Reliability Characterization for Embedded System with N-Version Programming

Manju Kaushik, Gireesh Kumar

***Abstract*—This investigation deals with an embedded system considering both the software and hardware reliability. In this model, we have considered a specified number of subsystems composed of software and hardware components arranged in series. According to N-Version programming, N versions of software are running at same time for the execution of the same task. The Markov model is established for the purpose of reliability analysis under the consideration of reboot. Also, we have assumed that all rates are exponentially distributed. A numerical illustration is also carried out to validate the proposed model.**

***Index Terms*– embedded system; N-version programming; markov model; redundancy; software/hardware components.**

I. INTRODUCTION

Reliability is probably the most important characteristics inherent in the concept of "software quality". Software reliability helps to the software developers and users for increasing the system efficiency. For providing guarantees and reducing the cost, the reliability analysis of the software is employed which enhance the quality of software. The concept of software reliability has attracted the more researchers in concern with academia and industry due to scarcity of applications in industry, military, embedded system including telecommunication, distributed system etc.. Jelinski and Moranda [1] discussed the basic software reliability concepts in his study. Keiller and Miller [2], Downs and Scott [3] explored the uses and performance of software reliability models. Satoh and Yamada [4] incorporated the discrete equations and software reliability growth models in their study. A software reliability growth model for a distributed developed environment has been suggested by [5].

An embedded system is a computer system which control many devices in common use today. In general, embedded systems can be defined as all computing systems other than general purpose computer. Further, embedded system plays a vital role in our daily life which can be found in cell phones, microwave ovens, digital cameras, camcorders, portable video games, home security systems, washing machines and many more devices. The embedded system is dedicated to specific-

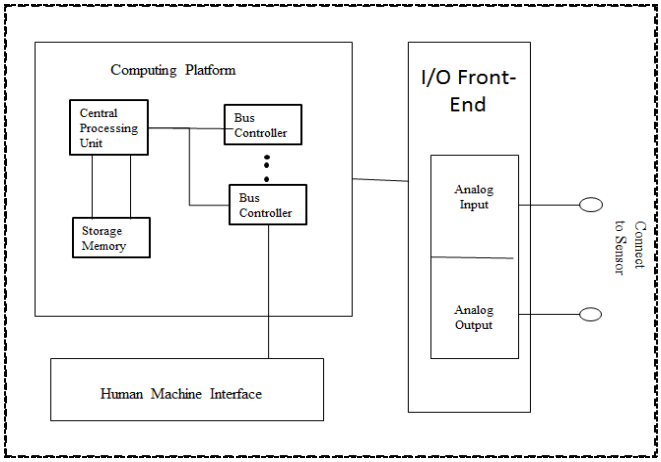

Fig 1. Embedded System.

Tasks according to which design engineers can optimize it to reduce the size and cost of the product and increase the reliability and its performance. Reference [6] discussed the characteristics of the embedded software. They have analyzed that the embedded software are real time and embedded which are the difference between embedded software and ordinary software. Wattanapongsakorn and Levitan [7] presented four models for optimizing the reliability of embedded systems considering both software and hardware reliability under cost constraints, and one model to optimize system cost under multiple reliability constraints. In fig. 1, we represent the internal position for embedded system.

To improve the quality of the software, the concept of N-version programming (NVP) has been applied by many researchers in their study. The concept of NVP through comparison of a redundancy method was used by [8] for hardware designers. According to this method, N-fold computation is carried out by using N independently designed software modules called versions which provide a single decision by decision algorithm. Moreover, the independent generation of $N \geq 2$ software versions are termed as NVP. NVP for fault tolerance was briefly suggested by [9]. The main advantage of NVP is to minimize the probability of similar errors at decision points, different algorithms, programming languages, environments and tools wherever possible. Recently, the work on NVP was done by [10] in their study.

In this paper, NVP has been applied for embedded system. On the other hand, the concept of reboot based recovery system is also taken in to consideration. According to this, we try to reduce the downtime caused by reboots as much as possible. Reference [11] has been applied the concept of rebooting for operating system in their study. The rest of the paper is organized as follows. Section II presents the model description for embedded system with N-version programing. Sections III provides the state governing

Manju Kaushik, Department of Computer Science & Engineering, JECRC University, Jaipur-303905, India
Gireesh Kumar, Department of Computer Science & Engineering, JK Lakshmipat University, Jaipur -302026, India.

equations of the given model. Section IV discusses the analysis of given model and V describe numerical results for validation purpose. Finally, paper comes to end with concluding remarks in section VI.

II. Model Description

In this paper, we have considered an embedded system in which software and hardware units are arranged together. The Markov model is created for the purpose of reliability analysis under the consideration of reboot. Now, we describe some notations and assumptions for modelling purpose which are given as follows:

Notations:

$P_{0,i}$: Steady state probability that the subsystem is in fully working state in i[th] (i=1, 2, ….,n) version

$P_{S_{j,i}}$: Steady state probability that j[th] (1, 2) software unit is working state in the sub system of i[th] (i=1, 2, ….,n) version

$P_{H_{j,i}}$: Steady state probability that j[th] (1, 2) hardware unit is working state in the sub system of i[th] (i=1, 2, ….,n) version

$P_{F,i}$: Steady state probability that the sub system is in the failed stated of i[th] (i=1, 2, ….,n) version

λ_1 : Transition rate from state 0 to state H_1

λ_2 : Transition rate from state H_1 to state H_2

μ_1 : Transition rate from state 0 to state S_1

μ_2 : Transition rate from state S_1 to state S_2

γ : Transition rate from state H_2 and S_2 to failed state F

β : Transition repair rate from the failed state H_2 to restoring the system to working state 0.

Assumptions:

- There are N-version of each subsystems of the entire system. In each subsystem there are two alternates primary and secondary of each software versions and two hardware units operating and spare in each version execution. If any one of these version is failed, the subsystem can be restored by the reboot.

III. Governing Equations

On the basis of the state transition diagram as depicted in fig. 2, using inflow and outflow rates, the steady state governing equations of a i[th] (i=1, 2, ….,n) version of the embedded system can be written as given below:

$$(\lambda_1+\mu_1)P_{0,1}=\beta P_{F,1} \quad (1)$$

$$\lambda_2 P_{H_{1,i}}=\lambda_1 P_{0,i} \quad (2)$$

$$\gamma P_{H_{2,i}}=\lambda_2 P_{H_{1,i}} \quad (3)$$

$$\mu_2 P_{S_{1,i}}=\mu_1 P_{0,i} \quad (4)$$

$$\gamma P_{S_{2,i}}=\mu_2 P_{S_1,i} \quad (5)$$

$$\beta P_{F,i}=\gamma\left(P_{H_{2,i}}+P_{S_{2,i}}\right) \quad (6)$$

and the normalizing condition for i[th] version (i=1, 2, ….,n) is given by

$$(P_{0,i}+P_{H_{1,i}}+P_{H_{2,i}}+P_{S_{1,i}}+P_{S_{2,i}}+P_{F,i})=1 \quad (7)$$

IV. Analysis

For the analysis purpose, recursive method has been applied. The steady state probabilities are obtained by solving the steady state equations (1)-(7) with the help of recursive method. Then, these steady state probabilities are used to evaluate the availability of the embedded system.

Thus, the availability of a i[th] (i=1, 2, ….,n) version is denoted by A_{V_i} and obtained by:

$$A_{V_i}=P_{0,i}+P_{H_{1,i}}+P_{H_{2,i}}+P_{S_{1,i}}+P_{S_{2,i}}=1-P_{F,i} \quad (8)$$

$$A_{V_i}=\frac{(\beta-\lambda_1-\mu_1)(\lambda_2\mu_2\gamma)}{\beta[\beta\lambda_1\lambda_2\mu_2\gamma+\lambda_2\mu_2\gamma(\lambda_1+\mu_1)+\gamma\lambda_1\mu_2+\lambda_1\lambda_2\mu_2+\mu_1\lambda_2\gamma+\mu_1\mu_2\lambda_2]} \quad (9)$$

Thus, the total availability of the entire embedded system is given by

$$AV=\sum_{i=1}^{n}A_{V_i}=\frac{(\beta-\lambda_1-\mu_1)(\lambda_2\mu_2\gamma)}{\beta[\beta\lambda_1\lambda_2\mu_2\gamma+\lambda_2\mu_2\gamma(\lambda_1+\mu_1)+\gamma\lambda_1\mu_2+\lambda_1\lambda_2\mu_2+\mu_1\lambda_2\gamma+\mu_1\mu_2\lambda_2]}\times n \quad \textbf{(10)}$$

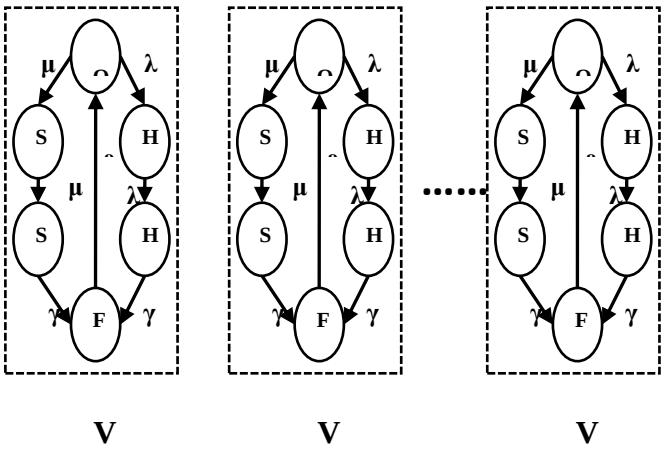

Fig. 2: State Transition Diagram.

V. Numerical Results

In this section, we have provided the numerical results for the total availability of the entire embedded system by varying transition rates and repair rate. For this purpose, the default parameters of the system are fix as λ_1=0.5, λ_2=2.0, μ_1=1.0, μ_2=1.0, γ =1.0, β=2.0 and i=5 for figs 3-6.

Fig. 3 shows that as the values of transition rate λ_1 increases the total availability of the entire embedded system decrease. Similar trend has been observed from fig 4 for the increasing values of rate λ_2. Moreover, it can be easily observed form fig. 5 that total availability (AV) shows the decreasing trend for the increasing values of transition rate μ_1. From fig. 6, we observed that as the values of repair rate (β) takes the higher values, the total availability of the system decreases which is quite observed. On the other hand, AV decreases as we increase the values of γ which can be observed from figs 3-6.

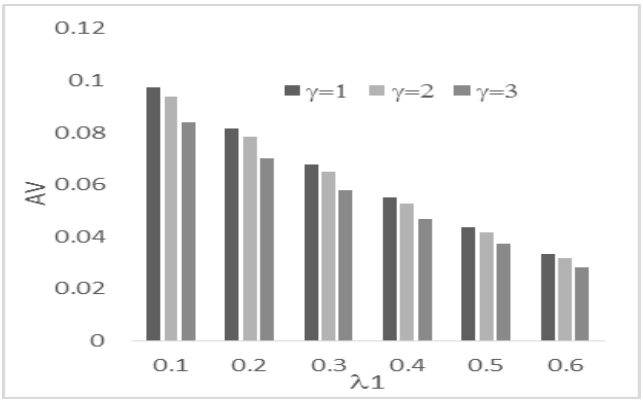

Fig. 3. Effect of λ_1 and γ on Availability.

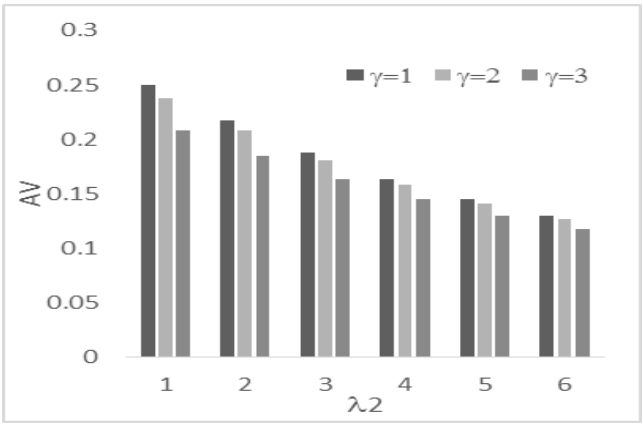

Fig. 4. Effect of λ_2 and γ on Availability.

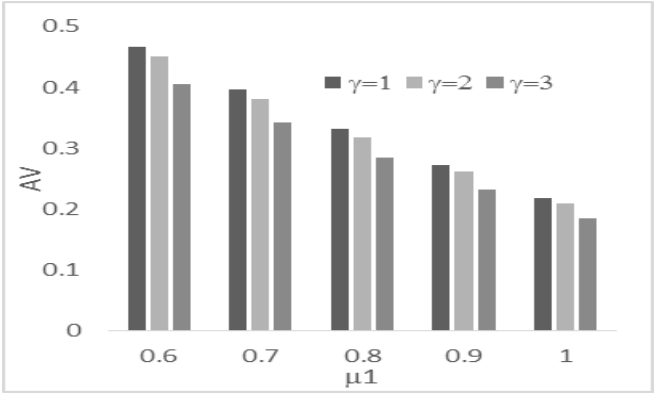

Fig. 5. Effect of μ_1 and γ on Availability.

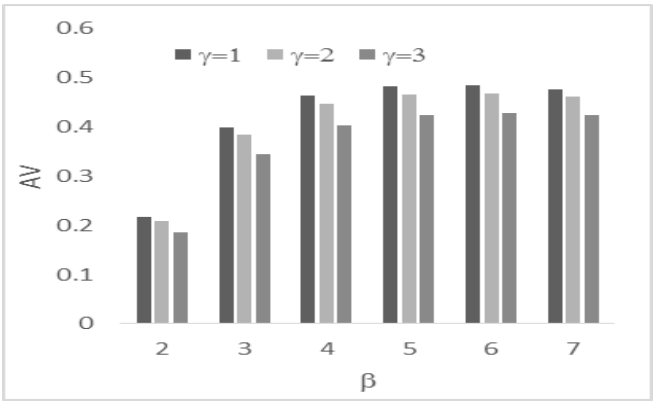

Fig. 6. Effect of β and γ on Availability.

Overall, on the basis of above observation, we finally concluded that the availability of the system can be increased up to certain label if repair rate attains the higher values.

VI. Conclusion

In this present study we have considered an embedded system in which software and hardware units are composed together with their standby units. The system is divided into N- version of subsystems. Also, we have considered the concept of reboot in this paper. The availability of the embedded system is evaluated in the terms of the availabilities of subsystem of N-versions. The proposed work may be applicable to real time embedded system in which the improvement is needed for smooth running of the software applications under techno-economic constraints. We hope that our investigation will be fruitful for the system managers to improve reliability of embedded computer systems.

References

[1] Z. Jelinski and P. B. Moranda,"Software reliability research," Statistical Computer Performance Evaluation ,W. Freiberger Ed., Academic Press, New York, 1972, pp. 465-484.

[2] P. A. Keiller and D. R. Miller, "On the use and he performance of software reliability growth models," Reliability Engineering & System Safety, vol. 32, no. 1-2, 1991, pp. 95-117.

[3] T. Downs and A.Scott, "Evaluating the performance of software reliability models," IEEE Transactions on Reliability, vol. 41, no. 4, 1992, pp. 532-538.

[4] D. Satoh and S.Yamada, "Discrete equations and software reliability growth models," 12th International Symposium on Software Reliability Engineering, 2001, pp.176-184.

[5] S. Yamada, Y. Tamura and M. Kimura, "A software reliability growth model for a distributed development environment," Electronics & Communication in Japan Part 3, vol. 83, 2003, pp. 1446-1453.

[6] Y. Wu, S. Wang and Z. Yu, "Embedded software reliability testing and its practice," International Conference on Computer Design and Applications, vol. 2, 2010, pp. 24-27.

[7] N. Wattanapongsakorn and S. P. Levitan, "Reliability optimization models for embedded systems with multiple applications," IEEE Transactions on Reliability, vol. 53, no. 3, 2004, pp. 406-416.

[8] A. Avizienis and L. Chen, "N-version programming: a fault-tolerance approach to reliability of software operation," Proceedings of FTCS 8, 1978, pp. 3-9.

[9] A. Avizienis and L.Chen, "On the implementation of NVP for fault tolerance," Proceeding of COMPSAC 77, 1977, pp. 149 – 155.

[10] Min Xie, Chengjie Xiong and Szu-Hui Ng, "A study of N-version programming and its impact on software availability," International Journal of Systems Science, 2013, pp. 1-13.

[11] K. Yamakita, H. Yamada, and K. Kono, "Phase-based reboot: reusing operating system execution phases for cheap reboot-based recovery," Proceedings of IEEE/IFIP 41st International Conference on Dependable Systems & Networks, 2011, pp. 169-180.