Measurement of Software Reliability Based on Coupling and Cohesion Rate by External and Internal View of Classes

Mahdie khadame, Sima Emadi*
Department of Computer, Yazd Branch, Islamic Azad University, Yazd, Iran

Abstract
One of the most significant parameters of software quality that is essential for the creation of valid and secure systems is reliability. Therefore, assessment of the software after its development prior to unveiling of the software, and its release into the market is an important issue that has generated immense interest. In addition, an increase in the use of software in different fields has precipitated an increase in the demand for software with high reliability value. . A suitable design and high reliability is an essential prerequisite for an increased acceptance of any product. Coupling and cohesion among the components in the design induces some challenges. Suitable coupling and cohesion among the classes can reduce the complexity in software development as well as improve reliability in software. In this research, software reliability was assessed using fuzzy computing approach on internal coupling and cohesion of classes because of the non-deterministic nature of coupling, which allows its fuzzification to increase the precision and concentration of the model. Thus, the rate of reliability and error that occurred in the system would be more accurately computed. This approach can also make it possible for software engineers to consider computing other non-functional parameters such as security, readability and reusability. By observing the results closely, it is obvious that the reliability value is variable and depends on the value of primary concepts. It was also observed that the increase in the speed line of reliability rose sharply because the basic concepts applied in reliability were used with high frequency. The rise can also be attributed to the known fact that low frequency concepts hardly affect the results. From these results, it can be stated that the pre-examined concepts have high reliability values.

Keywords:
Coupling, Cohesion, Reliability, Software, Class, object oriented

1. Introduction

Object oriented techniques provide the platform for development of useful applications including reusable components along with good communications among heterogeneous computing platforms. If a design model has high coupling and low cohesion, implementation, testing and maintenance of the system will be very hard. Therefore, to improve the software, we need to introduce or utilize modules that provide appropriate coupling and cohesion [1]. In a favorable design, measurement of two metrics of coupling and cohesion statically and dynamically is of great importance. Cohesion and coupling are among the most imperative metrics used in assessing the structural integrity of object oriented systems and assessment of the quality of design. Cohesion means the consistency of the general performance of a class while coupling denote the rate of dependency of a class on other classes in an object oriented system [2]. In a well-designed software system, classes have low coupling and high cohesion. Measurement of cohesion and coupling examines the rate of association among various elements of design like methods and features of the object oriented software systems owing to the fact that these two metrics are useful in different tasks such as evaluation of the quality of design, analysis of pressure, and prediction of software quality and defects. Reliability is one of the most important software quality parameters that are essential in the creation of valid and secure systems because reliability problems can induce significant changes at each stage of the software life cycle especially in the early stages of the software development (architecture, design or requirements phases). Different studies that show the stages of software development processes that affect each quality features have been conducted. In this regard, it was shown that reliability could be assessed in the early stages of development. Coupling and cohesion affect reliability among classes; therefore, they can be used to assess reliability. The purpose of this study was to use static and dynamic metrics of coupling and cohesion to measure the reliability of the software. In line with this measurement, in metric of cohesion, some associations between classes were also considered and measurement of coupling was done using fuzzification in addition to internal association of classes. On the other hand, owing to the uncertainty of coupling in this study, fuzzy computations approach was used to measure the reliability on internal coupling and cohesion of classes in order to improve the accuracy of the model. It was also observed that cohesion of components within a class along with their direct and indirect associations affected the reliability of the software and measurement of the software reliability based on coupling and cohesion of a class because of the associations of its internal components with other classes, which produced more accurate results.

The structure of the article is as follows: In section two, the works carried out on the application of coupling and cohesion among the components are presented. In the

Manuscript received November 5, 2016
Manuscript revised November 20, 2016

third part, the proposed method is given and in the fourth part, a case study is used to evaluate the proposed model. Finally, conclusion is presented in the fifth part.

2. Literature Review

Coupling and cohesion have been used in measuring applications and quality features. Some of such works are presented below:

Ouyang et al., used coupling and cohesion measures to assess ontologies, making it possible for a suitable ontology to be employed in the design of Semantic Web [3].

In another study, it was shown that some features of modular structures such as coupling and cohesion are useful in determining the appropriateness of modularity techniques. The present study has examined modular ontology in form of coupling and cohesion metrics because ontology is one of the essential components of the Semantic Web. Therefore, modular ontologies are of great importance. Among the available modularities, the one that produces the best system and performance should be used. In this study, a method was provided for analysis of coupling and cohesion of modular ontology so that a comprehensive model is produced by computing the structural dependencies in ontology [4].

In his model, Sartipi used component association, which is the generalized form of cohesion and coupling to assess and classify software to help users have a better understanding of the system for repairing and reversing engineering activities. In this study, component association was defined as a measure of the general dependency of high level components of system, and associations were identified using the methods of data mining on data and dependencies of control flow extracted from the software system [5].

Rathore et al., used coupling metric to demonstrate the difference between inheritance programming and interface programming. In this study, the concepts of coupling between object (CBO), number of associations between classes (NASSocC), number of dependencies in metric (NDepIN), number of dependencies out metric (NDepOut) and number of children (NOC) in object oriented programming was used to rate the efficacy of the inheritance programming and interface programming among classes for developers of c # . When CBO is reduced, reusability (a feature for measuring the difference between class inheritance and interface inheritance) increases [6]

In order to assess the design of systems in service oriented architecture , Alahmari et al., mentioned service granularity as an important factor in design features that affect the structural features of internal SOA (ie coupling, cohesion and complexity) [7] .

Tosi et al., focused on defining dynamic measures of software and provided a framework for defining dynamic measure that can be used to acquire these features dynamically. In this framework, attention was only given to coupling, and cohesion was not considered among the components [8].

Hasan et al., provided an appropriate measure that pointed out the importance of reducing coupling in software design. They classified coupling measures into two groups, ratio oriented and ratio less and introduced inconsistencies among the measures based on the coupling measures [9].

Iyapparaja et al., provided an intelligent Risk Analysis Model (IRAM) for reusability of software components. This model performs surfing operation for Java modules by studying the modules of various projects and providing a list of appropriate modules. These modules were tested based on coupling and cohesion analysis test in order to facilitate the determination of individual power modules and binding capacity of modules before the regression test [10].

In their study, Prasad et al., first investigated the relationship between object oriented metrics (coupling and cohesion) and procedure oriented metrics (cyclomatic complexity, line of code and knot metric). Thereafter, they proposed a strategy of analysis and inspection for reusability and comparability. In this study, a new set of operational measures for conceptual coupling of classes showed that these metrics provided new dimensions in coupling measurement compared to the existing structural metrics [11].

Ramasubbu et al., analyzed two key dimensions of software structure that are coupling and cohesion by design according to efforts required in maintaining software, and showed that these two measures have important role in the maintenance of software [12].

Dixit et al., used coupling metrics to determine the difference between inheritance and interface programming. In this study, coupling between objects, number of dependencies among classes and the number of dependencies in both dynamic and static state were measured. The concept, which was well suited for use by the developer of the packages, was demonstrated [13].

In another study, two new concept metrics were provided for measurement of coupling and cohesion of software systems. The first measure was conceptual coupling among the classes of objects based on CBO coupling metric; that is, rate of coupling of one class to other classes. The second is lack of conceptual cohesion on methods based on cohesion metric of LCOMS. One of the benefits of these two conceptual measures is that they are similar to the structural metrics and are computed in a simple way. These two conceptual measures have been used in predicting the defect of classes in a large open source system [14].

In another study, coupling and cohesion were used to assess the reusability of Java components. Measures introduced in this research are different from previous measures in two ways; one is that they reflect the rate of coupling of entities and second is that they estimate indirect coupling or similarities . Measures introduced in this research have been shown to be appropriate in measuring reusable components [15].
In another study, coupling and cohesion metrics were introduced for comprehension of state diagram, which shows the dynamic behavior of systems and objects. These metrics are ACOS (average cohesion of states) and ASSOS (average coupling of similar states) and are used to assess comprehension of the state diagram. This research shows that these two measures can be used to calculate comprehension more precisely than the existing metrics based on structure [16].
Cho et al., noted the importance of object clustering in determining the overall performance of distributed applications and client/server, and suggested that higher coupling objects should be grouped into cluster so that each cluster can have higher consistency. Their proposed method has been employed in CORBA-based applications. They demonstrated that existing design metrics deal with static coupling and cohesion and focuses only on association, composition and inheritance among classes. They also showed that these measures were not suitable for measuring traffic load of messages of objects that are dependent on system performance. They provided a set of metrics that allocated weight to associations among the classes and estimated static and dynamic flow of messages between the objects. They showed that this metrics could help to improve the definition of clusters [17].
Apel et al., showed the importance of high cohesion in software product line and stated that the development of feature oriented software in the encapsulation of features to provide cohesion of units contributed to the completeness and reusability of the programs [18].
Anquetil et al., stated that it seems that restructuring of the legacy systems can improve cohesion and reduce coupling. However, by measuring two metrics of coupling between java package and eclipse plugin, they showed that cohesion was not a suitable criterion for modularity of these systems, and that restructuring does not improve modularity [19].

3. The Proposed Method

Choosing a model is a very important factor in predicting the enhancement of software reliability because input data is published to predict the information. Final decision for allocation of resources can be done with high precision. The basis for the task of predicting reliability is shown in Figure 1.

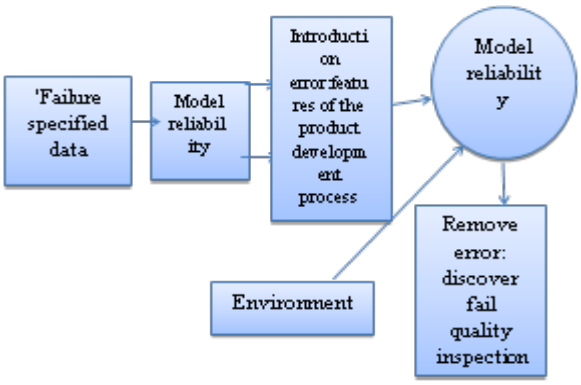

Figure 1. Brief review of the reliability model [20]

In addition, software reliability model is classified based on the different stages of the software development life cycle. The life cycle of software development processes and activities of software reliability are depicted in Figure 2.

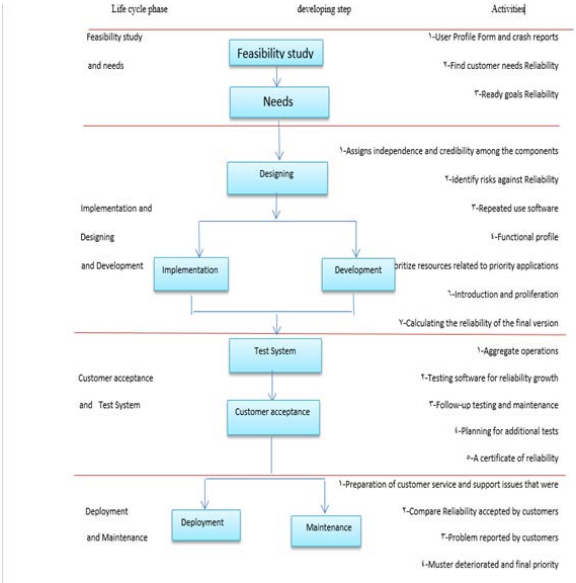

Figure 2. Software reliability engineering in the software development life cycle [20]

Prediction of software reliability using fuzzy logic produces more accurate results than traditional statistical methods because fuzzy logic presupposes that ambiguity is in the nature of science. Unlike others who believe approximations should be close to increase productivity, Lotfizadeh believes that models should be built to make ambiguity a part of the model system [1, 2].
The proposed method is a sequence process, which is shown in Figure 3. According to the figure, first coupling was computed using fuzzy logic and thereafter total

cohesion was computed according to the following steps that will be used to measure reliability:

Determination of the type of coupling in the software system in accordance with the method proposed in [1] and according to the Myers model (Data, Stamp, Control, and Common Content). Myers coupling model calculation is as follows:

To measure coupling, a metric was provided using Myers classification where n is the number of coupling of x and y, and I is equivalent to the level of Myers model in Equation (1).

$$C(X, y) = i + n / (n + 1) \quad (1)$$

Another metric that was specified for coupling is shown in Equation (2), [15]:

$$CoupD\ (i, j) = \frac{|MVij|}{|MV\ i| + |Vi| + |Mi|} \quad (2)$$

In this relation, M is the method and V is the variable and MVj is the set of methods and variables Which are called class Cj. MVj is the set of methods and variables in Ci class that are called class Cj. In $|MV_i| = |V_i| + |M_i|$ this relation, and as a result $0 \le coupD(i, j) \le 1$. Therefore, D is independent of the class size. In addition, to measure the indirect coupling, Equation (3) was used [15]:

$$coupT(i,j,p) = \prod_{s,t \in p} coupD(s,t) = \prod_{s,t \in p} \frac{|MV_{st}|}{|MV_s| + |V_s| + |M_s|} \quad (3)$$

CoupT (i, j, p) is the transitive coupling between classes Cj and Ci due to the presence of the special path p. When a transitive association is created between modules, indirect coupling occurs which has multiple coupling in indirect routes. Therefore, according to equation (3), a path with the greatest coupling was considered for computing the overall coupling design [15]:

$$coup(i, j) = coupT(i, j, p_{max}(i, j)) \quad (4)$$

Where p max (i, j) arg max p∈Π CoupT (i, j, p) and Π are the number of possible routes between classes Cj and Ci. To compute the coupling in a system (total coupling), Equation (5) was used, where m is the number of classes in the system [15].

$$Coupling = \frac{\sum_{i,j=1}^{\infty} coup(i, j)}{m^2 - m} = \frac{\sum_{i,j=1}^{\infty} \frac{A_{ji}}{B_i}}{m^2 - m} \quad (5)$$

Aij and Bj were determined in a similar way according to Table 1 [1].

• Calculation of the total coupling according to fuzzy logic and usage of the fuzzy rules.

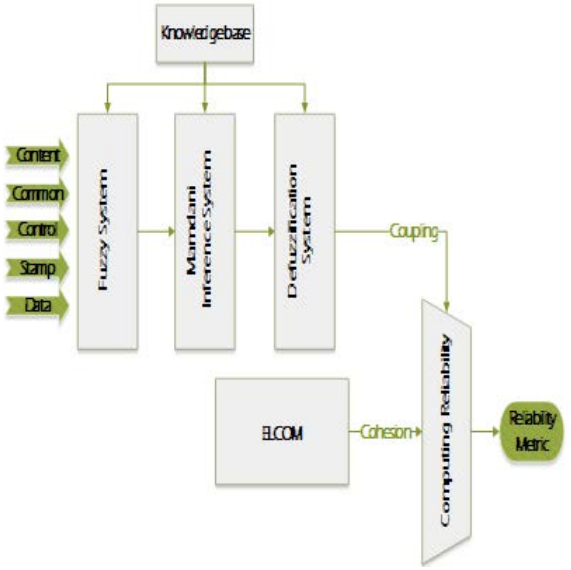

Figure 3. The proposed model

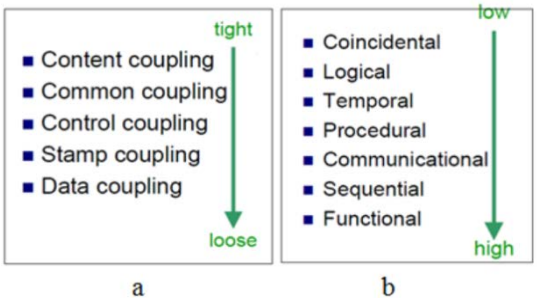

Figure 4. A: Classification of coupling, B: Classification of cohesion

By classifying input and output parameters, the total coupling with fuzzy structure is displayed in Table 2.

TABLE 1. Used parameters in coupling relation [1]

	A_{ji} B_{ji} common	
Content coupling (j,i)	method and variable from i that change its j	Methods and variables that change j, methods j, variables j
Common coupling (j,i)	methods and variables from i that j accessed them	Common methods and variable that j accessed them, methods j, variables j
Control coupling (j,i)	The number of control variable in i that j can change them.	The number of control variable that j can change them
Stamp coupling (j,i)	I's methods and variables (data) that send to j and j need them.	i's methods and variables (data) that sent to j.
Data coupling (j,i)	I's methods and variables (data) that send to j and j need them.	Methods and variables that j need them, methods j, variables j.

TABLE 2. Coupling was computed in the system through data fuzzification [1]

Parameters	Type	Data	Linguistic variables
Content	Input	[0 1]	low, medium, high
Common	Input	[0 1]	low, medium, high
Control	Input	[0 1]	low, medium, high
Stamp	Input	[0 1]	low, medium, high
Data	Input	[0 1]	low, medium, high
Total	Output	[0 1]	v-low, medium low, high, v-high

In the present research; Lee [1] Fuzzy Inference System of Mamdani was used

• Determination of the rate and type of cohesion in software systems according to Myers model (Figure 4). Calculation of the coherency model is as follows:

LCOM as internal cohesion for class c is defined in Equation (6) [21]:

$$LCOM(c) = 1 - \frac{|U(c)|}{|M(c)||V(c)|} \quad (6)$$

Where $M(c)$ the set of methods is, $V(c)$ is the set of instance variables and $U(c)$ is the set of ordered pairs $(m, v) \in M(c) \times V(c)$ so that the method m will use the variable v. In the new definition with respect to external view, $M(c)$ is the set of all public method (inherited and local), $V(c)$ the set of instance variable (inherited and local) and U (c) represents the indirect uses of the considered class [21].

If the external cohesions are defined for class c, what do other classes expect from it and how do they use the features of class c? In other words, by defining the external cohesion, association between the client classes and the actual instance variables of class c was determined. As a result, Equation 6 was changed into Equation 7 as follows [21]:

$$ELCOM(c) = 1 - \frac{|CU(c)|}{|CM(c)||V(c)|} \quad (7)$$

Where $CM(c)$ is the set of client classes that used c, $V(c)$ is the set of instance variable (local and inherited), and $CU(c)$ is the set of ordered pairs $(c', v) \in CM(c) \times V(c)$ so that most of the methods of class c' could use the instance variables of class c [21]. The overall cohesion of the system in the k classes was computed as follows:

$$ELCOM = \frac{\sum_{i=1}^{k} ELCOM(c_i)}{k} \quad \mathbf{(8)}$$

4. Evalution of Reliability in Object Oriented System

Software reliability was assessed through the measurement of frequency of failures, failure intensity and accuracy of the output results. Therefore, reliability is the amount of time the software is available and has error tolerance. Researches show that complexity is a factor in decreasing reliability. If the software system has high cohesion and low coupling, complexity will reduce [1]. Therefore, the likelihood of errors will reduce and reliability will increase. Thus, cohesion has an inverse relationship with complexity and a direct relationship with reliability. In the calculation of reliability, there was an inverse relationship between coherency and coupling. Therefore, in defining fuzzy membership functions, inverse function is required, which implies the actual addition of extra load to the system. In this thesis, inverse proportion to the coupling is provided so that we can produce the reliability relationship. If the final cohesion and coupling of the output fuzzy system is Total_coh and Total_coup, the reliability of the software system is as follows [1]:

$$R \propto \frac{\alpha \times (T_{coh})}{\beta \times (T_{cup})} \quad (9)$$

Where $0 \le \alpha, \beta \le 1$ indicates the importance of coupling and cohesion in the system, and has been identified as an effective factor in different software systems. Thus, reliability of the software design is calculated as shown in relation 10 [1]:

$$R_{design} = \frac{TotalCohesion + (1 - TotalCoupling)}{2} \quad (10)$$

5. Case Study

In order to show the importance of cohesion with external view in the assessment of reliability, two different projects were selected and their information is shown in Table 3.

6. The First Project

Agent Markup Language (DAML) is one of the programs developed by DARPA for semantic web. In this project, a number of concepts, ontology and classes were used and are shown in Table 3 [23].

Table 3. Details of the first project

Related external class	The number of classes	Number of connections	The number of concepts	Ontology
38	132	124	189	The first project

7. The Second Project

SWEET ontologies are written in the OWL ontology language and are available to the public. SWEET 2.3 is highly modular with 6000 concepts in 200 separate ontologies. You can view the entire concept space from an OWL tool such as Protege by reading from sweetAll.owl. SWEET 2.3 consists of nine top-level concepts/ontologies. Some of the next level concepts are shown in Figure 5 [24]. Table 4 shows the details of this project.

TABLE 4. Details of the second project.

Related external class	The number of classes	Number of connections	The number of concepts	Ontology
46	269	247	436	The first project

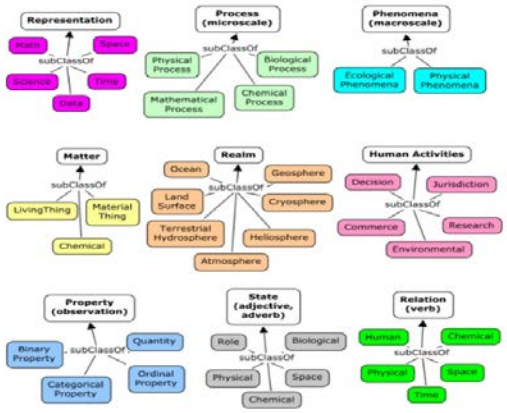

Figure 5. Association of classes in the second group [24]

By entering information into the fuzzy set for calculating coupling, we achieved some results. Fuzzy functions were applied on both groups of case study and the following results were obtained. Fuzzy architecture that was used as case study is shown in Figure 6.

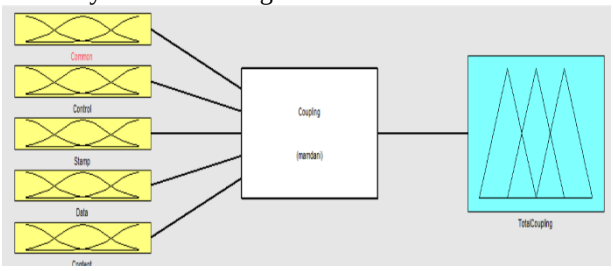

Figure 6. Fuzzy architecture for obtaining coupling

Figure 7 is related to the fuzzy membership functions common variable. Input values in the range 0 to 1 were substituted into this function and its coupling value was calculated.

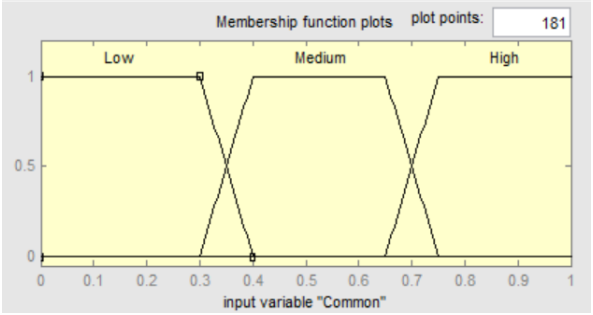

Figure 7. Fuzzy membership function of common variable

Figure 8 is related to the fuzzy membership function of the control variable. Input values in the range of 0 to 1 were substituted into this membership function and its coupling value was calculated.

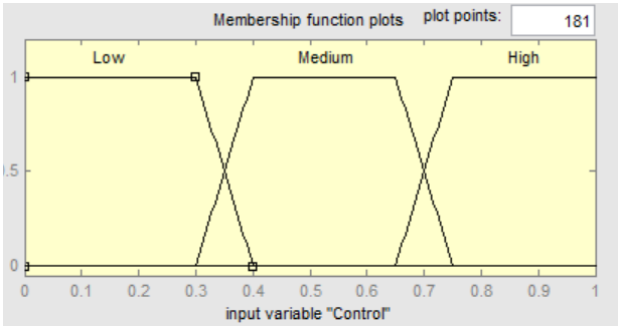

Figure 8. Fuzzy membership function of control variable

Figure 9 is related to the fuzzy membership function of the variable stamp. Input values in the range of 0 to 1 were entered into this membership function and the value of its coupling was computed.

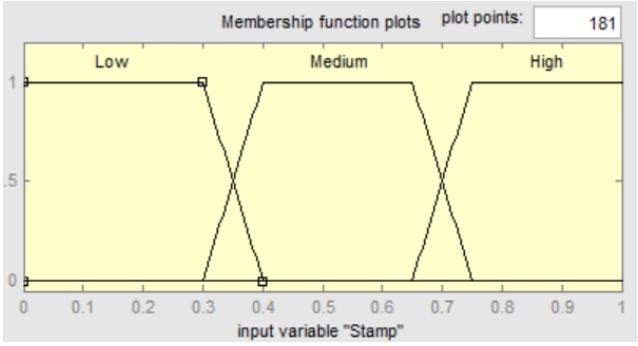

Figure 9. Fuzzy membership function of variable stamp

Figure 10 is related to the fuzzy membership function of the variable data. Input values in the range of 0 to 1 were substituted into this membership function and its coupling value was calculated.

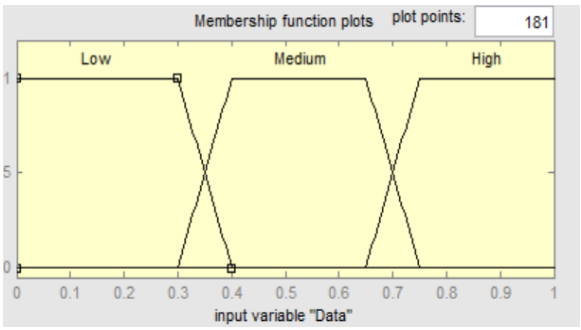

Figure 10. Fuzzy membership function of variable data

Figure 10 is related to the fuzzy membership function of the variable content. Input values in the range of 0 to 1 were substituted into this membership function and its coupling value was calculated.

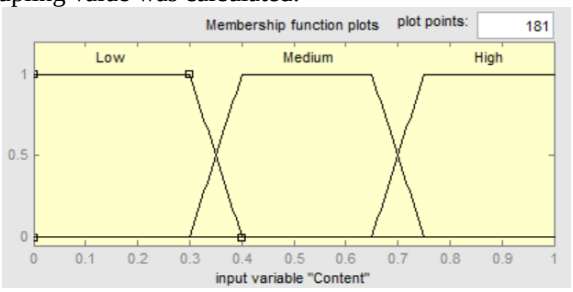

Figure 11. Fuzzy membership function of variable content

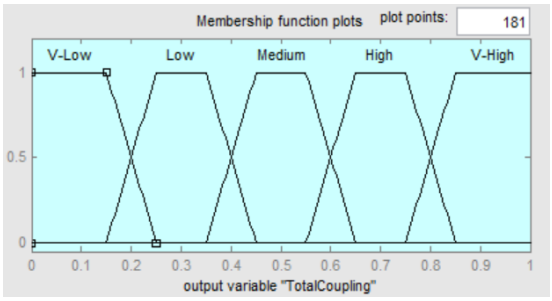

Figure 12. Fuzzy membership function of total output coupling

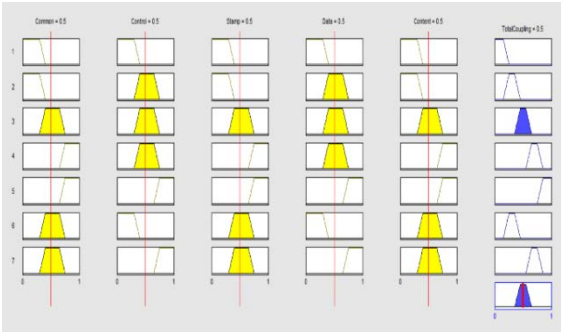

Figure 13. Rules of the used fuzzy membership function.

After calculating the fuzzification values in the first stage, cohesion was calculated. After numerical calculation of this value and its substitution into the formula obtained in the previous part, reliability of these values was obtained (Figure 14 and 15).

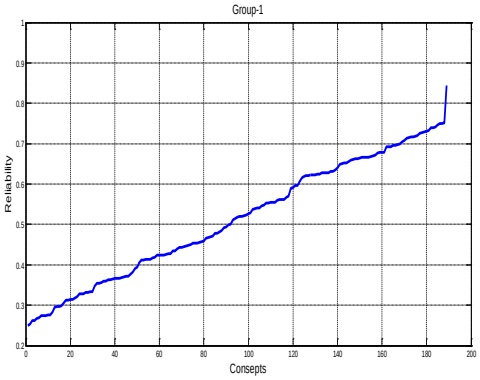

Figure 14. Value of reliability in Group 1

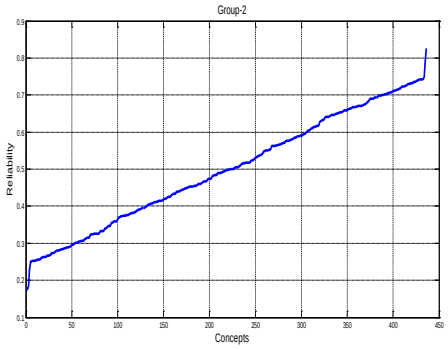

Figure 15. Value of reliability in Group 2

In the values obtained, tests showed that small ontologies may produce high reliability values due to the choice of a valid set. Great ontology may produce low reliability because of the choice of improper configuration.

By observing the results, we concluded that the reliability of ontology is variable and depends on the values of their basic concepts. At the beginning of the test, we saw that speed line was rising sharply because quite common basic concepts were used, and low frequency concepts hardly affect their result. From these results, it can be stated that the pre-assessed concept do not have high reliability values.

TABLE 5. Overall assessment

reliability	cohesion	coupling	
0.5068	0.5200	0.4936	**project1**
0.49585	0.4941	0.4976	**Projec2**

According to the table above, by using a number of concepts, the quality of ontology can be measured because

reliability should be estimated in comparison with the set of coupling and cohesion values. A number of concepts in the set, which is definitely more than those in the ontology, are effective in the estimation of reliability. If the number of concepts in this ontology is high, the reliability value will be high. In other words, the value of reliability depends on the original scale. The second point is that the environmental conditions of each group can affect ontology.

8. Discussion and Conclusion

Given the important role that software plays in systems today, the quality of software components is of great importance. One of the important aspects of quality is reliability and for this reason, software reliability engineering is very germane. Designing great software is a very complex and difficult task. In fact, the process of design is one of the most difficult tasks in the process of software development. Researchers in the process of software development seek to increase the level of reliability; the reason for this is that reliability enables the design process to be performed with high reliability. With increasing development in the design of software, many methods have been developed to improve software reliability by researchers. However, no method that can fully guarantee the reliability of software has been developed to date. Thus, the purpose of this study was to use static and dynamic cohesion and coupling metrics in measuring software reliability. Class cohesion or rate of association within a class in object oriented systems was crucial. The higher class cohesion in object oriented systems will also improve reliability, variability, flexibility, maintainability and reusability. The importance of identifying cohesion metrics is that it measures class cohesion quantitatively and this measurement can be used as the basis for increasing reliability. Class cohesion was investigated structurally and conceptually. Most of the existing metrics measure class cohesion structurally or conceptually. However, for designers, a metric is appropriate if it mainly considers the conceptual aspect of class cohesion. On the other hand, research has shown that high complexity is a factor in decreasing reliability. If the software system has high cohesion and low coupling, complexity will decrease. Therefore, the likelihood of errors will decrease and reliability will increase. Therefore, cohesion has inverse relationship with complexity and direct relationship with reliability. In this research, to obtai mor effects of cohesion, external view was used to measure cohesion metric.

References

[1] Kalantari, S., Alizadeh, M. and Motameni, H.," Evaluation of reliability of object-oriented systems based on Cohesion and Coupling Fuzzy computing". Journal of Advances in Computer Research, 6(1), 2015, pp.85-99.

[2] Michael, N., "Artificial Intelligence A Guide to Intelligent Systems". ISBN, 321204662, 2005, pp.1-18.

[3] Ouyang, L., Zou, B., Qu, M. and Zhang, C., July. A method of ontology evaluation based on coverage, cohesion and coupling. "In Fuzzy Systems and Knowledge Discovery (FSKD"), 2011 Eighth International Conference on (Vol. 4,, 2011, pp. 2451-2455). IEEE.

[4] Sukalikar, S., Kumar, S. and Baliyan, N., September. "Analysing cohesion and coupling for modular ontologies. In Advances in Computing, Communications and Informatics (ICACCI, 2014 International Conference on, 2014, (pp. 2063-2066). IEEE.

[5] K. Sartipi., "A software evaluation model using component association views". In IWPC, 2001, (pp. 259-268).

[6] Rathore, N.P.S. and Gupta, R., A Novel Coupling Metrics Measure difference between Inheritance and Interface to find better OOP Paradigm using C#. In Information and Communication Technologies (WICT), World Congress on 2011, (pp. 467-472). IEEE.

[7] Alahmari, S., Zaluska, E. and De Roure, D.C., July. A metrics framework for evaluating soa service granularity. In Services Computing (SCC), 2011 IEEE International Conference on (pp. 512-519). IEEE.

[8] Tosi, D., Lavazza, L., Morasca, S. and Taibi, D., September. On the definition of dynamic software measures. In Proceedings of the ACM-IEEE international symposium on Empirical software engineering and measurement, 2011,(pp. 39-48). ACM.

[9] Hasan, K.A. and Hasan, M.S., May. Principal component analysis of coupling measures for developing high quality object oriented software. In Computer and Communication Engineering (ICCCE), International Conference on 2010(pp. 1-6). IEEE.

[10] Iyapparaja, M. and Sureshkumar, S., December. Coupling and cohesion metrics in Java for adaptive reusability risk reduction. In Sustainable Energy and Intelligent Systems (SEISCON 2012), IET Chennai 3rd International on 2012, (pp. 1-6). IET.

[11] Prasad, L. and Nagar, A., Experimental analysis of different metrics (object-oriented and structural) of software. In Computational Intelligence, Communication Systems and Networks, 2009. CICSYN'09. First International Conference on 2009, (pp. 235-240). IEEE.

[12] Ramasubbu, N., Kemerer, C.F. and Hong, J., Structural complexity and programmer team strategy: An experimental test. Software Engineering, IEEE Transactions on, 2012, 38(5), pp.1054-1068.

[13] Dixit, V. and Vishwkarma, R., September. Comparison of class-level versus object-level static and dynamic coupling and cohesion measures in object oriented programming. In Wireless and Optical Communications Networks (WOCN), Eleventh International Conference on 2014, (pp. 1-5). IEEE.

[14] Újházi, B., Ferenc, R., Poshyvanyk, D. and Gyimóthy, T., September. New conceptual coupling and cohesion metrics for object-oriented systems. In Source Code Analysis and Manipulation (SCAM), 10th IEEE Working Conference on 2010,(pp. 33-42). IEEE.

[15] Gui, G. and Scott, P.D., Measuring software component reusability by coupling and cohesion metrics. Journal of computers, 4(9), 2009, pp.797-805.
[16] Bae, J.H., Jeong, Y.J., Chae, H.S. and Chang, C.K., Semantics based cohesion and coupling metrics for evaluating understandability of state diagrams. In Computer Software and Applications Conference (COMPSAC), 2011 IEEE 35th Annual, 2011, (pp. 383-392). IEEE.
[17] Cho, E.S., Kim, C.J., Kim, S.D. and Rhew, S.Y., Static and dynamic metrics for effective object clustering. In Software Engineering Conference, 1998. Proceedings. 1998 Asia Pacific, 1998, (pp. 78-85). IEEE.
[18] Apel, S. and Beyer, D., May. Feature cohesion in software product lines: an exploratory study. In Software Engineering (ICSE), 33rd International Conference on 2011,(pp. 421-430). IEEE.
[19] Anquetil, N. and Laval, J., March. Legacy software restructuring: Analyzing a concrete case. In Software Maintenance and Reengineering (CSMR), 15th European Conference on 2011,(pp. 279-286). IEEE.
[20] Bhuyan, M.K., Mohapatra, D.P. and Sethi, S., 2014. A survey of computational intelligence approaches for software reliability prediction. ACM SIGSOFT Software Engineering Notes, 39(2), pp.1-10.
[21] Makela Sami ‹Lepp¨anen Ville ‹Client based Object-Oriented Cohesion Metrics, Computer Software and Applications Conference, COMPSAC. 31st Annual International, 2007, pp.743 - 748.
[22] Saxena, V. and Kumar, S., 2012. Impact of Coupling and Cohesion in Object-Oriented Technology. Journal of Software Engineering and Applications, 5(09), p.671.
[23] http://www.cs.umd.edu/projects/plus/DAML/onts/base1.0.daml
[24] http://sweet.jpl.nasa.gov/2.3/reprDataServiceAnalysis.owl