# Software Reliabilty for Distributed systems

## *Industry Perspective*

Dr. Amit Gupta* , Renu Garg#
*Maharaja Agersen Institute of technology,
Guru Gobind Singh Indraprastha University
Rohini, Delhi , India
#Research Scholar, Mewar University,Chittorgarh
Raiasthan, India

Abs*tract*— ***"Reliability is probably the most important of the characteristics inherent in the concept "Software quality." Software reliability concerns itself with how well the software functions to meet the requirements of the customer". by J. MUSA AND OTHERS, 1990. Over the past two decades, many models have been developed to predict a software system's failure rate and were used as management tools to evaluate software reliability. In this paper we have discussed Weibull model, Rayleigh, exponential and Musa model. This study introduces the graphical representation of these models by implementing in MATLAB. The objective of this paper is to tell that in Industry perspective the only measure of software reliability is customer satisfaction. If customer is satisfied with particular software, the software will be considered reliable and no further testing is required.***

***Keywords— Customer satisfaction, Software reliability , MATLAB, Software quality.***

## I. Introduction

Software reliability is defined as "the probability of failure-free operation of a computer program in a specified environment for a specified period of time"[1]. Failure means the software does not perform its intended function. An error, which is an incorrect portion of software, is the potential cause of failure. Probability of failure-free operation for a specific time period or input set under a specific environment. Software reliability is inversely related to the complexity of software but it is directly related to other important factors in software quality, especially functionality, capability. Software Reliability is hard to achieve, because the complexity of software tends to be high.
Relaibility can be expressed by many ways:

- **Failure intensity:** the number of failures per unit time . Failure intensity is a way of expressing reliability.
- **Mean-Time-To-Failure (MTTF):** Expected value of a failure interval.
- **Expected total failures m(t):** The number of failures expected in a time period t.

## II. Software Reliability Models

After fitting a model describing the failure process we can estimate its parameters, and the quantities such as the total number of faults in the code, future failure intensity and

| Mobile company- A (Dec'2011- Aug'2012) | | |
|---|---|---|
| **Months** | **Defects** | **Cumulative Defects** |
| **Development Phase** (Dec'2011- April '2012) | | |
| 1 | 300 | 300 |
| 2 | 400 | 700 |
| 3 | 400 | 1100 |
| 4 | 250 | 1350 |
| 5 | 200 | 1550 |
| **Testing Phase**(May'2012- July'2012) | | |
| 6 | 200 | 1750 |
| 7 | 150 | 1900 |
| 8 | 100 | 2000 |
| **Market Release**(Aug.'2012- Sept'2012) | | |
| 9 | 30 | 2030 |
| 10 | 20 | 2050 |

additional time required to achieve a failure intensity objective. Data from various sources has been collected and observed on various models.

Some of the models are mentioned below:

- ***Musa's Basic Model***
- ***Weibull Distribution Model***
- ***The Rayleigh Model***
- ***Exponential Model***


ISSN: 2278 – 1323

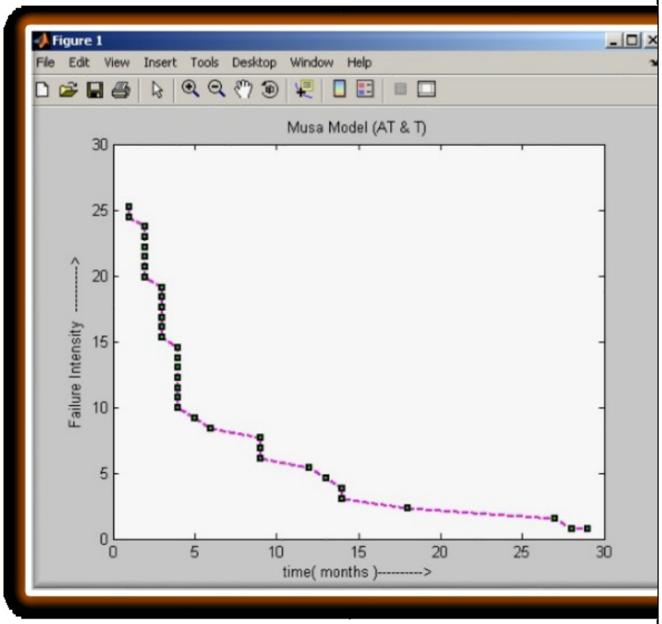
| AT & T | | |
|---|---|---|
| **Months** | **Defects** | **Cumulative Defects** |
| **Development Phase** | | |
| 1 | 3 | 3 |
| 2 | 6 | 9 |
| 3 | 6 | 15 |
| 4 | 7 | 22 |
| 5 | 1 | 23 |
| 6 | 1 | 24 |
| 7 | 0 | 24 |
| 8 | 0 | 24 |
| 9 | 3 | 27 |
| 10 | 0 | 27 |
| 11 | 0 | 27 |
| **Testing Phase** | | |
| 12 | 1 | 28 |
| 13 | 1 | 29 |
| 14 | 2 | 31 |
| 15 | 0 | 31 |
| 16 | 0 | 31 |
| 17 | 0 | 31 |
| 18 | 1 | 32 |
| 19 | 0 | 32 |
| **Maintenance Phase** | | |
| 20 | 0 | 32 |
| 21 | 0 | 32 |
| 22 | 0 | 32 |
| 23 | 0 | 32 |
| 24 | 0 | 32 |
| 25 | 0 | 32 |
| 26 | 0 | 32 |
| 27 | 1 | 33 |
| 28 | 1 | 34 |
| 29 | 0 | 34 |

[1]

*A. Musa's Basic Model*

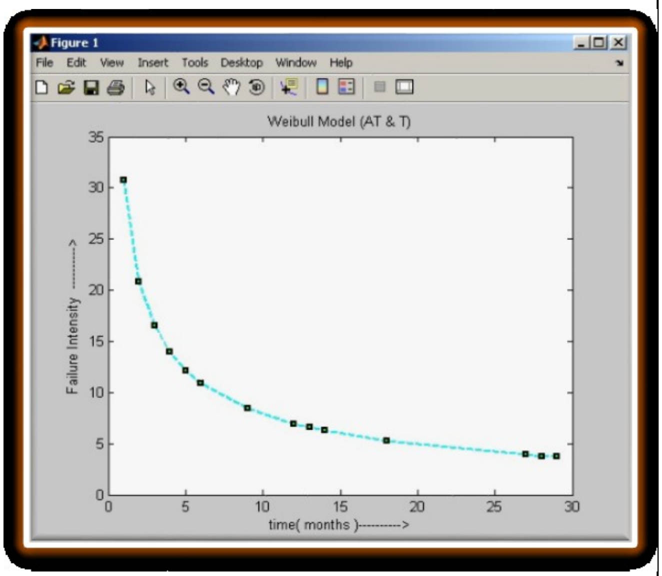
- Assumption: Decrement in failure intensity function is constant
- Result: Failure intensity is function of average number of failures experienced at any given point in time (= failure probability)
  - $\lambda(\mu)$: failure intensity
  - $\lambda_0$: initial failure intensity at start of execution
  - $\mu$: average total number of failures at a given point in time
  - $v_0$: total number of failures over infinite time

$$\lambda(\mu) = \lambda_0 \left[ 1 - \frac{\mu}{v_0} \right]$$




*B. Weibull Distribution Model*

The Weibull model does not make any assumptions about the parameters. It does not require any form of approximation to compute the mean time to failure. Estimates of reliability provide guidelines to stop testing, thus saving time and money. The sequence of λ





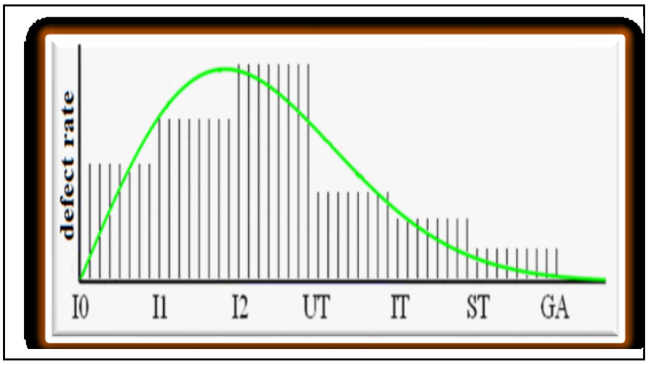

ISSN: 2278 – 1323 

is probabilistically decreasing reflecting tester intent to improve the program upon discovering software faults.
Probability distribution function
m - shape parameter
c - scale parameter
t - time

$$PDF: f(t) = \frac{2}{t}\left(\frac{t}{c}\right)^2 e^{-(t/c)^2}$$

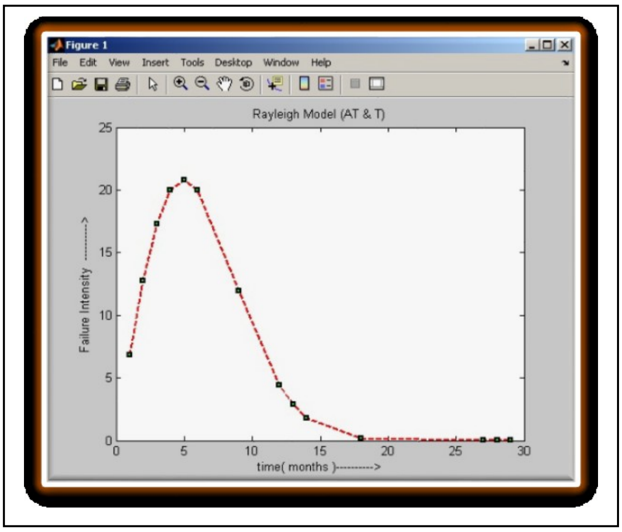
*C..The Rayleigh Model [12]*

Rayleigh model is the particular case of the Weibull distribution. Rayleigh took shape parameter m = 2 . Probability density grows until a moment $t_m$, then decreases asymptotically to 0.




Applying the Rayleigh model in software quality relies on two basic assumptions

- The defect rate observed during the development process is positively correlated with the defect rate in the field.
- If more defects are discovered during the early phases, fewer defects remain in the later phases.

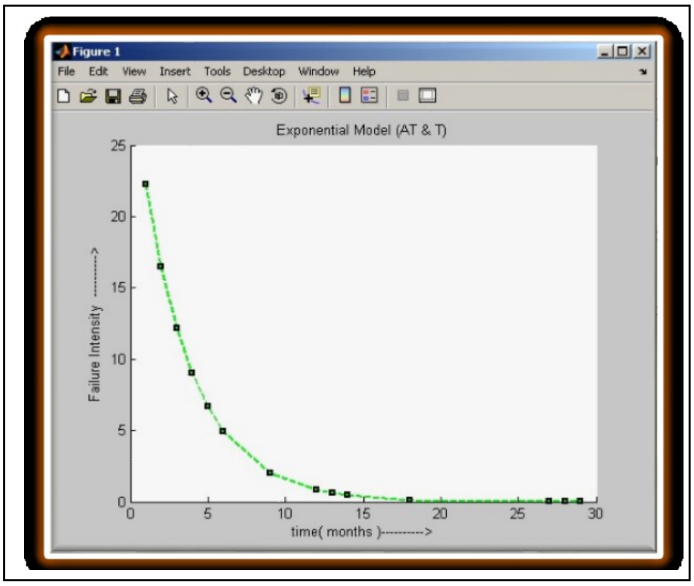



*D. Exponential Model*

$$PDF: f(t) = \frac{1}{c}e^{-t/c} = \lambda e^{-\lambda t}$$

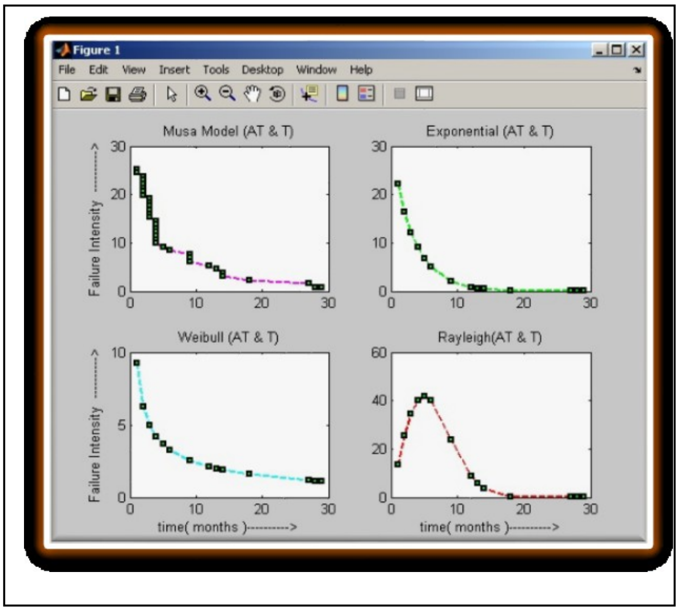
$\lambda = 1/c$ - instantaneous failure rate

the parameters to be estimated for describing the model: K (total number of defects) and λ.




## III. RESULT AND DISCUSSION/ ANALYSIS

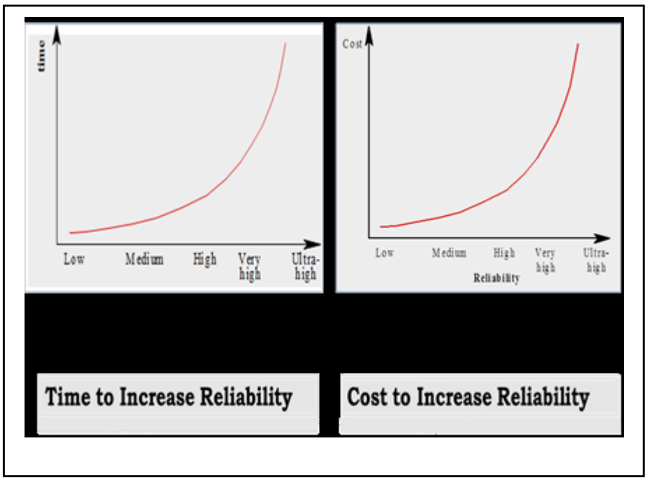
Various model can be compared on AT&T data[1]. Behavior of models is represented by graph.





ISSN: 2278 – 1323 

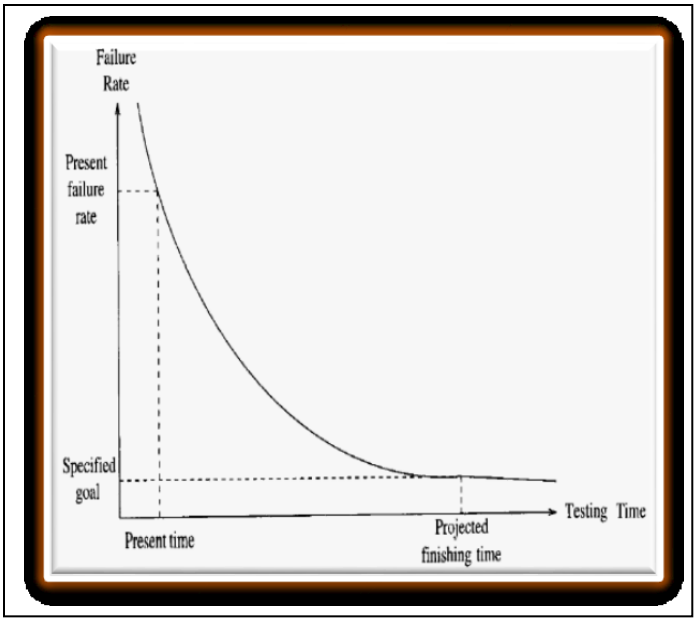
## IV. LIMITATIONS




Increasing the reliability of a system does not always mean ecreasing the losses from failures. An inappropriate increase of the reliability of the system may lead to a simultaneous increase of time therefore increase the cost. d In other words, a system reliability improvement, not supported by analysis of the losses from failure does not necessarily reduce the losses from failures. Limitations to achieve high reliability

## V. CONCLUSIONS

Behaviour of software failure is very complex owing to the debugging process, the sequence of input and the operational environment. In this study, we implement weibull , Rayleigh, Musa and exponential model in MATLAB for AT&T data. Software reliability is a key part in software quality. Reliability must be quantified so that we can compare software systems. Measurement is very important for finding the correct model.Statistical testing should be used but it is not easy again. There are no generic models. To recollect the success, the projects where this has been used for predicting out going quality and has matched to the prediction. A system is reliable if it is used according to its specific parameters. If the access is forced beyond limits by the employee, the system can become unreliable. As per Industry perspective, various parameters can be added like Level, Time and stage of failure. Customer satisfaction is the direct measure of Software reliability .




The ultimate goal is – To decide when to stop testing by observing the minimal risk.

## REFERENCES

[1] Loan Pham , Hoang Pham . Software Reliability Models with Time-Dependent Hazard Function Based on Bayesian Approach. IEEE Transactions on Systems , Man and Cybernatics- Part A: Systems and Humans, Vol. 30, No. 1, Jan 2000

[2] Mettas, A. and Zhao, W. Modeling and Analysis of Complex Repairable Systems, Technique Report, ReliaSoft Corporation, 2004.

[3] H. Roberts. Predicting the Performance of Software Systems via the Power Law Process. Ph.D. thesis, University of South Florida, Tampa, FL, 2000.

[4] E. Gamma, R. Helm, R. Johnson, J. Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software, Reading, MA: Addison-Wesley, 2005.

[5] J. D. Musa, A. Iannino, and K. Okumoto. Software reliability. McGraw-Hill,New York, 1987

[6] Kaminskiy, M. and Krivtsov, V. "A Monte Carlo approach to repairable system reliability analysis." Probabilistic Safety Assessment and Management, New York: Springer; p. 1063-1068, 1998.



ISSN: 2278 – 1323 

[7] Sy-Yen Kuo, Fellow IEEE, Chin- Yu Huang, and Michael R. Lyu, Senior Member, IEEE. Framework for Modeling Software Reliability, using various Testing-efforts and Fault- Detection Rates. IEEE Transactions on Reliability, vol 50, No. 3, Sept. 2001

[8] M. R. Lyu. Handbook of Software Reliability Engineering. McGraw-Hill, 1996

[9] S. E. Rigdon and A. P. Basu. *Statistical Methods for the Reliability of Repairable Systems*. John Wiley and Sons, New York, 2000.

[10] T. A. Mazzuchi and R. Soyer. A Bayes Empirical-Bayes Model for Software Reliability. *IEEE Trans. Reliability*, 37(2), pp.248{254, 1988.

[11] Wang, H. and Pham, H. "Some maintenance models and availability with imperfect maintenance in production systems." *Annals of Operations Research*, 91: 305-318, 1999

[12] http://students.info.uaic.ro/~andrei.amariei/Teme/Sem%209/CSS/curs10.ppt

[13] www.site.uottawa.ca/~nelkadri/SEG%203202/Reliability-1.ppt

AUTHORS

**Dr Amit Gupta** is working as Associate Professor in Maharaja Agrasen Institute of Technology, affiliated to Guru Gobind Singh Indraprastha University, Delhi. He obtained his Ph.D degree from University of Delhi. He has published extensively in Indian Journals and abroad in the area of Reliability, optimization, Innovation in ICT and maintenance and software reliability. He had guided M.Phil/ Ph.D theses in Computer Science as well in Operational Research,

**Renu Garg** is a research scholar of Mewar university, Chittorgarh, Rajasthan, India. She has completed MCA from Gurukul Kangri University- Hardwar, M. Tech from DOEACC( C Level). She has 14 years academic experience. Her research interests are in improving software reliability.



ISSN: 2278 – 1323 