Volume 4, Issue 4, April 2014 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Software Reliability Estimation and Distributing Testing Effort in Component Based Software

Abhinav Choudhary*, **Dr. Pradeep Tomar**
ICT, Gautam Buddha University
India

Abstract— ***Present models and methodologies available for reliability estimation have mainly focused on the using only the component reliability for calculating the system reliability and neglecting other factors. All the available models have parameters and a number of assumptions. Also they are applicable to limited domains. Some models focused on collecting the past failure data for predicting the future reliability of software. Interfaces efficiency are critically important for efficient working of any software because interactions is based upon them. This study identifies three major factors component reliability, interface reliabilities and component attachment ratio for reliability estimation. The proposed mathematical model calculates the reliabilities of all executable profiles which will exist in the system and then overall system reliability. Finally a method of distributing the testing effort throughout the system.***

Keywords— ***Component Based Software Systems (CBSS), Software Reliability, Component Based Reliability, Interface Reliability, Component Attachment Ratio (CAR).***

I. INTRODUCTION

Component Based Software Engineering (CBSE) is widely applicable in the software industry today. CBSS contributes to a large part of the software industry i.e. a large portion of the software development is based upon the CBSE. CBSE provides the facility of reusability of existing parts and features into new system which is done with a single independent entity known as component. A software component is a software element that conforms to a component model and can be independently deployed and composed without modification according to a composition standard [1]. Component is a independently executable entity made up of executable objects and have interfaces through which it communicates with the other components, all its functionalities and dependencies are defined through these interfaces. Components provide a service without regard to where the component is executing or its programming language. This technology has provided the industry a break through by reducing the development time and reducing cost. CBSS provided various facilities in the design, testing, implementation and maintenance of the software system.

This technology has emerged from the failure of object-oriented technology to support effective reuse as the single object and classes are too detailed but still Component Based Development (CBD) uses object oriented technology to develop components. The basic idea behind the CBSE is reusability, easily managing and using components to build new functionality.

II. RESEARCH PROBLEM DEFINATION

With the growth of the industry, scale using of the components in software development has risen very high. Nowadays in the industry software developed are very complex in architecture and include a large number of functionalities. For building up of complex system it requires a large number of components which is hard to manage and use. This has arisen the question on reliability of the components *i.e.* how reliable a component is? ANSI defined the reliability as the probability of failure free operation of a system for a particular time in a particular environment [2]. Estimation of reliability of software system has been important research area in the past and a lot of work has been done over it but the area is still open for the research. Reliability provides a quantitative figure which tells us about the behavior of the system, it is useful for the development team as well as for the testing team [3]. This work is related to provide a quantitative value for reliability of the system based upon the reliability of the individual components, interfaces and CAR. This will give a better idea to the developer about the system and to make the necessary changes in the system and also distributing the testing effort by dissipating the reliabilities of different executing profiles. Consider a CBSS consists of N number of software components, such as C1 ,C2...Cn. and interfaces I1, I2,….In. The goal of this study is to estimate the reliability of a software depending on the component reliability and related factors.

i) In this study it is assumed that the reliability of various components and interfaces are known for calculation purpose.

ii) Finding the CAR.

iii) Estimating the overall system reliability through the reliability of constitutes components, interactions among the components and architecture of the system.

An easy way to comply with the conference paper formatting requirements is to use this document as a template and simply type your text into it.

III. RELATED WORK

The first attempt to estimate the reliability was done by *Jelinsky and Moranda* long back in 1972. Software reliability estimation has been a popular research area, many model and methods have been proposed in the past for estimation of reliability. Most the models are based on failure data which is extracted during testing or collected from data of past failures of similar software. Most of the models developed based upon architecture used past failure data [4], [5], [6], [7]. Also there comes many other models like *Curve Fitting Model*[8], *Time Series Models*[9] which were used for quite a long time for estimating the reliability of software. All these models were very useful but due to a number of assumptions involved they were applicable only to a limited number of domains. The architecture of software became complex at a fast pace which involves a lot of functionalities, so thereafter these models were not enough to support these type of architecture. But CBS comes with a more enhanced architecture based upon a reusable unit called component. Reliability estimation of CBSS have attracted a large number of researchers and many models have been proposed in this field. *Szyperski* [10] defines a component precisely by enumerating the characteristics of a component. A software component is a unit of composition with contractually specified interfaces and explicit context dependencies only. Considering the system as a single unit and estimating the reliability is not suitable in case of the CBS. It required a more firmed analysis of the architecture of the system for identifying the behavior of the components and there interaction within the system to more precisely estimate the reliability [11]. There are mainly three methods used to measure the software reliability *Profile Based*: It uses the information in the regard of system usage by the user which is represented through the operational profiles. *State Based*: This technique assume the transition processes *Markov* property for reliability estimation and *Path based*: It estimates the reliability of the software after implementation [1].

This study proposes the estimation of reliability by considering the reliabilities of the individual components, reliabilities of the interfaces and the CAR. All the three factors have a potential impact on the system functionality. Every component is a part of the system and its failure affects the whole system. An interface failure may lead to the failure of the component which again affects the whole system. CAR identifies that to how many components a component is attached, there may be situation where a component is attaches to large number of components and if it fails the whole system may crash down. Identifying the CAR gives a better idea to testing team to identify critical parts of the system through a quantitative reliability value. It will provide the benefit to put more testing efforts to those parts which will help in reducing the testing effort , reducing cost and time.

Mohamed and Zulkerine [12] proposed a methodology for reliability assessment, which considers component failure dependency and enables failure type awareness. They incorporated component failure dependency in the reliability qualification by analyzing failure propagation among system components.

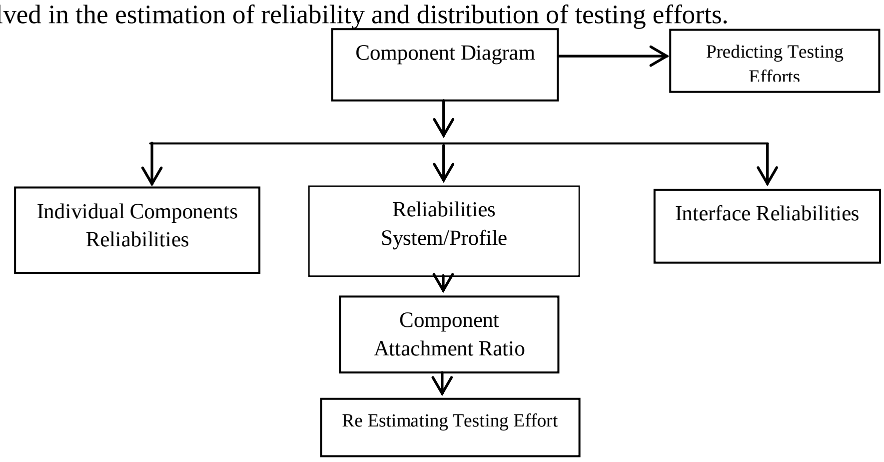
Seth and Shukla [13] proposed a enhanced approach for reliability analysis of CBSS by considering that a component failure affects system as well as other components also and different failures have different impact on overall system.

Sherif M. Yacoub et al. [14] provides methodology for assessing of reliability in early stages of development. They used a heuristic risk assessment methodology based upon the dynamic metrics, complexity and coupling matrices.

Voas and Payne [15] based on statistical approaches to testing have proposed mathematical models of component reliability, but have not defined models of composing reliability measures.

IV. PROPOSED WORK

This study assumes that the complete system reliability is a combination of the components individual reliabilities, interfaces reliabilities, and CAR. Many times components come with interfaces from the developer and some times it is required to build new interfaces according to requirements. There are two types of interfaces associated with the components first is known as provide interface which is associated with the component which initiate the action and second is known as required interface which is associated with the component which is affected by the action. Both these interface are critically important for software to work efficiently. As the failure of interfaces may lead to the failure of components and thus resulting in the failure of whole system, they must be taken into consideration. Figure (1.1) shows flow of process involved in the estimation of reliability and distribution of testing efforts.



Fig(1.1)

First of all component based architecture of the system is required which represents how all components and interface are connected to each other in the system. Then a initial estimation of testing effort is performed. Then individual reliabilities of components and interfaces are collected. There are some methods available to calculate the reliability of components one of them is by identifying the operational, logical errors and then predicting component reliability based upon them [16]. Then reliabilities of different executable profiles and system are estimated. Then CAR is calculated for each component. Establishing the relation based upon these parameters the reliabilities of all executable profiles and overall system can be estimated. The quantitative values of the reliability will provide a better way distributing the testing efforts throughout the system, thus improving the testing time and reducing the testing efforts. The proposed work is based upon some assumptions which are listed below.

- The overall system is designed using component based software technology.
- Each individual component is physically independent from all other components.
- Transfer of control between the components is governed by the *Markov* process.
- All the components are connected to each other through interfaces.
- Reliabilities of the components and interfaces are known.

For a given software system architecture, say there are N number of components in the system and each component have some reliability say Cr. Ir is the reliability of interfaces where Pi is the reliability of the provide interface and Ri is the reliability of the required interface.

Say there are 1, 2, ……X executable profile in the system and each executable profile x have some number of components say n. Then a relation can be established for reliability of each executable profile Rx and say there are 1, 2,….p direct relations in the system.

$$Rx = \frac{\sum_{i=1}^{n} Cr*Ir}{P}$$

$$Rs = \frac{\sum_{i=1}^{X} Rp}{X}$$

CAR, it is the ratio of number of components attached to a component to the total number of components in the system.

$$\sum_{i=1}^{N} CAR \geq 1$$

Let T be the total testing time or effort estimated to test the overall system and tp is the estimated time for testing each executable profile

T= *t1*+*t2*……+*ti*+…….+*tp*.

Where *tp*= $CAR * T$

V. RESULT AND ANALYSIS

In the context of analyzing the results from proposed method it is applied to software component system a company inventory system shown in the figure (1.2), [17]

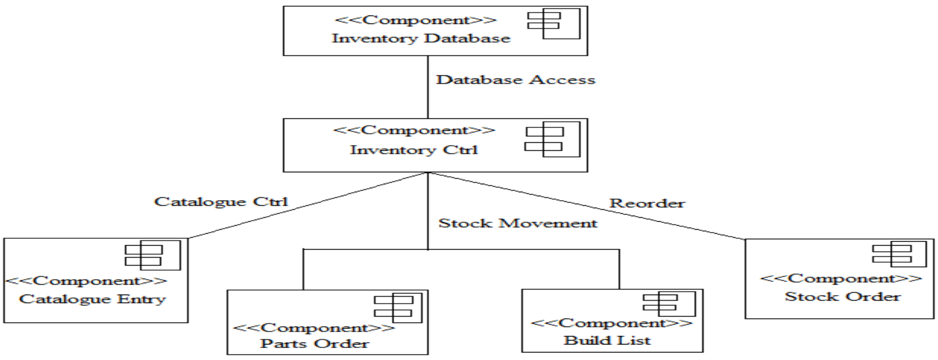


Fig (1.2)

The component diagram of the system showing all the interactions between the components follows is shown below

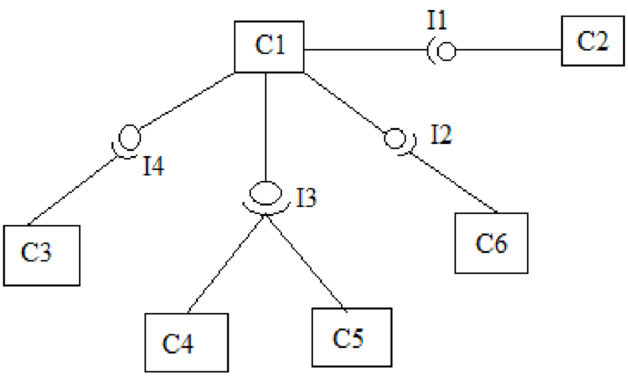


Fig (1.3)

Data Set Table(1.1)

Components	Component Reliability	Interface Reliability Provide/ Required C1	C2	C3	C4	C5	C6
C1	.991	x	.970	.990	.960	.960	.975
C2	.995	.960	X	X	X	X	X
C3	.996	.980	X	X	X	X	X
C4	.994	.965	X	X	X	X	X
C5	.993	.993	X	X	X	X	X
C6	.992	.985	X	X	X	X	X

There are five components in the system C1,……C5 and four Interfaces I1,…..I4. It is assumed that the reliabilities of the component and interfaces are already known.

There exists three executable profile on the system C2-C1-C3 say X1, C2-C1-C4-C5 say X2 and C2-C1-C6 say X3. Now deploying the methodology on the given data set and obtaining the results. Considering the profile X1

Rx1 = .96841 , similarly for profiles then

Rs = .96281 i.e. system reliability.

Calculating CAR for C1=1/6 and similarly for all components.

Results Obtained: Table (1.2)

Profile	Reliability Estimates	CAR	System Reliability
X1	.96841	.333	.96281
X2	.95508	.666	
X3	.96495	.333	

It is also assumed that in the earlier phase the testing effort is estimated for the system, say T= 60 hours of testing are estimated for the overall system testing and 20 hours is estimated for testing each profile say t1=20, t2=20 and t3=20 hours. Analyzing the results it is clear that the profile X2 have low reliability and high CAR so there must be more emphasis given to test X2 profile i.e. t2=40 hours and similarly t1=20 hours and t3=20 hours for X1 and X3 profiles respectively.

VI. CONCLUSION AND FUTURE SCOPE

Reliability estimation, representation of the executable profile and the overall system with the parameters of component reliability, interface reliability along with CAR provides a more accurate quantitative value to the development team and the testing team. This value helps better understanding of the system and provides the facility to make better changes in the architecture of the system, making the system work better. Reliability estimation of CBSS is still a wide area where a lot of work is to be done. In future this work can be extended to executable profiles which don't follow *Markov* process because most of the real world architectures does not follow the *Markov* process.

REFERENCES

[1] Mao X. and Deng Y., "A General Model for Component-Based Software Reliability", Conference on Software Engineering and Advanced Applications, pp. 395-398, 2003.
[2] Michael R. Lyu, "Handbook of Software Reliability Engineering", 1995.
[3] Michael R. Lyu, "Software Reliability Engineering: A Roadmap", In the Proceedings of International Conference on Future of Software Engineering, 2007.
[4] K. Goseva-Popstojanova, and K. S. Trivedi, "Architecture-based Approach to Reliability Assessment of Software Systems," Performance Evaluation Journal, vol. 45(2), 2001, pp. 179–204
[5] Kaiyuan Cai , Chenggang Bai, and Xiaojun Zhong, "Introduction to Reliability Models of Component-Based Software System," Journal of Xi'an Jiaotong University, vol. 37(6), 2003, pp. 560–564.
[6] Swapna S. Gokhale, "Architecture-Based Software Reliability Analysis: Overview and Limitations," IEEE Transactions on Dependable and Secure Computing, vol. 4(1), 2007, pp. 32–40.
[7] R. C. Cheung, "A user-oriented software reliability model," IEEE Transactions on Software Engineering, vol. 6(2). 1980, pp. 118-125.
[8] J. D. Musa, A. Iannino, and K. Okumoto, "Software Reliability: Measurement, Prediction, Application", McGraw-Hill, 1987.

[9] V. Almering, M.V. Genuchten, G. Cloudt, and P. J. M. Sonnemans, "Using Software Reliability Growth Models in Practice," IEEE Software, Vol. 24, No. 6, pp. 82–88, 2007.

[10] Szyperski, C., "Component Software. Beyond Object-Oriented Programming", Addison-Wesley, 1998.

[11] Daeeui Hong , Taewan Gu and Jongmoon Baik "A UML Model based White Box Reliability Prediction to Identify Unreliable Components", 5th International Conference on Secure Software Integration & Reliability Improvement Companion, pp.152 – 159, 2011.

[12] Mohamed, A. and Zulkernine, M. "Failure Type software Reliability Assessment with Component Failure Dependency", Fourth International Conference on Secure Software Integration and Reliability Improvement, pp. 98 – 105, 2010.

[13] Shukla.S and Seth.K." Failure Dependent Reliability Analysis for Component Based Software Systems", Third International Conference on Computer and Communication Technology, pp. 149 – 153, 2012.

[14] Sherif M.Yacoub and Hany H.Ammar "A Methodology for Architecture-Level Reliability Risk Analysis", IEEE Transactions on Software Engineering, Vol. 28, No. 6, pp. 529-547, 2002.

[15] Voas, J., and J. Payne, "Dependability Certification of Software Components", Journal of Software Systems, Vol. 52, pp. 165-172, 2000.

[16] Singh, A. P. And Tomar P., "A Proposed Methodology for Reliability Estimation of Component-Based Software", International Conference on Optimization Modelling and Applications, 2012.

[17] TechnologyUK Unified Modeling Language. , http://www.technologyuk.net/computing/uml/component_diagrams.shtml.