New Method to Analyze the Impacts on Reliability and Reusability in Component-Based Software Development

Deepak Panwar*
Pradeep Tomar*
Nasib S. Gill**
Arvind Kumar*

Abstract

Component-Based Software Engineering (CBSE) is an approach which is used to enhance the quality and productivity to deliver software in the market with less development time and cost. CBSE enhance the reusability with the development of component-based software from the pre-existing software components which is already tested. But when reuse a pre-existing software component, reusability and reliability play important role before the testing phase of development life cycle. Analyze of both the reusability and reliability factors for Component-Based Software Development (CBSD) is very active and challenging field for researchers and practitioners. This paper discusses the software quality models viz. McCall Quality Model and ISO-9126. On the basis of these models and Halsted Software Science, this paper presents a new method which can be used to measure and detect the faults before testing phase by the following factors likely software requirement change, design change, code change, component complexity, software complexity. This paper also summarizes the impacts on reliability and reusability due to the changes in the software requirement, design, code, component complexity, software complexity.

Keywords: CBSD, Quality Model, Software Reliability, Reusability.

1. Introduction

Component-Based Software Engineering (CBSE) is a technology used to develop a complex software system using reusable software component according to user requirements which aims to satisfy the user. CBSE is based on reusable software component that can be replaced and updated easily without affecting the software equality. So CBSE play an important role to enhance the quality through reusability to build component-based software from the pre-fabricated reusable and testable component. Quality means the degree of excellence thus, the mean of software quality is "how much the functions of that particular software are according to the requirements of the customers" or in short software quality is Conformation to Requirements [1]. Software quality can be discussed, felt, and judged, but can not be weighed or measured. Software crisis is very big problem to maintain the software quality and to handle this problem CBSE which is very helpful to improve the productivity. To maintain the quality it is necessary to measure the attributes of the software to make it of reliable and reusable. In CBSE, to make reliable and reusable software it is very necessary to check the quality at every phase before testing to detect the faults at early stage of software development lifecycle.

*School of Information and Communication Technology, Gautam Buddha University, Greater Noida, Uttar Pradesh, INDIA
**Department of Computer Science & Applications Maharshi Dayanand University, Rohtak, Haryana, INDIA

2. Software Quality Models

Software quality engineering uses the software quality models to make the quality software according to customer requirements. These models are based on different attributes like reliability, reusability, portability, functionality etc; McCall Quality Model and ISO-9126 models have different sets of attributes as shown in figure 1 and 2 [2], [3].

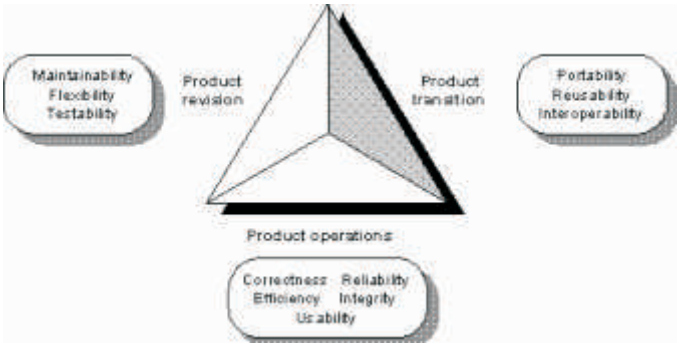

Figure 1: McCall Model

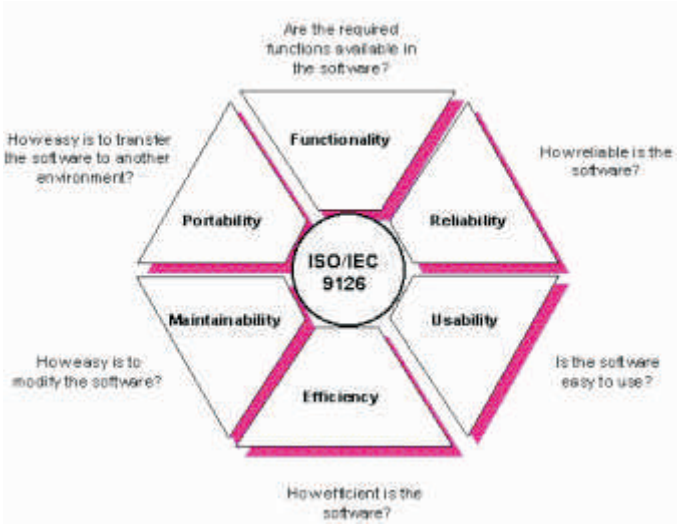

Figure 2: ISO 9126

Quality component based software can be produced by predicting the reliability and reusability of the components. Model ISO-9126 which contributes six qualities attributes: functionality, portability, efficiency, reliability, maintainability and usability. But as researchers and practitioners know that the reusability of code or the reusability of component is basic principle for component-based software development. So the main focus of this paper is to measure the general factors of reliability and reusability. The general factors for reliability and reusability are software requirement change, design change, code change, component complexity, software complexity. Metrics and models are helpful for measuring the reliability and reusability with more general factors viz. coupling, cohesion, software complexity, component complexity [5], [6]. This paper uses Halstead's Software Science to predict the reliability and reusability for component based software. A computer program is a collection of tokens according to software science and token can be classified as either operators or operands. In 1972 Halstead Model gives an idea of software science [4], [6], [7]. Primitives Halsted model are shown below in equation 1 and 2.

$$\text{Volume}(V) = N \log_2 (n_1 + n_2) \quad \ldots\ldots 1$$

$$\text{Fault } B = V/S0 \quad \ldots\ldots 2$$

Here n1 is no of distinct operators in a program, n2 is no of distinct operands in a program, N1 is no of operators occurrences and N2 is no of operands occurrences. B is the number of faults in program and V is the volume in terms of operators and operands. S0 is the mean number of mental discriminations (decisions), S0 = 3000 according to Halsted Software Science.

3. Proposed Method

This paper focus on main factors like reliability and reusability to make quality software, because of reliability and reusability factors analyzed before testing phase then it would be beneficial for both developer and customer because developer wants to give the delivery of software with low bugs & high quality and customer wants software with good quality and low cost. If these factors can predict in early phases of a software development then it would be possible to decrease the percentage of software crisis. By using this new proposed method, this paper summarizes the affects on software reliability and reusability due to the changes in the software requirement, design, code, component complexity, software complexity. This paper use equation 2 as base for new proposed method and apply this method before the testing phase of our development process because these equation works before the testing according to Halsted Software Science. According to the Table 1 and 2 general factors which affect the reliability and reusability in increasing way will be taken as nominator and the factors which have adverse effect will be taken as denominator according to the [6] and after that this study conclude and calculate the following two variable first one is reliability (SRE) and second one is reusability (SREU) with the following equation. Here Software Reliability is denoted by SRE and Software Reusability is denoted by SREU

Table 1: General Factors to Analyze Reusability during CBSD

Sr. No.	Factors for Analyzing Reusability of Software	Impacts on Reusability	
1	Requirements	Change	▲
		No Change	▼
2	Design	Change	▲
		No Change	▼
3	Code	Change	▲
		No Change	▼
4	Component Complexity	Increase	▼
		Decrease	▲
5	Software Complexity	Increase	▼
		Decrease	▲

Table 2: General Factors to Analyze Reliability during CBSD

Sr. No.	Factors for Analyzing Reliability of Software	Impacts on Reliability	
1	Requirements	Change	▲
		No Change	▼
2	Design	Change	▲
		No Change	▼
3	Code	Change	▼
		No Change	Constant
4	Component Complexity	Increase	▼
		Decrease	▲
5	Software Complexity	Increase	▼
		Decrease	▲

SRE = SRC * SDC / SCC *SCC * SC-RE…….. 3

SREU = SRC * SDC *CCC / *CC * SC-REU..... 4

SRC is Requirement Change in software, SDC is Design Change in software, SCC is Code Change in Software, CC is Component Complexity, SC-RE is software complexity for analysis of reliability factor according to integration dependency, SC-REU is software complexity for analysis of reusability factor at different levels, CCC is Code Change in Component. To calculate the above variables for equation 3 and 4 following methods and metrics are used according to [9], [10].

SRC is Requirement Change in software
Variable SRC counts the changes in requirements.

SDC is Design Change in software
Variable SDC counts the changes in different level of design.

SCC is Code Change in Software
Cyclometic Complexity = E-N+2P
E = Number of edges of the graph
N = Number of nodes of the graph
P = Number of connected components
SCC is Software Components Complexity
IDC = I/ImaxIN
IDC = Interaction Density Metrics for Component Integration
I = Actual Interactions
ImaxIN = Maximum Available Interactions

SC-RE is Software Complexity for analysis of reliability factor according to integration dependency
SC-RE -AID = (IDC1+IDC2+……+IDCn) / Components
SC-RE -AID = Average Interaction Density
Components = Total Number of Available Components

CCC is Component Code Change
CCC = CC+NCOM
CC = Cyclometic Complexity
NCOM = Number of Calls to Other Methods

CC is Component Complexity
CC = NIMC+NOIC+NCOC
NIMC = Number of Internal Method Calls
NOIC = Number of Interfaces of a Component
NCOC = Number of Couplings to Other Components

SC-REU is Software Complexity for analysis of reusability factor at different levels
SC-REU = (NLBC+DOCT) + CCC + CC
NLBC = Number of Links between Components
DOCT = Depth of the Composition Tree

Put values of SRE and SREU in equation 2 and get a new equation 5, which help in calculating the no of faults before the testing phase. Reliability (SRE) and Reusability (SREU) factors are selected for analyzing the reliability and reusability of component-based software because both the factors are the main factors in component-based software development approach.

B = SRE * SREU * V / S0…….....…..... 5
Where V is no. of software code line and S0 is constant as discussed above, its value comes from project history data, and with the help of equation 5 we can measure the faults by analyzing the reliability and reusability to develop a good quality software.

4. Conclusion

Proposed method helps in reducing the cost and risk in CBSD process according to Halsted Software Science. According to the Halsted Software Science, it would be very beneficial because the testing phase consume more time in compression of all the phases and by applying this method researchers and practitioners can analyze the software reliability and reusability with general factors viz. software requirement, design, code, component complexity, software complexity. So this method will help to make a reliable component-based software based system and thus this method can be helpful in measuring the quality of component-based software.

5. References

1. *Pressman, R. S. (2006). "Software Engineering: A Practitioner's Approach", 7th Edition, McGraw Hill.*
2. *www.cse.dcu.ie/essiscope/sm2/charact.html.*
3. *www.cse.dcu.ie/essiscope/sm2/9126ref.html.*
4. *Lyu, M. R. (2007). "Software Reliability Engineering: A Roadmap", proceedings of Future of Software Engineering'07, IEEE Conference, pp. 153-170.*
5. *Kan, S. H. (2002). "Metrics and Models in Software Quality Engineering", 2nd Edition, Pearson Education.*
6. *Oliveira;, M. F. S.; Redin, R. M.; Carro, L.; da Cunha Lamb, L. and Wanger, F. R. (2008). "Software Quality Metrics and their Impact on Embedded Software", proceedings of Model-based Methodology for Pervasive and Embedded Software '08, IEEE Conference, pp. 68.*

7. *Parnas, D. L., (2002). "Software Engineering Programs are not Computer Science Programs". Software IEEE Journal, Vol.16, No.6, pp-19.*
8. *Daniel Galin, (2009). "Software Quality Assurance", 1st Edition, Pearson Education.*
9. *Narasimhan, V .L and Hendradjaya, B (2004) "A New Suite of Metrics for the Integration of Software Components", http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.88.681*
10. *Salman, N. (2005). "Metric and metric validation approaches for component orientation", www.pdffactory.com*