International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2013): 4.438

State- Based Approach to Analyze the Reliability of Component Based Software

Pooja Gupta[1], Amanpreet Kaur Boparai[2]

[1]Research Scholar, Chandigarh University, Gharuan-Mohali, Punjab, India

[2]Assistant Professor, Chandigarh University, Gharuan-Mohali, Punjab, India

Abstract: ***Software Reliability is defined as the "Probability of failure free operation of a computer program in a specified environment for a specified time". The conventional approaches which are used for software Reliability systems are black-box based. In this approach, the software is considered as a whole without looking into its internal architecture there is an only interaction with the outside world are modeled. Hence, this approach is insufficient to model the behavior of real software applications. Architecture-based analysis, try to determine the behavior of software application by considering the nature of its part and their interaction. Most of the research has been done in the area of architecture-based analysis to developing analytical models, and no effort cursed to establishing a framework (model) and however no attempt has been made to know how this framework might be applied to real world applications. In this paper, present an approach for determining the reliability of component-based software .Our Method is based on the state–Based approach to analyze the reliability of component-based software. In state-based models the architecture is represented either by discrete time Markov chain (DTMC) or a continuous time Markov chain (CTMC).***

Keywords: Software Reliability, Reliability Model, Component-Based Software, Architecture based software Reliability, State-Based model.

1. Introduction

In Component-Based Software Engineering, Software system can be developed by selecting appropriate Off- The - Shelf components (COTS) and then assembling then with well defined software architecture [9].

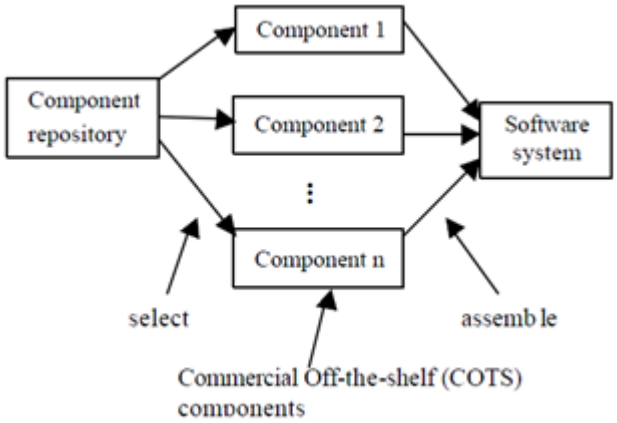


Figure 1: Component Based software development

It is very difficult to achieving highly reliable software because software is composed in a heterogeneous way, in which each component has its own workload and according to workload it has different failure.

Following are the major classes of Software Reliability:-

Black box reliability analysis: in which estimation of software reliability based on failure observation from testing. It is called black-box approach because internal details of the software are not considered.

Software metric based reliability analysis: in which reliability evaluation based on the static analysis of software (e.g., LOC, complexity) or its development process and conditions.

***Architecture-based reliability analysis*:** in which reliability evaluation based on software component reliability and the system architecture (the way the system is composed of the components). This approach is also called as white-box approach or grey or component-based reliability estimation (CBRE) [10].

There are several techniques and models have been proposed to analyze the reliability of component based applications [4]. A Software Reliability models are used to assess a software product's reliability or to estimate the number of latent defects when it is available to the customers.

Reliability models can be broadly classified into two categories:- static models and dynamic models.

A static model uses other attributes of the project or program modules to estimate the number of defects in the software. A dynamic model, usually based on statistical distributions, uses the current development defect patterns to estimate end-product reliability.

Dynamic software reliability models, further classified into two categories: those that model the entire development process and those that model the back-end testing phase. The former is represented by the Rayleigh model. The latter is represented by the exponential model and other reliability growth models (SRGM).

The Faults and failures are two important factors which are generally exist in our software during the development phases. Fault is also known as an error or bug which is injected during the development phase. As the many user use the software application so there is a probability that the no of failure from fault increases. The no of fault increases if

affect the reliability of the software. Because if software has less no of errors then we can say that it is reliable software. Due to this, the software technology has failed to steep in various fields like in quality, productivity as compared to hardware.

Software structure greatly impact on its reliability and correctness has been highlighted in 1975-1976 by Parnas and Shooman[7].There are several drawbacks of conventional approach after that a average-sized software application is developed using a "Divide and conquer" technique in which several intermediate parts are integrated after some time this technology generates a renew concept in "architecture-based analysis" its aim is according to component behavior and the architecture of the software application characterize the reliability and performance behavior of application. The main aim of architecture analysis is that it can analyze how different components are interacting with each other and how they are combined to make a software system.

The existing architecture-based model is classified into three broader categories viz. state-based, Path-based and additive [11]

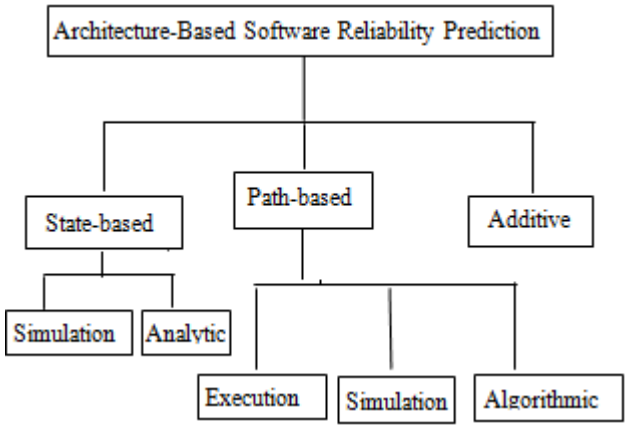


Figure 2: Classification of architecture-based software reliability model

The layout of the paper is as follows. Section I introduced the basic concept of software reliability. Section II introduced the role of state-Based model in software reliability. Section III describes Analysis and classification. Section IV describes the metrics which are used for Software Reliability. Section V presents conclusion

2. Role of State-Based Model in Software Reliability

State-based model uses the control flow graph to represent the software architecture and to examine reliability analytically. There is an assumption that when control transfer among the component it follow the Markov property. To model the software architecture it uses the Discrete-time Markov chain (DTMC) and Continuous-time Markov Chain (CTMC).

In state-based model there is still a question that how this model will be implemented in real world because on which there is no perception that how input is accepted and output is produced by the model. To apply these models in real world some parameter values are needed these values describe the architectural nature of an application and the failure characteristic of each component. The architecture-based models apply during the design phase whereas the Black-box based approach applied during the testing phase of the development life cycle. Based on these software artifacts we gave a conclusion that which model is appropriate to predict the reliability at each phase of the software development cycle. It is the responsibility of architecture-based approach is that which components should be picked off the shelf, and which components should be developed in house. Whether the components are reliable or not all these aspects are handled by the design phase of the development process. This technique will not be useful for testing phase, but also be useful in applying those applications of software which are in operational mode or that has already been established.

3. Analysis and Classification of State-Based Model

The Analysis of State-based model on the basis of architecture and failure behavior

a) *Architecture of the Application*: This is the way in which different components of the Software are interacts, and is given by the intercomponent transition probabilities. Mathematically the transition probability is given by:

$$\boldsymbol{P\ xn-1, xn\ =\ P\{X(tn)\ =\ xn\ |\ X(tn-1)\ =\ Xn-1\}}$$

The architecture of the application can be designed by using Discrete Time Markov chain (DTMC) and Continuous Time Markov Chain (CTMC). In which the state transitions are represented by transfer control among the components. The DTMC characterized by its one-step transition Probability matrix, **P = [pij]**, where i and j are the two states. Since we are considering a terminating application, the DTMC of interest will have one or more absorbing states.

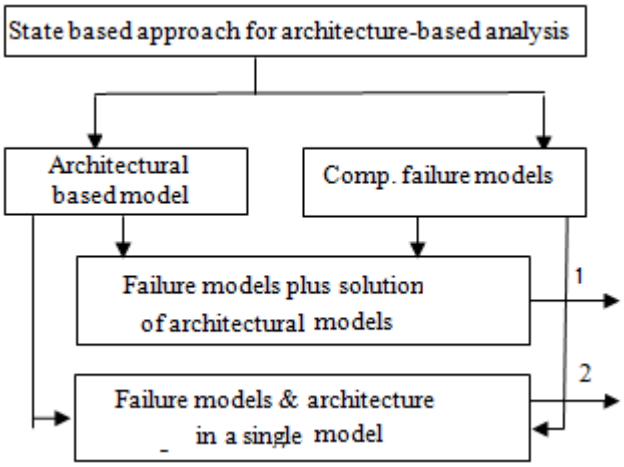
b) *Failure Behavior of the component and interfaces*:
Failure may occur during the execution of any component or during the control transfer between two components. The failure behavior of the components may be specified in terms of the probability of failure (or reliability), time-independent failure rate or time-dependent failure intensity. It is clear that interface failures or failures that occur during the transfer of control between two components should be considered separately from individual component failures. In this case we use many software metrics. The information of the architecture of the application with failure behavior of the component and interface with the component can be combined in two different ways to predict the reliability and application performance of the system. Following are the two ways:

i. Hierarchical Method.
ii. Composite Method.

In the "*hierarchical method*," an estimate of the application reliability is obtained in two steps. In the first step, the model representing the application architecture is solved to obtain the architectural statistics of the application. The architectural statistics include the mean and the variance of the number of visits to each component, In the second step,

the architectural statistics are combined with the failure parameters of the components to obtain an analytical reliability function. The hierarchical method describe the concept of intracomponent in which same component are independent of one another and it can lead to pessimistic reliability estimate. Krishnamurthy and Mathur [12] resolve the issue of intracomponent dependency.

Composite Method: The Composite method provides an exact reliability estimate and it is not depend on the concept of intracomponent dependency.



1. Hierarchical method of reliability analysis
2. Composite method of reliability analysis.

Figure 3: Analysis methods in state-based approaches.

4. Metrics Used for Software Reliability

The Reliability requirements for different categories of software products may be different. A good reliability measure should be observer-independent, so that different people can agree on the degree of reliability a system has. There are several metrics that are correlate with reliability as follows:

- ***Rate of Occurrence Of Failure (ROCOF):*** ROCOF measure the frequency of occurrence of failures. ROCOF measure of a software product can be obtained by observing the behavior of a software product in operation over a specified time interval and then calculating the value of ROCOF value.
- ***Mean Time To Failure (MTTF):*** MTTF is the time between two successive failures averaged over a large no of failures. To measure MTTF, we can record the failure data for n failures. It is important to note that only run time is considered in the time measurement.
- ***Mean Time To repair (MTTR)***: Once failure occurs, some time is required to fix the error. It measure the average time it take to track the error and causing failure and to fix them.
- ***Mean Time Between failure (MTBF):*** it is the combination of MTTR and MTTF.
- ***Probability Of Failure On Demand (POFOD):*** It measure the likelihood of the system failing when a service request is made. For example, a POFOD of 0.001 would mean that 1 out of every 1000 service request would result in a failure.
- ***Availability***: It measure how likely would the system be available to use over a given period of time. Availability, or more specifically, instantaneous availability, is typically defined as the fraction of time during which a component or system is functioning acceptably, i.e., the uptime over the total service time

5. Related Work

Several Reliability models and estimation techniques have been suggest to assess the reliability of Component based applications.

Sherif Yacoub and Gokhale [4] discuss the discrete-event simulation to analyze component-based applications. In which program based procedure is used which gives the inter-failure arrival time of a given component. This approach assume the existence of control flow graph of a program. It also assume constant execution time of component interaction and ignores failures in component interfaces and links.

After that Gokhale et al.[5] proposed a unification framework. However the development of State-based approach is ad hoc with little or no effort toward the establishment of component so unification framework is proposed which compare and contrast the models.

Michael R. Lyu and Zibin Zheng [6] uses a service-oriented Architecture (SOA).The SOA is major software framework for building complex distributed systems. Reliability of the service-oriented system heavily depend on the remote web services as well as the unpredictable internet and they proposed a collaborative Reliability prediction approach, which employs past failure data of other similar user, without requiring real world web service invocations. In large-scale real- world experiments are conducted and the experimental results shows that our collaborative reliability predictions approach obtain better reliability.

Swapna S.Gokhhale[7] introduces the various limitations of software Reliability analysis. Like Modeling limitation which discusses the limitations of models used for architecture-based analysis. There are several problems arise in modeling are concurrent execution in which state-based model assume that only one component is executing at any instant of time. The other limitations is Non- markov transfer of control in which state-based model assume that transfer of control follow only first-order Markov property.

Poor[8] proposed an approach in which component model is used to represent the system as a combination of components with transition probabilities. He does not consider how components are interacting and how its parameters are obtained. He only assumed that the analyst will construct the model based on domain experience.

Krishnamurthy et al.[9] uses the CBRE technique to assess the reliability of component-based applications. This approach is based on the test information and test cases. In which to run the test cases each execution path is defined. In

which component interface fault is not considered because our main consideration on the test cases but component interface part is the main factor in reliability analysis of component-based application.

6. Conclusion

At the first software engineering (SE) conference in 1968, Doug McIlroy introduced the concept of software components during his keynote speech, "Mass-Produced Software Components." Since 1968, components have played a role in both SE research and practice. For example, components have been an important part of software architecture from its early days. In 1998, the International Conference on Software Engineering introduced component-based software engineering (CBSE) as a specific area within SE at the first workshop on CBSE. CBSE aims to build software from preexisting components, build components as reusable entities, and evolve applications by replacing components. This requires significant changes in the development paradigm, from both technical and business viewpoints. The Software reliability is a key part in software quality. The Software reliability is defined as the probability of failure-free operation in a defined environment for a specified period of time. Reliability can be associated with both hardware and software. The hardware Reliability can easily be evaluated since hardware get wear out but in case of software it be very difficult. Further, we used various Reliability-models to predict the reliability of the system. So at last, main purpose of this paper is to understand the concept of software reliability using the state-based approach.

References

[1] Gokhale,S.S,.Architecture-based software reliability analysis: Overview and limitations. IEEE Transactions on dependable and secure computing, 4(1),32-40, 2007.
[2] Eklund, P. Dealing with the complexity of CBSE-Fundamental Environmental needs.
[3] Cortellessa, V., & Grassi, V. "A modeling approach to analyze the impact of error propagation on reliability of component-based systems", in Springer Berlin Heidelberg , (pp. 140-156) , 2007.
[4] Yacoub, S., Cukic, B., & Ammar, H. H. "A scenario-based reliability analysis approach for component-based software. Reliability", IEEE Transactions on, 53(4), 465-480,2004.
[5] Reussner, R. H., Schmidt, H. W., & Poernomo, I. H."Reliability prediction for component-based software architectures". Journal of Systems and Software, 66(3), 241-252 , 2004.
[6] Zheng, Z., & Lyu, M. R." Collaborative reliability prediction of service-oriented systems". In Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering-Volume 1 (pp. 35-44). ACM , 2010.
[7] Gokhale, S. S., & Trivedi, K. S. "Analytical models for architecture-based software reliability prediction: A unification framework. Reliability" ,IEEE Transactions *on*, *55*(4), 578-590 , 2006.
[8] Koziolek, H., Schlich, B., & Bilich,."A large-scale industrial case study on architecture-based software reliability analysis". In Software Reliability Engineering (ISSRE), 2010 IEEE 21st International Symposium on (pp. 279-288). IEEE , 2010
[9] Xia, C., & Fu, A. "Component-Based Software Engineering: Technologies, Quality Assurance Schemes, and Risk Analysis Tools". In Seventh Asia-Pacific Software Engineering Conference, 2003.
[10] Khoshgoftaar, T. M., Allen, E. B., Hudepohl, J. P., & Aud, S. J."Application of neural networks to software quality modeling of a very large telecommunications system". Neural Networks, IEEE Transactions on, 8(4), 902-909 , 1997.
[11] K. Goseva-Popstojanova and K. S. Trivedi, "Architecture-based approach to reliability assessment of software systems," Performance Evaluation, vol. 45, no. 2–3, June 2001.
[12] S. Krishnamurthy and A.P. Mathur, "On the Estimation of reliability
[13] Shanmugam, L., & Florence, "A Comparison of Parameter best estimation method for software Reliability Models".International Journal of Software Engineering & Applications, 3(5) , 2012.
[14] Tyagi, K., & Sharma, " Reliability of component based systems: a critical survey". ACM SIGSOFT Software Engineering Notes, 36(6), 1-6. , 2011.
[15] Crnkovic, I., & Larsson, M. P. H. (Eds.)."Building reliable component-based software systems". Artech House.2002