Package Level Cohesion Metric for Object-Oriented Design

Sandip Mal[1], Kumar Rajnish[2], Sanjeev Kumar[3]
[1]Dept. of CSE, BIT, Mesra, Ranchi, India
Sandip.mal1987@gmail.com
[2]Dept. of IT, BIT, Mesra, Ranchi, India
krajnish@bitmesra.ac.in
[3]Dept. of IT, Siddhant College of Engineering,Pune, India
sksaisan@gmail.com

***Abstract*— This paper presents a new package cohesion metric (CohP), which is based on the properties of elements of a package and dependencies with in the package elements. The proposed metric has been validated theoretically against Briand properties as well as empirically using packages taken from two open source software systems. An attempt has also been made to present a positive Spearman correlation between CohP values and Average Effort require to extend the packages. The results indicate that proposed metrics is used to predict extendibility of a software system.**

Keyword- Object-Oriented, cohesion metric, package, quality factor, extendibility.

I. INTRODUCTION

Software engineering is an engineering discipline that is concerned with all aspects of software production. Software products consist of developed programs and associated documentation. Essential product attributes are modifiability, dependability, understandability and usability. Cohesion measures the degree of interaction and relationships among modules, such as classes, methods, attributes, and packages within a block. Cohesion measure has important applications in software development and maintenance. In another way coupling with in a block is called cohesion. They are used to help developers, testers and maintainer's reason about software complexity and software quality attributes. One of the main goals behind OO analysis and design is to implements a software system where elements of a package have good interaction among them. This paper presents a package level cohesion metric (CohP) and shows how elements are dependent with each other. A theoretical validation and empirical validation of the metric also present in this paper.

For OO systems, most of the cohesion metrics have been defined up to class level [1-9] and only a few metrics exist for measurement of cohesion at the higher levels of abstraction in OO systems [10-11]. Other work related to packages or other higher abstraction levels has been carried out in [12-16] [24].

The rest of the paper is organized as follows. Section 2 describes some basic definitions related to package. New package cohesion metric (CohP) with a working example has been described in section 3. Section 4 provides theoretical validation of CohP against Briand properties. Section 5 presents a case study on open source software system. Section 6 presents conclusion and future work.

II. BASIC DEFINITIONS

This section presents some basic definitions related to the packages and properties regarding package structure. The measure of CohP is based on packages and their structure. These definitions and properties are useful for theoretical validation and empirical evaluation.

***Package*:** Packages may consist of classes, sub packages and interfaces (Java/C# application) as their elements. Further, these sub-packages also may contain classes, sub packages and interfaces as their elements. This leads to a hierarchical structure of packages in a software system. A package at a hierarchical level I contains elements and relations with other packages. It can be represented as $p_i = (E_{i+1}, Ri_{+1})$. Where E_{i+1} represents set of elements of package p_i present at level $i+1$, which may be classes or sub-packages or interfaces, and R_{i+1} is a set of relationships on E_{i+1} at hierarchical level $i+1$.

As shown in Figure1, for a package at hierarchical level i, p_i is represented as (E_{i+1}, R_{i+1}). Where E_{i+1} represents a set of elements of package p_i present at level $i+1$, which are classes (C^1_{i+1}, C^2_{i+1}, C^3_{i+1}) and a sub package (p^1_{i+1}), and R_{i+1} is a set of relationships on E_{i+1} at hierarchical level $i+1$ which are the relationship between C^1i+1 and C^3i+1 and relationship between p^1_{i+1} and C^1_{i+1}.

***Sub Package*:** For any package p_i in a system, sub package of p_i denotes an element of p_i which is package itself and is present at level $i+1$ in the hierarchy. From figure 1, the package $p^1_{i+1} = (E_{i+2}, R_{i+2})$ is said to be a sub package of package $p_i = (E_{i+1}, R_{i+1})$ if $p^1_{i+1}$is the element of set E_{i+1}.

***Disjoint Packages*:** For any two packages p^1_{i+1}, p^2_{i+1} are present in the same level i+1 in figure 1. Then packages p^1_{i+1}, p^2_{i+1} are said to be disjoint packages if $p^1_{i+1} \cap p^2_{i+1} = \emptyset$.

***Empty Package*:** A package (p^2_{i+1}) that have no elements in it and hence, there is no relations with other packages. It is denoted by (Ø, Ø).

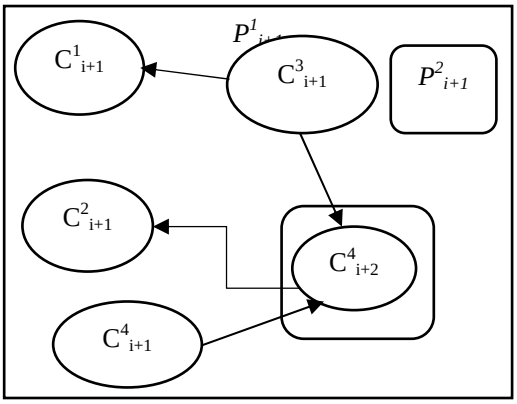


Fig. 1. Example of package

III. PACKAGE COHESION METRIC AND WORKING EXAMPLE

Package in an OO system have set of elements and relationships between these elements. These set of elements are nothing but classes, sub packages or interfaces. The relationship between the elements of two different elements of a package is denoted by r (e_i, e_j). It means there is a relation of an element ei to an element j of a package. If an element e_i is related to an element e_j, then it is not necessary that element e_j is also related to element e_i and there is a directional connection [18-21] between e_i and e_j and denoted by e_i->e_j. r (e_i, e_j) =1 if there is a directional connection between e_i and e_j, otherwise 0. Four different type of connection have been discussed below.

- *Class-Class Connection:* If one class (or interfaces) of a package is related with a class (or interfaces) of same package, then there exists a class-class type of connection between them.
- *Sub Package-Sub Package Connection: P*ackages may consist of sub-packages as its elements at the next level. While elements of a sub package of a package is related with a element of a sub package of same package.
- *Class – Sub Package Connection:* This type of connection exists between elements of a package with the elements of a sub package of same package. That means elements (class) of level i+1 of a package p^1_i is related with the elements of level i+2 of a sub package p^3_{i+2} of same package p^1_i.
- *Sub Package-Class Connection:* This type of connection exists between elements of a sub package a package with the elements of same package. That means elements of level i+2 of a sub package p^3_{i+2} of a package p^1_i is related with the elements (class) of level i+1 of a package p^1_i.
- *Sub Package- Sub Package Connection:* This type of connection exists between elements of a sub package of a package with the elements of another sub package of same package. That means elements of level i+2 of a sub package p^3_{i+2} of a package p^1_i is related with the elements of a sub package p^4_{i+2} of level i+2 of a package p^1_i.

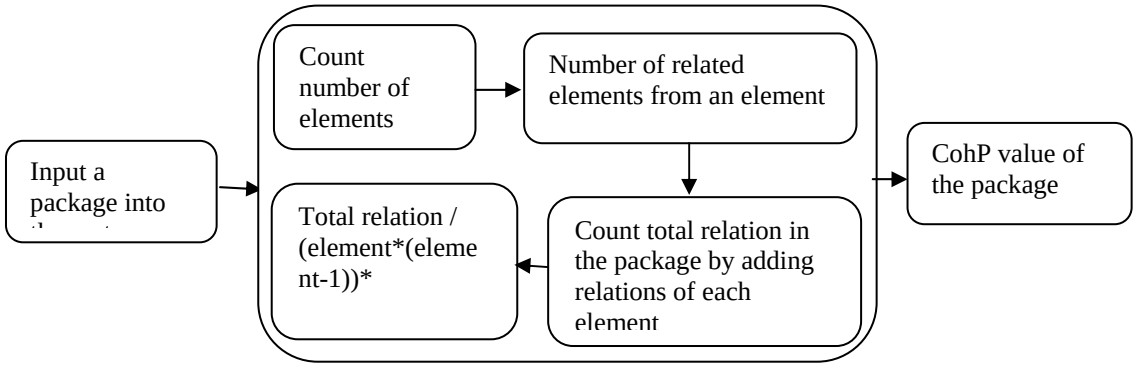
Proposed cohesion metric is defined as follows:

CohP (I) = (Number of relations between the elements of a package) / n (n-1)

Where n is the no of element of package I. So CohP value is between 0 and 1. A Package with high cohesive value indicates that, CohP value near to one and low cohesive indicates that CohP value near to zero.

In figure 1, within a single package (P_i) 5 elements are there. So CohP (P_i) = 4/ (5*4) = 0.2

Tool for measuring proposed cohesion metric:



We have developed a system using java named CohP, which takes a package as an input and fids out the CohP value of the package. The tool is applied on two open source software system for our case study.

IV. THEORETICAL VALIDATION

The proposed cohesion measures are validated theoretically by analysing their mathematical properties. For this purpose, five properties given by Briand *et al*. in [17] are used and these properties provide a useful guideline in construction and validation of coupling measures in a precise manner and these properties are necessary to prove the usefulness of a cohesion measure although not completely sufficient.

Property 1: Non-Negativity

The value of cohesion of a package as defined by our measures in an OO system will always be non-negative.

CohP (I) $\geq$0

Thus, CohP satisfy Property 1.

Property 2: Null Value

If the number of elements in a package is zero or there is no relationship between the elements of a package, then cohesion will be null of the package, then the value of CohP will be null for that package. So CohP satisfies property 2.

Property 3: Monotonicity

If an additional relationship is added between two elements of a package, then according to this property the cohesion of the package must not decrease. If we add an additional relationship between the elements of a package, then the CohP will increase or at least remain the same, but can never decrease in any case. So CohP also satisfies property 3.

Property 4: Merging of Packages

This property states that merging of two elements of a package must not increase CohP value because some of the relationships may disappear on merger. Let *P* be a package, and e1, e2, e3, e4.be the packages in *P*. Let e5be the element that is obtained by merging of e1 and e2. Then, in any case, *CohP* (*P*) value before merging of e1 and e2 $\geq$ *CohP* (*P*) value after merging of e1 and e2. Thus, CohP also satisfies Property 4.

Property 5: Merging of Unconnected Packages

This property states that merging of two unconnected elements of a package must not increase CohP value of the package. When two or more elements having no relationships between them are merged, cohesion cannot increase because apparently unconnected elements are being encapsulated together in a single element. Let e*1*and *e2*be two elements of a package *P*. Let *e1+e2* be the element, which is the union of e1and *e2*. If no relationships exist between elements e*1* and e*2*, then CohP value before merging is always greater than CohP value after merging. Thus, CohP satisfy this property.

V. CASE STUDY OF COHP ON OPEN SOURCE SOFTWARE SYSTEM

Two open source software projects have been chosen for case study. XGen [22] Source Code Generator, that creates Java source code from a simple XML document and its main function is to generate JDBC compliant beans that allow object level persistence to relational databases and The Byte Code Engineering Library (Apache BCEL) [23] is intended to give users a convenient way to analyse, create, and manipulate (binary) Java class files (those ending with .class).The basic data about these two projects are given in Table 1. For CohP analysis 4 package of BCEL and 7 package of XGen have been taken. BCEL have 367 classes in 4 packages and

XGen have 73 classes in 7 packages. Table 2 and Table 3 list the names of packages of BCEL, XGen and the number of classes contained in each package.

TABLE 1: Information about Project Taken for Case Study

Software Project	**BCEL 5.1**	**XGen 0.5.0**
No of Package	4	7
No of Classes	367	73

A. Results

The CohP has been applied to seven packages taken from XGen and four packages taken from BCEL software systems. The CohP value of packages is given in Table 2. It may not be always true that a package with the large number of classes have more connections with in the package, as an example, package org.apache.bcel.generic and workzen.xgen.ant. Three teams of three members each have been set up and assigned these packages to three teams. These members are well experienced of Java programming. First, calculate the effort required to fully understand the functionality and extend of these packages by these three teams and rank the effort from 1 to 10. A higher rank indicates that more effort spent on extending the package. Table 2 shows the effort required by each team and average effort. Then all the packages have been given to most experienced team to modify the package. The teams add some classes to extend the packages.

TABLE 2: Effort and CohP values of seven packages Taken from two open source system

Sl. No.	**Name of Package**	**Team**			**Average Effort**	**No. of Classes**	**No. of Classes added**
1	workzen.xgen.ant	7	6	6	6.33	5	3
2	workzen.xgen.engine	1	2	1	1.33	2	2
3	workzen.xgen.loader	4	3	5	4.00	7	0
4	workzen.xgen.model	2	2	1	1.67	17	1
5	workzen.xgen.test	2	3	1	2.00	23	4
6	workzen.xgen.type	1	1	1	1.00	15	0
7	workzen.xgen.util	4	6	5	5.00	4	2
8	org.apache.bcel.classfile	4	5	4	4.33	51	3
9	org.apache.bcel.generic	2	1	3	2.00	225	6
10	org.apache.bcel.util	4	5	5	4.67	28	0
11	org.apache.bcel.verifier	2	4	3	3.00	63	4

TABLE 3: Number of classes and COP values of four packages Taken from Apache BCEL

Sl. No.	**Name of Package**	**CohP**
1	workzen.xgen.ant	0.65
2	workzen.xgen.engine	0.00
3	workzen.xgen.loader	0.12
4	workzen.xgen.model	0.01
5	workzen.xgen.test	0.04
6	workzen.xgen.type	0.00
7	workzen.xgen.util	0.25
8	org.apache.bcel.classfile	0.20
9	org.apache.bcel.generic	0.01
10	org.apache.bcel.util	0.15
11	org.apache.bcel.verifier	0.08

B. Empirical Validation

This study shows a positive Spearman correlation (0.072) between Average Effort and the number of classes added to extend the system. It is also observed that number of classes added and CohP gives a positive Spearman correlation (0.067). From this transitive relation we can say that, CohP value is the good predictor of the effort require for extending software system. This property of CohP indicates the usefulness of the proposed metrics.

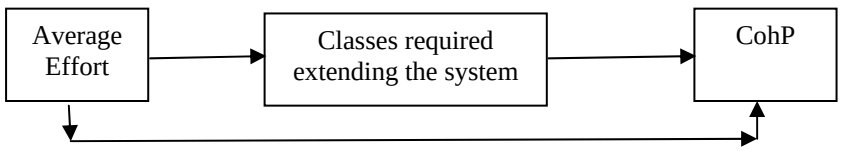


Fig 2: Transitive relation of CohP and Average Effort.

VI. CONCLUSION AND FUTURE WORK

In this paper, an attempt has been made to propose a new package cohesion metric, which is based on formal definitions, properties and relations of elements of a package. The proposed metrics has been validated theoretically as well as empirically. The theoretical validation of CohP satisfies all the properties presented by Briand. In addition to the proposal and theoretical validation, this paper has also presented empirical data on CohP from two open source software system (Apache BCEL, XGen 0.5.0). Both systems developed in java. So, this study clearly provided that CohP is the valid indicator of external quality attributes of the software such as extendibility. This firmly believes us that this work will encourage other researchers and developers to use the results obtained from this study to predict and measure several other software quality attributes.

The future scope includes some fundamental issues

- To analyze the nature of proposed metric with performance indicators such as design, maintenance, effort and system performance.
- Another interesting study would be together different coupling metric at various intermediate stages of the project. This would provide insight into how application reusability, maintainability, testability evolves and how it can be managed and controlled through the use of metrics.

REFERENCES

[1] S. R. Chidamber, C. F. Kemerer, "Towards a metrics suite for object oriented design." *In Proc. the 6th ACM Conf. Object- Oriented Programming: Systems, Languages and Applications (OOPSLA), Phoenix, AZ,* Oct. 6-11, 1991, pp.197-211.

[2] S. R. Chidamber, C. F. Kemerer, A metrics suite for object oriented design. IEEE Transactions on Software Engineering, 1994, 20(6): 476-493.

[3] J. Eder, G. Kappel, M. Schrefl "Coupling and cohesion in object-oriented systems." *Technical Report, University of Klagenfurt,* 1994.

[4] K. Rajnish and V. Bhattacherjee, "Class Cohesion: An Empirical and Analytical Approach", *International Journal of Science and Research (IJSR), Victoria, Australia,* Vol.2, No. 1, 2007, pp.53-62, http://www.international.au.in.

[5] S. Mal, K. Rajnish, Applicability of Weyuker's Property 9 to Inheritance Metric. *International Journal of Computer Applications, Foundation of Computer Science, USA,* Volume 66– No.12, March 2013.

[6] B. H. Sellers, L. L. Constantine, I. M. Graham, "Coupling and Cohesion towards a valid metrics suite for object oriented analysis and design", *Object oriented systems,* vol. 3, 143-158, 1996.

[7] Y. S. Lee, B. S. Liang, S. F. Wu, F. J. Wang "Measuring the coupling and cohesion of an object-oriented program based on information flow". *In Proc. International Conference on Software Quality, Maribor, Slovenia,* Nov. 6-9, 1995, pp.81-90.

[8] B. Xu, Z. Chen, J. Zhao, "Measuring cohesion of packages in Ada95". *In Proc. Annual ACM SIGAda International Conference on Ada: The Engineering of Correct and Reliable Software for Real-Time &Distributed Systems Using Ada and Related Technologies, San Diego, California, USA,* Dec. 7-11, 2003, pp.62-67.

[9] G. Gui, P D. Scott, "Coupling and cohesion measures for evaluation of component reusability". *In Proc. International Workshop on Mining Software Repositories, Shanghai, China,* May 22-23, 2006, pp.18-21.

[10] S A. Ebad, and M. Ahmed,"An Evaluation Framework for Package-Level Cohesion Metrics", *International Conference on Future Information Technology, Singapore* vol.13 (2011), pp-239-243.

[11] E Allen, T Khoshgoftaar. "Measuring coupling and cohesion of software modules: An information theory approach." *In Proc. the Seventh International Software Metrics Symposium, London, UK,* April 4-6, 2001, pp.124-134.

[12] T Xu, K Qian, X. He. "Service oriented dynamic decoupling metrics." *In Proc. the 2006 International Conference on Semantic Web and Web Services (SWWS'06), Las Vegas, USA,* June 26-29, 2006, pp.170-176.

[13] F B Abreu, G Pereira, P Sousa. "A coupling-guided cluster analysis approach to reengineer the modularity of object-oriented systems. " *In Proc. the 4th European Conference on Software Maintenance and Reengineering (CSMR'2000), Zurich, Switzerland,* Feb. 29-March 3, 2000, p.13.

[14] X Franch, J P Carvallo. "A quality-model-based approach for describing and evaluating software packages". *In Proc. IEEE Joint International Conference on Requirements Engineering (RE'02), Essan, Germany,* Sept. 9-13, 2002, pp.1-8.

[15] H Washizaki, H Yamamoto, Y Fukazawa. "A metrics suite for measuring reusability of software components". *In Proc. the Ninth International Software Metrics Symposium (METRICS'03),* 2003.

[16] Li W, Henry S. "Object-oriented metrics that predict maintainability". *Journal of Systems and Software,* 1993, 23(2): 111-122.

[17] Briand L, Morasca S, Basili V. Property-based software engineering measurement. IEEE Transactions of Software Engineering, 1996, 22(1): 68-86.

[18] Northcott M, Vigder M. "Managing dependencies between software products. Lecture Notes in Computer Science 3412", *Franch X, Port D (eds.), ICCBSS 2005, Springer-Verlag, Berlin/Heidelberg,* 2005, pp.201-211.

[19] Martin R. "Object oriented design quality metrics: An analysis of dependencies". *ROAD,* 1995, 2(3).

[20] L Grunske, B Kaiser. "An automated dependability analysis method for COTS-based systems". *Lecture Notes in Computer Science 3412, Franch X, Port D (eds.), ICCBSS 2005, Springer-Verlag, Berlin/Heidelberg,* 2005, pp.178-190.

[21] K K Aggarwal, Y Singh, J K. Chhabra, "Complete dependency matrix for object-oriented software". *International Journal of Management and Systems (IJOMAS),* 2003, 19(1): 43-54.

[22] http://sourceforge.net/projects/xgen/

[23] http://jakarta.apache.org/

[24] V. Gupta, J. K. Chhabra "Package coupling measurement in object-oriented software". *Journal of computer science and technology* 24(2): 273-283 Mar. 2009.

About Authors:

Mr. Sandip Mal received B.Tech degree in the department of Computer Science and Engineering from West Bengal University of Technology in the year 2008. He has also Completed ME (Software Engineering) from Birla Institute of Technology, Mesra, Ranchi, Jharkhand, India in the year of 2012. Currently, he is pursuing Ph.D. on Software Quality Metrics from Birla Institute of Technology, Mesra, Ranchi, Jharkhand, India. His Research area is Object-Oriented Metrics, Software Engineering, Database System, and Image Processing.

Dr. Kumar Rajnish is an Assistant Professor in the Department of Information Technology at Birla Institute of Technology, Mesra, Ranchi, Jharkahnd, India. He received his PhD in Engineering from BIT Mesra, Ranchi, Jharkhand, India in the year of 2009. He received his MCA Degree from MMM Engineering College, Gorakhpur, State of Uttar Pradesh, India. He received his B.Sc Mathematics (Honours) from Ranchi College Ranchi, India in the year 1998. He has 23 International and National Research Publications. His Research area is Object-Oriented Metrics, Object-Oriented Software Engineering, Software Quality Metrics, Programming Languages, and Database System.

Sanjeev Kumar is an Asst. Professor in Department of Information Technology at Siddhant College of Engineering, Pune. He did his B.Tech. in Information Technology and Masters in Quality Engineering in the Department of Management at the Birla Institute of Technology, Mesra, Ranchi, India. He has 4 International Journal publications. He has 2 years of teaching experience and did his masters project at ITR, DRDO, Chandipur, Orissa.