International Journal of Application or Innovation in Engineering & Management (IJAIEM)
Web Site: www.ijaiem.org Email: editor@ijaiem.org
Volume 3, Issue 4, April 2014 ISSN 2319 - 4847

Analysis of Cohesion Metrics for Aspect Oriented System

Himanshu Dua[1], Puneet Jai kaur[2]

[1]Research Scholar UIET Panjab University, Chandigarh, India
[2]Assistant Professor, UIET, Panjab university, Chandigarh, India

Abstract

During development and maintenance phase when software grows, they may lose their quality. With the help of software quality metrics overall or partial quality of a system can be measured. Various metrics have been proposed to measure software quality. Cohesion is most widely used metric for measuring software quality in terms of relatedness among the members of class. In literature, many cohesion metrics have been proposed for object-oriented system but a few for aspect-oriented system. Aspect–oriented programming (AOP) is a new programming technique by which crosscutting concerns can be handled efficiently with core concerns. Crosscutting concerns are implemented in AOP by a new modular unit called 'Aspect'. Higher the cohesion higher is the quality. This paper presents the widely used cohesion metrics proposed for AOP. Cohesion metrics for AOP can be easily extended from cohesion metrics of object oriented, since AOP is derived from Objection oriented. This paper addresses the development and implementation of various cohesion metrics for AOP design paradigm.

Keywords: Aspect-oriented programming (AOP), Object - oriented programming (OOP), Aspect, Lack of Cohesion in operations (LCOO).

1. INTRODUCTION

Software metrics plays important role in software engineering. Cohesion, coupling, polymorphism and complexity are several main software attributes by which software quality of a system are measured. Cohesion is the most important factor in software development. It provides high maintainability and reusability in a software [1][2]. Higher class cohesion means that maximum members of class are related to each other and it should be difficult to split a class. This also promotes the encapsulation. Low cohesion indicates members of class are not related to each other and it represents the poor design and structure of class and also increases probability of errors during development phase. So cohesion is a desired goal for good design [3].

Many software programming techniques have been developed. In procedural oriented programming data is exposed to whole program therefore the problem of security is encountered. Then OOP introduced which divides the programming tasks into the objects and also provides encapsulation, abstraction, polymorphism and inheritance. Real world problems are difficult to implement in OOP having completely modularize model.

Kiczales, first introduced the concept of AOP by which one can separate the concerns [3]. AOP provides more modularization of crosscutting concerns that helps the programmers to reuse the code [13]. At low level, concerns can be primitive like adding a variable but at higher level concerns are coarser such as transaction management. In library management, main focuses are on account management and book management, these are the core concerns. Other concerns, like security, logging are crosscutting concerns. In OOP core concerns can be handled easily but if crosscutting concerns are handled with OOP then code become more tangled and scattered [12].

Some of the authors believe AOP design can even lead to new types of faults [9].But further research in this area has proved effectiveness and feasibility of AOP approach [10]. AOP can also be used to analyse the legacy system during runtime to trace the system behaviors [14][15].

Here the various cohesion metrics for AOP have been discussed. High cohesion among Aspects shows high maintainability and reusability. Maintainability refers to have modification in the software after delivery and reusability refers to use module again to reduce the coding. Proper separation of crosscutting concerns in AOP indicates high cohesion.

2. COHESION METRICS FOR ASPECT-ORIENTED SYSTEM

There are many metrics that have been proposed for OOP like LCOM1, LCOM2, LCOM3, Coh, TCC, LCC, CBMC etc. Following are several metrics which have been developed for AOP.

- LCOO proposed by Sant Anna in 2003.
- Dependencies metrics proposed by Zhao and Xu in 2004.
- ACOH proposed by Gelinas in 2006.

Each metrics is discussed one by one and its pros and cons are discussed:

A. LCOO

This metrics measure the lack of cohesion in components of software. This metric is extension of LCOM metric which is proposed by C&K [5] in 1994. Cohesion refers to the degree of relatedness in between the internal components of a system. In this method, instance variable is taken into account by which the cohesion of a system is measured.
To calculate the value of LCOO, Let class C has n methods and advices i.e. { M_1, M_2.....M_n } which has { I_1,I_2....I_n } set of instance variable respectively. First calculate the intersection of sets of the instance variables. Let |P| number of null intersections and |Q| is number of non-empty intersection between sets of instance variables.

$$\text{LCOO}=|P| - |Q|, \text{ if } |P|>|Q|. \qquad (1)$$
$$\text{LCOO} = 0 \text{ otherwise.}$$

Lower the value of LCOO denotes the higher cohesion. Higher the value of LCOO denotes that components are difficult to maintain and reuse.
LCOO is extension of LCOM metric which is widely used in OOP. LCOO only takes instance variables into account for calculating the cohesion and it neglect any other type of attributes. It also neglects the nested methods and attributes used in class. So LCOO value doesn't always reflect the true cohesion value. It may lead to some difficulties in interpretation of results.
Consider 3 cases presented in Fig.1

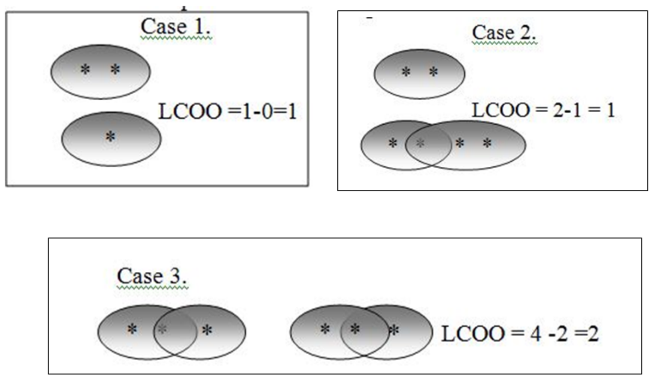

Figure1: Calculation of LCOO

Values of LCOO in case 1 and case 2 are same which is difficult to understand and in case 3 value of LCOO has been increased which show some conflicting results. Cohesion metrics LCOO should not be used as changeability indicators [11].

B. Dependency metrics

This metrics is proposed by Zhao and Xu to measure the cohesion on basis of relatedness among the attributes and modules. There are three types of dependencies used in this metrics which are inter-attribute dependencies, inter-module dependencies and module-attribute dependencies. Aspect cohesion has three tuples(□□$_{a,}$ □$_{ma}$ and □$_m$) and to calculate total cohesion of a system the result of all three dependencies is combined.
Inter-attribute cohesion refers to the relatedness among the attributes in aspects and classes. To measure the inter-attribute cohesion (□$_a$), assume a set $D_{a,}$ as record of attributes on which attribute *a* depends. Then inter-attribute cohesion of Aspect A is given as:

$\gamma_a(A)=0$ if k=0
$\gamma_a(A)=1$ if k=1

$$\gamma_a(A)= \frac{1}{k}\sum\frac{|D_a(a_i)|}{k-1} \qquad (2)$$

where k is number of attributes in A.

$\frac{|D_a(a_i)|}{k-1}$ represents the degree on which a_i depends on other attributes in A.

a.

Module –attribute (γ_{ma}) cohesion refers to tightness among modules and attribute in an aspect. To measure the module-attribute cohesion of aspect A for each module m, assume $D_{ma}(m)$ is set of attributes in aspect A that are used in module m and $D^0_{ma}(m)$ is set of attribute of A that used in module m and also referred in other modules.

$\gamma_{ma}=0$ if n=0

γ_{ma}=1 if n=1and $|D_{ma}(m_i)|\neq 0$

$$\gamma_{ma}= \frac{1}{n}\sum\frac{|D^{\circ}_{ma}(m_i)|}{|D_{ma}(m_i)|} \text{ for others} \qquad (3)$$

It is not necessary that if modules are not related by its attribute, it is not related at all. When one module is called by other module this type of cohesion is inter-module cohesion.

$\gamma_m(A)$=0 if n=0;
$\gamma_m(A)$=1 if n=1;

$$\gamma_m(A)=\frac{1}{n}\sum_{i=1}^{n}\frac{|D_m(m_i)|}{n-1} \text{ if } n>1; \qquad (4)$$

n is number of modules in Aspect A.

$\frac{|D_m(m_i)|}{n-1}$ represents the relatedness among the modules.

Cohesion can be measured by 3 factors as a 3 tuple or combining the three facets as a whole.□□$_{a,}$ □$_{ma}$ and □$_m$ are used to compute the aspect cohesion independently and can be combined as single unit

$$x=\gamma_a*\beta 1+\gamma_{ma}*\beta 2+\gamma_m*\beta 3$$
$$\beta\iota+\beta 2+\beta 3=1 \qquad (5)$$

The dimensions inter-attribute dependencies, inter-module dependencies and module- attribute dependencies are a good measure for assessing the aspect cohesion. This metrics doesn't consider the application environment and is focused only on the features of an aspect. It doesn't take aspect inheritance into account.

C. ACoh

This metrics is based on the relatedness among the modules inspired by class cohesion measurement [5].This metrics reflects the quality of design of a system. Higher value of ACoh denotes the good design of software. ACoh consider two types of connection that are modules-data connection and modules-modules connection [6].

Modules-data connection considers all attribute used directly and indirectly. Directly attributes means those attributes which are referred in module. Indirectly attributes means those attributes which are not directly referred in a module but used in another module which is called in it. Consider an example with help of a relationship figure between modules and attributes. A_1**,** A2 are the attributes and $M_{1,}$ M2, M3 are the modules used in an aspect. A_1 is used by M_2 and M_1 directly, also used indirectly by M_3**.** A_2 is used directly by M_2 and also used indirectly by M_3.

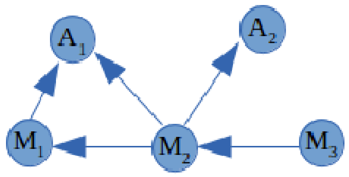

Figure2:

In module -module connection we consider all modules used directly or indirectly by a module. It can be easily understand it with the help of example. Consider a relationship diagram between modules.

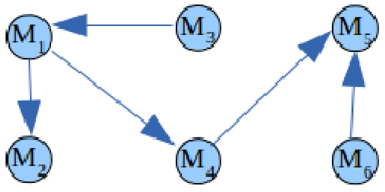

Figure3:

Suppose M_1, M_2, M_3, M_4, M_5, M_6 are the modules in an aspect and there are some connections between them. M_2, M_4, M_5 are called in M_1 or can say used in M_1.M_4 and M_2 are directly used by M_1. M_5 is indirectly connected with M_1 because M_5 is directly connected by M_4 which is in direct connection of $M_{1.}$ ACoh metrics value gives us the degree of relatedness between the aspect members. Two modules can be connected by sharing attributes or by sharing modules or by sharing both. Consider an undirected graph with modules as vertices and there is edge between two vertices if they are related by either way. NC represent number of edges in a graph and NM represents the maximum number of connection possible in between the modules which can be given by N*(N-1)/2 where N is number of modules.

$$\text{ACoh} = \frac{NC}{NM} € [0, 1]. \qquad (6)$$

Value 1 of the ACoh metrics denotes the highest cohesion in an aspect. Value 0 of ACoh represents that no component of aspect is related in any way. ACoh metrics reflects design problems accurately but it doesn't consider the indirect relationship between the aspect members. It also doesn't take aspect inheritance into account.

3. CONCLUSION

Three main cohesion metrics used in AOP are mentioned above. Every metrics has its own different parameters. LCOO gives the relation among class and aspects. ACOH deals with dependencies criteria in module–data connection and module-module connection. The dependency metrics defines cohesion by relating the attributes and modules. It deals with inter-attribute dependencies, inter-module dependencies and module- attribute dependencies. But there are several limitations of these metrics. As LCOO does not specify how cohesion can be evaluated empirically. The approach to Dependency metric is complicated and it is difficult to use in real world problems. Table1 below gives the overview of the cohesion metrics

Table1: Description Of cohesion Metrics

AUTHOR	METRIC	STUDY
Sant Anna	LCOO	It only takes the instance variables into account .It don't include the nested component.
Zhao and Xu	Dependency metric	It doesn't take aspect inheritance into account though it is most efficient metrics.
Gelinas	ACoh	It reflects design problems accurately but it doesn't consider the indirect relationship between the aspect members

REFERENCES

[1] J.M. Bieman and B.K. Kang, "Cohesion and reuse in an object-oriented system", Proceedings of the Symposium on Software Reusability, Seattle, WA, pp. 259–262, April 1955.
[2] L.C. Braind, J. Daly and J. Wust, "A unified framework for cohesion measurement in object-oriented system", in proceeding of 4th international Software metrics symposium, pp. 43-53, 1997.
[3] G. Larman, Applying UML and Design Patterns,An introduction to object-oriented analysis and design and the unified process,second edition,Prentice Hall,2002.
[4] G. Kiczales, J.Lamping, A.Mendhekar, C.Maeda, C.V. Lopes, J.-M. Loingtier, J. Irwin , "Aspect-oriented programming", in Proceedings of the 11th European Conference on Object-Oriented Programming (ECOOP'97), Springer, 1997, pp. 220–242.
[5] Martin Hitz, Behzad Montazeri, "Chidamber & Kemerer's metrics suite: a measurement theory perspective", IEEE Transactions on software engineering, vol 22, No. 4, pp. 267-271, April 1996.
[6] Jean-Francois Gelinas, Mourad Badri, Linda Badri, "A Cohesion measure of Aspects", Journal of object technology, vol 5, no 7, pp.75-95, sept-Oct 2006.
[7] Jianjun Zhao, Baowen Xu, "Measuring Aspect Cohesion", 7th international Conference, FASE 2004, pp.54-68, April 2004.
[8] Kanika Arora, Abhishek Singhal, Avadhesh kumar, "Study of Cohesion for aspect oriented systems", International journal of engineering science and advanced technology, vol 2, issue 2, pp-332-337 ,March 2012.
[9] M. Mortensen and R. T. Alexander, "An approach for adequate testing and AspectJ programs", in 1st WTAOP, Chicago, USA, 2005.
[10] Cafeo B.B.P. , Masiero P.C., "Contextual Integeration testing of object oriented and aspect oriented programs: a structural approach for java and AspectJ", in proceedings of 25th Brazilian symposium on software engineering, pp. 214-223, 2011.
[11] Hind Kabaili , Rudolf K. Keller and Francois Lustman, "Cohesion as Changeability Indicator in Object- Oriented Systems", in proceedings of 5th European conference of software maintenance and reengineering, pp. 39-46,2001.

[12] Christina Chavez, Uira Kulesja and et al, "The AOSD research community in Brazil and its crosscutting impact", in proceedings of 25th Brazilian symposium on software engineering,pp.72-80,2011.
[13] Pawan kumar verma, Deepak dahiya and Pooja jain, "Using aspect oriented software architecture for enterprise systems development" in proceeding of 5th international conference of Digital information management,pp.448-453,2010.
[14] Liangyu chen,Jianlin wang, Ming Xu, Zhen bing zeng, "Reengineering of java legacy system based on aspect oriented programming" in proceedings of 2nd international workshop on Education technology and computer science, pp.220-223, 2010.
[15] Varun gupta, Jitender kumar chhabra, "Dynamic Cohesion measures for object oriented software", in journal of systems architecture, pp.452-462, 2011.

AUTHOR

Himanshu Dua currently pursuing M.E from Punjab university,Chandigarh and doing his research on dynamic cohesion metrics in aspect oriented programming

Puneet Jai Kaur currently working as an assistant professor in the Panjab University and pursuing the degree of Phd.