Monitoring Burr Type III Software Quality Using SPC

Smitha Chowdary. Ch [#1], Satya Prasad. R [*2], Sobhana. K [#3]

[#1] Research Scholar
Dept. of Computer Science, Krishna University, Machilipatnam, Andhra Pradesh. INDIA-521001
[1] smithacsc@gmail.com

[*2]Associate Professor
Department of Computer Science & Engineering, Acharya Nagarjuna University, Nagarjuna Nagar, Andhra Pradesh, India, PIN: 522502
[2] profrsp@gmail.com

[#3] Research Scholar
Dept. of Computer Science, Krishna University, Machilipatnam, Andhra Pradesh. INDIA-521001
[3] msobhana@yahoo.com

Abstract— The ability of software in satisfying its functional requirements successfully is measured as software reliability, making it one of the most important characteristics of software quality. Improving software processes employed during the software development life cycle is essential to produce reliable software systems of assured quality. Software Reliability Growth Models (SRGMs) aid software engineers and managers in tracking and measuring the growth in reliability as software is being developed for quality assurance. Software quality is improved by continuously monitoring and controlling the software process. Statistical Process Control (SPC) is the application of statistical methods on software process data presented graphically to quickly and easily identify anomalies that enable the developer to address software failures. In this paper we proposed a SPC mechanism of control charts for time domain data using Burr Type III based on Non Homogenous Poisson Process (NHPP) and parameters are estimated by Maximum likelihood Estimation (MLE) method.

Keywords- Software reliability, Non Homogenous Poisson Process (NHPP), Burr Type III, Maximum likelihood estimation (MLE), Statistical Process Control (SPC), Control Charts, Control Limits.

I. INTRODUCTION

Software reliability is probability of fault free operations provided by the software product under consideration over a specified period of time in a specified operational environment [1]. The system's reliability improves as software design flaws are detected and corrected over time. A number of software reliability models (SRM's) are proposed in order to assess the reliability of a software system.

SRM's can be used to predict the future behaviour of software system from known or assumed characteristics of the software, such as past failure data [2], [3]. The software reliability growth is one of the fundamental techniques to assess software reliability quantitatively [4]. In Software Reliability Growth Model failure specification is taken as input and the reliability of the software is provided as output [5].The models used during the testing phase are called Software Reliability Growth Models (SRGM's). The specifications used must be the number of failures within an interval and the time between two successive failures. There are two types of failure data: time-domain data and interval-domain data. The time-domain data records the individual times at which the failures have occurred. The interval-domain data counts the number of failures occurring during a fixed time period. Always with current existing software reliability models time-domain data provides better accuracy in the estimation of parameters, but involves more data collection efforts [6].

Research activities in software reliability engineering have been conducted, to assess the reliability of software by developing a number of Non Homogenous Poisson Process (NHPP) software reliability growth models. NHPP based SRGM's are generally classified into two groups. The first group contains models that use the execution time (i.e., CPU time) or calendar time and are called continuous time models. The second group contains models that use the number of test cases as a unit of fault detection period and are termed as discrete time models [7]. The discrete time models predict the reliability of software by assuming that the debugging process reduces the future fault occurrence count characterized by its mean value function.

The focus of NHPP model lies in determining an appropriate mean value function to denote the expected number of failures experienced up to a certain point in time. The model, factors in different functional forms of the mean value based on various assumptions [8].

Burr Type III functional form is taken as NHPP's mean value function. Once the analytical solution for m(t) is known for a given model, the model parameters are estimated using the Maximum Likelihood Estimation (MLE) method. MLE methods are versatile, so that they can be applied to most of the models and for different

types of data. Though the methodology for maximum likelihood estimation is simple, its implementation is mathematically strong.

With the ever growing demand to deliver quality software, software development organizations need quality assurance in software processes. Software process monitoring is recommended by several authors by using Statistical Process Control (SPC). Potential pitfalls in the use of SPC were highlighted by some authors [9]. For more than a decade, SPC has been widely used among others, in manufacturing industries for the purpose of controlling and improving processes. Our aim is to apply SPC techniques in the software development process to improve software reliability and quality [10]. It is reported that SPC can be successfully applied to several processes for software development, including software reliability process [11]. The utilization of SPC for software reliability has been the subject of several research studies. A few of these studies are based on reliability process improvement models [12]. Some of the studies furnish guidelines in the use of SPC by modifying general SPC principles to suit the special requirements of software development [13], [14]. In doing so, they zero in on control charts as efficient and appropriate SPC tool specific to software process. SPC is a method of process management through application of statistical analysis, which involves and includes the defining, measuring, controlling, and improving of the processes [15].

II. Related Research for the Model

In this section we present the theory that underlies NHPP models, the SRGMs under consideration and maximum likelihood estimation for complete data which is ungrouped. Assuming that 't' is taken as a continuous random variable with probability density function (pdf): $f(t;\theta_1, \theta_2,\ldots. \theta_k)$ and cumulative distribution function (cdf): F(t) where $\theta_1, \theta_2,\ldots. \theta_k$ are k unknown constant parameters which need to be estimated then the mathematical relationship between the pdf and cdf is given by: f(t) = F '(t). If 'a' is represented as the expected number of faults that would be detected when infinite testing time is given then, the mean value function and the failure intensity function in case of NHPP models can be written as: m(t) = aF(t) and $\lambda(t)$ = aF '(t) where F(t) is a cumulative distribution function[8].

A. NHPP model.

Reliability of software is not deterministic as a faulty program may produce correct output in certain cases. Hence, reliability is best measured probabilistically. Numerous software reliability growth models are available for use according to probabilistic assumptions for NHPP in practical software reliability engineering. To assess reliability of software process, model parameters need to be estimated and can be estimated by using maximum Likelihood Estimate (MLE). NHPP model formulation is described below.

A software system is randomly subject to failures caused by errors present in the system. The fault detection process of software has been widely formulated by using counting process [16]. A counting process $\{N(t), t \geq 0\}$ is said to be a non-homogeneous poisson process which represents the cumulative number of failures by time 't', where 't' is the failure intensity function which is proportional to the residual fault content . Suppose N(t) is known to have a Poisson probability mass function with parameters m(t) i.e.,

$$\Pr\{N(t)=x\} = \frac{[m(t)]^x}{x!} e^{-m(t)}, x = 0,1,2...\infty \quad (1)$$

Then N(t) is called an NHPP. Thus N(t) describes the stochastic behaviour of software failure phenomena. The stochastic failure process by an NHPP is described in various time domain models which differ in the mean value function m(t) .

The mean value function m(t) representing the expected number of faults detected by time 't', is a finite valued, non decreasing, non negative function that is bounded with the boundary conditions

$$m(t) = \begin{cases} 0, t = 0 \\ a, t \to \infty \end{cases}$$

Here 'a' represents the expected number of software failures eventually detected.

B. Burr Type III the proposed model description.

In this paper, we proposed monitoring of software quality using SPC based on Burr Type III model. Burr [17] introduced twelve different forms of cumulative distribution functions for modeling data. The probability density function of a three-parameter Burr type III distribution has the form: $f(t,b,c) = \frac{bct^{bc-1}}{[1+t^c]^{b+1}}$ where b,c are shape parameters. The corresponding cumulative distribution function is: $F(t) = [1+t^{-c}]^{-b}$. It is most notably being used to model insurance claim sizes [18].Therefore mean value function and the failure intensity function of Burr type III NHPP model are as follows

$$m(t)=\int_0^1 \lambda(t)dt = a\left(1+t^{-c}\right)^{-b} \tag{2}$$
$$= aF(t)$$

$$\lambda(t)=\left[\frac{abc}{t^{c+1}\left(1+t^{-c}\right)^{b+1}}\right] \tag{3}$$
$$= af(t)$$

Where t>0, a>0, b>0 and c>0 and 'a' denotes the expected number of faults that would be detached given infinite testing time in case of finite failure NHPP models. In order to assess the reliability of software unknown parameters a, b and c need be estimated by using Newton-Raphson method.

III. PARAMETER ESTIMATION USING MLE.

In order to model data, out of the twelve different forms of cumulative distribution functions introduced by Burr, Burr type III is considered in this paper. The task of building a mathematical model is incomplete until the unknown parameters of the model are estimated and validated on actual software failure data sets. In this section we developed expressions for estimating the parameters of the Burr type III model based on time domain data.

Parameter estimation is given primary importance in software reliability prediction and is achieved by applying a technique of MLE which is the most important and widely used estimation technique. Failure data sets are usually collected in one of two common ways, time domain data and interval domain data. The failure data taken into consideration here is time domain data.

The mean value function and intensity function of Burr type III [19] are given by

$$m(t)=a\left[1+t^{-c}\right]^{-b}, t>0, a, b, c>0 \tag{4}$$

$$\lambda(t)=\left[\frac{abc}{t^{c+1}\left(1+t^{-c}\right)^{b+1}}\right] \tag{5}$$

The constants 'a', 'b' and 'c' appearing in the mean value function m(t) of NHPP and λ(t) intensity function (error detection rate) are parameters of the model. These parameters 'a', 'b' and 'c' are to be known in order to assess the quality of software and they are to be estimated from software failure data sets.

Expressions are now delivered for estimating 'a', 'b' and 'c' for Burr Type III model using the log likelihood function to obtain N independent observations $t_1, t_2, t_3, \ldots, t_n$.

$$LLF=\sum_{i=1}^{n}\log\left[\lambda(t_i)\right]-m(t_n) \tag{6}$$

$$LogL=\sum_{i=1}^{n}\log\left[\frac{abc}{t_i^{c+1}(1+t_i^{-c})^{b+1}}\right]-\frac{a}{\left[1+t_n^{-c}\right]^b} \tag{7}$$

$$LogL=\frac{-a}{\left(1+t_n^{-c}\right)^b}+\sum_{i=1}^{n}\left[\log a+\log b+\log c-(c+1)\log t_i-(b+1)\log\left(1+t_i^{-c}\right)\right] \tag{8}$$

In the above function equation (6) is called the log likelihood function for the given failure data. Values of 'a', 'b' and 'c' that may maximize L are called maximum likelihood estimators (MLEs) and the method is called maximum likelihood (ML) estimation method. Accordingly 'a', 'b' and 'c' would be solutions of the equations.

$$\frac{\partial LogL}{\partial a}=0, \frac{\partial LogL}{\partial b}=0, \frac{\partial LogL}{\partial c}=0, \frac{\partial^2 LogL}{\partial c^2}=0$$

The expressions for m(t), λ(t) given by equations (4) and (5) are substituted in equation (8) by taking logarithms and differentiating them with respect to 'a', 'b', 'c' and equated to zero, after some joint simplification we get

$$\Rightarrow a=n\left(1+t_n^{-c}\right)^b \tag{9}$$

$$\Rightarrow b = \frac{n}{\sum_{i=1}^{n} \log\left(1 + t_i^{-1}\right) - n\log\left(1 + t_n^{-1}\right)} \tag{10}$$

The parameter 'c' is estimated by Newton-Raphson iterative Method using $c_{i+1} = c_i - \frac{g(c_i)}{g'(c_i)}$ where g(c) and g′(c) are expressed as follows

$$\Rightarrow g(c) = \frac{-n\log(t_n)}{1 + t_n^c} + \frac{n}{c} + \sum_{i=1}^{n} \log t_i \left[-1 + \frac{2}{1 + t_i^c} \right] \tag{11}$$

$$\Rightarrow g'(c) = \frac{n(\log t_n)^2 t_n^c}{\left(t_n^c + 1\right)^2} - \frac{n}{c^2} - \sum_{i=1}^{n} \frac{2t_i^c (\log t_i)^2}{\left(t_i^c + 1\right)^2} \tag{12}$$

IV. MONITORING TIME DOMAIN FAILURE DATA SETS USING CONTROL CHARTS OF SPC

Statistical Process Control is an analytical decision making tool that allows monitoring of a process and gives a perspective on when a process is working correctly and when it is not. Variation is present in any process and the key to quality control is observing the variation which helps in deciding when the variation is natural and when it needs correction. A process is said to be statistically "in-control" when it operates with only chance occurrence of variation. On the other hand, when variations are out of admissible bounds, then we say that the process is statistically "out-of-control". [20]

Variation in the process is observed by using the control charts that is one of the seven tools for quality control. Control charts monitor processes to show how the process is performing and how the process and capabilities are affected by changes to the process. They are capable to create an alarm when a shift in the level of one or more parameters occurs. This information is then used to make quality improvements.

There are many charts which use statistical techniques and choosing of the best chart is important for the given data, situation and need [21]. Much effective statistical analysis is provided as there are advances in charts and basic types of advances in charts are the variable and attribute charts that depend on the type of data.

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.

Attribute data is based upon discrete distinction as attributes are characteristics of a process which are stated in terms of good or bad, accepted or rejected, etc. Attribute charts are not sensitive to variation in the process as variables charts. Control charts for attributes are p-charts, c-charts, np-charts, and u-charts.

A procedure based on monitoring cumulative quantity was proposed by Chan et al, [22] that has the following 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.,[23] 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 Lower Control Limit (LCL) or greater than Upper Control Limit (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 [24][25].

There are number of control charts which are commonly used. They differ slightly depending on their data, but all have the same general fundamentals. The four key features of control charts are:

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 Center Line(CL) 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 UCL is drawn above the CL. This is often called the "+ 3 sigma" line.

4) The LCL is drawn below the CL. This is called the "- 3 sigma" line.

The x and y axes should be labeled and a title specified for the chart.

The control chart of this paper is named as Mean value Control Chart and it helps in assessing the software failure phenomena on the basis of the given inter-failure time data.

A. Control Limits for Burr Type III Time Between Failures.

Software system is susceptible to failure at times due to inherent analysis or design problems or inadequate testing. Even if we know that software contains errors, we generally do not know their exact identity. In this paper Burr type III is adapted to time between failures data cumulatively for reliability monitoring and software quality is determined by detecting failures at an early stage.

We compute the software failures process through Mean Value Control chart. The m(t) function of Burr Type III from equation (4) is given as

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

'a', 'b' and 'c' are MLEs of unknown parameters for the considered model. The MLEs values are computed using iterative method for the given cumulative time between failures data. The data set of software errors considered here is acquired from software development project [8]. The data named as AT&T data are summarized in the Table I. In this paper we used numerical conversion data (Failure Time (hours)*0.01) in order to facilitate the parameter estimate [19], [26], [27], [28].

TABLE I. Time Between Failures of Software

Failure index	Inter-failure time	Cumulative Inter-failure time	Failure Time(hours)*0.01
1	5.5	5.5	0.055
2	1.83	7.33	0.0733
3	2.75	10.08	0.1008
4	70.89	80.97	0.8097
5	3.94	84.91	0.8491
6	14.98	99.89	0.9989
7	3.47	103.36	1.0336
8	9.96	113.32	1.1332
9	11.39	124.71	1.2471
10	19.88	144.59	1.4459
11	7.81	152.4	1.524
12	14.6	167	1.67
13	11.41	178.41	1.7841
14	18.94	197.35	1.9735
15	65.3	262.65	2.6265
16	0.04	262.69	2.6269
17	125.67	388.36	3.8836
18	82.69	471.05	4.7105
19	0.46	471.51	4.7151
20	31.61	503.12	5.0312
21	129.31	632.43	6.3243
22	47.6	680.03	6.8003

The Values of Parameter Estimates obtained by ML method for the data set AT&T are

$a = 26.839829$

$b = 1.658692$

$c = 1$

Using 'a' ,'b' and 'c' values we can compute m(t) for T_U, T_L, T_C i.e. UCL, LCL, CL. The control limits can be obtained by assuming an acceptable probability of false alarm of 0.27% [23]. These are calculated by taking the standard values 0.00135, 0.99865 and 0.5.

The limits are given below:

$$T_u = \left[1 + t^{-c}\right]^{-b} = 0.99865$$

$$T_c = \left[1 + t^{-c}\right]^{-b} = 0.5$$

$$T_l = \left[1 + t^{-c}\right]^{-b} = 0.00135$$

TABLE III. Mean Successive Difference of Burr Type III

Failure index	Failure Time(hours)*0.01	M(t)	Successive Difference
1	0.055	0.199922	0.112959
2	0.0733	0.312882	0.196035
3	0.1008	0.508916	6.561219
4	0.8097	7.070136	0.311273
5	0.8491	7.381409	1.111709
6	0.9989	8.493118	0.241995
7	1.0336	8.735113	0.664188
8	1.1332	9.399301	0.70758
9	1.2471	10.10688	1.115878
10	1.4459	11.22276	0.401173
11	1.524	11.62393	0.699934
12	1.67	12.32387	0.505689
13	1.7841	12.82955	0.768647
14	1.9735	13.5982	2.119171
15	2.6265	15.71737	0.001095
16	2.6269	15.71847	2.635528
17	3.8836	18.354	1.148998
18	4.7105	19.50299	0.005528
19	4.7151	19.50852	0.360998
20	5.0312	19.86952	1.169818
21	6.3243	21.03934	0.337798
22	6.8003	21.37714	

These limits are converted to $m(t_U)$, $m(t_C)$ and $m(t_L)$ form and their values are as follows.

$$m(t_u) = 26.8036$$
$$m(t_c) = 0.036234$$
$$m(t_l) = 13.41991$$

These control limits are used to find whether the software process is in control or not by placing the points in Mean Value Control chart shown in Fig.1.

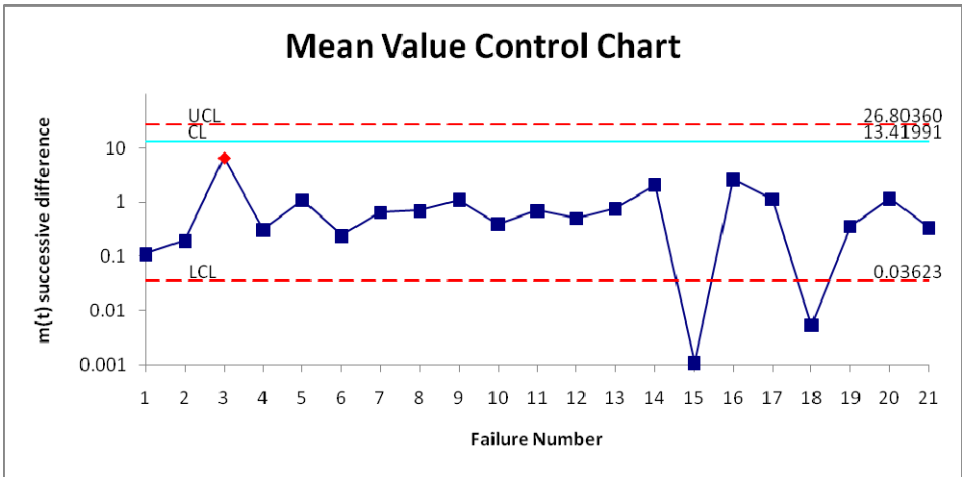

Fig. 1. Mean Value Control Chart.

A point below the control limit $m(t_L)$ indicates an alarming signal. A point above the control limit $m(t_U)$ indicates better quality. If the points are falling within the control limits, it indicates the software process is in stable condition [29]. The mean value control chart shows all the successive differences.

Placing the time between failures cumulative data i.e., m(t) successive differences shown in Table II on y axis and failure number on x axis and the values of control limits on Mean Value Control chart we obtain Fig.1. The failure process is identified by the Mean Value control chart at 15^{th} point (failure number) indicating that the failure data has fallen below $m(t_L)$. It is significantly early detection of failures using Mean Value Control Chart. The software quality is determined by detecting failures at an early stage.

V. Conclusion

Software reliability is an important measure of software quality that quantifies the software failures. Determining software reliability needs collecting of accurate and complete failure data that serves as a measure of software quality. Burr Type III software reliability growth model is used for estimating and monitoring software reliability, viewed as a measure of software quality. Equations of MLE are developed to obtain the MLEs of the parameters based on time domain data.

SPC aids in monitoring the process right from the initial stages of development, leading to early failure detection through its tools. Thus the quality of the software can be improved using SPC's control charts. Estimated parameter values and cumulative time between failures are used for constructing the mean value control chart.

Analysis of Mean Value Control Chart shows that the AT&T data has out of control signals i.e., below the LCL. The adopted method of estimation and the control chart are giving a positive recommendation for their use in finding out preferable or desirable control process.

Mean value control chart detected failure situation at 15^{th} point of Fig.1 which is an out of control situation. Hence this method of model validation is very simple and convenient for practitioners of software reliability as the early detection of software failure will improve the software reliability. In conclusion this model is among the better choices for an early detection of software failures.

References

[1] Marinos, Swapna S. Gokhale Peter N., and Kishor S. Trivedi. "Important Milestones in Software Reliability Modeling". In Proceedings of Software Engineering and Know ledge Engineering (SEKE' 96), Lake Tahoe, NV, pp.345-352.1996.
[2] R. Lai and M. Garg, "A Detailed Study of NHPP Software Reliability Models", Journal of Software, 2012.
[3] Sonia Deswal, Renu Dalal, "A Study of Various Reliability Growth Models". International Journal of Advanced Research in Computer Science and Software Engineering, Volume 4, Issue 4, pp. 1213-1219, April 2014.
[4] Michael R. Lyu, "Handbook of Software Reliability Engineering", IEEE Computer Society Press and McGraw-Hill Book Company, 2005.
[5] Dr. William H. Farr, "A survey of Reliability Modeling and Estimation", Sept1983.
[6] K.B.Misra. "Handbook of Performability Engineering". Springer. 2008.
[7] Omar Shatnawi, "Discrete Time NHPP Models for Software Reliability Growth Phenomenon", The International Arab Journal of Information Technology, Vol. 6, No. 2, April 2009.
[8] Pham. H., 2006. "System software reliability", Springer.
[9] N. Boffoli, G. Bruno, D. Cavivano, G. Mastelloni. "Statistical process control for Software: a systematic approach". ACM 978-1-595933-971-5/08/10. October 2008.
[10] K. U. Sargut, O. Demirors. "Utilization of statistical process control (SPC) in emergent software organizations: Pitfalls and suggestions". Springer Science + Business media Inc. 2006.
[11] Dr. R Satya Prasad, K Ramchand H Rao, Dr. R.R. L Kantham. "Software reliability with SPC". International Journal of Computer Science & Emerging Technologies (E-ISSN: 2044-6004). Volume 2, Issue 2, April 2011

[12] Kumari, K. Sita; Prasad, R. Satya. "Assessing Software Quality with Time Domain Pareto Type II using SPC". International Journal of Computer Applications; Vol. 85, p46. Jan2014.
[13] Burr,A. and Owen ,M. "Statistical Methods for Software quality" . Thomson publishing Company. ISBN 1-85032-171-X. 1996.
[14] Carleton, A.D. and Florac, A.W. "Statistically controlling the Software process". The 99 SEI Software Engineering Symposium, Software Engineering Institute, Carnegie Mellon University. 1999.
[15] Mutsumi Komuro. "Experiences of Applying SPC Techniques to software development processes". ACM 1-59593-085-x/06/0005. 2006.
[16] S. Chatterjee, J. B. Singh, "A NHPP based software reliability model and optimal release policy with logistic–exponential test coverage under imperfect debugging". International Journal of System Assurance Engineering and Management, Volume 5, Issue 3, pp 399-406. September 2014.
[17] Burr, I.W, "Cumulative Frequency Functions", Annals of Mathematical Statistics, Volume13, pp. 215-232, 1942.
[18] Hee-cheul Kim. "Assessing Software Reliability based on NHPP using SPC", International Journal of Software Engineering and its Applications, vol.7, No.6, pp.61-70. 2013
[19] Ch.Smitha Chowdary, Dr R.Satya Prasad, K.Sobhana, "Burr Type III Software Reliability Growth Model," IOSR-JCE Volume17, Issue 1, Jan-Feb 2015.
[20] Ved Parkash, Deepak Kumar, Rakesh Rajoria, "STATISTICAL PROCESS CONTROL", International Journal of Research in Engineering and Technology eISSN: 2319-1163 | pISSN: 2321-7308, Volume: 02 Issue: 08 | Aug-2013.
[21] Ronald P.Anjard. "SPC CHART selection process". Pergaman 0026-27(1995)00119-0Elsevier science ltd.
[22] Chan, L.Y, Xie, M., and Goh. T.N. "Cumulative quality control charts for monitoring production processes". Int J Prod Res; 38(2):397-408. (2000).
[23] M.Xie, T.N. Goh, P. Rajan. "Some effective control chart procedures for reliability monitoring", Elsevier science Ltd, Reliability Engineering and system safety 77.pp143- 150. 2002.
[24] Swapna S. Gokhale and Kishore S.Trivedi. "Log-Logistic Software Reliability Growth Model", The 3rd IEEE International Symposium on High-Assurance Systems. 1998.
[25] Kumari, K.S, Amulya.B, Prasad.R.S, "Comparative study of Pareto Type II with HLD in assessing the software reliability with order statistics approach using SPC". International Conference on Circuit Power and Computing Technologies (ICCPCT), Page(s): 1630 – 1636. IEEE Conference publication. 2014.
[26] N. R. Barraza., "Parameter Estimation for the Compound Poisson Software Reliability Model", International Journal of Software Engineering and Its Applications, http://www.sersc.org/journals/IJSEIA/vol7_no1_2013/11.pdf, vol. 7, no. 1, , pp. 137-148. January 2013.
[27] I. Inayat, M. Asim Noor and Z. Inayat, "Parameter Successful Product-based Agile Software Development without Onsite Customer: An Industrial Case Study", International Journal of Software Engineering and Its Applications,http://www.sersc.org/journals/IJSEIA/vol6_no2_2012/1.pdf, vol. 6, no. 2l, pp. 1-14, April 2012.
[28] Hassan Najadat and Izzat Alsmadi., "Enhance Rule Based Detection for Software Fault Prone Modules", International Journal of Software Engineering and Its Applications, Vol. 6, No.1, pp. 75-86, http://www.serc.org/journals/IJSEIA/vol6_no1_2012/6.pdf. January 2012.
[29] Satya Prasad, R., "Half logistic distribution for software reliability growth model", Ph.D thesis, 2007.

AUTHOR PROFILE

Mrs. Ch. Smitha Chowdary received MCA from Kakatiya University in 2003 and M.Tech., (Computer Science & Engineering) from Acharya Nagarjuna University in 2010. Now she is pursuing Ph.D.,in Computer Science & Engineering from Krishna University as Part-Time Research Scholar under the guidance of Dr. R.Satya Prasad. Currently, she is working as a Lecturer at Post Graduate Centre of P.B.Siddhartha College of Arts & Science, Vijayawada, AP, India. Her research interest lies in Software Reliability Engineering, Data Warehousing and Data Mining.

Dr.R Satya Prasad received Ph.D.degree in Computer Science in the Faculty of Engineering in 2007 from Acharya Nagarjuna University, Andhra Pradesh, India. He received gold medal from Acharya Nagarjuna University for his outstanding performance in master's degree. He is currently working as Associate Professor in the department of Computer Science & Engineering, Acharya Nagarjuna University. He performed various academic roles like practical examiner, project adjudicator, external member of board of examiners for various universities and colleges in and around Andhra Pradesh. He received Dr.Abdul Kalam Life Time Achievement Award for his remarkable achievements in the field of Teaching, Research and Publications. His current research is focused on Software engineering, Image processing & Database Management system. He has published 70 research papers in National & International Journals.

Mrs. K. Sobhana received MCA from Acharya Nagarjuna University in 2005 and M.Tech., (Computer Science & Engineering) from Acharya Nagarjuna University in 2010. Now she is pursuing Ph.D.,in Computer Science & Engineering from Krishna University as Part-Time Research Scholar under the guidance of Dr. R.Satya Prasad. Currently, she is working as a Lecturer at Post Graduate Centre of P.B.Siddhartha College of Arts & Science, Vijayawada, AP, India. Her research interest lies in Software Reliability Engineering and Artificial Intelligence.