Oriental Journal of Computer Science & Technology Vol. **4**(1), 135-140 (2011)

Reliability through Simulation: Goals and Limitations

S.M.K. QUADRI and AASIA QUYOUM

Department of Computer Sciences, University of Kashmir, J&K (India)

(Received: December 04, 2010; Accepted: January 13, 2011)

ABSTRACT

Software Reliability is an important component of software quality. A number of software reliability models have been proposed since 1970s, but there is no single model that can be used in all the situations. To reduce the risk, it is better to experiment with the model of the system rather than with the system itself. Simulation, offers an attractive alternative to analytical models as it describes a system being characterized in terms of its artifacts, events, interrelationships and interactions in such a way that one may perform experiments on the model, rather than on the system itself. Simulation strives for achieving its goals but it does have certain limitations. This research paper focuses on the goals and limitations of using simulation in software reliability.

Key words*:* Simulation, Software Reliability, Model.

INTRODUCTION

To study a system, it is possible to experiment with the system itself or with the model of the system, but experimenting with the system itself is very expensive and risky. The objective of many system studies, however is to predict how a system will perform before it is built. Consequently, system studies are generally conducted with a model of a system. A model is not only a substitute of a system; it is also a simplification of the system.

A number of software reliability models have emerged as people try to understand the attributes of how and why software fails, and try to quantify software reliability. Over 200 models have been proposed since 1970s, but how to quantify software reliability still remains unsolved. There is no single model that can be used in all the situations. No model is complete; one model may work well for a set of certain software, but may be completely off track for other kinds of problems.

Most existing analytical methods to obtain reliability measures for software systems are based on the Markovian models and they rely on the assumption on exponential failure time distribution. The Markovian models are subject to the problem of intractably large state space. Methods have been proposed to model reliability growth of components which can not be accounted for by the conventional analytical methods but they are also facing the state space explosion problem. A simulation model, on the other hand offers an attractive alternative to analytical models as it describes a system being characterized in terms of its artifacts, events, interrelationships and interactions in such a way that one may perform experiments on the model, rather than on the system itself, ideally with indistinguishable results.

Simulation is the process of constructing a model of a system which contains a problem and conducting experiments with the model on a computer for a specific purpose of experimentation to solve the problem.

Software reliability

Software Reliability is defined as the probability of the failure free software operation for a specified period of time in a specified environment [ANSI91] [Lyu95].

Unreliability of any product comes due to the failures or presence of faults in the system. As software does not 'wear-out" or "age", as a mechanical or an electronic system does, the unreliability of software is primarily due to bugs or design faults in the software.

Reliability is a probabilistic measure that assumes that the occurrence of failure of software is a random phenomenon. Randomness means that the failure can't be predicted accurately. The randomness of the failure occurrence is necessary for reliability modeling.

Software reliability activities

The reliability process in generic terms is a model of the reliability- oriented aspects of software development, operations, and maintenance. Quantities of interest in a project reliability profile include artifacts, errors, defects, corrections, faults, tests, failures, outages, repairs, validation, and expenditures of resources, such as CPU time, manpower effort and schedule time. The activities relating to reliability are grouped into classes:

Construction

Generates new documentation and code artifacts

Combination

Integrates reusable documentation and code components with new documentation and code components

Correction

Analyzes and removes defects in documentation and code using static analysis of artifacts.

Preparation

Generates test plans and test cases, and readies them for execution.

Testing

Executes test cases, where upon failure occurs.

Identification

Makes fault category assignment. Each fault may be new or previously encountered.

Repair

Removes faults and possibly introduces new faults.

Validation

Performs inspections and checks to affirm that repairs are effective

Retest

Executes test cases to verify whether specified repairs are complete if not, the defective repair is marked for repair. New test cases may be needed.

Simulation

Simulation refers to the technique of imitating the character of an object or process in a way that permit us to make quantified inferences about the real object or process. In the area of software reliability, simulation can mimic key characteristics of the processes that create, validate, and revise documents and code. A simulation model describes a system being characterized in terms of its artifacts, events; inter relationships, and interactions in such a way that one may perform experiments on the model, rather than on the system itself, ideally with indistinguishable results. Simulation presents a particularly attractive computational alternative for investigating software reliability because it averts the need for overly restrictive assumptions and because it can model a wider range of reliability phenomena than mathematical analyses can cope with. Simulation does not require that test coverage be uniform, or that a particular fault-to-failure relationship exist, or that failures occur independently, if these are not actually the case.

Simulation goals

The fourteen goals presented herein are described based on the already published literature. The goals are listed below in no particular order.

Data Availability

Reliability modeling ultimately requires good data. But software projects do not always collect data sets that are comprehensive, complete, or consistent enough for effective modeling research or model application. Since good data sets are so scarce, one purpose of simulation is to supply carefully controlled, homogeneous data.

Understanding assumptions of analytic models

The actual software artifacts (such as faults in computer programs) and processes (such as failure and fault removal) often violate the assumptions of analytic software reliability models, simulation can provide a better understanding of such assumptions and may even lead to a better explanation of why some analytic models work well in spite of such violations.

When the analytic solution is expensive

Simulation is also used even when an exact analytic solution is possible, but it is too expensive in terms of computation time.

Mimic the behavior the system

Simulation can mimic key characteristics of the processes that create, validate, and revise documents and code. It can mimic faulty observation of a failure when one has, in fact, occurred, and, additionally, can mimic system outages due to failures. Furthermore, simulation can distinguish faults that have been removed from those that have not, and thus can readily reproduce multiple failures due to the same as-yet un-repaired fault.

Relates model-pertinent resource dependencies

Some reliability sub-processes may be sensitive to the passage of execution time (e.g., operational failures), while others may depend on wall-clock, or calendar, time (e.g., project phases); still others may depend on the amount of human effort expended (e.g., fault repair) or on the number of test cases applied. A simulator can relate model-pertinent resource dependencies to a common base via resource schedules, such as workforce loading and computer utilization profiles.

Risk reduction

Simulation is a risk reduction method. Uncertainty is reduced and replaced with certainty about the expected operation of a new system or about the effects of changes to an existing system.

Complex estimation

Simulations can be used to explore and gain new insights into new technology, and to estimate the performance of systems too complex for analytical solutions.

Reduce computation complexity

By using simulation, the computation complexity of reliability analysis can be reduced to a great extent.

Detection of unpredicted Problems

When a system is simulated prior to installation and found to work in concept, the model is often refined to include finer details. The detailed simulation may reveal unforeseen problems or bugs that may exist in the system's design. By discovering these problems prior to installation, debug time and rework costs can be avoided. In addition, improvements to system operation may be discovered.

Perform experiments on the model rather than on the system

A simulation model describes the system being characterized in terms of its artifacts, events, interactions and interrelationships in such a way that one may perform experiments on the model rather than on the system itself.

Increase in Knowledge on System

A primary benefit of the simulation process is an increase in overall system knowledge. At the start of a simulation project, especially in modeling of complex systems, knowledge is often dispersed among many different people, so channels for information gathering process need to be established which will speed up the process considerably and allow data to flow from individual experts to simulation.

Present a computational alternative for investigating software reliability

Simulation presents a computational alternative for investigating software reliability as it averts the need for overly restrictive assumptions

and can model a wider range of reliability phenomenon.

Eliminate the requirement of uniform test data coverage

Simulation does not require that test coverage be uniform, or that a particular fault to failure relationship exists.

Speed in Analysis

After a model has been developed, it is possible to run the simulated system at speeds much greater than would be attainable in the real world.

Make quantified inferences about the real object

Simulation imitates the character of an object or process in a way that permits one to make quantified inferences about the real object or process.

Enhances Creativity

Simulation can enhance creativity in the design of a system. If a model of a system is available all the potential solutions for a particular problem could be tried and compared without risk of failure.

Generate more realistic forecasts

Simulation can generate more realistic forecasts than analytic models do.

Allows Experimentation without disruptions to the existing system

Testing new ideas in already existing systems may be costly and vary difficult. Simulation allows a model to be developed, so any change can be made to the model first, the effect on the system examined, and then decision to implement the changes in the actual system can be made.

Idea can be tested prior to Installation

A computer simulation allows concepts to be tested prior to the installation of new systems. This testing may reveal unforeseen design flaws and give designers a tool for improvement. If the same flaws were discovered after installation, changes to the system might end up being very costly or even impossible to implement.

Simulation limitations

Limitation limits the extent of something. Simulation also has some limitations that should be taken into account to set realistic expectations about its benefits. Simulation has following limitations:

- While simulation may be useful for creating data sets for studying other, more conventional reliability models, it cannot provide the necessary attributes of the phenomenon being modeled without real information derived from real data collected from real projects, past and present.
- Simulation model validation verification and testing (VV&T) is difficult and requires creativity and insight. One must thoroughly understand the whole simulation model so as to design and implement effective tests and identify adequate test cases. Knowledge of the modeling methodology and prior modeling and VV&T experience are required for successful testing
- Simulation model credibility can be claimed only for the prescribed conditions for which the model is tested.
- Simulation yields only approximate answers as it mostly relies on the use of random number generators to provide model input. Since the input has a random element, some uncertainty is also associated with the output.
- It is expensive to build an effective simulation model.
- A large number of effects in the real world are still unexplained or very hard to model. This means the programmer may not be able to model everything accurately, especially in real time.
- It is difficult to validate. Validation is the process of making sure the computer model accurately represents the system being studied. When the system does not yet exist, this can become a difficult task.
- Simulation required large number of experimentations or runs under a given set of conditions. Any deviation in these conditions may not justify the simulation results. Therefore, each simulation model provides a unique solution.
- With increase in parameters, simulation becomes very complex to the model.

- The creation of a computer model often can be an expensive method of analysis.
- In most cases, data collection, model development, analysis, and report generation will require considerable amounts of time.

CONCLUSION

Simulation is a risk reduction method. Uncertainty is reduced and replaced with certainty about the expected operation of a new system or about the effects of changes to an existing system as it allows us to perform experiments on the model, rather than on the system itself. It presents a particularly attractive computational alternative for investigating software reliability because it averts the need for too restrictive assumptions and because it can model a wider range of reliability phenomena than mathematical analyses can cope with. Simulations are becoming an increasingly powerful tool that scientists and engineers can employ to aid in solving challenging problems. A further consideration when planning or implementing computer simulations is to take into account the computational resources available and the length of time those resources will remain available. Simulations can be utilized to guide researchers in the proper direction and to investigate phenomena that are currently beyond our capacity to investigate using empirical means, but they should not be utilized as a replacement for empirical methods

REFERENCES

1. C. T. Lin, C. Y. Huang, and C. C. Sue, "Measuring and Assessing Software Reliability Growth Through Simulation-Based Approaches," Proceedings of the 31st IEEE Annual International Computer Software and Applications Conference (COMPSAC 2007), pp. 439-446, Beijing, China, (2007).
2. J. Lo, S. Kuo, M.R. Lyu, and C. Huang, "Optimal Resource Allocation and Reliability Analysis for Component-Based Software Applications," *Proc. 26th Ann. Int'l Computer Software and Applications Conf. (COMPSAC),* pp. 7-12 (2002).
3. John D. Musa, "Operational Profiles in Software-Reliability Engineering," IEEE Software, **10**(2): 14-32 (1993).
4. Kishor S. Trivedi, "Probability and statistics with reliability, queuing and computer science applications," John Wiley and Sons Ltd., Chichester, UK (2001).
5. K. Kanoun M. Kaaniche C. Beounes J.C. Laprie and J. Arlat, "Reliability Growth of Fault-Tolerant Software," IEEE Trans. Reliability, **42**(2): 205-219 (1993).
6. Kumar, M., Ahmad, N., Quadri, S.M.K., "Software reliability growth models and data analysis with a Pareto test-effort", *RAU Journal of Research*, **15**(1-2): 124-8 (2005).
7. Norman F. Schneidewind, "Fault Correction Profiles, "Proceedings of the 14th International Symposium on Software Reliability Engineering, p.257 (2003).
8. Quadri, S.M.K., Ahmad, N., Peer, M.A., "Software optimal release policy and reliability growth modeling", Proceedings of 2nd National Conference on Computing for Nation Development, INDIACom-2008, New Delhi, India, 423-31 (2008).
9. R.C.Tausworthe, "A General Software Reliability Process Simulation Technique, "NASA JPL Publication, 91-7 (1991).
10. R.C. Tausworthe and M.R, Lyu, "A generalized technique for simulating software reliability, "IEEE Software," **13**(2): 77-88 (1996).
11. Robert C. Tausworthe , Michael R. Lyu, "A Generalized Technique for Simulating Software Reliability, "IEEE Software, **13**(2): 77-88 (1996)
12. S.Gokhale, M. R.Lyu, and K. S. Trividi," Reliability Simulation of Component-Based Software Systems, "Proceedings of the 19th International Symposium on Software Reliability Engineering, pp. 192-201, Paderborn, Germany, (1998).
13. S.Gokhale, Michael R. Lyu, and K.S. Trivedi." Reliability Simulation of Fault-Tolerant Software and Systems". In Proc. of Pacific

Rim International Symposium on Fault-Tolerant Systems (PRFTS '97), pp. 197-173, Taipei, Taiwan, (1997).

14. S. Krishnamurthy , A. P. Mathur, On The" Estimation Of Reliability Of A Software System Using Reliabilities Of Its Components," Proceedings of the Eighth International Symposium on Software Reliability Engineering (ISSRE '97), 146 (1997).
15. Swapna S. Gokhale , Kishor S. Trivedi, "A time/structure based software reliability model," Annals of Software Engineering, **8**(1-4) 85-121 (1999).
16. Swapna S. Gokhale , Kishor S. Trivedi , Michael R. Lyu, "Reliability Simulation of Fault-Tolerant Software and Systems," Proceedings of the 1997 Pacific Rim International Symposium on Fault-Tolerant Systems, 167 (1997).
17. S. Y. Kuo, C. Y. Huang, and M. R. Lyu, "Framework for Modeling Software Reliability, Using Various Testing-Efforts and Fault-Detection Rates," IEEE Transactions on Reliability," **50**(3): 310-320 (2001).
18. Tausworthe, Robert C., "A General Software Reliability Process Simulation Technique," Technical Report 91-7, Jet Propulsion Laboratory, Psaadena, CA, (1991).
19. Wood, A., "Predicting software reliability", IEEE Computers, **11**: 69-77 (1996).
20. Von Mayrhauser, A.,Malaiya, Y.K.,Keables, J.,and Srimani, P. K., "On the Need for Simulation for better Characterization of Software Reliability, "International Symposium on Software Reliability Engineering," Denver, CO (1993).