ISSN : 2230-7109 (Online) | ISSN : 2230-9543 (Print)

Software Reliability Measurement using Sequential Bayesian Approach

[1]D.Gowtham Chakravarthy, [2]K.Aravindhan
[1,2]Dept. of CSE, SNS College of Engineering, coimbatore, Tamilnadu, India

Abstract
This paper presents a model for sequential Bayesian techniques for the software reliability engineering with respect to predicting reliability and other elements of software quality such as size, complexity and statistical models that can be used for reliability prediction for the analysis of failure data. This technique and the proposing model evaluates for integrating approach as well as all the types of historical information and opinion from experts in the form of distribution with respect to parameters.

Keywords
Software Reliability Engineering (SRE), sequential Bayesian technique probabilistic models, predicting reliability, operational profile

I. Introduction

Software reliability Engineering is focused on engineering techniques for developing and maintaining software systems whose reliability can be evaluated. Software reliability is defined by the probability of failure free operations for a specified period of time with specific environment [4]. One of the important attributes of software reliability, software quality together with 8 dimensions named functionality, usability, performance, serviceability, capability; install ability, maintainability, and documentation. Software reliability is hard to achieve since the complexity of software tends to be hard. As a result, reliability places an essential role in customer satisfaction for most IT related companies and other organization.

The main objective of the paper is to determine the reliability of software. The requirements specification defines operations interface perform quality assurance requirements of the reliability of software.

The paper describes the entire design constraints that are to be considered when the system is to be designed and other factors which are related to requirements of the software in full description. This particular paper is to focus on analysis of software reliability using Bayesian technique [1, 5, 7-8, 11].

II. Software Reliability

Fig. 1 shows an SRE framework in current practice [1]. First reliability objective is determined from the customer view point to maximize customer satisfaction, with diagram.

The software reliability [9-10], provides real time information about the software that how much it is reliable on client needs & how much he can be dependable on it. The Product functions are more or less the same as described in the product perspective.

III. Reliability Topology

This is the relationship between failures of an individual function to the failure of the aggregate system. The software functions or operations are described as series topology [The failure of the function results is the failure of the entire system]. The software fault tolerant techniques are one of the major approaches to highly reliable software for survives. The failure has two different fault tolerant techniques namely single and multi version technique. Here in this paper, we have not described in depth of the software fault tolerance.

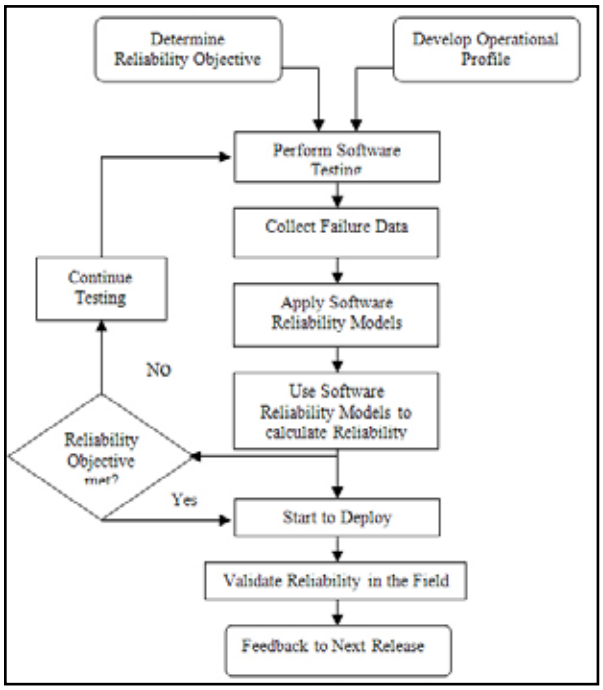

Fig. 1: Software Reliability Engineering Process Overview

IV. Types of Faults

The following types of faults are being found while determining the reliability of software.

Table 1:

Activity introducing fault	Fault type or root cause
Requirements	Missing requirements. Misinterpreted requirements. Requirements not clear. Changed requirements. Conflicting requirements.
Design	Design not to requirements. Missing design. Top level design logic. Low level design logic. Design not robust.
Code	Code not implemented to design. Code not implemented to requirements. Missing code. Initialization error. Storing error. Mismatched parameters. Math operations not robust. I/O operations not robust. Memory errors. Domain errors.
Maintenance & Corrective actions	New fault generated in maintenance.

V. Operational Profiles

The operational profiles describes the reliability of the software based products depends on both computer and other external elements involved in it. The operational profile is a quantitative characterization in any software reliability engineering applications. The operational profile is a set of independent operations that a software system performs with respect to associated probabilities. The five operating profiles are as follows.

1. Find the customer profile
2. Establish the user profile
3. Define the system-mode profile
4. Determine the functional profile
5. Determine the operational profile itself

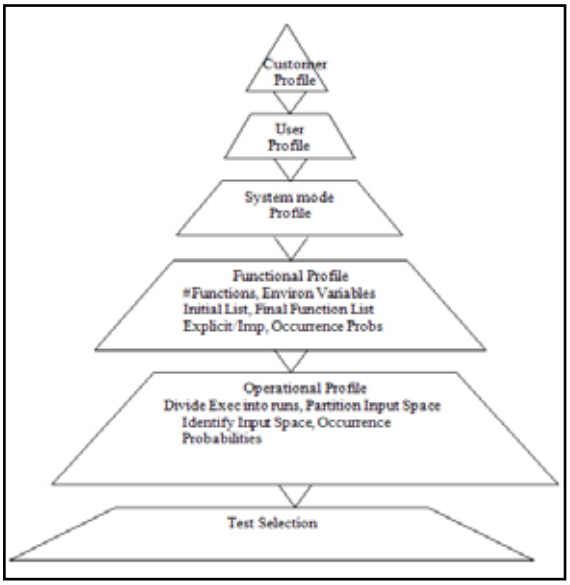

Fig. 2: Operational Profile Development

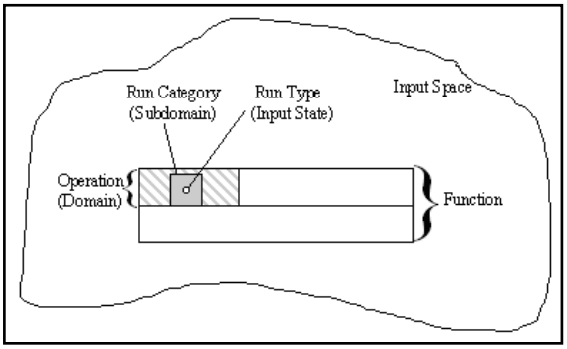

Fig. 3: Operational Elements

Fig. 3, shows the elements involved in determining operational profiles from functions. A function may comprise several operations. In turn, operations are made up of many run types. Grouping run types into operations partitions the input space into domains. A domain can be partitioned into sub domains, or run categories. To use the operational profile to drive testing, first choose the domain that characterizes the operation, then the sub domain that characterizes the run category, and finally the input state that characterizes the run.

VI. Bayesian Diagram-An Approach for Software Reliability Measurement

A Bayesian Net (BN) [2], is a directed acyclic graph (such as the example shown in fig. 4, where the nodes represent random variables and the directed arcs define causal influences or functional relationships. Nodes without parents (such as the "Probability of finding defects" and "Defects in" nodes in fig. 4) are defined through their prior probability distributions. Nodes with parents are defined through Conditional Probability Distributions (CPDs). For some nodes, the CPDs are defined through deterministic functions of their parents (such as the "Defects out" node in fig. 4, others (such as the "Defects found" node in fig. 4, are defined as standard probability distribution functions.

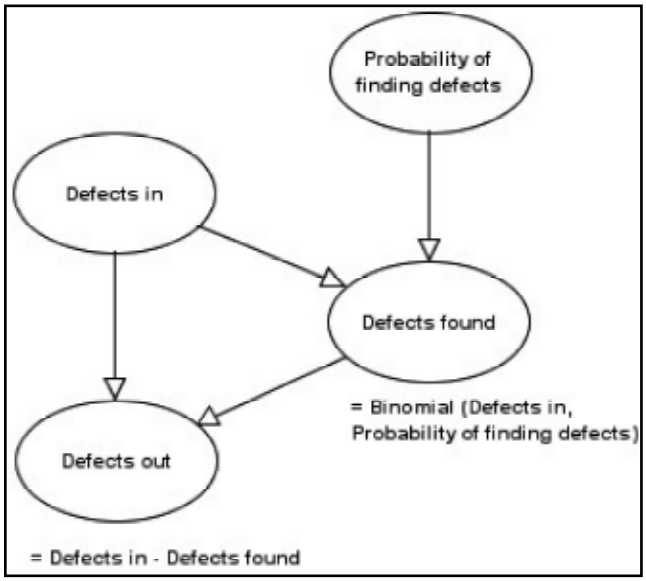

Fig. 4: An Example of a Bayesian Network

Conditional Independence (CI) relationships are implicit in the directed acyclic graph: All nodes are conditionally independent of their non descendents given their parents. This general rule makes it unnecessary to list CI relationships explicitly.

As explained by Fenton and Neil [6], BN models have several advantages over regression-based models. BNs do not rely on point values of parameters that have been derived through some "best fit" procedure. Instead, the whole distribution of a variable is included. Similarly, BN models do not just predict a single value for a variable; they predict its probability distribution. By taking the marginal distributions of variables of interest, we get a ready-made means of providing quantitative risk assessment.

When a variable is actually observed, this observation can be entered into the model. An observation reduces the marginal probability distribution for the observed variable to a probability of 1 for the observed state (or a small interval containing the value in the continuous case—in practice, the discretization chosen must be a compromise between the accuracy of the evidence added and computational feasibility) and zero otherwise. The presence of an observation updates the CPD of its children and, through Bayes theorem, the distributions of its parents. In this way, observations are propagated recursively through the model. BN models can therefore update their beliefs about probable causes and so learn from the evidence entered into the model. More information on BNs and suitable propagation algorithms can be found in [2-3].

A sequential Bayesian technique [5, 7-8, 11] is a posteriori estimation procedure based on Bayesian approach is presented here. The Bayesian approach is one of the types where prior information can be used at the maximum. Let us consider a model equation

$$Y = B_0 + \sum_{j=1}^{q-1} B_i X_i + \varepsilon. \quad (1)$$

The equation for the Bayesian estimation of the model parameters,

$\hat{B}$ is given as

$$\hat{B} = M + PXTQ-1(Y - XM), \quad (2)$$

Where P is the covariance matrix of estimators (q x q), given as

$$P= (X^TQX^{-1}+V^{-1}) \quad (3)$$

$\hat{B}$: Estimated parameter vector (q x 1) M: mean value of parameter vector (q x 1) known from the prior information

X: independent variable matrix (n x 1)

V: covariance matrix of B known from prior information

Q: covariance matrix of errors.

Substituting

$$Bi = Bi+1, M = Mi, Y = Yi+1, P = Pi+1, V = Pi,$$
$$X = Xi+1 \text{ and } Q = Ci+1$$

We get the recursive form of equation (2) and (3). Here C is a m x m diagonal covariance matrix of error and m is the number of observations. Substituting the above expressions in equation (2) and (3) we get

$$B_{i+1}=B_i + P_{i+1}X^T_{i+1}C_{i+1}(Y_{i+1}-X_{i+1}B_i) \quad (4)$$

$$P_{i+1} = (X^T_{i+1}C^{-1}_{i+1}X_{i+1} + P^{-1}_i)^{-1} \quad (5)$$

From matrix inversion theorem we know that,

$$(A + BCD)^{-1} = A^{-1} - A^{-1}B (C^{-1} + DA^{-1}B)^{-1}DA^{-1}$$

Hence equation (5) may be written as follows

$$P_{i+1} = P_i - P_iX^T_{i+1}(X_{i+1}P_iX^T_{i+1} + C_{i+1}) \quad (6)$$

Let $R = P_iX^T_{i+1}C_{i+1}$ & $H = X_{i+1}$

Then the following matrix identity holds

$$(I + RH)^{-1}R = R (I + HR)^{-1}$$

Therefore, substituting the values of R and H we get,

$$(I + P_iX^T_{i+1} C_{i+1} X_{i+1})^{-1}P_iX^T_{i+1} C_{i+1} = P_iX^T_{i+1} C^{-1}_{i+1} (I + X_{i+1} P_iX^T_{i+1} C^{-1}_{i+1})^{-1}$$

$$(X^T_{i+1}C_{i+1}X_{i+1} + P^{-1}_i)^{-1}P_iX^T_{i+1}C_{i+1} = P_iX^T_{i+1}C^{-1}_{i+1}(I + X_{i+1}P_iX^T_{i+1}C^{-1}_{i+1})^{-1}$$

$$(X^T_{i+1}C^{-1}_{i+1}X_{i+1} + P_i)^{-1}P^{-1}_i X^T_{i+1}C^{-1}_{i+1}= P_iX^T_{i+1}C^{-1}_{i+1}C_{i+1}(C_{i+1} + X_{i+1}P_iX^T_{i+1})^{-1}$$

$$P_{i+1}X^T_{i+1}C^{-1}_{i+1}=P_iX^T_{i+1}(X_{i+1}P_iX^T_{i+1}+ C^{-1}_{i+1})^{-1} \quad (7)$$

Substituting equations (6) & (7) in (4) & (5) we get,

$$A_{i+1} = P_iX^T_{i+1} \quad (8)$$
$$D_{i+1} = C_{i+1} + X_{i+1}A_{i+1} \quad (9)$$
$$K_{i+1} = A_{i+1}D_{i+1} \quad (10)$$
$$E_{i+1} = Y_{i+1} - X_{i+1}B_i \quad (11)$$
$$B_{i+1} = B_i + K_{i+1}E_{i+1} \quad (12)$$
$$P_{i+1} = P_i - K_{i+1}A^T_{i+1}. \quad (13)$$

Equations (8) to (13) are the governing equations for the sequential estimation procedure of the parameters. If the number of observations is one then no matrix inversion is involved and the computation becomes efficient. Thus for one observation, equation (8) to (13) may be rewritten as follows:

$$A_{i+1} = \sum P_{uk,i}X_{k,i+1} \quad (14)$$
$$D_{i+1} = \sigma^2_{i+1} + \sum X_{k,i+1}A_{k,i+1} \quad (15)$$
$$K_{u,i+1} = \frac{A_{u,i+1}}{D_{i+1}} \quad (16)$$
$$E_{i+1} = (Y_{i+1} - \sum X_{k,i+1}B_{k,i}) \quad (17)$$
$$B_{u,i+1} = B_{u,i} + K_{u,i+1}E_{i+1} \quad (18)$$
$$P_{uv,i+1} = P_{uv,i} - K_{u,i+1}A_{v,i+1}, \quad (19)$$

Where u = 1, 2, 3,. . . q, v = 1, 2, 3, . . . , q, q is the number of parameters and σ^2_{i+1} is the variance of Y_{i+1}. Here in equation (15) S is used instead of σ^2_{i+1} to denote the error variance obtained from linear regression method. So equation (15) becomes

$$D_{i+1} = S + \sum X_{k,i+1}A_{k,i+1}. \quad (15\text{-A})$$

VII. Estimation Of Model Parameters

In the following paragraphs the description of the model is followed by the parameter estimation using proposed algorithm.

Let $X_t = X^{\theta}_{t-1\delta}$ where θ is constant and values of θ > 1 mean growth of reliability and θ < 1 means decay of reliability. δ is the error due to some uncertainty in power law. Taking natural logarithm on both sides we get

$$\log X_t = \theta \log X_t - 1 + \log \delta \quad (20)$$
$$\text{or } Y_t = \theta Y_t - 1 + B_1, \quad (21)$$
$$\text{Where } Y_t = \log X_t, B_1 = \log \delta.$$

To apply the above-mentioned algorithm for general sequential procedure given in equations (14) to (19) the expression (21) becomes

$$Y = B_1X_1 + B_2X_2,$$

Where Y is Y_t, $B_1 = \log \delta$, $B2 = \theta$, $X_2 = Y_t - 1$ and X_1 is a dummy variable taking a constant value 1. Here t denotes the stage of testing and Xt denotes the time between failures.

VIII. Software Reliability Prediction

Software reliability predictions [2, 10] are made during the software development phases that precede software system test, and are available in time to feed back into the software development process. The predictions are based on measurable characteristics of the software development process and the products produced by that process.

The final outcomes of a software reliability prediction include:

- Relative measures for practical use and management.
- A prediction of the number of faults expected during each phase of the life cycle.
- A constant failure rate prediction at system release that can be combined with other failure rates.

Fig. 5, shows the software reliability prediction process. Product and process metrics are collected and used to predict the initial failure rate and fault content. From these quantities, the reliability growth model parameters are predicted, and then the growth model is used to obtain estimates of the test time and resources needed to meet reliability objectives.

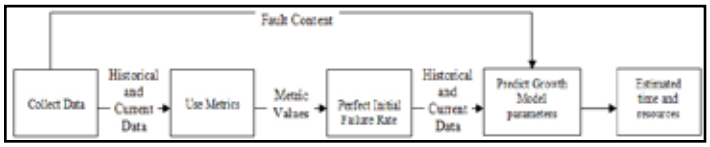

Fig. 5: Software Reliability Prediction Procedure

IX. Software Reliability Growth Testing

Software reliability growth testing [12, 22] takes place during the software system test phase, after the software has been fully integrated. During growth testing, the software is executed in an environment with inputs that most closely simulate the way the software is expected to be used in the field. In particular, the inputs are randomly selected in accordance with the software's operational profile. The quality of testing is directly related to reliability growth and is a function of various system level tests that validate the software from more than one perspective.

System tests can validate domains, paths, states, transaction flow, error handling, etc. The quality of testing is also related to testing the functionality that is executed most often by end user, most critical to end user, and most error prone. An operational profile associates each input state or end-user function with a probability of occurrence. Testing according to the operational profile is

efficient with respect to failure intensity reduction, because it reveals those faults that the user is most likely to encounter in use, those faults that contribute most to the program failure rate. When a failure is observed, the execution time, among other information, is recorded. The observed failure times are used as input to a statistical estimation technique that determines the parameters of the software reliability growth model. This way, the current reliability can be measured and the future reliability can be forecasted. Fig. 6 depicts a failure intensity curve.

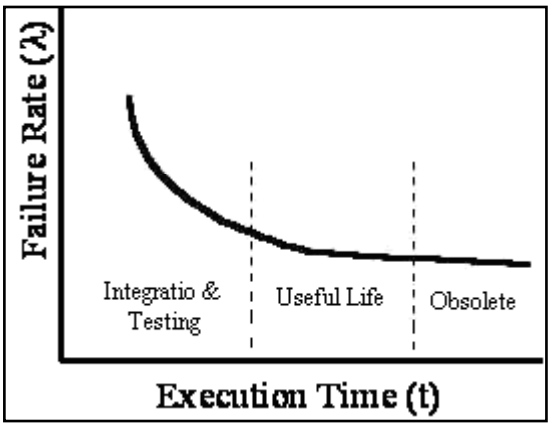

Fig. 6: Software Failure Intensity Curve

X. Conclusion

This research paper attempts to focus on analysis of reliability of software using Sequential Bayesian Technique. The document describes the design constraints that are to be considered when the system is to be designed, and other factors necessary to provide a complete and comprehensive description of the requirements for the software.

References

[1] Basu, S., Ebhrahimi, N.,"Bayesian software reliability models based on martingale processes", Technometrics, 45, pp. 150–158, 2003.

[2] F.V. Jensen,"Bayesian Networks and Decision Graphs", Springer-Verlag, 2001.

[3] S.L. Lauritzen, D.J. Spiegelhalter,"Local Computations with Probabilities on Graphical Structures and Their Application to Expert Systems (with Discussion)", J. Royal Statistical Soc. Series B, Vol. 50, No. 2, pp. 157-224, 1988.

[4] "Institute of Electrical and Electronics Engineers", ANSI/IEEE Standard Glossary of Software Engineering Terminology, IEEE Std. 729-1991, 1991

[5] Jeske, D., Qureshi, M., Muldoon, E.,"A Bayesian methodology for estimating the failure rate of software", International Journal of Reliability, Quality, and Safety Engineering, 7, pp. 153-168, 2000.

[6] N.E. Fenton, M. Neil,"A Critique of Software Defect Prediction Models", IEEE Trans. Software Eng., Vol. 25, No. 4, pp. 675-689, 1999.

[7] Kuo, L., Yang, T.Y.,"Bayesian computation for nonhomogeneous Poisson processes in software reliability", Journal of the American Statistical Association, 91, pp. 763-773.

[8] Littlewood B, Verrall J. L.,"A Bayesian reliability growth model for computer software", Appl.Statist. 22: pp. 332–346, 1973.

[9] Musa, J. D.,"A theory of software reliability and its application", IEEE Trans. Software Eng. SE-1, pp. 312–327.

[10] Musa, J. D., Iannino A. Okumoto, K.,"Software reliability easurement, Prediction, Application", McGraw-Hill Int. Ed., 1987.

[11] Schick, G. J.,"Wolverton R W 1978 An analysis of competing software reliability model", IEEE Trans. Software Eng. SE-4, pp. 104–120.