© 2016 IJSRSET | Volume 2 | Issue 5 | Print ISSN: 2395-1990 | Online ISSN : 2394-4099
Themed Section: Engineering and Technology

A Comparative Approach of Cohesion and Coupling for Reducing Complexity

Vinay Mishra[1], Rahul Bakshi[2]

[1]Mtech Student, [2]Assistant Professor, Computer Science Department, United Institute of Technology Naini, Allahabad, Uttar Pradesh, India

ABSTRACT

Object-oriented technology is becoming increasingly popular in industrial software development environments. This technology helps in the development of a software product of higher quality and lower maintenance costs. Thus, measuring the relationships has become a prerequisite to develop efficient techniques for analysis and maintenance. The extent of cohesion in an object-oriented system has implications for its external quality. In this thesis Cohesion Metrics Tight Class Cohesion (TCC) and Loose Class Cohesion(LCC) on the programs to calculate the cohesion value and compare the result. On the basis of result we differentiate between complexities of inheritance and interface.

Keywords :- Cohesion, Coupling, Interface, Inheritance

I. INTRODUCTION

The Object-Oriented (OO) software development technology was initially introduced in the early 1990s. OO technology employs Classes together with Objects and their interdependencies to design and implement systems. OO introduced various underpinning approaches to software development which distinguish OO from traditional software development paradigm. It is used to encapsulate a set of closely related functionality in a structured hierarchy where common functionality is added in one class and more specialized functionality of that class is added in other classes.

Object-oriented technology is becoming increasingly popular in industrial software development environments [7]. This technology helps in the development of a software product of higher quality and lower maintenance costs. Since the traditional software metrics aims at the Procedure-oriented software development so it cannot fulfill the requirement of the object-oriented software, as a result a set of new object oriented software metrics came into existence. Object Oriented Metrics are the measurement tools adapted to the Object Oriented paradigm to help manage and foster quality in software development [7]. OO technology introduced various underpinning approaches like concept of classes, interfaces etc. to the software development which distinguish it from traditional software development paradigm.

Object/instance is a run time structure with state and behavior. Object state is stored in its fields (variables) and behavior as its methods (functions).Class is static description of object [6]. Inheritance is one of the most widely used concept of OO paradigm. It is used to encapsulate a set of closely related functionality in a structured hierarchy where common functionality is added in one class and more specialized functionality of that class is added in other classes. The specialized classes inherit the common functionality from their super class and add their own extra functionality. The primary concern of inheritance is to promote reusability in a system.

To meet the above objectives, the following steps are taken:

- Set of 22 metrics is first defined and then explained using examples. Their values are computed for three standard projects and interpretations are drawn regarding their inter-relationships from the metric values. The fault

prone classes are also identified based on earlier empirical investigations.

- To find whether all these metrics are independent or are capturing same underlying property of the object being measured, distribution of the metric values is computed and principal component analysis on these metric values is done.
- The relationship between design measures and size of the class is analyzed to determine empirically whether the measure, even though it is declared as a coupling, cohesion, or inheritance measure, is essentially measuring size.

When a software system program is modularized, its tasks area unit divided into many modules supported some characteristics. As we know, modules area unit set of directions place along so as to attain some tasks. They are although, thought-about as single entity however could confer with one another to figure along. There is a unit measures by that the standard of a style of modules and their interaction among them is often measured. These measures area unit known as coupling and cohesion.

II. LITERATURE SURVEY

In Year -2010, V. Krishnapriya and Dr. K. Ramar have measured interface concept by using coupling metrics on design based and have proved interface is more effective in use then inheritance to increase reusability of a code in object oriented programming [1]. In this paper, measurement of inheritance and interface is calculated using cohesion metrics using a example and prove the usage of interface increased the reusability James m. Bieman and byung kyoo kang published a Paper on Cohesion and reuse in object oriented system explaining the TCC and LCC on C++ Program [4].

There are many metrics to find class cohesion but no standard metric or definition has been generally accepted, out of available [Fenton & Pfleeger 1998], [Counsell et al. 2002] and [Etzkorn et al. 2004]. A reasonable metric to measure class cohesion should give an insight to the relatedness among the methods of a class while considering the impacts of inheritance paradigm on local class cohesion.

Author Name / Title	**Journal**	**Strength**	**Weakness**
N. Rajkumar1 ”Measuring Cohesion And Coupling In Object Oriented System Using Java Reflection”	ARPN Journal of Engineering and Applied Sciences	This paper proposes a set of new measures to find coupling and cohesion in a developmental system using Java reflection components to assess the usability. It will predict the fault in an object-oriented system.	Next version will calculate coupling and cohesion metrics for UML representations
Martin Hitz “Measuring Coupling and Cohesion In Object-Oriented Systems “	http://www.isys.uni-klu.ac.at/PDF/1995-0043-MHBM.pdf	This distinction refers to dynamic dependencies between objects on one hand and static dependencies between implementations.	important aspects of software quality at run-time and during the maintenance phase, respectively.
Aaron B. Binkley “A classical view of object-oriented cohesion and coupling”	http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.99.4519&rep=rep1&type=pdf	Evidence is starting to accumulate that this paradigm is indeed as effective as has been suggested	Most of the metrics used in conjunction with the object-oriented paradigm are, fact, classical metrics.
Mr. Kailash Patidar “Coupling and Cohesion Measures in Object Oriented Programming ”	International Journal of Advanced Research in Computer Science and Software Engineering	A large numbers of metrics have been built and proposed for measuring properties of object-oriented software such as size, inheritance, cohesion	To achieve consistent and satisfying results, empirical data obtained from reallife software engineeri

		and coupling. The coupling is an important aspect in the evaluation of reusability and maintainability of components or services.	ng projects
Shweta Sharma "A review of Coupling and Cohesion metricsin Object Oriented Environment"	International Journal of Computer Science & Engineering Technology (IJCSET)	This paper focuses on two very significant factors of complexity measurement of software which are coupling and cohesion. An extensive study of approximately all types of coupling and cohesion metrics has been reported in this paper	Very little work has been done in areas of dynamic coupling and cohesion metrics and need further more investigations

Table 1 Literature Survey

III. METHODS AND MATERIAL

Object oriented design is becoming more popular in software development environment and object oriented design metrics is an essential part of software environment. Metrics measure certain properties of software system by mapping them to numbers (or to other symbols) according to well-defined, objective measurement rules. Design Metrics are measurements of the static state of the project's design and also used for assessing the size and in some cases the quality and complexity of software. Analysis and maintenance of Object-Oriented (OO) software is expensive and difficult.

We take two C# programs one implemented with inheritance and one with interface. Then we apply Cohesion Metrics Tight Class Cohesion(TCC) and Loose Class Cohesion(LCC) on the programs to calculate the cohesion value and compare the result. On the basis of result we differentiate between complexities of inheritance and interface.

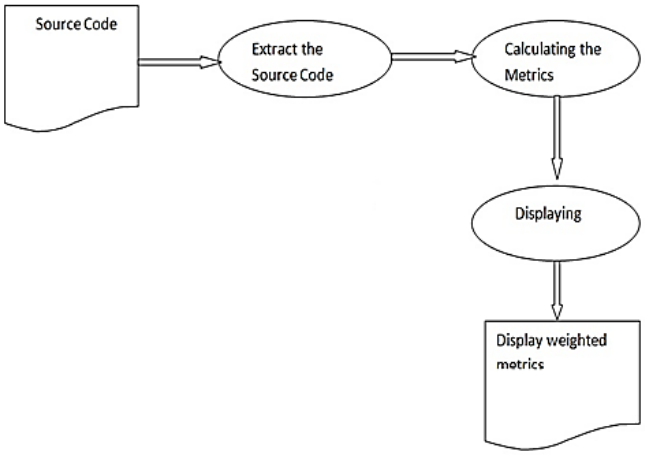

Figure 2. Proposed System Architecture

Thus, measuring the relationships has become a prerequisite to develop efficient techniques for analysis and maintenance. The extent of coupling and cohesion in an object-oriented system has implications for its external quality. Various static coupling and cohesion metrics have been proposed and used in past empirical investigations; however none of these have taken the run-time properties of a program into account. As program behavior is a function of its operational environment as well as the complexity of the source code, static metrics may fail to quantify all the underlying dimensions of coupling and cohesion.

IV. RESULTS AND DISCUSSION

Software functionality very well, and also how can we use the software functionality in new environment thus we can find our purpose with few fault and few pace. And it also increases the ratio since we utilized software functionality effectively to receive the desire purpose of the project.

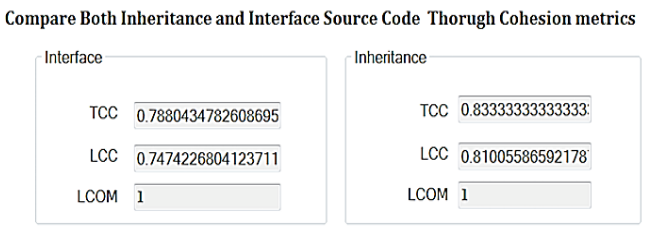

Figure 3. Calculate TCC ,LCC and LCOM metric for Inheritance and Interface Program

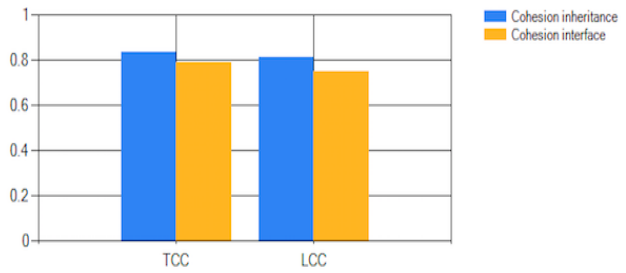

Figure 4. Graph show TCC, LCC and LCOM metric for Inheritance and Interface Program

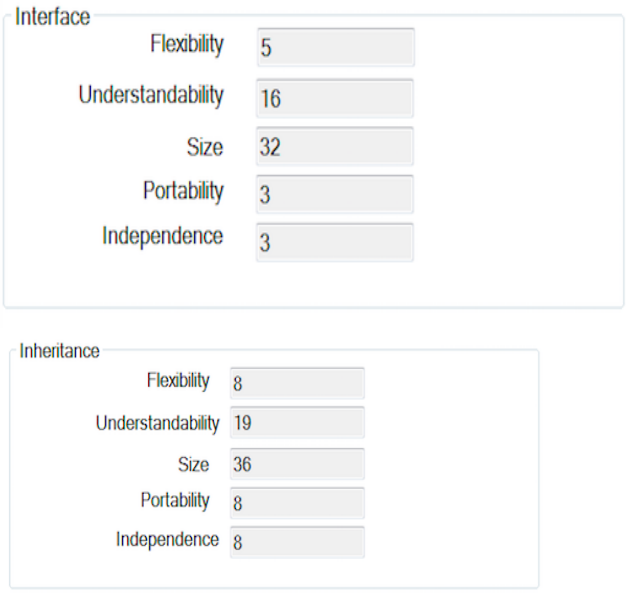

Figure 5. Calculate Size, Flexibility, Portability and Indecency metric for Inheritance and Interface Program

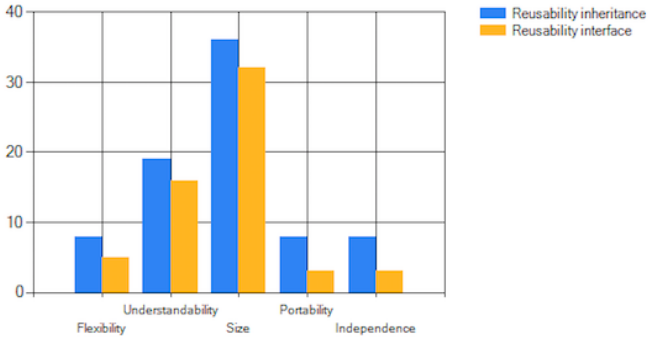

Figure 6. Graph shows Size, Flexibility, Portability and Indecency metric for Inheritance and Interface Program

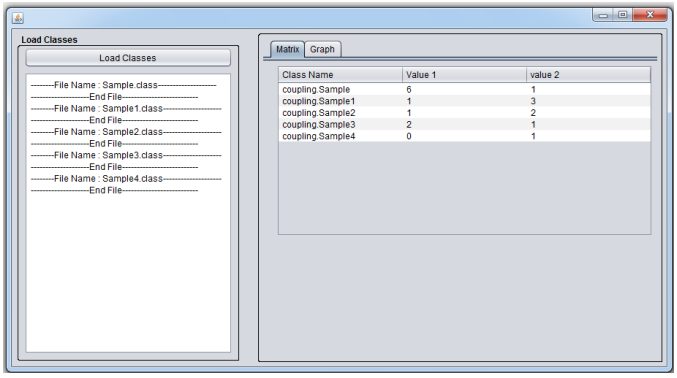

Figure 7. Calculate CBO

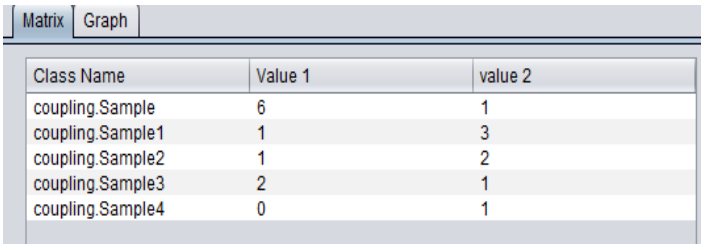

Figure 8. No of Association

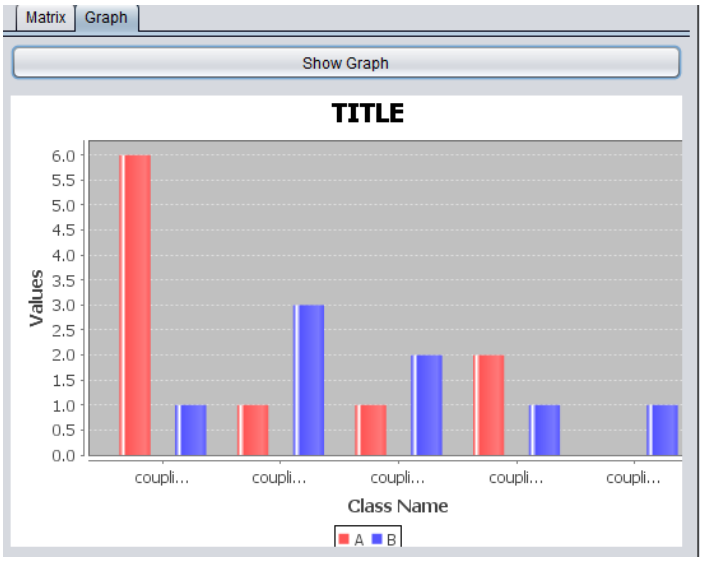

Figure 9. Number of Dependencies In metric and Number of Dependencies out metric for Inheritance Program

V. CONCLUSION

To improve modularity and encapsulation the inter class cohesion measures should be larger. By using more interfaces compared to inheritance the coupling measures are reduced. Good abstractions typically exhibit high cohesion. In Comparison of cohesion in between inheritance and interface for the modules, functions, attributes, classes in oops through cohesion metrics is done, and interface is calculated as more reusable code than inheritance. The more independent a class it is easier to be reused by another application."

VI. REFERENCES

[1] V. Krishnapriya, K. Ramar, "Exploring the Difference Between Object Oriented Class Inheritance and Interfaces Using Coupling Measures," ace, pp.207-211, 2010 International Conference on Advances in Computer Engineering, 2010

[2] K.K.Aggarwal, Yogesh Singh, Arvinder Kaur, RuchikaMalhotra. "Empirical Study of Object-Oriented Metrics",2006

[3] Martin Hitz, BehzadMontazeri."Measuring Coupling and Cohesion.In Object-Oriented Systems" in Angewandte Informatik (1995)

[4] James M. Bieman andByung-kyookang."Cohesion and Reuse in Object Oriented System" Department of Computer Science, Colorado State University Fort Collins,Colorado,1995

[5] Shyam R. Chidamberand Chris F. Kemerer" A Metrics Suite For object Oriented Design" IEEE Transactions on software Engineering, Vol. 20, No. 6, June 1994

[6] KrishnaprasadThirunarayan." Inheritance in Programming Languages" Department of Computer Science and Engineering ,Wright State University ,Dayton, OH-45435

[7] ArtiChhikara Maharaja Agrasen College, Delhi, India. R.S.Chhillar "Applying Object Oriented Metrics to C#(C Sharp) Programs"Deptt. Of Computer Sc. And Applications, Rohtak, India. SujataKhatriDeenDyalUpadhyaya College, Delhi, India(2011)

[8] Christopher L. Brooks, Chrislopher G. Buell, "A Tool for Automatically Gathering Object-Oriented Metrics", IEEE, 1994

[9] Friedrich Stiemann, Philip Mayer and Andreas Meibner, "DecouplingClasses with Inferred Interfaces", Proceedings of the 2006 ACMSymposium on Applied Computing, P.No:1404 – 1408.

[10] Pradeep Kumar Bhatia, Rajbeer Mann, " An Approach to Measure Software Reusability of OO Design", Proceedings of 2nd International Conference on Challenges & Opportunities in InformationTechnology(COIT-2008),RIMT-IET,MandiGobissndgarh, March 29, 2008.

[11] Fried Stiemann, Wolf Siberski and Thomas Kuhne, " Towards the Systematic Use of Interfaces in Java Programming", 2nd Int. Conf. on the Principles and practice of Programming in Java PPJ 2003, P.No:13-17.

[12] Girba, T.; Lanza, M.; Ducasse, S. (2005) Characterizing the Evolution of Class Hierarchies. Proceedings of the 9th European International Conference on Software Maintenance and Reengineering. Manchester, UK, pp.2-11.

[13] Gilb, T. (1976) Software Metrics. Chartwell-Bratt, Cambridge, MA.

[14] Hall, T., Rainer, A., Jagielska, D. (2005) Using software development progress data to understand threats to project outcomes. Proceedings of the 11th IEEE International Software Metrics Symposium (METRICS 2005). Como, Italy, 10 pages.

[15] Harrison R., Counsell S. and Nithi R.: "Experimental Assessment of the Effect of Inheritance on the Maintainability of Object-Oriented Systems", the Journal of Systems and Software, vol. 52, pp. 173-179, 2000.

Author Profile:

Vinay Mishra is currently working as Assistant Teacher in SMN Inter College, Allahabad. He has done his Bachelor of from Allahabad Institute of Engineering & Technology from Allahabad (U.P.) in the year 2012.