A Model for Reliability Estimation of Software based Systems by Integrating Hardware and Software

Shelbi Joseph
Division of Information Technology
Cochin University of Science and Technology (CUSAT) cochin
Kerala, INDIA

Shouri P.V
Department of Mechanical Engineering
Model Engineering College Cochin
Kerala, INDIA

Jagathy Raj V. P
School of Management Studies,
Cochin University of Science and Technology (CUSAT) cochin
Kerala, INDIA

ABSTRACT
Software reliability engineering has gained considerable momentum in the recent past. A variety of models are available for the evaluation of software reliability. However the models, as such, cannot be applied for software based systems as any given system is comprised of both hardware and software. The present paper brings up an algorithm for determining reliability of software based systems by integrating both hardware reliability and software reliability. The open source software data is made use of and the methodology for evaluating the software reliability involves identifying a fixed number of packages at the start of the time and defining the failure rate based on the failure data for these preset number of packages. The defined function of the failure rate is used to arrive at the software reliability model. The hardware reliability is obtained using constant hazard model.

General Terms
Software Engineering, Open Source Software, Software Reliability.

Keywords
Failure rate, hardware, reliability, software .

1. INTRODUCTION
Reliability can be defined as the probability that an item can perform a required function for a specified period of time under the specified operating conditions [1-5]. Along with the increasing of the demand of the software system in our society, the damage caused by the failure of the software becomes more serious than ever before [6]. So it is important that accurate deterministic methods be developed for estimation of reliability of software based systems. Moreover, as the software reliability problems are predominant during the developmental stage, the reliability model should be developed by considering this aspect.

Software reliability is defined as the probability of failure-free software operation for a specified period of time in a specified environment [7, 8]. Open Source Software (OSS) has attracted significant attention in recent years [9]. It is being accepted as a viable alternative to commercial software [10]. OSS in general refers to any software whose source code is freely available for distribution [11]. However the OSS development approach is still not fully understood [12]. Reliability estimation plays a vital role during the developmental phase of the open source software. In fact, once the package has stabilized (or developed) then chances of further failure are relatively low and package will be more or less reliable. However, during the developmental stage failures or bug arrival are more frequent and peaks at the code inspection phase and get rather stabilized in the system test phase [13].

Typical software reliability models include Jelinski-Moranda [14], Littlewood [15], Goel-Okumoto [16] and Musa-Okumoto [12]. For software projects that have not been in operation long enough, the failure data collected may not be sufficient to provide a decent picture of software quality, which may lead to anomalous reliability estimates [17, 18] Weibull function is also used for reliability analysis and the function has been particularly valuable for situations for which the data samples are relatively small [19].

Even though a variety of models are available for software reliability prediction, a deterministic model is presently not available. Or in other words, none of these models quantifies reliability. Also while evaluating the reliability of software based systems the hardware reliability is no taken into consideration. The present paper develops a model for evaluation of software reliability based on the available failure data of open source software and also incorporates the hardware reliability.

2. MODEL DEVELOPMENT
Please Evaluation of reliability of a computing or software based system involves computation of hardware reliability and software reliability. Most of the earlier works were merely focused on software reliability with no consideration for hardware part or vice versa. However, a complete estimation of reliability of a computing system requires these two elements (hardware and software) to be considered together and thus demands a combined approach. The present work attempts to develop a systematic procedure for evaluating reliability of open source software based computing system. The model is based on the following assumptions:

1. The hardware components exhibit a constant hazard rate
2. The failures of hardware components as well as the various packages involved in the software are assumed to be independent of each other.
3. The software analyzed is an open source.

4. As the open source software is made up of a very large community the environmental changes are not considered.
5. The total number of packages at the beginning of the analysis is assumed to remain constant and is taken as the initial population.
6. The model is developed for evaluation of the software reliability at the developmental stage and the packages that fail during this period are not further considered. It is further assumed that by the end of developmental stage the bug associated with the failed packages would be eliminated and will be stable further.
7. The reliability of the software is inversely proportional to the number of bugs reported at any point of time.
8. The beginning of the time period after which the bug arrival or failure rate remains constant marks the culmination of the developmental stage and the software will be stabilize.

The reliability estimation involves considering the computing system as two subsystems, one comprising of hardware components and the other the various software modules or packages. These two subsystems are then considered to be connected in series as shown in Fig.

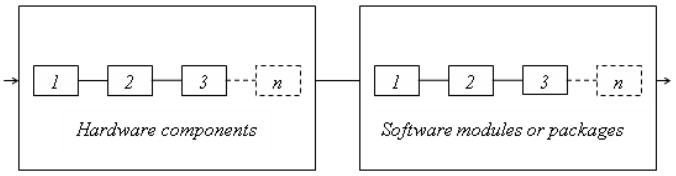

Figure 1. Reliability block diagram for the computing system

Combining the failure rates of both hardware and software the computational system reliability at time *t* can be expressed as:

$$R(t) = R_{hardware} \times R_{software} \quad (1)$$

where, $R_{hardware}$ = Hardware reliability at time *t*

$R_{software}$ = Software reliability at time *t*

Further, if the hardware system components are assumed to have a constant failure rate, the reliability of the hardware part can be expressed as:

$$R_{hardware} = e^{-\sum_{i=1}^{n} \lambda_{h_i} t} \quad (2)$$

where, $\lambda_{h_1}, \lambda_{h_2} \ldots\ldots\ldots\ldots \lambda_{h_n}$ represents the failure rate of different hardware components involved in the system.

Equation (2) assumes that all the hardware components are essential for the success of the system and as such these components are connected in series in the Reliability Block Diagram (RBD).

For software reliability estimation, the methodology involves identifying a fixed number of packages at the start of the time and defining the failure rate based on the failure data for these preset number of packages. The defined function of the failure rate is used to arrive at the software reliability model. For the computation of software reliability a 6 – step algorithm is developed as detailed below:

1. Identify the total initial population. This corresponds to the total number of packages existing at the beginning of the time period. That is, at the start of analysis.
2. Define a time period and find out the bugs reported during this time interval. As the failure can occur anywhere between the time intervals, indicate the reported failures between the time intervals while tabulating the data.
3. Calculate the cumulative failures and thereby the survivors and different points in time.
4. Estimate the failure rate associated with the time intervals by dividing the number of failures associated with the given unit time interval by average population associated with the time interval. Average population associated with a given time interval is the average of survivors at the beginning and end of the time period.
5. Plot the graphs defining the relation between failure rate and time and obtain the equation defining the relation between failure rate and time.
6. Obtain the expression for reliability of the software by substituting the equation of failure rate in equation(3) given as:

$$R_{Software} = e^{-\int_0^t Z(t)dt} \quad (3)$$

where, *Z(t)* if the failure rate equation in terms of time, *t*

3. RESULTS AND DISCUSSION

A total of 1880 packages were available at the start of the analysis as per the details available from the official website of Debain [20]. This is taken as the initial population. A time interval of 1 month is fixed and the bug arrival rate during this interval is noted. The reported errors at different time intervals are given in the Table 1. The observations are taken for 1 year after which the bug arrival is negligible indicating that the software has more or less stabilized.

Table 1. Software Failure data analysis

Time	No. of Failures	Cumulative Failures	Survivors	Failure Rate (per month)
Feb-08		0	1880	
	25			0.013386881
Mar-08		25	1855	
	61			0.033433817
April-08		86	1794	
	340			0.209359606
May-08		426	1454	
	49			0.034277719
Jun-08		475	1405	
	55			0.039927405
Jul-08		530	1350	
	214			0.172164119
Aug-08		744	1136	
	136			0.127340824
Sept-08		880	1000	
	37			0.037697402
Oct-08		917	963	
	40			0.042417815
Nov-08		957	923	
	48			0.048929664
Dec-08		1005	875	
		Average		0.075893525

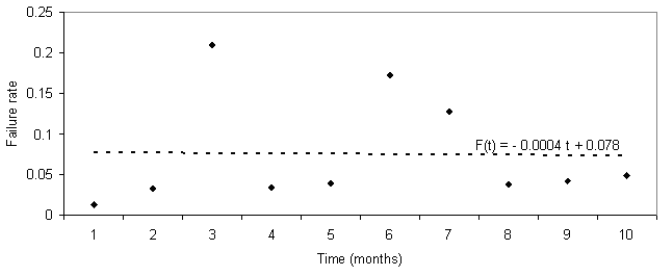

Figure 2. Software failure rate

The variation of software failure rate with respect to time is shown in Fig. 2. It can be seen that after the 8th month onwards the software has somewhat stabilized indicating the completion of developmental phase. The failure model corresponding to the failure rate can be expressed as:

$$Z(t) = -0.0004t + 0.078 \quad (4)$$

The corresponding reliability can be expressed as:

$$R_{Software} = e^{-\int_0^t (-0.0004t + 0.078)\,dt}$$

$$= e^{\frac{0.0004t^2}{2} - 0.078t} \quad (5)$$

Table 2. Hardware component failure rate

Hardware Component No.	Failure Rate (per month)
H_1	0.027778
H_2	0.025
H_3	0.028571
H_4	0.02381
H_5	0.016667
H_6	0.041667
H_7	0.034483
H_8	0.027778

The failure rate of the hardware components are indicated in Table 2. The values of the hardware failure rate can be substituted in equation (2) to get the hardware reliability equation and can be expressed as:

$$R_{hardware} = e^{-\sum_{i=1}^{n} \lambda_{h_i} t}$$

$$= e^{-0.225753t} \quad (6)$$

Thus the reliability of the computing system R(t) at any given time will be the product of equations (5) and (6) and can be expressed as:

$$R(t) = e^{\frac{0.0004t^2}{2} - 0.078t} \times e^{-0.225753t}$$

$$= e^{\frac{0.0004t^2}{2} - 0.303753t} \quad (7)$$

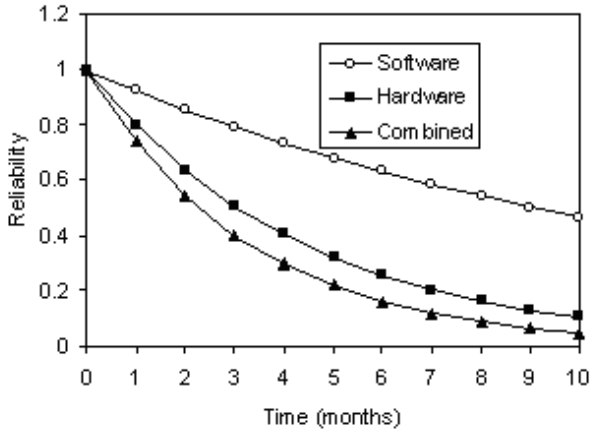

Figure 3. Variation of reliability with time

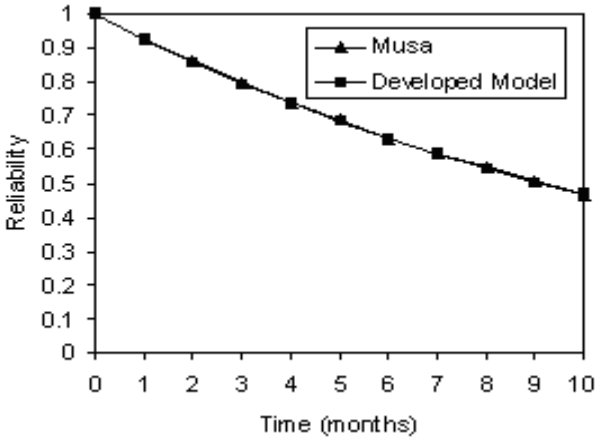

Figure 4. Comparison of developed and Musa model

Fig.3 shows the variation of reliability with time for the hardware part and software part separately and also the reliability values when the software and hardware are considered together. It is evident that if hardware and software are not treated together in the reliability analysis, the predicted values will be an overestimate and will be far from realistic. A measure of the error involved in the reliability calculations can also be obtained by comparison. Fig. 4 is a comparison of the software reliability obtained using the developed model and the conventional Musa model. It can be seen that the variation is virtually nil.

4. CONCLUSIONS

A new method for estimation of reliability of software based systems was developed. The methodology provides a far more realistic estimate of reliability in comparison with the traditional methods which focuses either on hardware or software alone rather than integrating these elements. The concept is very much in accordance with the systems approach. The developed method could prove to be a very effective tool for reliability analysis of the software based systems.

5. REFERENCES

[1] Kumar D, Klefsjo B, Kumar U. Reliability analysis of power-transmission cables of electric loaders using a proportional-hazard model. J Reliability engineering & system safety 1992; 37:217–22.

[2] Goel HD, Grievink J, Herder PM, Weijnen MPC. Integrating reliability optimization into chemical process synthesis. J Reliability Engineering and System Safety 2002; 78:247–258.

[3] Patrick DTO. Practical reliability engineering, 4th ed. England: John Wiley & Sons Ltd; 2002.

[4] Charles EE. Reliability and maintainability engineering.1st ed. New Delhi: Tata McGraw-Hill Publishing Company Ltd; 2000.

[5] Srinath LS. Reliability engineering. 3rd ed. New Delhi:Affiliated East West Press;1991.

[6] Weiguo Li,Yongfu Wang, He Huang A new model for software reliability Fifth international Joint Conference on INC, IMS and IDC, 2009 IEEE.

[7] Michael R Lyu Software Reliability Engineering: A Roadmap, Future of Software engineering (FOSE 07) May 2007 IEEE.

[8] J.D. Musa and A. Iannino. Software reliability modelling-accounting for program size variation due to integration or design changes. Proceedings of the 1981 ACM workshop/symposium on Measurement and evaluation of software quality, 1981. pp 129-130.

[9] Ying ZHOU,Joseph Davis. Open Source Software reliability model: an empirical approach, ACM 2005.

[10] Sharifah Mashita Syed-Mohamad,Tom McBride. A comparison of the Reliability Growth of Open Source and In-House Software. 2008 IEEE 15th Asia-Pacific Software Engineering Conference

[11] Obra Rahmani,Harvey Siy, Azad Azadmanesh. An Experimental Analysis of Open Source Software Reliability. Department of Defense/Air Force Office of Scientific Research

[12] Lars M.Karg, Michael Grottke, Arne Beckhausa . Conformance Quality and Failure Costs in the Software Industry: An Empirical Analysis of Open Source Software. 2009 IEEE.

[13] Kan H.S. Metrics and models in software quality engineering, 2nd edition, Addison-Wesley(2003).

[14] S.P. Leblanc, P.A.Roman Reliability Estimation of Hierarchical Software Systems. 2002 Proceedings annual reliability and maintainability symposium.

[15] J D Musa and K. Okumoto. A logarithamic Poisson execution time model for software reliability measurement 7th international conference on Software Engineering(ICSE),1984, pp. 230-238.

[16] H. Pham, Software Reliability Spriner-Veriag , 2000.

[17] Z. Jelinski and P.B. Moranda, Software Reliability research, in Stastistical Computer Performance Evaluation, W. Freiberger, Ed. New York: Academic Press, 1972, pp.465-484.

[18] B. Littlewood and J.L. Verrall, A Bayesian reliability growth model for computer software, Applied Stastistics, Vol 22, 1973, pp 332-346.

[19] A. L. Goel and K. Okumoto, A time-dependent error-detection rate model for software reliability and other performance measure, IEEE Transactions on Reliability, vol. R-28, 1979, pp. 206-211.