables, functions which are using different type of variables. The analysis of COVC and COMC reveals that by using moderate type variables, the strength of component will be high and cohesiveness of the component will be high and the possibility of reusability for developing new applications become high.

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 and 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] 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.

[2] Chen, Wang, Zhou (2011): Complexity Metrics for Component Based Software Systems, International Journal of Digital Content Technology and its Applications, Volume 5, Number 3, March 2011. Doi:10.4156/jdcta.vol5.issue3.24

[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] 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.

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

[6] 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.

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

[10] Jianguo Chen and Hui Wang (2011); Complexity Metrics for Component-based Software Systems; International Journal of Digital Content Technology and its Applications. Volume 5, Number 3

[11] 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

[12] Navneet Kaur, AshimaSingh(2013) "A Complexity Metrics for Black Box Components", International Journal of soft computing and engineering. Vol 3, issue 2,May 2013

[13] Pooja Rana, 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

[14] Rajender Singh Chhillar and Praveen Kajla(2012) "New Component Composition Metrics for Component Based Software Development", International Journal of Computer Application, Vol 60, No15, Dec 2012

[15] Rajender Singh Chhillar, PriyankaAhlawat and UshaKumari (2012) "Measuring Complexity of Component Based System Using Weighted Assignment Technique", 2nd International Conference on information Communication and Management(ICICM 2012).

[16] Sengupta, S., Kanjilal, A. (2011): Measuring Complexity of Component Based Architecture : A Graph Based Approach, ACM SIGSOFT Software Engineering Notes, 36 (1), pp. 1-10.

[17] Sharma, A., Grover, P.S., Kumar, R. (2009): Dependency Analysis for Component-Based Software Systems, ACM SIGSOFT Software Engineering Notes, 34 (4), pp. 1-6.

[18] Sonu Mittal and Pradeep Kr Bhatia(2013) "Predicting Quantitative Functional Dependency Metric Based Upon the Interface Complexity Metric In Component Based Software", International Journal of Computer Application, Vol 73, No 2, July 2013

[19] Tobias, Mwangi and Michael(2015) "Empirical Evaluation of Complexity Metrics For Component based systems", Journal of Theoritical and Applied Information Technology, ISSN 1817-3195, March 15,Vol 73 No2, Pg No-275-282

[20] Umesh Tiwari and Santosh Kumar(2014) "Cyclomatic Complexity Metric for Component Based Software", ACM SIGSOFT Software Engineering Notes page 1 vol 39 No1, Jan 2014

[21] Usha Chhillar, Sucheta Bhasin (2011): A Journey of Software Metrics: Traditional to Aspect-Oriented Paradigm, 5th National Conference on Computing For Nation Development, 10th -11th March, 2011, New Delhi, pp. 289-293.

[22] UshaKumari and Shuchita Upadhyaya(2011): An Interface Complexity Measure for Component-based Software Systems International Journal of Computer Applications (0975 – 8887) Volume 36– No.1

[23] V. Lakshmi Narasimhan, P. T. Parthasarathy, and M. Das (2009):Evaluation of a Suite of Metrics for Component Based Software Engineering (CBSE), Issues in Informing Science and Information Technology Volume 6, 2009

[24] W. Kozaczynski, G. Booch (1998), "Component-Based Software Engineering," IEEE Software Volume: 155, Sept.-Oct. 1998, pp. 34–36.

[25] Yadav and Tomar (2014) Design of Metrics for Component-Based Software System at Design Level, International Journal of Engineering and Technical Research, ISSN: 2321-0869, Vol 2, Issue-4, Apr-2014