RESEARCH ARTICLE OPEN ACCESS

Analyzing the Factors Affecting Reusability for Black–Box Component

Suchita Yadav[1], Amit Yadav[2], Surabhi Yadav[3]
School of Information and Communication Technology [1 & 3],
Gautam Buddha University,
Greater Noida

Department of Computer science [2],
MERI College of Engineering and Technology,
Bahadurgarh, Jhajjar,
Haryana

ABSTRACT

Component-Based Software (CBS) has emerged as more generalized approach for application development. In Component-Based Software Engineering (CBSE), it is required to measure the reusability of components in order to improve the reuse of components effectively because reusability is an effective way to increment the productivity in CBSD; it is required to calculate the reusability of components. However, due to the component's black-box nature the source code of the components are not available for the users, so it is difficult to use conventional metrics of Object-Oriented metrics in Component-Based Development (CBD) as these metrics require source codes of the components. Now here this study examines various factors to measuring software reusability. Knowing that what makes software "reusable" can help us learn how to build new reusable components and help us to identify potentially useful modules in existing programs and proposing three factors model that affects the reusability of the component directly.

Keywords:- Reusability, Component-Based Software, Reusability, Complexity, Adaptability, Reuse.

I. INTRODUCTION

"A *software component* is a unit of composition with contractually specified interface and explicit context dependencies only. A software component can be deployed independently and is subjected to composition by third parties."[1]

- **Component-Based Reusability**

Reusability is the main feature of soft component which differs from other software. Reusability is used to improve the degree of one component that can be reused; it evaluates the reusability of component in order to help the developer to recognize the useful parts from the legacy system. The information of component reusability can also provide helpful choice for component user, and select the components that can be used in future software system.

Component reusability is an effective way to improve productivity. Its development differs from the traditional way of software development where it affects software measures, therefore a new way of software reusability metrics is needed, and moreover new model of component reusability is needed to be established [2].

Here in this study we are analyzing and proposing a reusability factors model using some factors that directly affects the reusability of the component. These factors are as follows adaptability, reuse and complexity of a component, as well as we have seen that there are no existing metrics or model that has considered these few factors collectively. Whereas adaptability concerns with the inter and intra interfaces of components affecting the reusability of component, the reuse includes about the actual reuse of the component that can also be used to infer how usable and how easy to adapt the component. And the complexity discusses about the degree of cohesiveness and coupling between the components that bounds the component to be reused.

II. RELATED WORK

Component-based software engineering (CBSE) is a branch of software engineering that emphasizes the separation of concerns in respect of the wide-ranging functionality available throughout a given software system. It is a reuse-

based approach to defining, implementing and composing loosely coupled independent components into one single system. This practice aims to bring about an equally wide-ranging degree of benefits in both the short-term and the long-term for the software itself and for organizations that sponsor such software [2].

Gill [3] discusses the importance of component characterization for better reusability. Component can be characterized on the basis of its informal description, external and internal specifications. Informal description consists of age, source, level of reuse, context, intent etc. External category defines its interactions with other application artifacts with the platform on which the component resides, which may consist of interoperability, portability, technology and other characteristics. Internal aspects of a component may include the nature of component like function, data package etc. Paper discusses several benefits of component characterization, which includes improved cataloguing, improved usage, improved retrieval and improved understanding eventually for better reuse.

Reusability can also be measured indirectly. Complexity, adaptability and observability can be considered as a good measure of reusability indirectly. Rotaru [7] et. al. measure the reusability by proposing some metrics for adaptability and interface complexity of the component. Adaptability refers to the accommodation of the changes required in its environment. Cho et. al. [8] proposes a set of metrics for measuring various aspects of software components like complexity, customizability and reusability. The work considers two approaches to measure the reusability of a component. The first is a metric that measures how a component has reusability and may be used at design phase in a component development process. This metric Component Reusability (CR) is calculated by dividing sum of interface methods providing commonality functions in a domain to the sum of total interface methods. The second approach is a metric called Component Reusability level (CRL) to measure particular component's reuse level per application in a component based software development.

Hironori Washizaki et.al. [6] Proposes the importance of reusability of components in order to predict the reuse of components efficiently and propose a Component Reusability Model for black-box components from the viewpoint of component users. The model defines a set of metrics to define quality factors that affect reusability.

According to *Boxall et.al.*[10] the understandability of a software component's interface is a major quality factor for determining reusability. To measure this, they have defined a set of metrics, including values such as Interface Size, identifier Length or Argument Count. They have selected 12 components from different software systems in C and C++ to empirically validate their metrics and developed simple tools to automatically calculate them. The derived values have been compared against the expert knowledge of judging the reusability of these components.

According to Danial Hristov et.al. [14] The basic measurement factors are given in his study that can directly affect the reusability of component and a metrics is defined on the basis of all those factors compositely.

According to *Seungwon Lee et.al.*[9] they measure the reusability of software components, pieces of reusable software, in a system called component grid. They discuss about the three different cases of component reuse and a new method to measure component reusability in component grid. The method that will be used to the new measure is as follows.

- Black box: Reusability is in proportion to component reuse frequency.
- Glass box: Reusability is in proportion to degree of component understanding and reuse frequency.
- White box: Reusability is in proportion to the weighted sum of degree of understanding and rate of modification. It is also in proportion to component reuse frequency.

III. PROPOSED MODEL

There are so many reusability models that are proposed by many authors in their respective papers that we have already discussed in our study and review chapter. Here we are giving a reusability model on the basis of the three main factors affecting the reusability of components.

The three factors that are given below:

- **Adaptability**

The adaptability define the availability of a software component that can determine how easy and fast it is to retrieve it, this is not to be confused with the operational availability used in the context of long-running systems.

In aspect of reusability one important aspect of the adaptability is the programming language and the other one is the availability of appropriate methods and interfaces for adapting the component [14]. The first aspect is context-dependent, subjective and qualitative.

- **Complexity**

The complexity of a software component determines how usable it is, and how easy it is to adapt the software component in the new context of use. The rationale behind this is that if there are two components which provide the same functionality (which is the prerequisite for assessing their reusability),[14] then a lower component complexity would mean that functionality is

implemented more efficiently and can be reused in another aspect. Thus, it is likely that the implementation of this functionality in the component is of higher quality, is easier to understand by the developer and easier to maintain in the future, and it will be easier to adapt in a new context.

There are so many sub factors of complexity that can useful in finding the reusability and here we are going to access some of them for assessing the complexity of software components. The complexity intended here is a composite metric of the size of the component (e.g., in Lines of Code (LOC), excluding the LOC containing only documentation, i.e. comments) and complexity metrics for the classes, methods and parameters of the component, as well as their coupling and cohesion. It should be noted that the application of these metrics will be different for white-box and black-box components.

- **Reuse**

The actual reuse of the component can also be used to discuss how usable and how easy it is to adapt that component. The amount and frequency of reuse, especially in contexts similar to that of the developer can serve as reference points and she or he may select the component with the higher amount and frequency of reuse. It can determine by the amount and frequency of reuse. The amount is the overall number of reuses of the component.

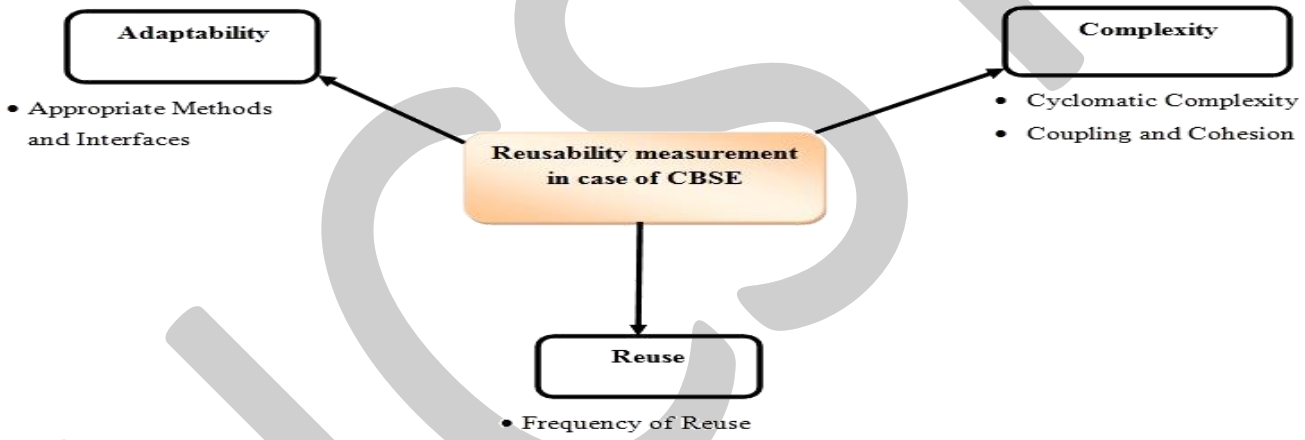

Figure: Factors Affecting Reusability

IV. CONCLUSION

In this paper as we have surveyed the available metrics for the reusability assessment of the code in object-oriented and component–based development. Here we have considered three main factors into consideration as the adaptability of component to the specific requirements of the new system, complexity of a component determines how usable it is and how easy to adapt the software component in a new context and the actual reuse of the component can also be used to infer how usable and easy to adapt it. And by analyzing these factors we can give effective reuse for the component in Component-Based Software.

REFERENCES

[1]. Jean-Guy Schneider: "Component Scripts and Glue: A Conceptual Framework For Software Composition" *Ph.D. thesis, Institute für Informatics (IAM), University Bern, Berne, Switzerland, 2003.*

[2]. Miguel Goulão: "CBSE: a Quantitative Approach" *PhD Workshop at ECOOP'2003, Darmstadt, Germany. July, 2003.*

[3]. Nasib S. Gill and P. S. Grover: "Necessary Guidelines for deriving Component Based Metrics". In ACM SIGPLAN SEN Vol 28, #6 Page: 30, 2003.

[4]. Arun Sharma, Rajesh Kumar and P S Grover, "Few Useful Considerations for Maintaining Software Components and Component-Based Systems", *ACM SIGSOFT Software Engineering Notes, Vol. 32, Issue 4, 2007.*

[5]. J. Poulin: "Measuring software reusability*",Proceedings of the 3rd International Conference on Software Reuse: Advances in Software Reusability, pp. 126–138, IEEE Computer Society Press, Los Alamitors, 1994.*

[6]. Hironori Washizaki, Hirokazu Yamamoto and Yoshiaki Fukazawa. "A Metrics Suite for Measuring Reusability of Software Components"*,Proceedings of the 9th International Symposium on Software Metrics September,IEEE Computer Society, Washington, DC, USA, 2003, pp.211-223, 2003.*

[7]. Rotaru O P andDobre M, "Reusability Metrics for Software Components", *In Proceedings of the ACS/IEEE 2005 International Conference on Computer Systems and Applications, pp: 24-I, 2005.*

[8]. EunSook Cho, Min Sun fim and Soo Dong Kim, "Component Metrics to Measure Component Quality", *Proceedings of the eighths Asia-Pacific Software Engineering Conference, 1530-1362/01,2005.*

[9]. Seungwon Lee and Ho-Jin Choi "Software Component Reusability Measure in Component" *Feb 15-18, 2009 the International Conference on Advanced Communication and Technology(ICACT) 2009.*

[10]. M.A.S. Boxall and S. Araban, "Interface Metrics for Reusability Analysis of Components", *Australian Software Engineering Conference (ACWEC'04), Melbourne, Australia, pp.40-50, 2004.*

[11]. A. Sharma, P.S. Grover, and R. Kumar, "Reusability assessment for software components", *SIGSOFT Software Engineering Notes, vol.34, No.2, , pp.1-6, February 2009.*

[12]. Arun Sharma, Rajesh Kumar, and P. S. Grover, "A Critical Survey of Reusability Aspects for Component-Based Systems", *World Academy of Science, Engineering and Technology Vol: 9,2007.*

[13]. Eun Sook Cho, Min Sun Kim and Soo Dong Kim, "Component metrics to measure component quality", *Eighth Asia-Pacific Software Engineering Conference, (APSEC), 2001.*

[14]. Danail Hristov, Oliver Hummel, Mahmudul Huq, Werner Janjic, "Structuring Software reusability Metrics for Component-Based Software Development", *ICSEA 2012 : The Seventh International Conference on Software Engineering Advances,2012.*

[15]. Li Yingmei, Shao Jingbo, Xia Weining, "On Reusability Metric Model for Software Component", *Software Engineering and Knowledge Engineering: Theory and Practice Advances in Intelligent and Soft Computing Volume 114, pp 865-870,2012.*

[16]. Gui Gui, "Component Reusability and Cohesion Measures in Object-Oriented Systems", *Information and Communication Technologies, 2006. ICTTA '06. 2nd (Volume:2), pp. 2878 – 2882, 2006.*