Software Defect Discovery and Resolution Modeling

Maskura Nafreen[1], Melanie Luperon[1], and Lance Fiondella[1]
[1]*Department of Electrical and Computer Engineering*
University of Massachusetts Dartmouth
North Dartmouth, MA 02747, USA
Email: {mnafreen, mluperon, lfiondella}@umassd.edu

Vidhyashree Nagaraju[2]
[2]*Tandy School of Computer Science*
University of Tulsa
Tulsa, OK 74104, USA
Email: vidhyashree-nagaraju@utulsa.edu

Ying Shi[3]
[3]*Goddard Space Flight Center*
National Aeronautics and Space Administration
Greenbelt, MD 20771, USA
Email: ying.shi@nasa.gov

Thierry Wandji[4]
[4]*Cyber Warfare Detachment*
Naval Air Systems Command
Patuxent River, MD, 20670, USA
Email: ketchiozo.wandji@navy.mil

Key Words: Software reliability, software reliability growth model, software defect lifecycle, defect resolution

Abstract- Traditional software reliability growth models only consider defect discovery data, yet the practical concern of software engineers is the removal of these defects. Most attempts to model the relationship between defect discovery and resolution have been restricted to differential equation-based models associated with these two activities. However, defect tracking databases offer a practical source of information on the defect lifecycle suitable for more complete reliability and performance models. This paper explicitly connects software reliability growth models to software defect tracking. Data from a NASA project was employed to develop differential equation-based models of defect discovery and resolution as well as a distributional model of defect resolution. Illustrations compare the predictive and computational performance of alternative approaches. The results suggest that the simple distributional approach achieves the best tradeoff between these two performance measures.

I. Introduction

Software reliability growth models (SRGM) [1] complement the software testing process in order to quantify the decreasing trend in times between the discovery of defects and the corresponding increase in reliability. Myriad papers call out the limitations of these models such as the assumption made by nonhomogeneous Poisson process (NHPP) SRGM that discovered defects are corrected immediately and no additional defects introduced. The relative magnitude of papers offering incremental advancements overshadows more innovative studies that attempt to establish concrete connections to the activities performed as part of software engineering processes. Despite these efforts, theory and practice have diverged, missing opportunities to drive meaningful advances. Marginal modeling efforts that do not explicitly validate underlying theories and poor data collection practices are two primary impediments. Modelers need to characterize the activities performed by processes and practitioners need to implement controls to capture data more consistently and accurately. The ideal outcome is a virtuous cycle in which modelers equip practitioners with practical inferences of high concern to inform process improvement and practitioners enhance data collection to support the livelihood of modelers.

Early studies that have attempted to characterize software defect discovery and resolution include the work of Schneidewind [2] who modeled defect discovery with a discrete exponential mean value function and resolution with a time lag. Xie and Zhao [3] subsequently extended Schneidewind's model, assuming a defect resolution rate proportional to the number of defects discovered but not yet resolved and also demonstrated the applicability of the Poisson thinning process to model the difference between defects discovered but not yet resolved. Ohba [4] proposed the inflexion S-shaped model to characterize the fact that some defects may need to be discovered and resolved before others can be discovered, while Yamada and Osaki [5] introduced the delayed S-shaped model, which incorporates a time delay to model this dependence. Yamada et al. [6] proposed two SRGM with imperfect debugging, where new defects are sometimes introduced when resolving others.

More recent studies include the work of Huang et al. [7] showed how several existing SRGM can be derived by applying the time-dependent delay function, while Lo and Huang [8] proposed an integrated defect discovery and resolution process modeling framework, in which the defect resolution process is expressed in terms of a time-varying resolution intensity and the difference between the number of defects discovered and resolved. Inoue and Yamada [9] proposed a modeling framework for alternative debugging scenarios based on an absorbing continuous-time Markov chain to characterize the time between defect discovery and resolution. Cinque et al. [10] proposed debugging-workflow-

aware SRGM to leverage debugging data managed by companies in bug tracking systems in order to improve accuracy when debugging fails to completely satisfy modelling assumptions. Vizarreta et al. [11] found that the inflection S-shaped model best characterized the defect resolution of four releases of an open source software defined network controller and subsequently [12] quantified the cumulative distribution function of time to resolution according to defect severity.

This paper develops defect resolution models based on a data set from a NASA project [13]. Specifically, a differential-equation based model from the framework of Lo and Huang [8] as well as a distributional model of the defect lifecycle. These models are created from information within the defect tracking database. The distributional model employs censoring to estimate model parameters when some defects have been discovered but not yet resolved, which is necessary to apply the models and make predictions during the testing and defect resolution process. Illustrations compare the predictive accuracy and computational efficiency of the proposed models. Our results indicate that the distributional model achieves the fastest run time and lowest predictive error, suggesting it may be a suitable alternative to models based on a system of differential equations.

The remainder of the paper is organized as follows: Section II explains the need to distinguish defect discovery and resolution times, while Section III covers defect discovery and resolution models. Section IV illustrates the defect resolution models whereas Section V concludes and suggests future research directions.

II. Defect Discovery and Resolution: Concepts

For clarity of exposition, Figure 1 shows the less detailed case of defect times data in order to explain many of the core concepts contained in past software defect discovery and resolution models.

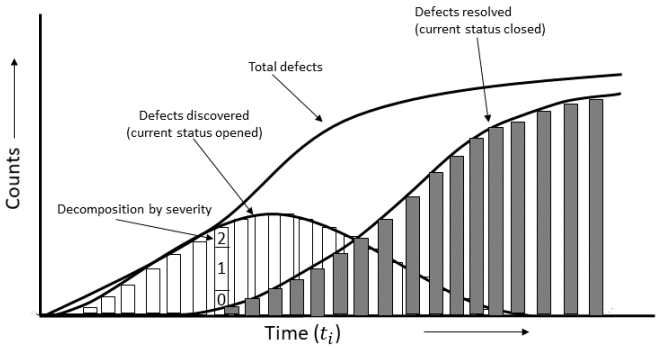

Fig. 1: Software defect discovery and resolution concepts

The upper curve in Figure 1 represents the total defects discovered by the end of the i^{th} interval, which is the cumulative number of defects ever discovered. This corresponds to the counting process $N(t)$ to which software reliability growth models are typically applied. However, basing reliability estimates on the non-discovery of additional defects makes strong implicit assumptions about the scope and completeness of testing. Therefore, a more pragmatic approach also employs the data in the bottom left of the figure with white bars, which represents the number of defects discovered but not yet resolved according to their severity, while the curve in the bottom right of the figure with grey bars represents the cumulative number of defects that have been both discovered and resolved $N_r(t)$. Ideally, the number of unresolved defects goes to zero soon after all defects have been discovered, since the goal of software testing is to both detect and remove defects. Thus, Figure 1 indicates that models that only consider $N(t)$ do not properly distinguish between detection and resolution and are therefore inadequate to assure software reliability. To support more detailed models of defect discovery and resolution a natural resource is the defect tracking database, which documents the lifecycle of individual defects, enabling explicit connections between the software process and activities performed. Toward this end, each model developed in the following section explains how data from the defect tracking database can be transformed in support of parameter estimation.

III. Defect Discovery and Resolution: Models

This section describes alternative methods to model the discovery and resolution of software defects recorded in a defect tracking database, including novel models based on the integrated defect discovery and resolution process modeling framework proposed by Lo and Huang [8] as well as a distributional resolution model developed here. These defect resolution models enable practical inferences of interest to the software practitioner that can be made throughout the software testing process, including the expected time required to remove (i) all defects discovered up to the present time t and (ii) all defects discovered up to the present time t as well as all defects anticipated to be discovered.

A. Integrated defect discovery and resolution processes

In the integrated defect discovery and resolution process modeling framework [8], the rate of defect discovery and resolution are respectively expressed as differential equations possessing forms

$$\frac{dm(t)}{dt} = \lambda(t)(a - m(t)) \tag{1}$$

and

$$\frac{dm_r(t)}{dt} = \lambda_r(t)(m(t) - m_r(t)) \tag{2}$$

where $m(t)$ ($m_r(t)$) denotes the mean value function of the number of defects discovered (resolved) by time t, $\lambda(t)$ ($\lambda_r(t)$) the defect discovery (resolution) rate, and $a > 0$ the number of defects that would be discovered with infinite testing. Thus, Equation (2) expresses the instantaneous rate of change in defect resolution as the product of the defect resolution intensity multiplied by the defects discovered but not yet resolved by time t.

Lo and Huang [8] derived general forms of the solutions for Equations (1) and (2) with initial conditions $m(t) = 0$ and

$m_r(t) = 0$ to produce expressions for the mean value function of defect discovery and resolution. They subsequently applied these general forms to explicitly show that $\lambda(t) = \lambda_r(t) = b$ is the special case where the defect discovery and resolution models are characterized by the Goel-Okumoto [14] and Yamada Delayed S-shaped models [5] respectively. They also derived a novel model with unequal discovery and resolution rates $\lambda(t) = b$ and $\lambda_r(t) = c$. However, additional discovery and resolution models for the inflection S-shaped [4] as well as exponential and Rayleigh [15] and Weibull [16] testing effort functions were only specified implicitly in terms of $\lambda(t)$ and $\lambda_r(t)$.

We applied the Goel-Okumoto, Weibull, Yamada Delayed S-shaped, inflection S-shaped, Jelinski-Moranda [17], and Geometric model [18] to a recent NASA data set [13] and found that the inflection S-shaped model best characterized the defect discovery process. Therefore, we derived two explicit forms of $m_r(t)$ with Equation (2), assuming defect discovery is modeled by the inflection S-shaped model, possessing mean value function

$$m(t) = a\frac{1 - e^{-bt}}{1 + ce^{-bt}} \tag{3}$$

where, b is the constant defect discovery rate, c is the inflection parameter

$$c = \frac{1 - r}{r}, \qquad r \in (0, 1]. \tag{4}$$

and r is the inflection rate. As r approaches 1.0, the inflection S-shaped model reduces to $m(t) = a(1 - e^{-bt})$, which is the form of the Goel-Okumoto model.

The first form of our defect resolution model assumes defect resolution intensity $\lambda_r(t) = b$, is the same as parameter b present in Equation (3), producing

$$m_r^b(t) = a\left(1 - e^{-bt} + (1 + c)\log\left(\frac{1 + c}{c + e^{bt}}\right)e^{-bt}\right) \tag{5}$$

The second form of our defect resolution model introduces an additional parameter through $\lambda_r(t) = d$, which implies that the defect resolution intensity is distinct from parameter b in Equation (3), producing

$$\begin{aligned} m_r^d(t) \quad = \quad & \frac{a}{c(b + d)}\left(d\ {}_2F_1\left(1, \frac{b + d}{b}; 2 + \frac{d}{b}; -\frac{1}{ce^{-bt}}\right)e^{bt}\right. \\ & -(b + d)\ {}_2F_1\left(1, \frac{d}{b}; \frac{b + d}{b}; -\frac{1}{ce^{-bt}}\right) \\ & +(b + d)\ {}_2F_1\left(1, \frac{d}{b}; \frac{b + d}{b}; -\frac{1}{c}\right)e^{-dt} \\ & \left.-d\ {}_2F_1\left(1, \frac{b + d}{b}; 2 + \frac{d}{b}; -\frac{1}{c}\right)e^{-dt}\right) \end{aligned} \tag{6}$$

where ${}_2F_1(a, b; c, z) = \sum_{k=0}^{\infty} \frac{(a)_k (b)_k}{(c)_k} \frac{z^k}{k!}$ is the hypergeometric function with $(q)_k = q(q + 1)\ldots(q + k - 1)$ for $k > 0$.

Given a defect database possessing discovery and resolution times, $\mathbf{T} = \langle t_1, t_2, \ldots, t_n\rangle$ and $\mathbf{T}^r = \langle t_1^r, t_2^r, \ldots, t_n^r\rangle$, the defect detection model in Equation (3) is applied with $\mathbf{T}$, while the defect resolution models in Equations (5) and (6) may be applied directly to $\mathbf{T}^r$.

B. Distributional approach

An alternative and potentially simpler method to model the time to correct defects is to compute the times between discovery and resolution and fit a distribution to these statistics. To make resolution time predictions possible throughout the software testing process, it is necessary to rely on the discovery and resolution times available up until time t. However, at any specific time, the number of defects resolved may be strictly less than the number of defects discovered. This discrepancy requires the use of maximum likelihood estimation techniques for censored data [19]. Specifically, the problem of identifying a distribution to best fit the defect resolution times, when only a subset of the defects discovered has been resolved, may be estimated from the following likelihood function

$$Lik = \prod_{i \in \mathcal{R}}^{k} f(t_{(i)}; \theta) \prod_{i \in \mathcal{D}}^{n-k} 1 - F(T - t_i; \theta) \tag{7}$$

where k is the number of defects in the set $\mathcal{R}$ discovered and resolved before time T, $(n - k)$ is the number of defects in the set $\mathcal{D}$ that have been discovered by time T but not yet resolved, and θ is the vector of parameters of the distribution being fit to the data. Moreover, $t_{(i)}$ denotes the time between discovery (t_i) and resolution (t_i^r) of the i^{th} defect such that $t_{(i)} = t_i^r - t_i$, whereas the term $1 - F(T - t_i)$ is the probability that a defect discovered at time t_i has not been resolved by time T.

In practice, Equation (7) can be maximized with multiple alternative distributions and the one achieving the best goodness of fit employed to make predictions. One may then compute the mean of this distribution of best fit, which can be interpreted as the mean time to resolve defects. In this manner, the MVF of the number of defects resolved by time t may be expressed as

$$m_r(t) = m(t - E[T_r]) \tag{8}$$

which translates the mean value function of the defect discovery process to the right by the mean time to resolve defects ($E[T_r]$).

IV. Illustrations

This section illustrates the approaches to defect resolution modeling developed in Section III in order to assess their relative predictive ability and computational efficiency.

To emphasize the need for software reliability modeling that can be used during the defect and resolution process, two cases are considered: (i) retrospective analysis, characteristic of many historical defect discovery modeling papers, which used all available data and (ii) online analysis, which periodically updates estimates throughout this process to refine estimates and track progress. Section IV-A presents a comparative retrospective analysis of the alternative defect resolution model and explains the steps taken to apply the

Distributional approach. Section IV-B presents online predictive analysis, discussing the tradeoffs between model accuracy and computational efficiency.

A. Defect resolution models

This example examines the accuracy of the alternative defect resolution modeling approaches.

1) Retrospective analysis: Figure 2 shows the defect discovery and resolution counting processes ($N(t)$ and $N_r(t)$) along with the corresponding fitted defect and resolution models as well as the distributional approach.

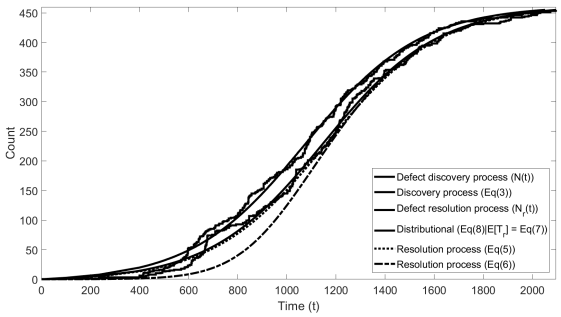

Fig. 2: Defect discovery and resolution processes with fitted defect models

The first counting process (left step function) indicates the times at which the defects were discovered and is accompanied by the fit of the inflexion S-shaped model. The second counting process (right step function) corresponds to the times at which defects were resolved.

For each of the defect resolution modeling approaches, Table I summarizes the sum of squares error (SSE) and runtime in seconds.

TABLE I: Comparative analysis of defect resolution intensity

Approach	SSE	Runtime
Resolution process $(m_r^b(t))$	3.77×10^4	175.2
Resolution process $(m_r^d(t))$	2.96×10^5	1627.2
Distributional	$\mathbf{3.43 \times 10^4}$	**150.1**

The model with the lowest SSE and runtime, namely the distributional approach, are shown in bold.

The maximum likelihood estimates of $m_r^b(t)$ were $\hat{a} = 465.13$, $\hat{b} = 0.004855$, and $\hat{c} = 101.605$, while the maximum likelihood estimates of $m_r^d(t)$ were $\hat{a} = 462.617$, $\hat{b} = 0.004191$, $\hat{c} = 167.91$, and $\hat{d} = 0.656773$. Thus, the unique resolution rate d in Equation (6) contributes to the poor quality of the model fit because of the higher masking term c, which corresponds to visible underprediction of the number of defects resolved in the interval $t = (600, 1000)$ of Figure 2.

The mean time from defect discovery to resolution of the distributional approach was $E[T_r] = 59.74$, which was determined with the special case of Equation (7), where all data up to the time of the resolution of the n^{th} defect was available. Thus, the accuracy and efficiency of the distributional approach may offer a competitive alternative to the resolution process $(m_r^b(t))$ of Equation (5).

2) Distributional approach: Seventeen possible distributions were considered, explaining why the calculation in Table I required 150 seconds. These distributions included the Beta, Birnbaum-Saunders, Exponential, Extreme value, Gamma, Generalized extreme value, Generalized Pareto, Inverse Gaussian, Logistic, Log-logistic, Lognormal, Nakagami, Normal, Rayleigh, Rician, t location-scale, and Weibull distributions. The generalized extreme value (GEV) distribution possessing the following maximum likelihood estimates $T_r \sim GEV(\hat{\mu} = 57.4437, \hat{\sigma} = 22.6722, \hat{\xi} = -0.0959)$ attained the best fit to the empirical times between defect discovery and resolution according to AIC [20] and BIC [21] measurements, as shown in Figure 3.

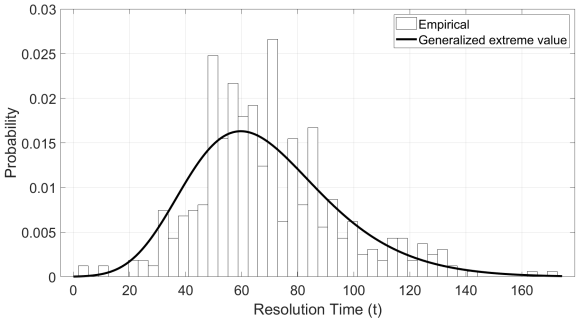

Fig. 3: Empirical distribution of time between defect discovery and resolution

However, this retroactive approach is only useful for model assessment after all of the data has been collected and remains a pervasive problem in the defect discovery modeling literature, namely the over reliance on in sample measures of goodness of fit and the lack of online procedures that can be employed during the defect discovery and resolution process, which are explicitly addressed in Section IV-B.

B. Comparison of online predictive accuracy and computational efficiency

To further compare the alternative defect resolution models, this section conducts an online assessment of predictive accuracy, which also explicitly considers the computational efficiency of each approach. Starting at time $t = 200$, the defect resolution processes of Equations (5) and (6) were fit directly to the resolution time data extracted from the defect tracking database, while the distributional approach first identified an SRGM that best fit the available defect detection data and then estimated the mean time to resolve defects ($E[T_r]$) with Equation (7). In this manner each model, was used to predict the cumulative number of defects that would be resolved 200 time units into the future or $t = 400$ and the percentage error computed between the actual number of defects observed and the predicted values. This process

was repeated periodically for each model. However, Table I determined that the model with unique defect resolution rate ($m_r^d(t)$) was over an order of magnitude slower than the distributional approach. Thus, the frequency of predictions made by each model was chosen to be inversely portion to their runtime. This method was implemented to allocate approximately equal computational time to each method.

Figure 4 shows the results of online assessment of predictive accuracy explicitly considering computational efficiency.

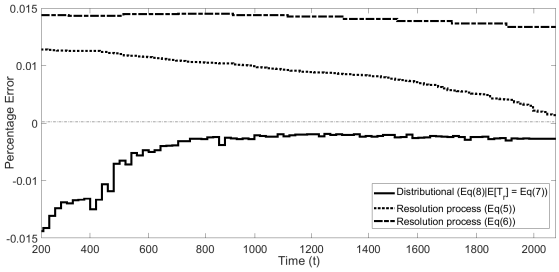

Fig. 4: Comparison of online predictive accuracy

Figure 4 indicates that the distributional approach achieves the lowest error as early as $t = 300$ and remains the most accurate throughout the remainder of the defect testing and resolution process. Moreover, the computational efficiency enables frequent updates, supporting regular online assessment throughout. Unlike the defect resolution processes, which rely solely on the defect resolution count data, the distributional approach explicitly incorporates information on defects discovered and resolved as well as those discovered but not yet resolved.

V. Conclusions and Future Research

This paper connect software reliability growth models to defect tracking databases. Differential-equation based and distributional approaches were developed. A NASA defect tracking database was employed for this purpose. Models were applied to the full data, referred to as retrospective analysis, and the steps taken to apply the distributional approach explained. Finally, the predictive accuracy and computational efficiency were assessed in an online manner. The results suggested that most of the models fit the defect resolution data well, but that the distributional approach achieved the best fit and also demonstrated the lowest runtime, enabling more frequent updates, which would support online tracking when defect detection and resolution are ongoing. The models demonstrate the potential benefit of efforts to collect high quality data related to the defect tracking lifecycle.

Possible future research includes developing queueing theoretic models of defect discovery and resolution and comparing their accuracy and computational performance to the models presented here.

Acknowledgments

This work was supported by the National Aeronautics and Space Administration (NASA) under Grant Number (#80NSSC20K0276) and National Science Foundation under Grant Number (#1749635). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

[1] W. Farr, *Handbook Of Software Reliability Engineering*. New York, NY: McGraw-Hill, 1996, chapter Software Reliability Modeling Survey, pp. 71–117.

[2] N. Schneidewind, "Analysis of error processes in computer software," in *Proc. ACM Sigplan Notices*, vol. 10, no. 6, 1975, pp. 337–346.

[3] M. Xie and M. Zhao, "The Schneidewind software reliability model revisited," in *Proc. IEEE International Symposium on Software Reliability Engineering*, 1992, pp. 184–192.

[4] M. Ohba, "Inflection S-shaped software reliability growth model," in *Stochastic models in reliability theory*. Springer, 1984, pp. 144–162.

[5] S. Yamada, M. Ohba, and S. Osaki, "S-shaped reliability growth modeling for software error detection," *IEEE Transactions on Reliability*, vol. 32, no. 5, pp. 475–484, 1983.

[6] S. Yamada, K. Tokuno, and S. Osaki, "Imperfect debugging models with fault introduction rate for software reliability assessment," *International Journal of Systems Science*, vol. 23, no. 12, pp. 2241–2252, 1992.

[7] C.-Y. Huang, C.-T. Lin, S.-Y. Kuo, M. Lyu, and C. Sue, "Software reliability growth models incorporating fault dependency with various debugging time lags," in *Proc. IEEE International Computer Software and Applications Conference*, 2004, pp. 186–191.

[8] J.-H. Lo and C.-Y. Huang, "An integration of fault detection and correction processes in software reliability analysis," *Journal of Systems and Software*, vol. 79, no. 9, pp. 1312–1323, 2006.

[9] S. Inoue and S. Yamada, "Debugging process oriented software reliability models and their goodness-of-fit," in *Proc. IEEE International Conference on Industrial Engineering and Engineering Management*, 2018, pp. 1150–1154.

[10] M. Cinque, D. Cotroneo, A. Pecchia, R. Pietrantuono, and S. Russo, "Debugging-workflow-aware software reliability growth analysis," *Software Testing, Verification and Reliability*, vol. 27, no. 7, p. e1638, 2017.

[11] P. Vizarreta, K. Trivedi, B. Helvik, P. Heegaard, A. Blenk, W. Kellerer, and C. M. Machuca, "Assessing the maturity of sdn controllers with software reliability growth models," *IEEE Transactions on Network and Service Management*, vol. 15, no. 3, pp. 1090–1104, 2018.

[12] P. Vizarreta, E. Sakic, W. Kellerer, and C. M. Machuca, "Mining software repositories for predictive modelling of defects in sdn controller," in *Proc. IFIP/IEEE Symposium on Integrated Network and Service Management*, 2019, pp. 80–88.

[13] H. Sukhwani, J. Alonso, K. Trivedi, and I. Mcginnis, "Software reliability analysis of NASA space flight software: A practical experience," in *Proc. IEEE International Conference on Software Quality, Reliability and Security*, 2016, pp. 386–397.

[14] A. Goel and K. Okumoto, "Time-dependent error-detection rate model for software reliability and other performance measures," *IEEE Transactions on Reliability*, vol. 28, no. 3, pp. 206–211, aug 1979.

[15] S. Yamada, H. Ohtera, and H. Narihisa, "Software reliability growth models with testing-effort," *IEEE Transactions on Reliability*, vol. 35, no. 1, pp. 19–23, 1986.

[16] S. Yamada, J. Hishitani, and S. Osaki, "Software-reliability growth with a Weibull test-effort: A model & application," *IEEE Transactions on Reliability*, vol. 42, no. 1, pp. 100–106, mar 1993.

[17] Z. Jelinski and P. Moranda, "Software reliability research," *Statistical computer performance evaluation*, pp. 465–484, 1972.

[18] P. Moranda, "Prediction of software reliability during debugging," in *Proc. Annual Reliability and Maintainability Symposium*, 1975.

[19] I. Gertsbakh, *Reliability Theory: With Applications to Preventive Maintenance*. Berlin: Springer, 2000.

[20] H. Akaike, "A new look at the statistical model identification," *IEEE Transactions on Automatic Control*, vol. 19, no. 6, pp. 716–723, 1974.

[21] G. Schwarz, "Estimating the dimension of a model," *The Annals of Statistics*, vol. 6, no. 2, pp. 461–464, 1978.