Research Paper
Impact Factor: 3.567
Peer Reviewed Journal

IJMDRR
E- ISSN –2395-1885
ISSN -2395-1877

SPC TO BURR TYPE XII SOFTWARE RELIABILITY- A STUDY

Dr. B. RamaDevi
Assistant Professor in Computer Science & Engineering,Vignan's Lara Engineering College Guntur, Andhra Pradesh, Iindia.

Abstract
Today, science and technology require high performance hardware and high quality software in order to make improvements and achieve breakthroughs. Software Reliability is an useful measure in planning and controlling the resources during the development process so that high quality software can be developed. Software Reliability was defined as the probability of software not causing failure of a system for a limited duration under specified conditions. Even though the definition looks very simple it constitutes a wide range of research activities with different sub activities. The sub activities are grouped in to number of fields; they are technological assessment of software reliability, quality concern of activity, management activity of project and selection of suitable software. There are many probabilistic and statistical approaches to modeling software reliability. Non Homogenous Poisson Process (NHPP) is a general class of well developed stochastic process model in reliability engineering.

Key words: NHPP, Burr type XII, SPC.

Introduction

Non-Homogenous Poisson Process (NHPP)

The main issue in the NHPP model is to determine an appropriate mean value function to denote the expected number of failures experienced up to a certain time. An NHPP is a realistic model for assessing software reliability and has a very interesting and useful interpretation in debugging and testing the software. There are two main types of software reliability models: the deterministic and the probabilistic. . Performance measures of the deterministic type are obtained by analyzing the program texture and do not involve any random event. Two well-known models are: McCabe's Cyclomatic complexity metric (McCabe, 1976) and Halstead's software metric (Halstead, 1977). The probabilistic model represents the failure occurrences and the fault removals as probabilistic events. The probabilistic software reliability models can be classified into different groups (Pham, 2000) such as, Error seeding, Curve fitting, Failure rate, Reliability growth, Markov structure, Time-series and NHPP. In this thesis, NHPP type of software reliability models and methods are used for estimating software reliability in the model under consideration in the paper is the Burr Type XII.

Objectives of the Study

1. To know the importance of Non-Homogenous Poisson Process (NHPP) models in software reliability**,** and
2. To offer appropriate suggestions for better utilisation of Burr Type XII models in software reliability.

NHPP Models- Discussions

The NHPP group of models provides an analytical framework for describing the software failure phenomenon during testing. These models are usually based upon various debugging scenarios, and can catch quantitatively typical reliability growth observed in the testing phase of software products. The NHPP based SRGMs are proved to be quite successful in practical software reliability engineering (Musa *et al*., 1987). Many of the SRGMs assume that each time a failure occurs, the fault that caused it can be immediately removed and new faults are not introduced. It is usually called perfect debugging. Imperfect debugging models have proposed a relaxation of the above assumption (Pham, 1993). If 't' is a continuous random variable with probability density function

(pdf): $f(t,\theta_1,\theta_2,\dots,\theta_k)$, and cumulative distribution function (cdf): $F(t)$. Where, $\theta_1,\theta_2,\dots,\theta_k$ are k unknown constant parameters. The mathematical relationship between the pdf and cdf is given as: $f(t) = F'(t)$

Let $N(t)$ be the cumulative number of software failures by time 't'. A nonnegative integer-valued stochastic process $N(t)$ is called a counting process, if $N(t)$ represents the total number of occurrences of an event in the time interval $[0,t]$ and satisfies these two properties.

1. If $t_1 < t_2$, then $N(t_1) \le N(t_2)$
2. If $t_1 < t_2$, then $N(t_2) - N(t_1)$ is the number of occurrences of the event in the interval $[t_1, t_2]$.

Where $m(t)$ is a finite valued non-negative and non-decreasing function of '*t* ' called the mean value function, such a probability model for *N* (t) is said to be an NHPP model.

The mean value function $m(t)$ is the characteristic of the NHPP model. There are two major classes of $m(t)$ used to describe different processes: increasing concave and S-shaped models (Ohba, 1984b; Yamada *et al*., 1984). A concave $m(t)$ describes the fault detection process with exponential decreasing intensity. S-shaped m*(t)* describes fault detection process with increasing-then-decreasing intensity. The derivative of $m(t)$ is called the failure intensity function λ(t) which is proportional to the residual fault content. In NHPP SRGM, the failure intensity λ(t) is proportional to the residual fault content $[a(t) - m(t)]$ and can be expressed as $\lambda(t) = b(t)[a(t) - m(t)]$.

Where $a(t)$ is the time-dependent fault content function which includes the initial and introduced faults in the program, $b(t)$ is the time-dependent fault detection rate. A constant $a(t)$ implies the perfect debugging assumption. A constant $b(t)$ implies the imperfect debugging assumption (Gokhale and Trivedi, 1999; Pham, 2007).

The NHPP models are further classified into Finite and Infinite failure models. Let 'a' denote the expected number of faults that would be detected if infinite testing time is given in case of finite failure NHPP models. Then, the mean value function of the finite failure NHPP models can be written as: $m(t) = aF(t)$. The failure intensity function λ(t) is given by: $\lambda(t) = aF'(t)$ (Pham,2006) . The model under consideration in the paper is the Burr TypeXII .

There are essentially two types of software reliability models-those that attempt to predict software reliability from design parameters and those that attempt to predict software reliability from test data. The first type of models are usually called "defect density" models and use code characteristics such as lines of code, nesting of loops, external references, input/outputs, and so forth to estimate the number of defects in the software. The second type of model is usually called "software reliability growth models".

Burr Type-XII SRGM

Burr Type XII based software reliability growth model with Interval domain data and study the reliability assessment with Maximum likelihood estimation.

The proposed mean value function $m(t)$ of Burr Type XII model is given by

The mean value function $m(t) = a\left[1-(1+t^c)^{-b}\right]$

The unknown parameters 'a','b' and 'c' are estimated by using New-Raphson method.

Newton Raphson iterative Method:

$b_{n+1} = b_n - \frac{g(b_n)}{g'(b_n)}$, which is substituted in finding 'a'. Where *g b* &*g ' b* are the Partial derivative of Log L w.r.t 'b' and equating to '0'.

g(b)=$\frac{\partial Log\, L}{\partial b} = 0$ and $g'(b) = \frac{\partial^2 Log\, L}{\partial b^2} = 0$

The parameter 'c' is estimated using Newton Raphson iterative Method

$$c_{n+1} = c_n - \frac{g(c_n)}{g'(c_n)}$$

Where $g(c)$ and $g'(c)$ are expressed as follows.

$$g(c) = \frac{\partial Log L}{\partial c}, \qquad g'(c) = \frac{\partial^2 Log L}{\partial c^2} = 0$$

Applicability of SPC

Software reliability growth models (SRGM"s) are useful to assess the reliability for quality management and testing progress control of software development. To improve reliability and quality the execution of software process must be controlled and the choice for monitoring software process is Statistical Process Control. The parameters estimated can be used to monitor the process through SPC concepts and methods over time, in order to verify that the process remains in the state of statistical control. SPC may help in finding assignable causes, long term improvements in the software process. Software quality and reliability can be achieved by eliminating the causes or improving the software process or its operating procedures . Variable control charts are used to control product or process parameters which are measured on a continuous scale. X-bar, R charts are variable control charts.

Control charts

Control charts, also known as Shewhart charts (Nelson, 1984) or process behaviour charts. A control chart is a specific kind of run chart that allows considerable change to be differentiated from the natural variability of the process. They separate common from special variation. They are graphical tools that help people to study the type and amount of variation present in a system. They can help identify special or assignable causes for factors that impede peak performance (Walter A.Shewhart).

A control chart consists of:

1) Data points are either averages of subgroup measurements or individual measurements plotted on the x/y axis and joined by a line. Time is always on the x-axis.
2) The Average or Centre Line is the average or mean of the data points and is drawn across the middle section of the graph, usually as a heavy or solid line.
3) The Upper Control Limit (UCL) is drawn above the centre line and often annotated as "UCL". This is often called the "+ 3 sigma‖ line.
4) The Lower Control Limit (LCL) is drawn below the centre line and often annotated as "LCL". This is called the "- 3 sigma‖ line.

A process is said to be statistically "in-control" when it operates with only chance causes of variation. On the other hand, when assignable causes are present, then we say that the process is statistically "out-of-control". Control charts are capable to create an alarm when a shift in the level of one or more parameters of a distribution occurs. Normally, such a situation will be reflected in the control chart by points plotted outside the control limits or by the presence of specific patterns. The most common non-random patterns are cycles, trends, mixtures and stratification. For a process to be in control the control chart should not have any trend or nonrandom pattern. The selection of proper SPC charts is essential to effective statistical process control implementation and use. The SPC chart selection is based on data, situation and need. Chan et al. proposed a procedure based on the monitoring of cumulative quantity. This approach has been shown to have a number of advantages: it does not involve the choice of a sample size; it raises fewer false alarms; it can be used in any environment; and it can detect further process improvement. Xie et al., proposed t-chart for reliability monitoring where the control limits are defined in such a manner that the process is considered to be out of control when one failure is less than LCL or greater than UCL. The traditional false alarm probability is to set to be 0.27% although any other false alarm probability can be used. The actual acceptable false alarm probability should in fact depend on the actual product or process.

Tu = a(1+t-c)-b = 0.99865
Tc = a(1+t-c)-b = 0.05

Tl = a(1+t-c)-b = 0.00135

These limits when converted to m(tU), m(tC) and m(tL) form will be used to find whether the software process is in control or not by placing the points in Mean value chart. A point below the control limit m(tL) indicates an alarming signal. A point above the control limit m(tU) indicates better quality. If the points are falling within the control limits, it indicates the software process is in stable condition.

IJMDRR
E- ISSN –2395-1885
ISSN -2395-1877

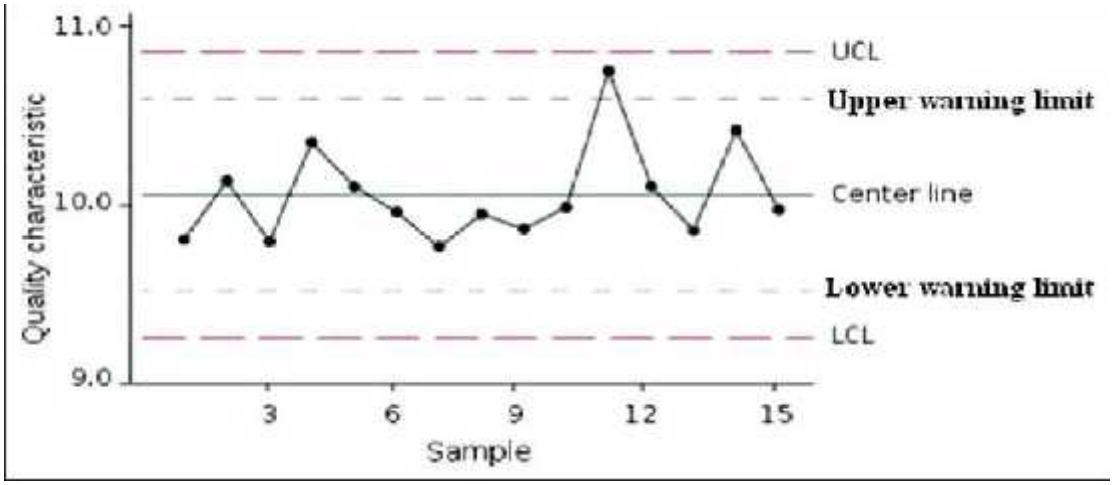

Figure 1.3.1: Example of a shewhart SPC chart.
CL= *Center Line,* LCL= *Lower Control Limit,* UCL= *Upper Control Limit.*

Conclusion

In this paper the author presented Burr type XII software reliability growth model which is primarily useful in estimating and monitoring software reliability, viewed as a measure of software quality. To improve quality of a process the execution of software process must be monitored and controlled. SPC is one such process for monitoring. Control charts are a tool of SPC that help in monitoring through which quality can be improved. The early detection of software failure will improve the software reliability. When the control signals are below the control limit, it is likely that there are assignable causes leading to significant process deterioration and it should be investigated. Hence, we conclude that our control mechanism will give a positive recommendation for its use to estimate whether the process is in control or out of control.

References

1. J. D. MUSA. Software Reliability Engineering.Wiley.1998.
2. J. D. MUSA, A. IANNINO, AND K. OKUMOTO. Software Reliability Measurement Prediction Application. McGraw-Hill, 1987. ISBN 0-07- 044093-X.
3. Pham. H., 2006. “System software reliability”, Springer.
4. Musa, J. D.; Iannino, A.; Okumoto, K. (1987). “Software Reliability - Measurement, Prediction,Application”, New York.
5. Burr (1942), "Cumulative Frequency Functions", Annals of Mathematical Statistics, 13, pp. 215- 232.
6. Ch.Smitha Chowdary, Dr.R.Satya Prasad, K.Sobhana (2015),” Burr Type III Software Reliability Growth Model”, IOSR Journal of Computer Engineering (IOSR-JCE) e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 17, Issue 1, Ver. I (Jan – Feb. 2015),PP 49-54.
7. Kimura, M., Yamada, S., Osaki, S., (1995). “Statistical Software reliability prediction and itsapplicability based on mean time betweenfailures”. Mathematical and Computer Modelling Volume 22, Issues 10-12, Pages 149- 155.
8. Koutras, M.V., Bersimis, S., Maravelakis,P.E., 2007. “Statistical process control using shewart control charts with supplementary Runs rules” Springer Science + Business media 9:207-224.
9. MacGregor, J.F., Kourti, T., 1995. “Statistical process control of multivariate processes”. Control Engineering Practice Volume 3, Issue 3, March 1995, Pages 403-414.
10. Chan, L.Y, Xie, M., and Goh. T.N., (2000), “Cumulative quality control charts for monitoring production processes. Int J Prod Res; 38(2):397-408.
11. Xie. M, T.N Goh and P.Ranjan. (2002). “Some effective control chart procedures for reliability monitoring”, Reliability Engineering and System Safety. 77, 143-150.
12. Swapna S. Gokhale and Kishore S.Trivedi, 1998. “Log-Logistic Software Reliability Growth Model”. The 3rd IEEE International Symposium on High-Assurance Systems.
13. MacGregor, J.F., Kourti, T., 1995. “Statistical process control of multivariate processes”. Control Engineering Practice Volume 3, Issue 3, March 1995, Pages 403-414
14. Dr.R.Satya Prasad,B.RamaDevi,G.Sridevi(2015) “Assessing Burr Type XII software reliability for interval domain data using SPC”Elixer computer engineering79(2015) :30335-30340.