ISSN 22773061

MONITORING PARETO TYPE IV SRGM USING SPC

R.Satya Prasad[1], G.Sridevi[2]
[1]Department of CSE, Acharya Nagarjuna University, Nagarjuna Nagar.
profrsp@gmail.com
[2]Department of CSE, Nimra Women's College of Engineering, Vijayawada.
sridevi.gutta2012@gmail.com

ABSTRACT

The Reliability of the Software Process can be monitored efficiently using Statistical Process Control (SPC). SPC is the application of statistical techniques to control a process. SPC is a study of the best ways of describing and analyzing the data and then drawing conclusion or inferences based on available data. With the help of SPC the software development team can identify software failure process and find out actions to be taken which assures better software reliability. This paper provides a control mechanism based on the cumulative observations of Interval domain data using mean value function of Pareto type IV distribution, which is based on Non-Homogenous Poisson Process (NHPP). The unknown parameters of the model are estimated using maximum likelihood estimation approach. Besides it also presents an analysis of failure data sets at a particular point and compares Pareto Type II and Pareto Type IV models.

Keywords

Software Reliability, NHPP, Pareto type IV distribution, Parameter Estimation, Interval Domain data, ML Estimation, Statistical Process Control, Mean value function, Control charts.

Academic Discipline And Sub-Disciplines

Computer Science

SUBJECT CLASSIFICATION

Software Engineering: Software Reliability

TYPE (METHOD/APPROACH)

Research Paper

Council for Innovative Research
Peer Review Research Publishing System
Journal: INTERNATIONAL JOURNAL OF COMPUTERS & TECHNOLOGY
Vol 11, No. 1
editor@cirworld.com
www.cirworld.com, member.cirworld.com

ISSN 22773061

1. INTRODUCTION

Software Reliability is the most dynamic quality characteristic which can measure and predict the operational quality of the software system during its intended life cycle. Software Reliability is the probability of failure free operation of software in a specified environment during specified duration [Musa 1998], Wood[1996], Satya Prasad[2007]. To identify and eliminate human errors in software development process and also to improve software reliability, the Statistical Process Control concepts and methods are the best choice. The NHPP based models are the most important models because of their simplicity, convenience and compatibility. The NHPP based software reliability growth models are proved quite successful in practical software reliability engineering [Musa *et al*., 1987].

SPC is concerned with quality of conformance. SPC can be divided into control charting and process capability study. Control charts provide a means of determining the type of variation (common cause or assignable cause) that is present in a process. Process capability study determines the ability of the "in control" process to produce product which meets specifications. The origin of SPC dates back to the 1920s and 1930s at the Western Electric Company and Bell Telephone Laboratories Walter Shewhart (1891-1967) recognized that variation in a production process can be understood and controlled through the use of statistical methods.

Here SPC concepts and methods are used to monitor the performance of a software process over time in order to verify that the process remains in the state of Statistical control. It helps 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 [4]. The most popular technique for maintaining process control is control charting. The control chart is one of the seven tools for quality control. Software process control is used to secure, that the quality of the final product will conform to predefined standards.

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”. SPC provides a real time analysis to establish controllable process baselines; learn, set and dynamically improve process capabilities; and focus business areas needing improvement. The early detection of software failures will improve the software reliability. 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 [5].

This paper presents Pareto type IV model to analyse the software system using SPC. The layout of the paper is as follows: Section2 describes the formulation and interpretation of the model for the underlying NHPP, Section 3 describes the Pareto type II software reliability growth model, Section 4 describes the proposed Pareto type IV software reliability growth model, Section 5 discusses parameter estimation of Pareto type IV model based on interval domain data. Section 6 describes the techniques used for software failure data analysis for a live data and Section 7 Conclusion. In conclusion it is proved that both models results the failure situation at the same point.

2. NHPP MODEL

There are numerous software reliability growth models available for use according to probabilistic assumptions. The Non Homogenous Poisson Process (NHPP) based software reliability growth models are proved to be quite successful in practical software reliability engineering [1]. NHPP model formulation is described in the following lines.

A software system is subjected to failures at random times caused by errors present in the system.

Let $\{N(t), t > 0\}$be a counting process representing the cumulative number of failures by time‘t’. Since there are no failures at t=0 we have

$$N(0) = 0$$

It is reasonable to assume that the number of software failures during non-overlapping time intervals do not affect each other. In other words, for any finite collection of times $t_1 < t_2 < \cdots t_n$. The n random variables $(t_1), \{N(t_2) -$ $Nt1, \ldots Ntn{-}Ntn{-}1$ are independent. This implies that the counting process {N(t), t>0} has independent increments. Let m(t) represent the expected number of software failures by time’s’. The mean value function m(t) is finite valued, non-decreasing, non-negative and bounded with the boundary conditions.

$$m(t) = 0, t = 0$$

$$= a,\ t \to \infty$$

Where ‘a’ is the expected number of software errors to be eventually detected.

Suppose N (t) is known to have a Poisson probability mass function with parameters m (t) i.e.,

$$P\{N(t) = n\} = \frac{[m(t)]^n . e^{-m(t)}}{n!}, n = 0,1,2 \ldots \infty$$

Then N(t) is called an NHPP. Thus the stochastic behaviour of software failure phenomena can be described through the N(t) process. Various time domain models have appeared in the literature (Kantam and Subbarao, 2009) which describe the stochastic failure process by an NHPP.

3. MODEL DESCRIPTION: PARETO TYPE II SRGM

We consider m (t) as given by

$$m(t) \;\; = a\,[1 \; - \frac{c^b}{(t+c)^b}]$$

Where [m (t)/a] is the Cumulative distribution function of Pareto type II distribution (Johnson *et al.,* 2004).

The parameter estimation of the above mean value function is already derived [7].

Using theses parameters we already proposed a problem namely Statistical Process Control (SPC) which is used for Monitoring the Software Reliability [16].

4. THE PROPOSED PARETO TYPE IV SRGM

We consider $m(t)$ as given by

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

Where [m (t)/a] is the cumulative distribution function of Pareto type IV distribution (Johnson et al, 2004) for the present choice.

$$P\{N(t) = n\} = \frac{[m(t)]^n . e^{-m(t)}}{n!}$$

$$\lim_{n\to\infty} P\,\{N(t) = n\} = \frac{e^{-a}.a^n}{n!}$$

This is also a Poisson model with mean ‘a’.

Let N (t) be the number of errors remaining in the system at time ‘t’

$$N(t) = N(\infty) - N(t)$$

$$\begin{aligned} E[N(t)] \;\; &= E[N(\infty)] - E[N(t)] \\ &= a - m(t) \\ &= \;\; a - a\left[1 - \left[1 + \left(\frac{t}{c}\right)\right]^{-b}\right] \end{aligned}$$

5. PARAMETER ESTIMATION BASED ON INTERVAL DOMAIN DATA

In this section we develop expressions to estimate the parameters of the Pareto type IV model based on interval domain data. Parameter estimation is of primary importance in software reliability prediction.

A set of failure data is usually collected in one of two common ways, time domain data and interval domain data. In this paper, parameters are estimated for the interval domain data.

The mean value function of Pareto type IV model is given by

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

In order to have an assessment of the software reliability, a, b and c are unknown parameters and estimated using Newton Raphson method. Expressions are now delivered for estimating ‘a’, ‘b’ and ‘c’ for the Pareto type IV model.

Assuming the given failure data set for the cumulative number of detected errors n_i in a given time interval (0, t_i) where i=1,2, ….. n and 0 < t_1< t_2< …t_n, then the logarithmic likelihood function (LLF) for interval domain data [12] is given by

$$Log\ L = \sum_{i=1}^{k} (n_i - \; n_{i-1}).\log[m(t_i) - m(t_{i-1})] - m(t_k)$$

$$Log\ L = \sum_{i=1}^{k} (n_i \; - n_{i-1}) \left[log\left[a\left[1 - \left[1 + \left(\frac{t_i}{c}\right)\right]^{-b}\right]\right] - \left[a\left[1 - \left[1 + \left(\frac{t_{i-1}}{c}\right)\right]^{-b}\right]\right]\right] - \left[a\left[1 - \left[1 + \left(\frac{t_k}{c}\right)\right]^{-b}\right]\right]$$

$$Log\ L = \; \textstyle\sum_{i=1}^{k} (n_i \; - n_{i-1}) \;\; Log\; \left[\left(\frac{at_i}{c}\right)^{-b} \; - \; \left(\frac{at_{i-1}}{c}\right)^{-b}\right] - \; \left(\frac{at_k}{c}\right)^{-b}$$

Differentiating Log L with respect to ‘a’ we have

$$\frac{\partial\ Log\ L}{\partial a} = 0$$

$$\sum_{i=1}^{k}(n_i - n_{i-1})\left[\left[\frac{-bt_i}{ac} + \frac{bt_{i-1}}{ac}\right] - \left(\frac{t_k}{c}\right)^{-b}\right] = 0$$

$$a = \sum_{i=1}^{k}(n_i - n_{i-1}) + \left[\frac{b(t_{i-1} - t_i)}{c}\right]\left(\frac{c}{t_k}\right)^{b}$$

The parameter ‘b’ is estimated by Newton Raphson iterative Method using the formula

$b_{n+1} = b_n - \frac{g(b)}{g'(b)}$, where g(b) and g’(b) are obtained as follows.

$$g(b) = \frac{\partial \log L}{\partial b} = 0$$

$$g(b) = \frac{\partial\, Log\, L}{\partial b} = \sum_{i=1}^{k}(n_i - n_{i-1})\Bigg[-Log(t_{i-1}+1) - Log(t_i+1) + \frac{1}{(t_i+1)^b\ {-(t_{i-1}+1)^b}}\left[(t_i+1)^b\, Log(t_i+1) - (t_{i-1}+1)^b\, Log(t_{i-1}+1)\right]\Bigg] + \left[\sum_{i=1}^{k}(n_i - n_{i-1}) + b(t_{i-1} - t_i)\, Log\left(\frac{1}{t_k+1}\right)\right] = 0$$

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

$$g^{'}(b) = \sum_{i=1}^{k}(n_i - n_{i-1})\left[\frac{2(t_{i-1}+1)^b\,(t_i+1)^b Log(t_i+1)\, Log\frac{(t_{i-1}+1)}{(t_i+1)}}{[(t_i+1)^b - (t_{i-1}+1)^b]^2}\right] + \sum_{i=1}^{k}(n_i - n_{i-1}) + b(t_{i-1} - t_i)\, Log\left(\frac{1}{t_k+1}\right)$$

Similarly the parameter ‘c’ is estimated by using the formula $c_{n+1} = C_n - \frac{g(c_n)}{g'(c_n)}$

Where g (c) and g^1(c) are obtained as follows.

$$g(c) = \frac{\partial LogL}{\partial c} = 0$$

$$g(c) = \sum_{i=1}^{k}(n_i - n_{i-1})\left[\left[\frac{1}{c} - \frac{1}{(t_{i-1}+c)} - \frac{1}{(t_i+c)} + \left(\frac{t_{i-1}-t_i}{c}\right)\left(\frac{c}{t_k}\right)\left[\frac{t_k}{(t_k+c)^2}\right]\right]\right] = 0$$

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

$$g^{'}(c) = \sum_{i=1}^{k}(n_i - n_{i-1})\left[\frac{-1}{c^2} + \frac{1}{(t_{i-1}+c)^2} + \frac{1}{(t_i+c)^2}\right] + \sum_{i=1}^{k}(n_i - n_{i-1})\left[\frac{t_{i-1} - t_i}{c^2}\left(\frac{1}{(t_k)}\right)\left(\frac{t_k}{(t_k+c)^3}\right)\right]$$

Solving the above equations simultaneously yields the point estimate of the parameters b and c. These equations are to be solved iteratively and their solutions in turn when substituted gives the value of ‘a’.

6. DATA ANALYSIS

In this section, we present the analysis of one software failure data set. The set of software errors analysed here is borrowed from a real software development project as published in Pham (2005), Alan Wood *Tandem Computers* .The data are named as Release 2 test data. The Release 2 test data is summarized in the below table.

Table1. Release2 Dataset (Wood 1996)

Week	CPU Hours	Defects found	Week	CPU Hours	Defects found
1	384	13	11	7229	95
2	1186	18	12	8072	100
3	1471	26	13	8484	104
4	2236	34	14	8847	110
5	2772	40	15	9253	112

6	2967	48	16	9712	114
7	3812	61	17	10083	117
8	4880	75	18	10174	118
9	6104	84	19	10272	120
10	6634	89	20	-	-

A set of failure data taken from Misra (1983), given in Table 2 consists of the observation time (week) and the number of failures detected per week are errors: major and minor.

Table2. Dataset 1(Misra 1983).

Week	Minor Errors	Week	Minor Errors	Week	Minor Errors
1	9	13	5	25	2
2	4	14	3	26	3
3	7	15	3	27	6
4	6	16	3	28	3
5	5	17	4	29	1
6	3	18	10	30	1
7	2	19	3	31	4
8	5	20	1	32	3
9	4	21	2	33	2
10	2	22	s4	34	11
11	4	23	5	35	9
12	7	24	2		

Solving equations in section 4 by Newton Raphson (N-R) Method for the Release 2 test data for the both the models, the iterative solutions for MLEs of a, b and c are

Table 3. Estimated parameters and the corresponding Control Limits for Release2 & Dataset1

Model	Data Set No	a	b	c	$m(t_u)$	$m(t_l)$	$m(t_c)$
Pareto Type II	Release 2	158.1535	0.977674	8.74581	157.940029	0.213507	79.076768
Pareto Type IV	Release 2	107.3145	0.979778	8.483201	107.1696	0.144875	53.65725
Pareto Type II	Data Set1	200.558922	0.985185	15.109772	200.5318465	0.270754545	100.279461
Pareto type IV	Data Set1	139.741418	0.98761	14.963992	139.5527671	0.188650914	69.870709

Table 4. Release2- Successive Differences of Cumulative Mean Values

Week	Cumulative Failures	m(t)	Successive Differences	Week	Cumulative Failures	M(t)	Successive Differences
1	13	64.13480181	8.003755618	11	95	98.0607854	0.418077135
2	18	72.13855743	8.016123376	12	100	98.47886253	0.307960809
3	26	80.15468081	5.021249418	13	104	98.78682334	0.423329575
4	34	85.17593023	2.687846616	**14**	**110**	**99.21015291**	**0.131832346**
5	40	87.86377684	2.703258951	15	112	99.34198526	0.127569847
6	48	90.56703579	3.076228171	16	114	99.46955511	0.18380551
7	61	93.64326396	2.250327539	17	117	99.65336062	0.059350235
8	75	95.8935915	1.090057964	18	118	99.71271085	0.115956552
9	84	96.98364947	0.519436859	19	120	99.8286674	---------
10	89	97.50308633	0.55769907				

Table 5. Dataset1-Successive Differences of Cumulative Mean Values

TT(Day)	CF	m(t)	Successive Differences	TT(Day)	CF	m(t)	Successive Differences
1	9	51.97116561	12.41075911	19	89	119.138914	0.193861184
2	13	64.38192472	14.92036007	20	90	119.3327752	0.376914305
3	20	79.30228479	8.751200026	21	92	119.7096895	0.71331277
4	26	88.05348482	5.556877794	22	96	120.4230022	0.822851656
5	31	93.61036261	2.792490043	23	101	121.2458539	0.309727289
6	34	96.40285266	1.680093927	24	103	121.5555812	0.299462856
7	36	98.08294658	3.677883954	25	105	121.855044	0.431039697
8	41	101.7608305	2.503226941	26	108	122.2860837	0.802275625
9	45	104.2640575	1.131136921	27	114	123.0883594	0.373944314
10	47	105.3951944	2.057709638	**28**	**117**	**123.4623037**	**0.120921212**
11	51	107.452904	3.061186633	29	118	123.5832249	0.119127033
12	58	110.5140907	1.851939892	30	119	123.7023519	0.459343491
13	63	112.3660306	1.002019201	31	123	124.1616954	0.327503491
14	66	113.3680498	0.930843246	32	126	124.4891989	0.210746155
15	69	114.298893	0.867004553	33	128	124.6999451	1.061808877
16	72	115.1658976	1.067575921	34	139	125.7617539	0.76275397
17	76	116.2334735	2.300961701	35	148	126.5245079	--------
18	86	118.5344352	0.60447879				

The control limits are calculated by the following equations taking the standard values 0.00135, 0.99865 and 0.5.

$$T_U = \left[1 - \left[1 + \left(\frac{t}{c}\right)\right]^{-b}\right] = 0.99865$$

$$T_L = \left[1 - \left[1 + \left(\frac{t}{c}\right)\right]^{-b}\right] = 0.00135$$

$$T_C = \left[1 - \left[1 + \left(\frac{t}{c}\right)\right]^{-b}\right] = 0.5$$

These limits are converted to $m(t_U), m(t_C)$ and $m(t_L)$ form. They are used to find whether the software process is within control limits or not. Graphical representations of both the models are given below.

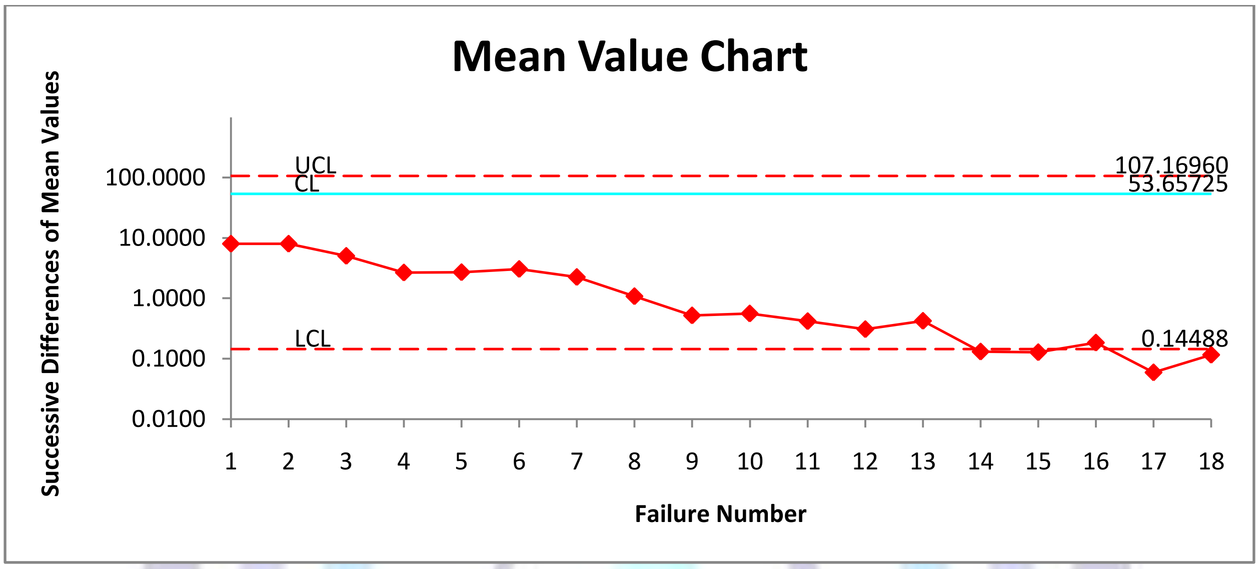

Fig 1. Mean Value Chart for Release2

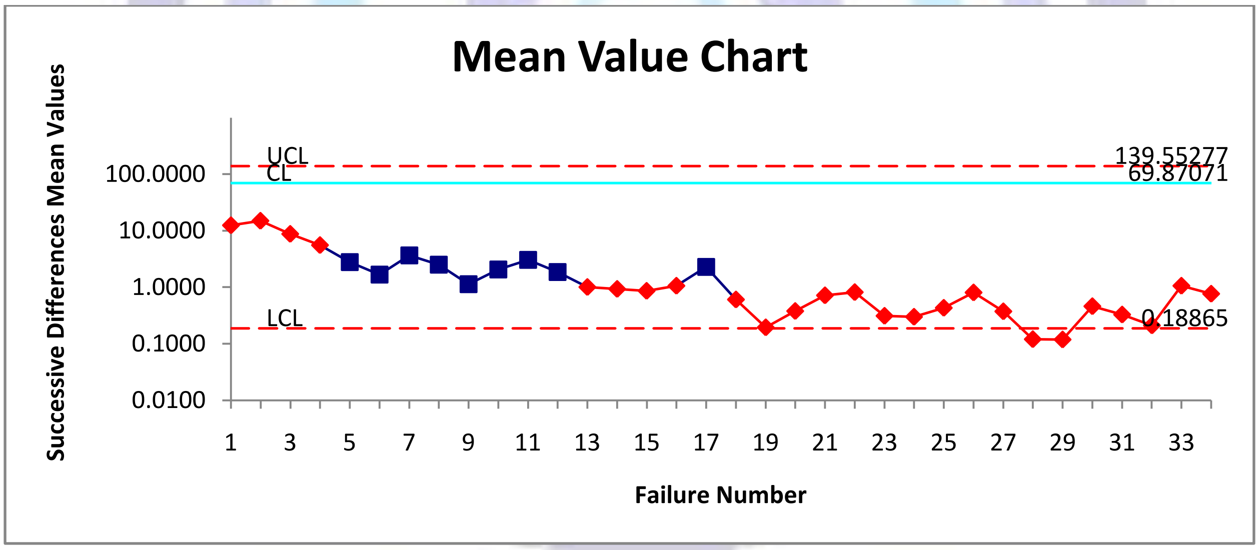

Fig 2. Mean Value Chart for Dataset1

By placing the successive differences on y axis and failure week on x axis the values of control limits are placed on Mean Value Chart. The Mean Value chart (Fig 1) shows that at 14th point the failure data has fallen below $m(t_L)$. Fig 2 shows that at 28th point the failure data has fallen below $m(t_L)$.This indicates that the failure process is identified. Thus Mean Value charts are significant in identifying early detection of failure data.

7. CONCLUSION

In this paper Pareto type IV software reliability growth model with SPC is proposed. Comparison is made between Pareto Type II and Pareto Type IV which shows that the failure points for both types is the same and fallen below $m(t_L)$ at 14th point and 28th point respectively as shown in figures 1 and 2. We conclude that our method of the control charts are giving a +ve recommendation for their use in finding out preferable control process or desirable out of control signal. The early detection of software failure will improve the software reliability. Therefore, we may conclude that both the models are equally best choice for an early detection of software failures.

ACKNOWLEDGEMENTS

Our thanks to Department of Computer Science and Engineering, Acharya Nagarjuna University for providing necessary details to carry out the research work.

REFERENCES

[1] A.L.Goel, K. Okumoto, "Time-dependent Error Detection rate model for software reliability and other performance measures", IEEE Trans. Reliab. 1979, R-28, pp. 206-211.

[2] Pham. H.. 2003. "Handbook of Reliability Engineering ", Springer.

[3] Musa J.D., Software Reliability Engineering MCGraw-Hill, 1998

[4] M. Kimura, S. Yamada, S. Osaki, "statistical software Reliability Prediction and its applicability based on mean time between failures", Mathematical and computer Modelling, 1995, Volume 22, Issues 10-12, Pages149-155.

[5] J.F. MacGaregor, T.Kourti, "Statistical process control of multivariate processes". Control Engineering Practise, Volume 3, Issue 3, March 1995, Pages 403-443.

[6] J.D.Musa, A.Iannino, K. Okumoto, "Software Reliability: Measurement Prediction Application ". MCGraw-Hill, New York. 1987

[7] Satya Prasad R and Geetha Rani N(2011), "Pareto type II Software Reliability Growth model", International Journal of Software Engineering, volume 2, Issue(4) 81-86.

[8] Wood, A(1996), "Predicting Software Reliability", IEEE Computer, 2253-2264.

[9] R.R.L. Kantam and R.Subba Rao, 2009, "Pareto Distribution: A Software Reliability Growth model ", International Journal of PerformabilityEngineering. Volume 5 , Number 3, April 2009, Paper 9,PP: 275-281

[10] Satya Prasad R and Geetha Rani N (2011) "Software Reliability Growth model using Interval Domain Data", International Journal Of Computer Applications (0975-8887), Volume 34-No 9, November 2011.

[11] Wood A., (1996). "Software Reliability Growth Models", Tandem Computers, Technical report 96-1.

[12] System Software Reliability, Springer, 2006, H.Pham.

[13] Lyu. M.R. (1996). "Handbook of Software Reliability Engineering", MCGraw-Hill, New York.

[14] S.M.K Quadri, N.Ahmed "SoftwareRreliability Growth Modeling with new modified weibull testing effort and optimal release policy 2010. " International journal of computer applications, volume 6 no 12, September 2010.

[15] Xie. M. Goh. T.N. Ranjan P.. (2002). "Some effective control chart procedures for Reliability monitoring", Reliability Engineering and System Safety 77 143-150.

[16] R.Satya Prasad and G.sridevi (2013),"Assessing Pareto Type II Software Reliability using SPC", International journal of computer applications(0975-8887), volume 62-No3, January 2013.

Author' biography

Dr. R. Satya Prasad received Ph.D.degree in computer science in the faculty of Engineering in 2007 from Acharya Nagarjuna University, Andhra Pradesh. 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 &Engg., Acharya Nagarjuna University. His current research is focused on Software engineering. He has published several papers in National & International Journals.

Mrs. G. Sridevi received M.Sc. and M.Tech degree from Acharya Nagarjuna University. She is currently pursing Ph.D at Department of Computer Science and Engineering, Acharya Nagarjuna University, Guntur, Andhra Pradesh, India. She is currently working as a Vice-Principal and Associate professor in the Department of Computer Science, Nimra Women's College of Engineering, Jupudi, Ibrahimpatnam, Vijayawada, Andhra Pradesh. Her research interests lies in Data Mining and Software Engineering.