Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X
IMPACT FACTOR: 5.258

IJCSMC, Vol. 5, Issue. 7, July 2016, pg.312 – 320

MEASURING COHESION METRICS IN SOA

Simardeep Kaur
Department of Information Technology, Adesh Institute of Engineering and Technology, Faridkot, Punjab, India
simardeep2789@gmail.com

Abstract*— *A service-oriented architecture (SOA) is an architectural pattern in computer software design in which application components provide services to other components via a communications protocol, typically over a network. The principles of service-orientation are independent of any vendor, product or technology. A service is a self-contained unit of functionality, such as retrieving an online bank statement. By that definition, a service is an operation that may be discretely invoked. However, in the Web Services Description Language (WSDL), a service is an interface definition that may list several discrete services/operations. And elsewhere, the term service is used for a component that is encapsulated behind an interface. Cohesion metrics play an important role in empirical software engineering research as well as in industrial measurement programs. The Cohesion metrics presented in this paper measure the difference between class inheritance and interface programming. The metric values of class inheritance and interface prove which program is good to use. Our goal is comparing the inheritance and interface concepts in object oriented programming through cohesion- metrics. This paper addresses the Cohesion of service operations for service-oriented systems from the perspective of a service provider. The requirements of this project include Java 7 SE, JRE (Java Runtime Environment) 6 or later Eclipse IDE. The final result shows the number of attributes Number of classes, Method, lines of code Nested Block Depth, Number of parameters, and Lack of cohesion of methods. This paper demonstrates the measuring Cohesion for a Secure Service Oriented Architecture.

Keywords*— *"SOA, WSDL, LOC, Cohesion Metrics, Inheritance, JRE"

I. INTRODUCTION

Service-Oriented Architecture (SOA) is emerging as a promising development paradigm, which is based on encapsulating application logic within independent, loosely-coupled stateless services, that interact via messages using standard communication protocols and can be orchestrated using business process languages, The notion of a service is similar to that of a component, in that services, much like components, are independent building blocks that collectively represent an application. However, services are more platform independent, business-domain oriented, and autonomous and hence decoupled from other services as compared with components. Service-oriented systems in conjunction with supporting middleware represent Service-Oriented Architecture (SOA), a more abstract concept which is founded on the idea of discovery and orchestration whereby a business process or workflow can identify at runtime the most suitable services for a particular scenario and dynamically compose them in order to satisfy a particular domain requirement. Moreover, in SOA, enterprises should consider services as enablers of business processes that reflect workflows within and between organizations, rather than treating them simply as interfaces to software functionality. Although SOA is becoming an increasingly popular choice for the development of enterprise software, service-oriented (SO) design principles are not well understood and documented, with contradicting definitions and guidelines making it hard for software engineers and developers to work effectively with service-oriented concepts . Consequently, service-oriented systems are often developed in an ad-hoc fashion potentially resulting in lower-quality software being produced.

Cohesion

In cohesion the single components are required to attain the same task. It was introduced within the environment of same design. The module is stated by reviewing the relationship between the pairs of its processing elements. This term was defined as an action performed by a module. In other informal definition the ordinal scale is developed for measurement that describes the amount to which any action performed by the module contributed to the unified function [11]. Basically there are seven kinds of it. The most desirable to the least desirable

II. STATIC AND RUNTIME METRICS

There are so many metrics proposed to calculate the quality of the object oriented design. There are two categories in which design metrics are classified: 1) static 2) dynamic/runtime. The task of static metrics is to measure, what happen when execution of program takes place and also measure the quantity and complexity of different features of the source code. The task of run time metrics is to measure actually what happen when there is an execution of program take place. They find the run time behavior, complexity and also find the characteristics of source code. Without affecting the structure of research work, the developers are able for developing or designing the quality metrics. In compare of static metrics the analysis of run time metrics is very expensive and very complex in case of performance [12]. For the object oriented design environment the developer giving less importance to the run time metrics. There are so many situations which the static coupling and cohesion does not measure the real situation of the software these cases like dynamic binding, polymorphism and unused code present in software while it is less related to the classes at run time. Most of the real time applications have very complex and dynamic behavior that provide a strong reason to move our attention towards run time metrics instead of existing static metrics. Now our work is to find out whether run

time measures of coupling and cohesion providing the more useful information in the object oriented design in compare to the information given by the simple static measures. Then only we will decide that it is beneficial to continue our research in the field of run time cohesion and coupling metrics or further more to find out their relationship with the external quantity [13].

III. OBJECT ORIENTED METRICS

In the development of software today's, the object-oriented development is becoming very common. There is a different methodology to design the object-oriented development and also requires a changed method to software metrics. Since the objects are used in object oriented technology and the fundamental building blocks are algorithms, object oriented program are using different approach for software metrics from the standard metrics set. For traditional functional/ procedural programs, lines of code and cyclomatic complexity are used as the metrics and become accepted as standard and were used to form an idea of object oriented environments at the opening of the object-oriented design uprising. However, procedural languages using outmoded approach are not suitable for assessing object oriented software, mostly because the basic elements like polymorphism inheritance, classes and object are not measure to design. The analyze of the object-oriented software are only used to internment a small part of such software and thus provide a fragile quality suggestion[8] [9]. In the literature many object oriented metrics are proposed. Now the question arises is that which type of proposed metrics is used in our project. With the comparison of other software, the object oriented software is more complex concept in which there is not a single software quality is measured. The developer must define the characteristics of software to improve the software quality and then decide how the quality of the software is to be measure. For others who can understand your view point you can make the quality measured in a very easy way. Coupling and cohesion is used to calculate the seminal methods of object oriented design.

IV. PROGRAMMING IN SOA

SOA is much different from point-to-point architectures. SOA comprise loosely coupled, highly interoperable application services. These services can be developed in different development technologies (such as Java, .NET, C++, PERL, PHP), the software components become very reusable i.e. the same C# (C Sharp) service may be used by a Java application and / or any other programming language. WSDL defines an standard, which encapsulates / hides the vendor / language specific implementation from the calling client / service.

Development of Java in SOA

Most developers often think web services and SOA are synonymous. Many also think it's not possible to build service-oriented applications without using web services. To clarify, SOA is a design principle, whereas web services is an implementation technology. You *can* build a service-oriented application without using web services--for example, by using other traditional technologies such as Java RMI. The main theme behind SOA is to find the appropriate modularity and achieve loose coupling between modules. You can build an application where the modules don't have too much tight coupling between interacting components, such as an application where the JSP presentation layer is not tightly integrated with the data model and accesses it appropriately via an EJB. It's worth mentioning that Jini had long established the concept of SOA prior to the rise in popularity of web services. What web services bring to the table are the platform-independent standards such as HTTP, XML, SOAP, and UDDI, thus allowing interoperability between heterogeneous technologies such as J2EE and .NET. In this article, we'll focus on web services as the enabling technology for building service-oriented applications.

V. LITERATURE REVIEW

Ashutosh Mishra and Vinayak Srivastava[1] described the software maintenance, cohesion plays very major role to determine the relationship among different software attributes such as class, method, function-type etc. There are many method have been used in this context such as method based on syntactically keyword count in source code. We have used the semantic value computation for the specific keyword occurs in distinct common method within the different classes for an open source code project. We have also computed the conceptual relation metric to analysis the cohesion for the method within their class. Also, there is comparison between different semantic values for the keyword of common method in this context.

Rupinder.S and Hardeep.S[2] implement the research in service-oriented architecture (SOA) in general is well-past the early phase, the research in the area of architectural or quality metrics for SOA is still in the early stages. A number of formal models for SOA have been proposed in the literature, and many metrics have been derived from them. This paper serves to project that such a recurrence of formal-model-proposals is not of such a magnitude as to be characterized as multiplicity or excess, and that a certain degree of proliferation is a welcome sign, and will only aid an increased understanding of SOA, rather than impede the same in any way. To support this entire view, this paper discusses the important existing formal models, associated metrics, and demonstrates that these models themselves are a fertile ground to create newer metrics. Thereby, it presents many new metrics, and discusses future research possibilities.

Varsha Mishra[3] enhanced the measurement is fundamental to any engineering discipline. Cohesion metrics play an important role in empirical software engineering research as well as in industrial measurement programs. The Cohesion metrics presented in this paper measure the difference between class inheritance and interface programming.. This paper presents a measurement to measure cohesion by Lack of Cohesion in Methods (LCOM1), LCOM2 in object oriented programming. A measurement is done for C# inheritance and interface programs. The metric values of class inheritance and interface prove which program is good to use and beneficial for C# developers.

Varun Gupta[4] described the Most of the object-oriented cohesion metrics proposed in the literature define static cohesion at class level. Measurement of object-level dynamic cohesion however gives better insight into the behavioural aspects of the system. In this paper, dynamic cohesion metrics are introduced which provide scope of cohesion measurement up to object level and take into account important and widely used object-oriented features such as inheritance, polymorphism and dynamic binding during measurement. A theoret-ical framework is introduced before defining the measures and a theoretic validation of the proposed measures is carried out to make them more meaningful. A dynamic analyser tool is developed using aspect-oriented programming (AOP) to perform dynamic analysis of Java applications for the purpose of collecting run-time data for computation of the proposed dynamic cohesion measures. Further, an experiment is carried out for the proposed dynamic cohesion metrics using 20 Java programs and this study shows that the proposed dynamic cohesion metrics are more accurate and useful in comparison to the existing cohesion metrics. Moreover, the proposed dynamic cohesion metrics are validated empir-ically using source code APIs of Java Development Kit (JDK) and the proposed metrics are found to be better indicators of change-proneness of classes than the existing cohesion metrics.

Himanshu Dua and Puneet Jai.K[5] Classes and Aspects are basic unit in AOP. These qualities influenced the readability, maintainability and also affect the degree of relatedness among classes and aspect members. High cohesion is desirable characteristic. Existing techniques for measuring cohesion do not demonstrates actual cohesion of all the AOP cohesion metric, none of them are dynamic in nature. In this paper, a new dynamic cohesion metrics which is extension of most widely used static metrics in OOP i.e. LCOM is presented. A dynamic evaluation tool is developed using Aspect J(JAVA) and Python to gather runtime data to estimate the dynamic cohesion. This new discussed cohesion metrics gives more appropriate vision to the behavioral aspects of the system.

VI. PROPOSED SYSTEM

Cohesion metrics are very helpful in the improvement of quality of object oriented application. The idea of this type of measurement is produced from the existing measures Cohesion is considered to be the most important attributes. Cohesion is the attributes which measure the degree or the strength of interaction and relationships among elements of the source code, for example classes, methods, and attributes in object-oriented (OO) software systems. One of the main objectives behind Object Oriented analysis and design is to implement a software system where classes have high cohesion amongst them. The Objectives are based on very significant factors of complexity measurement of software which in cohesion.

- To study the existing cohesion metrics.

- To study the importance of java programming on object orientated metrics.
- To enhance the object oriented metrics for java programming.

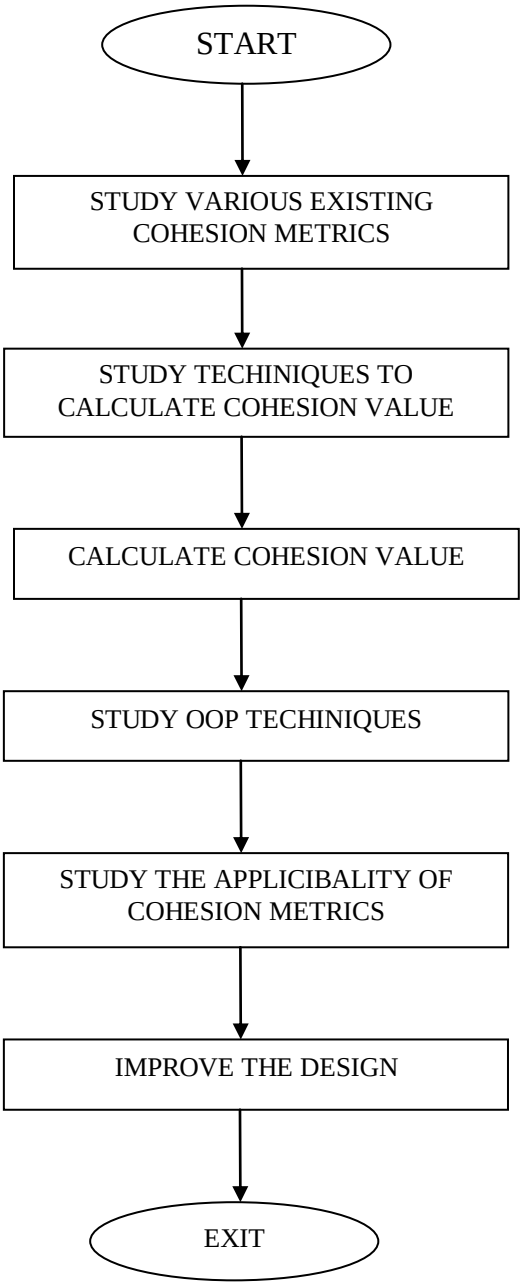

Fig 1: Flow diagram of proposed System

VII. RESULTS

In this section we will discuss the implementation of proposed technique. There are five types of metrics on which we are working. These metrics are Maintainability Index, Cyclomatic Complexity, Depth of Inheritance, class coupling, Lines of code.

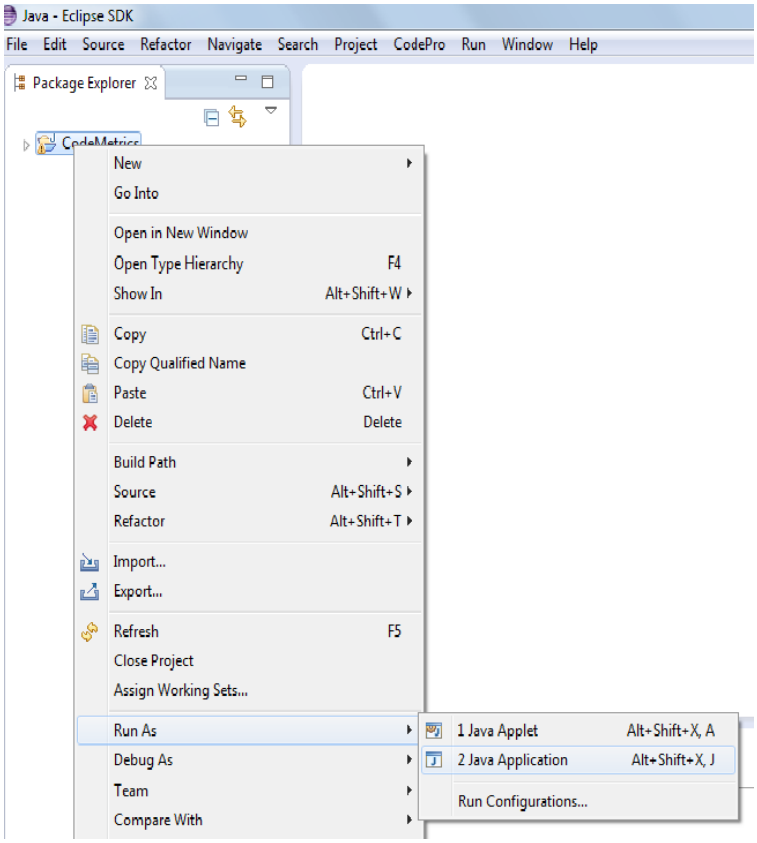

Fig 2 java eclipse launcher

This is the user interface; this interface is designed for the purpose of calculating the lines of codes in the project. In this interface, there are one option given where we can given the path of any project and then we apply go button and then the system calculate the number of lines of code present in the project.

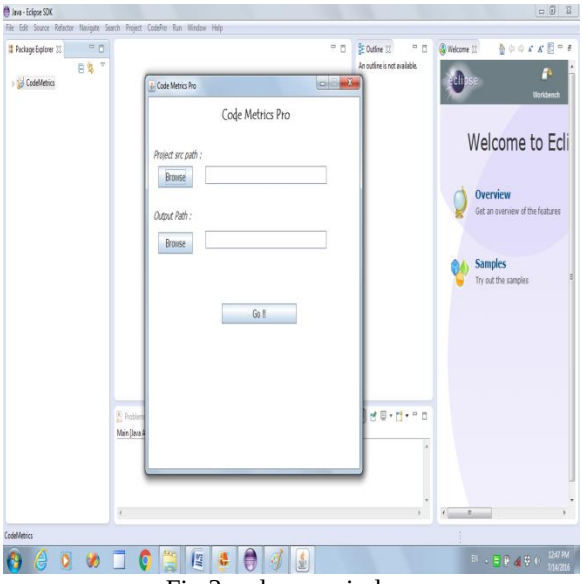

Fig 3 code pro window

When we run code pro metric file as a java launcher than code pro window open. In this window we entered the source path as a input file and out put path for saving final result as a xml documents in folder. Another popup is displayed , this time for selecting the Eclipse launcher jar. The launcher jar is located inside the eclipse installation directory as Eclipse\plugins\org.eclipse.equinox.launcher<some version number>.jar as given below

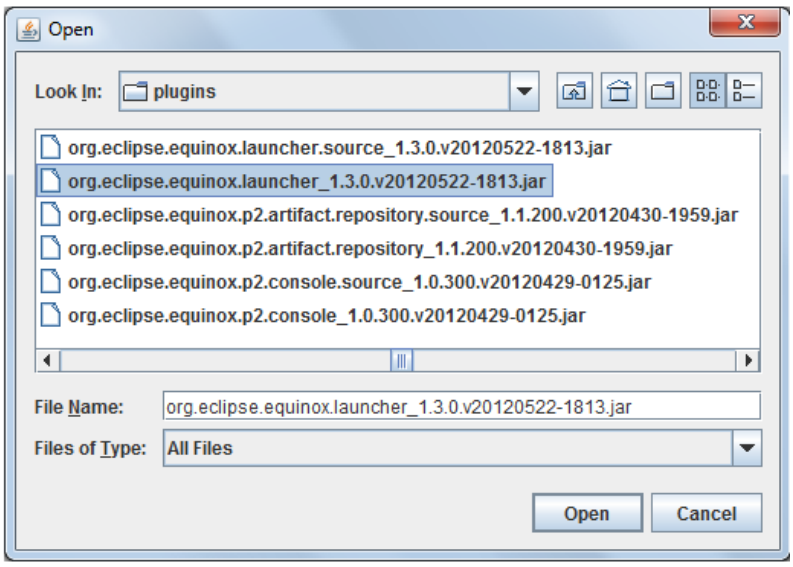

Fig 4: launcher window

After launcher , the final processing start. Than the final result is displayed as give below

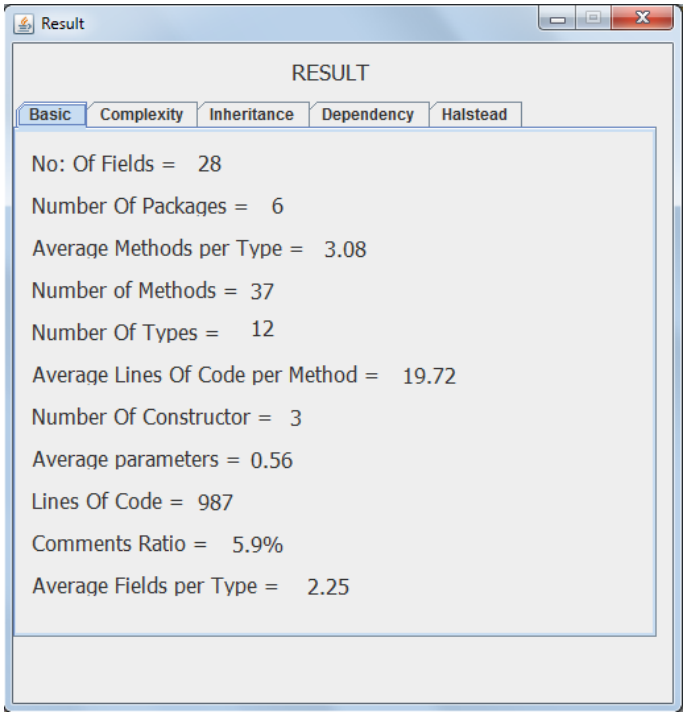

Fig 5 Final result window

Here the results of different metric are generated. Here the system generates the two types of results. The first is system generates the metrics results of complete project that is

No of field =28

No of packages= 6

Average Methods per type= 3.08

Number of Methods= 37

Number of types= 12

Average Line of Code per method= 19.72

Number of constructor = 3

Average parameters= 0.56

Line of code = 987

Comments Ratio= 5.9%

Average Fields per type = 2.25

VIII. CONCLUSION

In this dissertation work, we have used java eclipse software to find out the value of cohesion metrics value. It is reasonable to say that most common techniques and methods that are applicable to software architecture in general are applicable to SOA. Namely, quality models, scenario-based methods, formal models, metrics, quality computation models etc. This dissertation reflects that current research in metrics is at early stage, and detailed formal models that form the base of metrics development may need to be fine-tuned. Further, it demonstrates that even the existing models offer ample opportunities to formulate newer metrics. In this thesis we have built UI interface which allow the user to import java code packages and the metrics runs on imported java code packages. We get final results in the form of number of parameters, method lines of code, nested block depth and number of static attributes.

References

[1] Ashutosh Mishra,vinayak srivastva (2012) "*Conceptual and Semantic Measures for Cohesion in Software Maintenance* "International Journal of Computer Applications (0975 – 8887)Volume 47– No.22, June 2012

[2] Rupinder.S and Hardeep.S(2010),*"On Formal Models and Deriving Metrics for Service-Oriented Architecture",* JOURNAL OF SOFTWARE, VOL. 5, NO. 8, AUGUST 2010

[3] Varsha Mishra(2013), *"Better Object Oriented Paradigm Inheritance and Interface through Cohesion Metrics"*, International Journal of Computer Applications (0975 – 8887) *Volume 66– No.21, March 2013*

[4] Varun Gupta(2010), *"Dynamic cohesion measures for object-oriented software",* Journal of Systems Architecture 57 (2011) 452–462

[5] Himanshu Dua and Puneet Jai.K(2014), *"Dynamic cohesion metrics for aspect oriented programming"* , International Journal of Emerging Technologies in Computational and Applied Sciences (IJETCAS).

[6] Girish K. K.(2014), *"Conceptual Cohesion of Classes(C3) Metrics*", International Journal of Science and Research (IJSR) ,ISSN (Online): 2319-7064, Volume 3 Issue 4, April 2014

[7] Aine Mitchell and James F. Power(2011) , "*Run-Time Cohesion Metrics: An Empirical Investigation"* 16th International Conference on Automated on Software Engineering *(ASE '01)*

[8] Mohmmad Daghaghzadeh(2011), " *A Metric for Measuring Degree of Service Cohesion in Service Oriented Designs"* IJCSI International Journal of Computer Science Issues, Vol. 8, Issue 5, No 2, September 2011

[9] Sandip Mal and Kumar Rajnish(2014) , " *New Class Cohesion Metric : An Empirical View* " International Journal of Multimedia and Ubiquitous Engineering Vol.9, No.6 (2014), pp.367-376

[10] Taranjeet.K and Rupinder.K(2013) *"Comparison of Various Lacks of Cohesion Metrics",* International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249 – 8958, Volume-2, Issue-3, February 2013

[11] L.L Constantine and E. Yourdon. "*Structured Design" .Prentice-Hall, Englewood*Cli s, New Jersey USA, 1979.

[12] M .Page - Jones."The Practical Guide to Structured Systems Design". *Yourdon Press,* New York, NY, 1980.

[13] D .A .Troy and S .H. Zweben." Measuring the quality of structured designs".*The Journal of Systems and Software*, 2:112{120, 1981.

