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 works 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 \( i = 1, 2, \ldots, N \)

\( \phi = \) 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 \(i\)th 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 \(i\)th failure interval

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

MTTF for the \(i\)th failure \(= \frac{1}{\phi[N-(i-1)]} \)

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

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

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

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 \(i\)th 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:
\[ \lambda(t_i) = \phi \cdot N \cdot (i-1) \cdot t_i \]

Where \( \phi \) 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 \( i \)th failure interval is
\[ \lambda(t_i) = \phi \cdot N \cdot p \cdot (i-1) \]
\[ R(t_i) = e^{-\phi \cdot N \cdot p \cdot (i-1) \cdot t_i} \]

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 \( i \)th 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 Bayesian Model
This model assumes that times between failures are independent exponential random variables with a parameter \( \xi \), \( i = 1, 2, \ldots, n \) which itself has parameters \( \Psi(i) \) and \( \alpha \) reflecting programmer quality and task difficulty) having a prior gamma distribution.
\[ \lambda(t_i) = (\alpha-1) \cdot (N^2 + 2B\phi\alpha^2) \cdot \frac{1}{\alpha^2} \]

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
\[ \lambda_{SG}(n,t) = \psi(t) \cdot (N_0 - n) \]

Where \( \psi(t) \) 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 shows 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 replaces 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
\( \psi \) = the failure rate corresponding to a single failure quantum unit
\( w_j \) = the number of failure-quantum units of the \( i \)th fault i.e. the size of the \( i \)th 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.
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

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 $\theta$ is included in the JM model parameters estimation such that $\theta \in [\theta_L, \theta_U]$ 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 $\theta$ 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.
F(t_i)=1-\exp(-\phi[N-(i-1)(p-r)]t_i) \\
R(t_i)=1-F(t_i)= \exp(-\phi[N-(i-1)(p-r)]t_i) \\
MTTF = \frac{1}{\phi[N-(i-1)(p-r)]}

Reliability function at the ith failure interval

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

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


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.