JM MODEL AND ITS VARIATIONS IN SOFTWARE RELIABILITY MODELING

Dr. Kuldeep Singh Kaswan
Department of Computer Science & Engineering
PDM College of Engineering, Bahadurgarh
Haryana- India

Abstract- Jelinski-Moranda (JM) Model is considered as the first and best software reliability model. This model was developed with some assumptions, but in reality all the assumptions of this model is not practical. So the later authors tried to design it as the more practical model with some modifications. In this paper, we have discussed about the different existing variations of the JM model. In the last of this paper, we also summarize the work done by different researchers based on the decadal growth of this model.

Index Terms: **Software Reliability Models, JM Model, Imperfect Debugging.**

I. INTRODUCTION

Software engineering is an engineering discipline that is concerned with all aspects of software production. According to the definition given at NATO conference, software engineering is the establishment and use of sound engineering principles in order to obtained economically software that is reliable and woks efficiently on real machines.

Software reliability can be defined as the probability of failure-free software operation for a specified period of time in a specified environment [1, 2, 3]. When the software passes through the different software development phases as analysis phase, design phase, programming and maintenance phase a number of errors may be occurred. To detect and remove these errors, the software system is tested. The quality of software system in terms of reliability, which is an important quality attribute of software, is measured by the removal of these errors.

To detect the failure behavior of software, to predict the faults or failures for software, reliability models are developed. These models are sensitive to the situations for the different stages of Software development life cycle. On behalf of these models one can access the current and future reliability through testing, and can make decisions whether the software product can be released in its present state or requires further testing in order to improve the quality of software.

II. JELINSKI MORANDA MODEL

The Jelinski-Moranda (JM) model [4], which is also a Markov process model, has strongly influenced many later models which are in fact modifications of this simple model.

Characteristics of JM Model

1. It is a Binomial type model
2. It is possibly the earliest and certainly one of the most well-known black-box models.
3. J-M model always yields an over optimistic reliability prediction.
4. JM Model follows a prefect debugging process, i.e. the detected fault is removed with certainty

Model Assumptions

The assumptions made in the J-M model include the following:

(i) The number of initial software faults is unknown but fixed and constant.

(ii) Each fault in the software is independent and equally likely to cause a failure during a test.

(iii) Time intervals between occurrences of failure are independent, exponentially distributed random variables.

(iv) The software failure rate remains constant over the intervals between fault occurrences.

(v) The failure rate is proportional to the number of faults that remain in the software.

(vi) A detected fault is removed immediately and no new faults are introduced during the removal of the detected fault.

(vii) Whenever a failure occurs, the corresponding fault is removed with certainty.

Mathematical Formulation

Software Fault Rate

$\lambda(t_i) = \phi[N-(i-1)]$ where i=1,2,….,N

ϕ=a constant of proportionality denoting the failure rate contributed by each fault

N=the initial number of faults in the software

t_i -the time between (i-1)th and ith failure.

Failure Density Function

$F(t_i) = \phi[N-(i-1)]\exp(-\phi[N-(i-1)]t_i)$

Distribution Function

$F_i(t_i) = 1-\exp(-\phi[N-(i-1)]t_i)$

Reliability function at the ith failure interval

$R(t_i) = 1- F_i(t_i) = \exp(-\phi[N-(i-1)]t_i)$

MTTF for the ith failure $=1/ \phi[N-(i-1)]$

III. VARIATIONS IN JM MODEL

JM model was the first influential software reliability model. A number of researchers showed interest and modify this model, using different parameters such failure rate, perfect debugging, imperfect debugging, number of failures etc. now we will discuss about different existing variations of this model.

I. Lipow Modified Version of Jelinski-Moranda Geometric Model

Allow multiple errors removal in a time interval. The program failure rate becomes [8]

$$\lambda(t_i) = Dk^{n_i-1}$$

Where n_{i-1} is the cumulative number of errors found up to the (i-1)st time interval.

II. Sukert Modified Schick-Wolverton Model

Sukert [6] modifies the S-W model to allow more than one failure at each time interval. The program failure rate becomes

$$\lambda(t_i) = \phi[N-n_{i-1}]t_i$$

$$R(t_i) = e^{-\phi[N-n_{i-1}]\cdot\frac{t_i^2}{2}}$$

Where n_{i-1} is the cumulative number of failures at the (i-1)th failure interval.

III. Schick Wolverton Model

The Schick and Wolverton (S-W) model [5] is similar to the J-M model, except it further assumes that the failure rate at the ith time interval increases with time since the last debugging. Other assumptions are as follows:

- Errors occur by accident.

- The error detection rate in the defined time intervals is constant.
- Errors are independent of each other.
- No new errors are generated.
- Errors are corrected after they have been detected.

In the model, the program failure rate function is:

$\lambda(t_i) = \phi[N-(i-1)]t_i$

Where ϕ is a proportional constant, N is the initial number of errors in the program, and t_i is the test time since the (i-1) st failure.

IV. GO-Imperfect Debugging Model

Goel and Okumoto [7] extend the J-M model by assuming that a fault is removed with probability p whenever a failure occurs. The program failure rate at the ith failure interval is

$$\lambda(t_i) = \phi[N-p(i-1)]$$
$$R(t_i) = e^{-\phi[N-p(i-1)]\cdot t_i}$$

V. Jelinski-Moranda Geometric Model

This model assumes that the program failure rate function is initially a constant D and decreases geometrically at failure time. The program failure rate and reliability function of time between failures at the ith failure interval are [9]

$$\lambda(t_i) = Dk^{i-1}$$
$$R(t_i) = e^{-Dk^{i-1}t_i}$$

Where k is Parameter of geometric function, 0<k<1

VI. Little-Verrall Baysian Model

This model assumes that times between failures are independent exponential random variables with a parameter ξ, i = 1, 2… n which itself has parameters $\Psi(i)$ and α reflecting programmer quality and task difficulty) having a prior gamma distribution.

$\lambda(t_i) = (\alpha-1)[N^2+2B\varphi p(\alpha-I)]^{-1/2}$

Where B represents the fault reduction factor

VII. Shanthikumar General Markov Model

This model assumes that the failure intensity functions as the number of failure removed is as the given below

$\lambda_{SG}(n,t) = \psi(t)\,(N_0-n)$

Where ψ (t) is proportionality constant.

VIII. An Error Detection Model for Application during Software Development

Main feature of this new model is that the variable (growing) size of a developing program is accommodated, so that the quality of a program can be estimated by analyzing an initial segment.

Assumptions

This model has the following assumptions along with the JM model assumptions:

1. Any tested initial portion of the program represents the entire program with respect to the number and nature of its incipient errors.
2. The detect-ability of an error is unaffected by the "'dilution" incurred when the initially tested portion is augmented by new code.
3. The number of lines of code which exists at any time is known.
4. The growth function and the error detection process are independent.

IX. The Langberg Singpurwalla Model

This model show how several models used to describe the reliability of computer software can be comprehensively viewed by adopting a Bayesian point of view. [20]

This model provides an alternative motivation for a commonly used model using notions from shock models.

X. Jewell Bayesian Software Reliability Model

Jewell extended a result by Langberg and Singpurwalla (1985) and made a further extension of the Jelinski-Moranda model.

Assumptions

1) The testing protocol is permitted to run for a fixed length of time-possibly, but not necessarily, coinciding with a failure epoch.
2) The distribution of the unknown number of defects is generalized from the one-parameter Poisson distribution by assuming that the parameter is itself a random quantity with a Beta prior distribution.
3) Although the calculation of the posterior distributions of the parameters leads to complex expressions, we show that the computation of the predictive distribution for undetected errors is straightforward.
4) Although it is now recognized that the MLE's for reliability, growth can be very unstable, we show that, if a point estimator is needed, the predictive mode is easily calculated without obtaining the full distribution first.

XI. Quantum Modification to the JM Model

This model replace the JM Model assumption, each fault has the same contribution to the unreliability of software, with the new assumption that different types of faults may have different effects on the failure rate of the software. [10]

Failure Rate:

$\lambda_i = (Q-\sum_{j=1}^{i-1} w_j)\psi$

Where Q = initial number of failure quantum units inherent in a software

ψ = the failure rate corresponding to a single failure quantum unit

w_j = the number of failure-quantum units of the ith fault i.e. the size of the ith failure-quantum

XII. Optimal Software Released Based on Markovian Software Reliability Model

In this model, a software fault detection process is explained by a Markovian Birth process with absorption. This paper revised the optimal software release policies by taking account of a waste of a software testing time.

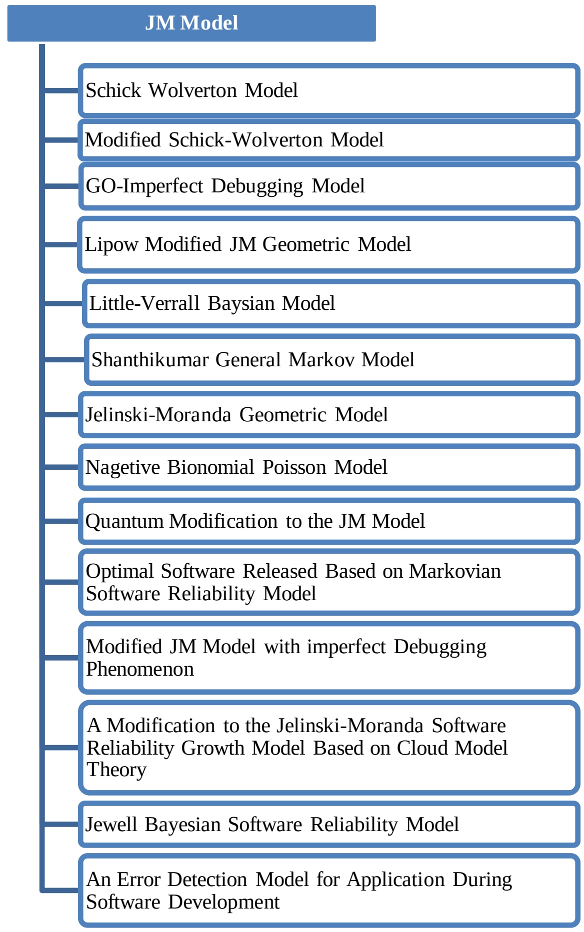

Fig. 1: Variations of JM Model

XIII. A Modification to the Jelinski-Moranda Software Reliability Growth Model Based on Cloud Model Theory

A new unknown parameter θ is included in the JM model parameters estimation such that $\theta \varepsilon [\theta_L, \theta_U]$ The confidence level is the probability value $(1-\alpha)$ associated with a confidence interval. In general, if the confidence interval for a software reliability index θ is achieved, we can calculate the numerical characteristics of virtual cloud $C(Ex, En, He)$, which can be switched to system qualitative evaluation by X condition cloud generator.

XIV. Modified JM Model with imperfect Debugging Phenomenon

The modified JM Model extend the J-M model by relaxing the assumptions of perfect debugging process and types of imperfect removal: (i) the fault is not removed successfully while no new faults are introduced and (ii) the fault is not removed successfully while new faults are created due to incorrect diagnoses.

Model Assumptions

The assumptions made in the Modified J-M model include the following:

(i) The number of initial software faults is unknown but fixed and constant.

(ii) Each fault in the software is independent and equally likely to cause a failure during a test.

(iii) Time intervals between occurrences of failure are independent, exponentially distributed random variables.

(iv) The software failure rate remains constant over the intervals between fault occurrences.

(v) The failure rate is proportional to the number of faults that remain in the software.

(vi) Whenever a failure occurs, the detected fault is removed with probability p, the detected fault is not perfectly removed with probability q and the new fault is generated with probability r. So it is obvious that $p+q+r=1$ and $q \geq r$

Software failure rate-

$\lambda(t_i) = \phi[N-(i-1)(p-r)]$

Failure Density Function

$F(t_i) = \phi[N-(i-1)(p-r)]\exp(-\phi[N-(i-1)(p-r)]t_i)$

Distribution Function

$F_i(t_i)=1-\exp(-\phi[N-(i-1)(p-r)]t_i)$

Reliability function at the ith failure interval

$R(t_i)=1- F_i(t_i)= \exp(-\phi[N-(i-1)(p-r)]t_i)$

$MTTF =1/ \phi[N-(i-1)(p-r)]$

IV. SUMMARIZATION

JM model played a vital role in software reliability model world. So the researchers showed much interest in this model. A brief summary of JM model variations in software reliability models have been summarized in this section as shown in table-1.

Table 1: Summary of different Software Reliability Models which is derived from JM Model

Sr. No.	Model Name	Author(s) Name	Date	Failure Rate Function	Data Set	Modification Based on	Reference
1	JM Model	Z. Jelinski, P. B. Model	1972	$\lambda(t_i)= \phi[N-(i-1)]$	---		[11]
2	Lipow Modified JM Model	Lipow	1974	$\lambda(t_i) = Dk^{n_i -1}$	---	No. of error removal	[8]
3	Modified Schick-Wolverton Model	A. N. Sukert	1977	$\lambda(t_i)= \phi[N-n_{i-1}]t_i$	---	No. of failures	[6]
4	Schick Wolverton Model	G. J. Schick. R. W. Wolvert	1978	$\lambda(t_i)= \phi[N-(i-1)]t_i$	W. L. Wagoner of The Aerospace Corporation	Failure Rate	[12]
5	GO-Imperfect Debugging Model	A. L. Goel, K. Okumoto	1979	$\lambda(t_i)= \phi[N-p(i-1)]$	Real-Time Control System	Imperfect debugging	[7]
6	Jelinski-Moranda Geometric Model	Z. Jelinski, P. B. Moranda	1979	$\lambda(t_i) = Dk^{ni-1}$	-----	Failure Rate	[9]
7	Little-Verrall Baysian Model	B Littlewood, JL Verrall	1979	$\lambda(t_i)= (\alpha-1)[N^2+2B\varphi p(\alpha-I)]^{-1/2}$	The "data" here are a stream of information from something of the order of 10,000 sensing devices, usually via analog/digital converters (ADCs).	Time Between failure	[13]
8	Shanthikumar General Markov Model	JG Shanthikumar	1981	$\lambda_{SG}(n,t)= \phi(t)[N_0-n]$	-----	No of failure removal	[14]
9	An Error Detection Model for Application During Software Development	Moranda	1981	----------------	Data from F. Akiyama	Dynamic Program size	[15]
10	The Langberg Singpurwalla Model	Langberg, N., and Singpurwalla, ND	1985	----------------	Real-Time Control System	Probability Distribution function	[20]
11	Jewell Bayesian Software Reliability Model	Jewell	1985	----------------	Data from J. D. Musa is analyzed-using their Bayesian mode.		[16]
12	Quantum Modification to the JM Model	Tsu-Fens Ho, WahChung Chan, Chyan Goei Chung	1991	$\lambda_i = (Q-\sum_{j=1}^{i-1} w_j)\psi$	B. Littlewood (1987) data set	Nature of Faults	[10]
13	Optimal Software Released Based on Markovian Software Reliability Model	K. Rinsaka, T. Dohi	2004	$z(xi) =\phi[a - (\boldsymbol{i} - l)]$	Koch and Kubat data set	Testing Time	[17]
14	A Modification to the Jelinski-Moranda Software Reliability Growth Model Based on Cloud Model Theory	Ziqiang Luo, Peng Cao, Guochun Tang and Lihua Wu	2011	$\lambda_i = \frac{1}{C(Ex3n, En3n, He3n)}$	Data set of a Naval Tactical Data System (NTDS)	Parameter Estimation Probability	[18]
15	Modified JM Model with imperfect Debugging Phenomenon	GS Mahapatra, P Roy	2012	$\lambda(t_i)= \phi[N-(i-1)(p-r)]$	Musa system 1 failure data set	Imperfect debugging	[19]

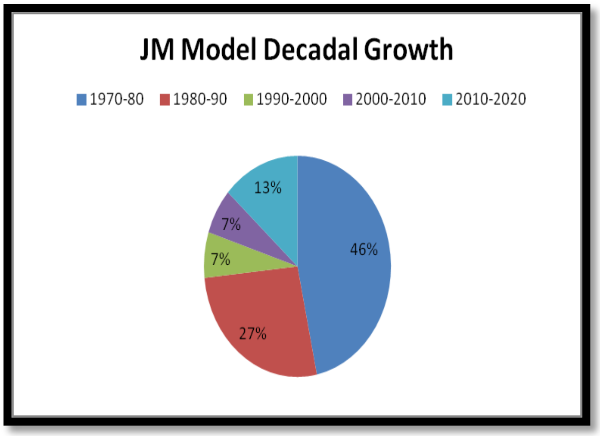

Fig. 2: JM Model Decadal Growth

From the summarization, it is concluded that in each decade there is a growth in JM model. In the 1970s decade the growth was maximum. Now in current decade there is again a good advancement in JM Model variants.

V. CONCLUSION

In this paper, we have discussed about the work done by the various researchers, with the endeavor made to include as many variations of JM model. Based on this paper, we thrash out some models as Lipow, Modified Schick-Wolverton Model, and GO-Imperfect Debugging Model etc. We emphasized on variations of JM model, with the reliance that it would serve as a reference to both old and new, incoming researchers in this field. These papers will support their understanding of current trends and assist their future research prospects and directions.

REFERENCES

[1] Kapur, P.K. and Garg, R.B., "Cost reliability optimum release policies for a software system with testing effort", Operations Research, Vol. 27, No. 2, pp. 109-116, 1990.

[2] Musa J.D., "Software Reliability Engineering: More Reliable Software, Faster Development and Testing", McGraw-Hill, 1999.

[3] Musa, J.D., Iannino, A. and Komodo, K., "Software Reliability: Measurement, Prediction and Application", McGraw-Hill, 1987.

[4] Littlewood, B. 1987. How good are software reliability predictions? Software Reliability: Achievement and Assessment. Blackwell Scientific Publications, pp. 154-166.

[5] Schick, G. J. and R. W. Wolverton. "An analysis of competing software reliability models," IEEE Transactions on Software Engineering, Vol. SE-4, No. 2, 1978.

[6] Sukert, A. N. "An investigation of software reliability models," Proceedings Annual Reliability & Maintainability Symposium, 1977.

[7] Goel, A.L., and Okumoto, K. "Time dependent error detection rate model for software reliability and other performance measures", IEEE T. Reliab. R-28(3), pp. 206-211, 1979.

[8] Lipow, M. "Some variation of a model for software time-to-failure", TRW Systems Group, Correspondence, ML-74-2260, August 1974.

[9] "An Error Detection Model for Application during Software Development", Moranda, P.B. (1979), IEEE Transactions on Reliability R-28, 5, pp. 325-329.

[10] Tsu-Feng Ho, Wah-Chun Chan and Chyan-Goei Chung, "A Quantum Modification to the Jelinski-Moranda Software Reliability Model", IEEE, Proceedings of the 33rd Midwest Symposium on circuits and systems, pp. 339-342, Vol. 1, 1990.

[11] Jelinski Z. and Moranda P. B., "Software reliability research", Statistical Computer

Performance, Evaluation. Academic Press: New York, pp. 465-484, 1972.

[12] George J. Schick and Ray W. Wolverton, "An Analysis of Competing Software Reliability Models", IEEE Transactions on Software Engineering, Vol. SE-4, NO. 2,pp. 104-120, MARCH 1978.

[13] Littlewood, B. and Verrall, JK, "A Bayesian Reliability Growth Model for Computer Software", Appl. Statist, 22:332.

[14] Shanthikumar General Markov Model, "A general software reliability model for performance prediction", Microelectronics Reliability Volume 21, Issue 5, 1981, pp. 671–682.

[15] Moranda, P.B. (1981). An error detection model for application during software development. IEEE Trans. Reliability, R-30, pp. 309-315.

[16] Jewell. W.S. (1985). Bayesian extensions to a basic model of software reliability. IEEE Trans. Software Eng.• SE-11, pp. 1465-1471.

[17] K. Rinsaka and T. Dohi, "Who Solved the Optimal Software Release Problems Based on Markovian Software Reliability Model?", The 47th IEEE International Midwest Symposium on Circuits and Systems, IEEE, pp. 475-478, 2005

[18] Ziqiang Luo, Peng Cao, Guochun Tang , Lihua Wu, "**A** Modification To The Jelinski-Moranda Software Reliability Growth Model Based On Cloud Model Theory", 2011 Seventh International Conference on Computational Intelligence and Security, pp. 195-198, IEEE, 2011.

[19] G. S. Mahapatra and P. Roy, "Modified Jelinski-Moranda Software Reliability Model with Imperfect Debugging Phenomenon", International Journal of Computer Applications (0975 – 888), Volume 48– No.18, pp. 38-46, June 2012.

[20] Langberg, N., and Singpurwalla, ND., "Unification of Some Software Reliability Models via the Bayesian Approach," SIAM Journal of Scientific and Statistical Computing,' Vol. 6, pp. 781- 790, 1985.

Kuldeep Singh Kaswan: has completed his Doctors Degree in Computer Science under the faculty of Computer Science at Banasthali Vidyapith, Rajasthan. He has obtained his Master Degree in Computer Science and Engineering from Choudhary Devi Lal University, Sirsa (Haryana) and Bachelor Degree from Kurukshetra University, Kurukshetra (Haryana). His area of interests includes software reliability and soft computing.