



Predicting Complexity and Reusability in the Component Based Software Systems

Sonam Rani¹, Chander Diwaker²
M.Tech. Scholar¹, Asst. Professor²
Computer Science and Engineering Department,
University Institute of Engineering and Technology,
Kurukshetra University, Kurukshetra, Haryana, INDIA

Abstract: From the last few decades, the idea to develop software has become very much popular with the concept of integration of reusable components. A component is an independent replaceable part of a system performing a specific functionality and component based software engineering focuses on the integration of reusable components for designing a complex system with less effort and cost. The paper has been divided into two parts. The first part measures the complexity of a component based system with the help of two complexity metrics. The second part determines the reusability based upon the specified parameters and methods.

Keywords: Component based software system, Complexity, Reusability, Complexity Metrics, and Reusability Metrics.

I. Introduction

Component based software development is the way of integrating different software components in an application such that these components can provide a specific functionality. Various metrics have been proposed by researchers considering various attributes like complexity, customizability, reusability etc. The main focus this paper is on the complexity and reusability attributes. Two metrics are used to measure complexity: Component own complexity metric and Component interface complexity metric. As shown in fig.1, a directed weighted graph has been used for getting the overall complexity of a component based system. Use of instance variables and methods also has been made in the graph for representing the relationship between different components [1].

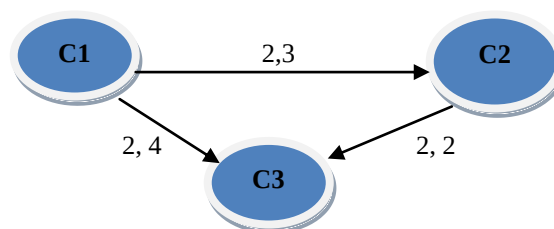


Fig. 1 Weighted Directed Graph

The first value is representing instance variables and second value representing methods. Moreover reusability has been found on the basis of various getters, setter methods and specified parameters. Matrix can also be used for specifying the relationship among components.

II. Literature Review

Gill [2] discussed the component reusability, cost and time saving by component reusability in a component based development. Author has made clear distinction on the reusability issues from both Technical and Management side. Cost-Benefit analysis for reuse purpose also has been made. Washizaki *et al.* [3] proposed a Suite of metrics for predicting the reusability of software components mainly for black box component where only restricted information is available about component. Metrics have been defined for observing the understandability of a component, adaptability and portability of a component. Confidence intervals also have been specified for getting more accurate and better result. But the tool developed accepts only JavaBeans components not works for such as ActiveX components. Yacoub *et al.* [4] presented a reliability model and reliability analysis technique for component based software systems. The technique is named scenario-based reliability analysis (SBRA). Using scenario for interaction of components a component dependency graph (CDG) is generated. Based on the graph, a reliability analysis algorithm has been generated to analyze the reliability. Gao *et al.* [5] described a systematic way to support the measurement for component based SOA

software at the system level which is the enhancement of component level performance measurement. The validation of system performance and measurement of non functional requirements in a component based SOA application becomes a challenge for software engineer. The paper provided a systematic solution using an event based functional transition for performance evaluation and its evaluation metrics such as throughput, reliability, availability, system processing time. *Singh et al.* [6] surveyed the concepts related to software component and also discussed the reusability matrices and models. Reusability measurement can be made directly or indirectly based upon black box and white box Components. In this paper, model has been used for describing relationship among metrics where metrics works as a quantitative indicator. *Sharp and Ryan et al.* [7] constructed a theoretical framework of the component based software development (CBSD) phases to differentiate between component and system development in the CBSD approach. The corresponding activities for each development phase: intelligence, design, choice and implementation phase for both component and system development has been presented. *Kumari and Upadhyaya* [8] defined the complexity attribute as an important attribute as it also affects other attributes like testability, reliability, and reusability etc. Authors described the interface complexity metric with the help of graphical notations. Interface complexity Metric is highly dependent on interaction complexity concluding that complexity increases with the increase of interaction among components as resulting in higher coupling. *Trivedi and Kumar* [9] presented a set of software matrix to find out the dependency between software component and application. Authors tried to answer the question that how much a component is beneficial for us in the terms of reusability. *Chillar et al.* [1] described the two metrics for measuring the complexity of interface and interface dependency of a component base system. The use of weighted assignment technique has been made to determine the strength of proposed metrics. The computed result was that the dependency and complexity increases with the increase in number of methods and instance variables invoked during an interface in a component based system. *Mittal and Bhatia* [10] determined the functional coupling in a component based software system. The strength of functional dependency metric has been calculated which is further based upon operation coupling complexity index and instance coupling complexity index. The use of analytical hierarchy process has been made for assigning weights to the instance variables and methods.

III. Proposed Work

From previous researches, conclusion has been made that complexity and reusability are the important attributes in a component based software system. Most of research regarding these attribute is based upon theoretical and manual interpretation but not in the practical manner. An effort has been made regarding automatic calculation of result for complexity and reusability attributes in a component based software system. As shown in table1, variables and methods have been categorized into three classes: simple, medium and complex. Variables of primitive data type such as character, integer etc. will be of simple Type. Variables of array, string, list, vectors will be of medium type and remaining like class, structure, and pointer etc. will be of complex Type. In the same manner, methods will be of simple, medium, complex type depending upon data type of arguments and return type of methods. According to category value has been given.

Data Type- No. of variables and methods	Simple	Medium	Complex
0-2	0.02	0.04	0.06
3-5	0.04	0.06	0.08
6-8	0.06	0.08	0.10
9-11	0.08	0.10	0.12
12-100	0.10	0.12	0.14

Table 1 Weighted table for variable and methods according to no. and type

a) Data Flow Diagram (DFD) for proposed solution

DFD is the graphical representation of a system working in a specified manner. As shown in fig. 2, firstly calculate the complexity and reusability of individual component and finally calculate overall weighted average complexity and reusability.

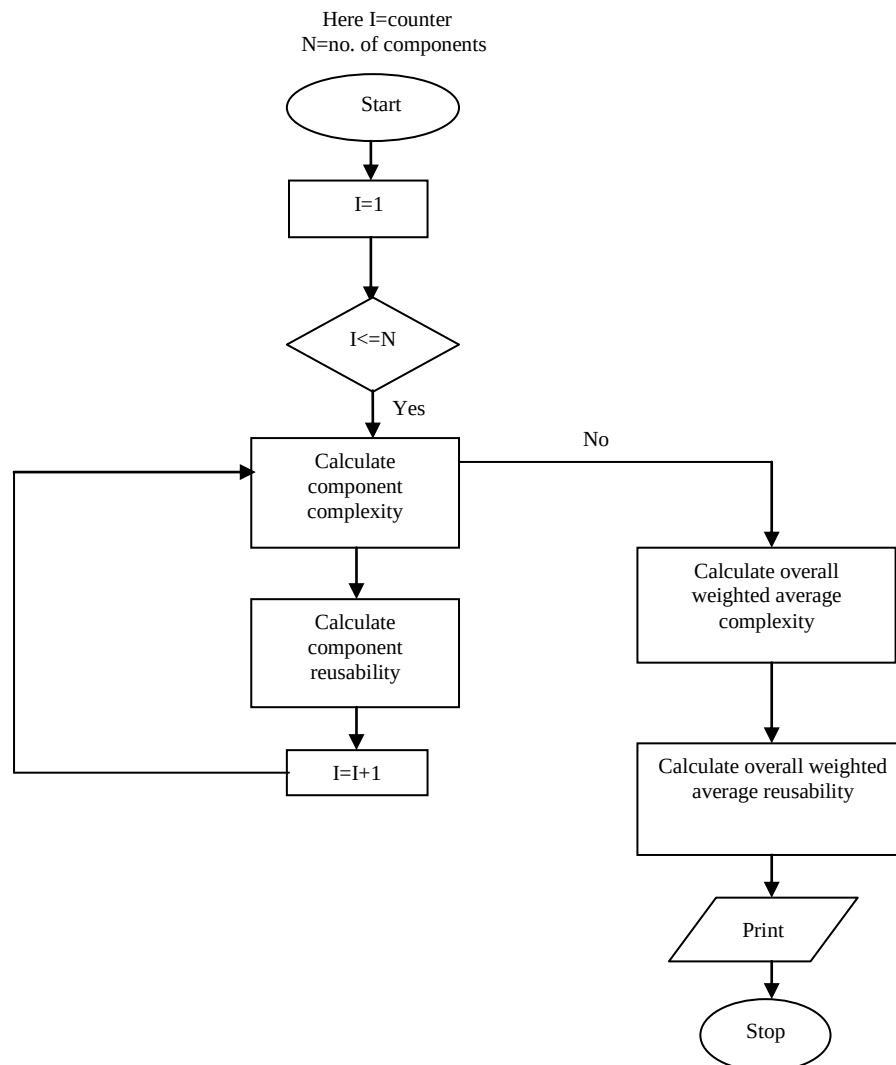


Fig.2 System Model Flowchart

b) Two important attributes calculated here are as

i) *Complexity attributes*

Component overall complexity = Component Own Complexity + Component Interface Complexity

Where component own complexity depends upon number of variables and methods and type of variable and methods. Corresponding weight can be seen from table 1 according to number and type. By multiplying the number and weight seen from the weighted table according to number of variables and methods, the required factor can be calculated. For calculating the component interface complexity, firstly find out all the interfaces of a component which is based upon graph and then calculate individual component own complexity and combine the individual complexity of all components.

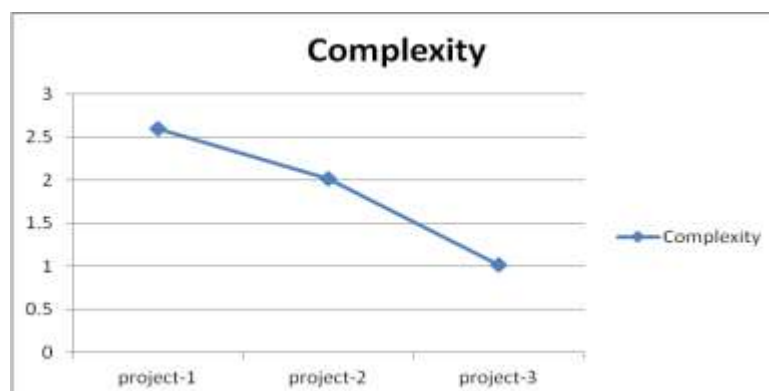


Fig. 3 Comparison of calculated complexity has been shown for three different projects.

ii) *Reusability attribute*:-Dependency of reusability has been made on the basis of 5 factors.

- i. *Existence of Info class*:- whether an Info class exists or not according to a particular component. Value is 1 if Info class exist otherwise 0. Info class is like the help class which provides extra information.
- ii. *Component Understandability*:- Calculated as number of getters/Total number of fields in a component. Understandability gives us the percentage of readable properties in a particular component.
- iii. *Component Configurability* - Calculated as number of setters/Total number of fields in a component. Configurability gives us the percentage of writeable properties in a particular component.
- iv. *External Dependency of Component based upon Return Value*:- Calculated as number of void methods/Total number of methods in a component. Resulting percentage of methods without any return value in a particular component.
- v. *External dependency of Component based upon Parameters*:- Calculated as number of Methods without parameters /Total number of Methods in a component. Resulting percentage of methods without any parameter in a particular component. Overall reusability can be calculated by combining the resulting value of above five factors [7].

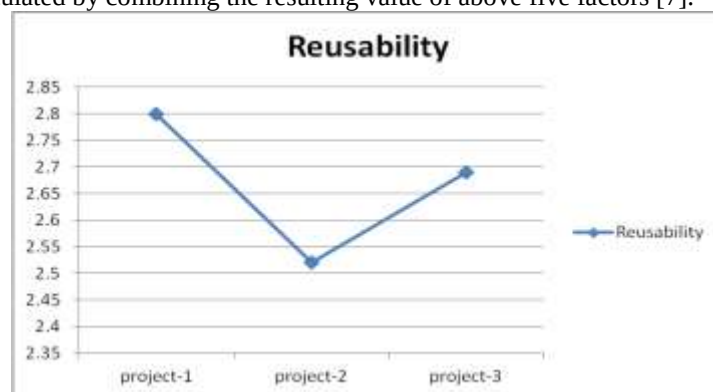


Fig. 4 Comparison of calculated reusability has been shown for three different projects.

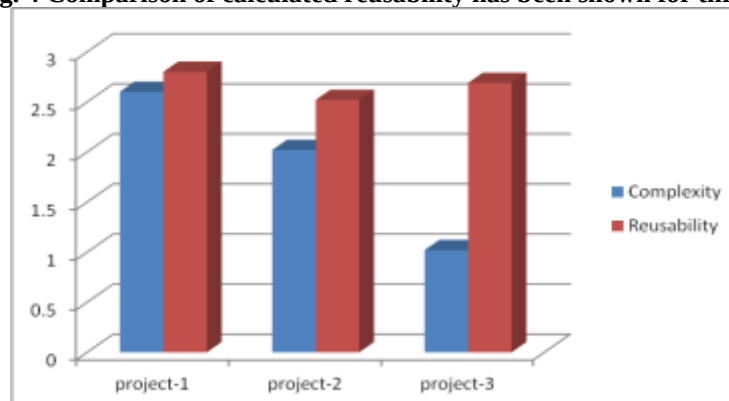


Fig. 5 Comparison of calculated complexity and reusability has been shown for three different projects.

IV. Conclusion

Complexity of a component has been calculated with the help of weighted graph. Reusability has been calculated on the basis of five factors. Complexity and Reusability of any component can be calculated by giving a component as an input. Also practical comparison for different projects has been shown for both complexity and reusability with the help of output graph. Further in future, other attributes like reliability, maintainability may be considered etc. in a component based software system.

V. References

- [1] Rajender Singh Chillar, Priyanka Ahlawat, and Usha Kumari ,“Measuring Complexity of Component Based System using Weighted Assignment Technique”, International Conference on Information Communication and Management, 2012, Vol. 55, pp. 19-27.
- [2] Nasib S.Gill, “Reusability Issues in Component-Based Development”, ACM SIGSOFT Software Engineering Notes, 2003, Vol. 28, Issue 4, pp. 1-5.

- [3] Hironori Washizaki, Hirokazu Yamamoto, and Yoshiaki Fukazawa, "A Metrics Suite for Measuring Reusability of Software Components", METRICS 03 Proceedings of 9th International Symposium on Software Metrics , IEEE Computer Society Washington, DC, USA, 2003, pp. 211-224.
- [4] Sherif Yacoub, Bojan Cokic, and Hany H. Ammar, "A Scenario-Based Reliability Analysis Approach for Component Based Software", IEEE Transactions on Reliability, 2004, Vol. 53, Issue 4, pp. 465-480.
- [5] Jerry Gao, Yumei Wu, Lee Chang, and Sigurd Meldal, "Measuring Component-Based Systems using a Systematic Approach and Environment", IEEE International Workshop on Service Oriented System Engineering , 2006, pp. 121-129.
- [6] Sarbjeet Singh, Manjit Thapa, Sukhvinder Singh, and Gurpreet Singh, "Software Engineering-Survey of Reusability based on Software Component", International Journal of Computer Applications, 2010, Vol. 8, No.12, pp.39-42.
- [7] Jason H. Sharp and Sherry D. Ryan, "A Theoretical Framework of Component- Based Software Development Phases", ACM - The Database for Advances in Information Systems, 2010, Vol. 41, No. 1, pp. 56-75.
- [8] Usha Kumari and Shuchita Upadhyaya, "An Interface Complexity Measure for Component-based Software Systems", International Journal of Computer Applications, 2011, Vol. 36, No.1, pp.46-52.
- [9] Prakriti Trivedi and Rajeev Kumar, "Software Metrics to Estimate Software Quality Using Software Component Reusability", International Journal of Computer Science Issues, 2012, Vol. 9, Issue 2, No 2, pp. 144-149.
- [10] Sonu Mittal and Pradeep Kumar Bhatia, "Predicting Qunatitative Functional Dependency Metric based upon the Interface Complexity Metric in Component based Software Systems: A New Approach", International Journal of Computer Applications, 2013, Vol. 73, No. 2, pp..21-28.