A Software Reliability Model with Dependent Failure and Optimal Release Time

Youn Su Kim 1, Kwang Yoon Song 1, Hoang Pham 2 and In Hong Chang 1,*

1 Department of Computer Science and Statistics, Chosun University, 309 Pilmun-daero, Dong-gu, Gwangju 61452, Korea; imk92315@chosun.kr (Y.S.K.); csssig84@gmail.com (K.Y.S.)
2 Department of Industrial and Systems Engineering, Rutgers University, 96 Frelinghuysen Road, Piscataway, NJ 08855-8018, USA; hopham@rci.rutgers.edu
* Correspondence: ihchang@chosun.ac.kr; Tel.: +82-62-230-6621

Abstract: In the past, because computer programs were restricted to perform only simple functions, the dependence on software was not large, resulting in relatively small losses after a failure. However, with the development of the software market, the dependence on software has increased considerably, and software failures can cause significant social and economic losses. Software reliability studies were previously conducted under the assumption that software failures occur independently. However, as software systems become more complex and extremely large, software failures are becoming frequently interdependent. Therefore, in this study, a software reliability model is developed under the assumption that software failures occur in a dependent manner. We derive the software reliability model through the number of software failure and fault detection rate assuming point symmetry. The proposed model proves good performance compared with 21 previously developed software reliability models using three datasets and 11 criteria. In addition, to find the optimal release time, a cost model using the developed software reliability model was presented. To determine this release time, four parameters constituting the software reliability model were changed by 10%. By comparing the change in the cost model and the optimal release time, it was found that parameter \( b \) had the greatest influence.

Keywords: non-homogeneous Poisson process; dependence failure; software reliability; software reliability model; cost model

1. Introduction

Software, one of the main components of a computer, plays an important role in the operation of physical devices. Software was originally developed with the ability to perform extremely small or simple functions. Currently, however, embedded systems that perform multiple functions are being developed. With the rapid development of the software market, technology has also developed, and software is now being used in all fields. Recently, the Internet of Things (IoT) based on the combination of various software, has been commercialized. Furthermore, AIoT (Artificial Intelligence of Things) combined AI (Artificial Intelligence) with IoT (Intelligence of Things) is developing [1]. It means that software has become a very important part not only in the industrial field but also in our daily life.

A software failure is caused by various faults (coding or system errors, etc.). In the past, software failures caused relatively small losses because the degree of software dependence was not as large. However, in today’s world, the degree of dependence on software is extremely high, and thus software failures can cause significant social and economic losses. Therefore, we measured the software reliability, which indicates the ability of a software program to avoid failure for a set period of time and refers to how long the software can be used without such a failure.
Early research on software reliability was conducted based on the assumption that software failures occur independently. Goel and Okumoto proposed the GO model, which is the most basic non-homogeneous Poisson process (NHPP) software reliability growth model [2]. The Hossain, Dahiya, Goel and Okumoto (HDGO) model further extended the GO model [3]. Yamada et al., Ohba, and Zhang et al. [4–6] proposed an NHPP S-shaped curve model in which the cumulative number of software failures increases to the S curve. In addition, Yamada et al. [7] proposed a new model in which the test effort invested during phase was reflected in the software reliability model. It is a model that reflects even the resources consumed for testing in the previously developed model. Furthermore, Yamada et al. [8] developed a software reliability model with a constant fault detection rate of \( b(t) = b \), assuming incomplete debugging, in which faults detected during the test phase were corrected and removed.

The model developed by extending the above approach involved a generalized incomplete debugging-error detection rate model. Here, the fault detection rate \( b(t) \) of the model is not a constant but rather a different function [9–14]. It started from the software error causing the failure being immediately eliminated and so a new error can be generated [9]. It progressed to that during the fault removal process, whether the fault is removed successfully or not, new faults are generated with a constant probability [13,14]. In addition, because the operating environment of the software is operated differently for each software program, a comparison is difficult to achieve. Therefore, in [15–17], a software reliability model was developed considering uncertain factors in the operating environment. Currently, research using non-parametric methods such as deep learning or machine learning is also being conducted [18–21].

Recently, finding the most optimal model for reliability prediction is an important concern. Through combination of analytic hierarchy method (AHP), hesitant fuzzy (HF) sets and techniques for order of preference by similarity to ideal solution (TOPSIS), Sahu et al., Ogundoyin et al., and Rafi et al. [22–24] found the most optimal software reliability model. However, software failures often occur in a dependent manner because the developed software is composed of extremely complex structures [25]. Here, the dependent failure means that one failure affects other failures or increases the failure probability of other equipment [26,27]. There are two main types of dependent failure. A common cause failure is when several pieces of software fail simultaneously due to a common cause, and a cascading failure is a case in which a part of the system fails and affects other software as well. A software reliability model assuming a dependent failure was developed from the number of software failures and the fault detection rate, which have a dependency relationship in a software reliability model assuming incomplete debugging [28]. In addition, Lee et al. [29] presented a model that assumes that if past software failures are not corrected well, they will continue to have an effect.

In this study, a new software reliability model is developed under the assumption that software failures occur in a dependent manner. It is suitable for a general environment. We show the superiority of the newly developed dependent software reliability model through a comparison under various criteria. In addition, determining the optimal release time of the developed software is also important. If the test period is long, the software will be reliable, but the software development cost will increase. If the test period is short, the reliability of the product may decrease. Therefore, it is important to find a balance between time to market and minimum cost taking into account the installation costs, test costs, and error removal costs, etc. We propose a cost model that combines the proposal software reliability model and the cost coefficient [30–33]. In addition, among the various parameters of the proposed model, we propose a parameter that has a significant influence on predicting the number of cumulative failures through a variation in the cost model for changes in the parameters [34,35]. Section 2 introduces a new dependent software reliability model and its mathematical derivation. Section 3 introduces the data and criteria, as well as numerical results. Section 4 describes the optimal release time, and finally, in Section 5, we present our conclusions.
2. New Dependent Software Reliability Model

Software reliability refers to the probability that the software will not fail a system for a certain period of time under certain conditions. In other words, it evaluates “how long the software can be used without failure”. The reliability function used to evaluate this is as follows:

\[ R(t) = P(T > t) = \int_t^\infty f(u)du \]  

(1)

This denotes the probability of the software operating without failure over a specific time \( t \). Here, the probability density function \( f(t) \) assumes the software failure time or lifetime as a random variable \( T \). When measuring reliability function \( R(t) \), it is assumed that it follows an exponential distribution with parameter \( \lambda \). In addition, it is assumed that the number of failures occurring in given unit time is a Poisson distribution with parameter \( \lambda \). When \( \lambda \) is a constant, it is the most basic form, and is called a homogeneous Poisson process. Extending this process, many researchers adapt a model where \( \lambda \) is an intensity function \( \lambda(t) \) that changes with time rather than a single constant by setting \( \lambda \) as a non-homogeneous Poisson process (NHPP) rather than as a homogeneous Poisson process.

\[ \Pr\{N(t) = n\} = \frac{(m(t))^n}{n!}e^{-m(t)}, n = 0, 1, 2, \ldots, t \geq 0 \]  

(2)

In Equation (2), \( N(t) \) is the Poisson probability density function with the time dependent parameter \( m(t) \). The \( m(t) \) is a mean value function which is the integral of \( \lambda(t) \) from 0 to \( t \) in Equation (3). The \( \lambda(t) \) is the intensity function indicating the number of instantaneous failures at time \( t \).

\[ m(t) = E[N(t)] = \int_0^t \lambda(s)ds \]  

(3)

A general class of NHPP software reliability models was proposed by Equation (9) to summarize the existing NHPP models as follows:

\[ \frac{dm(t)}{dt} = b(t)[a(t) - m(t)] \]  

(4)

where, the \( m(t) \) is calculated using the relationship between the number of failures \( a(t) \) at each time point and a fault detection rate \( b(t) \) assuming point symmetry in Equation (4). Various software reliability models have been developed based on the assumption that software failures occur independently.

However, software failures occur not only independently but also dependently. If the failure is not completely fixed, it will continue to affect the next failure. In addition, as the system becomes more complex, the relationship between failure and failure also shows the dependent relationship because of the dependent combination of several software. Therefore, in this study, we assume that failure is dependent on other failures. The mean value function \( m(t) \) based on NHPP software reliability model using the differential equation is as follows:

\[ \frac{dm(t)}{dt} = b(t)[a(t) - m(t)]m(t) \]  

(5)

In Equation (5), the \( m(t) \) is multiplied once more to assume that the failure occurring from 0 to \( t \) affects another failure. We assume:

\[ a(t) = a(1 + at), b(t) = \frac{b}{1 + ce^{-bt}} \]  

(6)

where, \( a(t) \) is the number of software failures at each time point and \( b(t) \) is the fault detection rate. Parameter \( a \) is the expected number of faults, \( a \) is the increasing rate of the number of faults, \( b \) is the shape parameter, and \( c \) is the scale parameter. When time \( t \) changes, the change according to the values of parameters \( b \) and \( c \) in the fault detection
rate \( b(t) \) are as shown in Figure 1. When \( b \) is 1, it is blue, and when it is 1.5, it is red. In addition, when \( c \) is 1, it is a dashed line, and when it is 2, it is a dotted line. It can be seen that the larger \( b \) is, the larger the \( b(t) \) is.

![Graph showing the changes in parameters \( b \) and \( c \).](image_url)

**Figure 1.** \( b(t) \) according to the changes in parameters \( b \) and \( c \).

When solving the differential equation by substituting \( a(t) \) and \( b(t) \) for \( m(t) \) in Equations (5) and (6), we obtain Equation (7):

\[
m(t) = \frac{\left( c + e^{bt} \right)^a e^{a(t)Li_2 \left( \frac{c}{c + e^{bt}} \right)}}{\int_0^t \left( c + e^{bt} \right)^a e^{a(t)Li_2 \left( \frac{c}{c + e^{bt}} \right)}} dt + C
\]

where \( Li_s(x) = \sum_{n=1}^{\infty} \frac{x^n}{n^s} \) is a polylogarithm when \( s = 2 \). At this time, \( \alpha = 0 \) in \( a(t) \).

\[
m(t) = \frac{h \left( c + e^{bt} \right)^a}{h \int_0^t \left( c + e^{bt} \right)^a e^{btx} dx + (1 + c)^a}
\]

where \( h \) is the number of initial failures. In Equation (8), \( \int_0^t \frac{(c + e^{bt})^a e^{btx}}{c + e^{bt}} dx \) is calculated through an integration using substitution. When \( u = c + be^{tx} \) and \( du = be^{tx}dx \), it is the same as in Equation (9).

\[
\int_0^t \frac{(c + e^{bt})^a e^{btx}}{c + e^{bt}} dx = \int_{c+b}^{c+be^{bt}} \frac{u^a}{u} du = \int_{c+b}^{c+be^{bt}} u^{a-1} du = \frac{u^a}{a} = \frac{(c + e^{bt})^a}{a}
\]

Substituting the result of the substitution integration into Equation (8), the final \( m(t) \) is given by Equation (10).

\[
m(t) = \frac{a}{1 + \frac{a}{a + h} \left( \frac{c + e^{bt}}{c + e^{bt}} \right)^a}
\]

This can be presented as a general model of a dependent failure occurrence in the software reliability model. When \( t = 0 \), \( m(0) = ah / (a + h) \). Table 1 shows the value of \( m(t) \) of the existing software reliability model and the model proposed in this study. From models 19–22, it is assumed that a failure occurs in a dependent manner.
Table 1. Software reliability models.

| No. | Model                  | Mean Value Function | Note     |
|-----|------------------------|---------------------|----------|
| 1   | Goel-Okumoto (GO) [2]  | $m(t) = a \left( 1 - e^{-bt} \right)$ | Concave  |
| 2   | Hossain-Dahiya (HDGO) [3] | $m(t) = \log \left( \frac{e^{(c-a)t}}{e^{-(c-a)t} - e^{-bt}} \right)$ | Concave  |
| 3   | Yamada et al. (DS) [4] | $m(t) = a \left( 1 - (1 + bt)e^{-bt} \right)$ | S-Shape  |
| 4   | Ohba (IS) [5]          | $m(t) = a\left((1-e^{-it}) \right)$ | S-Shape  |
| 5   | Zhang et al. (ZFR) [6] | $m(t) = \frac{a}{b} \left[ 1 - \frac{(1+\alpha)e^{-bt}}{1+\alpha e^{bt}} \right]^{t \left[p-\beta \right]}$ | S-Shape  |
| 6   | Yamada et al. (YE) [7] | $m(t) = a \left( 1 - e^{-\gamma t}(1-e^{-bt}) \right)$ | Concave  |
| 7   | Yamada et al. (YR) [7] | $m(t) = a \left( 1 - e^{-\gamma t}(1-e^{-bt/2}) \right)$ | S-Shape  |
| 8   | Yamada et al. (YID 1) [8] | $m(t) = \frac{ab}{\beta \gamma} \left( e^{\alpha t} - e^{bt} \right)$ | Concave  |
| 9   | Yamada et al. (YID 2) [8] | $m(t) = a \left( 1 - e^{-bt} \right) \left( 1 - \frac{\gamma}{\beta} \right) + aab$ | Concave  |
| 10  | Pham-Zhang (PZ) [9]    | $m(t) = \frac{(c+a)[1-e^{-bt}]-\frac{a\beta}{b}}{1+\text{e}^{bt}-e^{-bt}}\frac{e^{bt}}{1+\text{e}^{bt}}$ | Both     |
| 11  | Pham et al. (PNZ) [10] | $m(t) = a\left((1-e^{-it})(1+\frac{t}{\beta})+aat \right)$ | Both     |
| 12  | Teng-Pham (TP) [11]    | $m(t) = \frac{a}{b} \left[ 1 - \left( \frac{bt}{\beta + (t-q)\ln \left( \frac{bt}{\beta} \right)} \right)^a \right]$ | S-Shape  |
| 13  | Kapur et al. (KSIRM) [12] | $m(t) = \frac{A}{1-a} \left[ 1 - \left( \frac{bt}{\beta + (t-q)\ln \left( \frac{bt}{\beta} \right)} \right)^a \right]$ | S-Shape  |
| 14  | Roy et al. (RMD) [13]  | $m(t) = a\left( 1 - e^{-bt} \right) \left( \frac{bt}{\beta} + \left( e^{bt} - e^{-bt} \right) \right)$ | Concave  |
| 15  | Pham (IFD) [14]        | $m(t) = a \left( 1 - e^{-bt} \right) \left( 1 + bt + \frac{bt^2}{2} \right)$ | Concave  |
| 16  | Pham (VTub) [15]       | $m(t) = N \left[ 1 - \left( \frac{\beta}{\beta + (bt)^a} \right)^a \right]$ | S-Shape  |
| 17  | Chang et al. (TC) [16] | $m(t) = N \left[ 1 - \left( \frac{\beta}{\beta + (bt)^a} \right)^a \right]$ | Both     |
| 18  | Song et al. (3P) [17]  | $m(t) = N \left[ 1 - \left( \frac{\beta}{\beta + (bt)^a} \right)^a \right]$ | S-Shape  |
| 19  | Pham (DP1) [28]        | $m(t) = a(a + \beta t) \left( \beta+ \left( e^{-\beta t} - 1 \right) \right)$ | Concave, Dependent |
| 20  | Pham (DP2) [28]        | $m(t) = m_0 \left( \frac{2t+1}{2t+3} \right) e^{-\gamma(t-t_0)} + a(\gamma t+1)(\gamma t+1 - 1 - \gamma t_0) e^{-\gamma(t-t_0)}$ | Concave, Dependent |
| 21  | Lee et al. (DPF) [29]  | $m(t) = \frac{a}{1+t\left( \frac{bt}{\beta} \right)}$ | S-Shape, Dependent |
| 22  | Proposed Model         | $m(t) = \frac{a}{1+t\left( \frac{bt}{\beta} \right)}$ | S-Shape, Dependent |

3. Numerical Examples

3.1. Data Information

Datasets 1 and 2 are derived from the online communication system (OCS) of ABC Software Co. and uses data accumulated over a 12-week period. Datasets 1 and 2 show that the cumulative number of failures at $t = 1, 2, \cdots, 12$ is 14, 17, 17, 17, 17, 17, respectively [14]. Dataset 3 is the test data of a medical record system consisting of 188 software titles and data for one of three releases. It shows that the cumulative number of failures is 90, 107, 107, 204 for $t = 1, 2, \cdots, 3$, respectively [36]. Table 2 shows the accumulated failure data for datasets 1, 2, and 3. We compare the fit between the software reliability models with two failure datasets obtained from OCS and one dataset from Lee et al. [29], which showed good performance as the dependent models (DPF).
Table 2. Cumulative number of software failure datasets.

| Index | Dataset 1 Failures | Cumulative Failures | Dataset 2 Failures | Cumulative Failures | Dataset 3 Failures | Cumulative Failures |
|-------|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|
| 1     | 14                 | 14                 | 11                 | 11                 | 90                 | 90                 |
| 2     | 3                  | 17                 | 6                  | 17                 | 17                 | 107                |
| 3     | 4                  | 21                 | 0                  | 17                 | 19                 | 126                |
| 4     | 7                  | 28                 | 5                  | 22                 | 19                 | 145                |
| 5     | 7                  | 35                 | 5                  | 27                 | 26                 | 171                |
| 6     | 18                 | 53                 | 25                 | 52                 | 17                 | 188                |
| 7     | 8                  | 61                 | 10                 | 62                 | 1                  | 189                |
| 8     | 4                  | 65                 | 6                  | 68                 | 1                  | 190                |
| 9     | 2                  | 67                 | 2                  | 70                 | 0                  | 190                |
| 10    | 9                  | 76                 | 10                 | 80                 | 0                  | 190                |
| 11    | 1                  | 77                 | 0                  | 80                 | 2                  | 192                |
| 12    | 4                  | 81                 | 1                  | 81                 | 0                  | 192                |
| 13    |                    |                    |                    |                    | 0                  | 192                |
| 14    |                    |                    |                    |                    | 0                  | 192                |
| 15    |                    |                    |                    |                    | 11                 | 203                |
| 16    |                    |                    |                    |                    | 0                  | 203                |
| 17    |                    |                    |                    |                    | 1                  | 204                |

3.2. Criteria

This study compares various independent and dependent software reliability models and the proposed model introduced Table 1 using 11 criteria. Based on the difference between the actual observed value and the estimated value, we would like to find a better model by comparing it with criteria reflecting the number of parameters used in each model.

First, the mean squared error (MSE) is defined as the sum of squares of the distance between the estimated value and the actual value when considering the number of parameters and the number of observations [37].

\[
MSE = \frac{1}{n-m} \sum_{i=1}^{n} (\hat{m}(t_i) - y_i)^2 
\]  

where \( \hat{m}(t_i) \) is the estimated value of the model \( m(t) \), \( y_i \) is the actual observed value, \( n \) is the number of observations, and \( m \) is the number of parameters in each model.

Second, the mean absolute error (MAE) defines the difference between the estimated number of failures and the actual value considering the number of parameters and the number of observations as the sum of the absolute values [38].

\[
MAE = \frac{1}{n-m} \sum_{i=1}^{n} |\hat{m}(t_i) - y_i| 
\]  

Third, Adj_\( R^2 \) is the modified coefficient of determination of the regression equation and determines how much explanatory power it has in consideration of the number of parameters [39].

\[
R^2 = 1 - \frac{\sum_{i=1}^{n} (\hat{m}(t_i) - y_i)^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2}, \quad \text{Adj}_\text{\textit{R}^2} = 1 - \frac{(1 - R^2)(n-1)}{n-m-1} 
\]  

Fourth, the predictive ratio risk (PRR) is obtained by dividing the distance from the actual value to the estimated value by the estimated value in relation to the model estimation [40].

\[
\text{PRR} = \frac{1}{n-m} \sum_{i=1}^{n} \left( \frac{\hat{m}(t_i) - y_i}{\hat{m}(t_i)} \right)^2 
\]
Fifth, the predictive power (PP) is obtained by dividing the distance from the actual value to the estimated value by the actual value [41].

\[
PP = \frac{\sum_{i=1}^{n} \left( \frac{\hat{m}(t_i) - y_i}{y_i} \right)^2}{n-1}
\] (15)

Sixth, Akaike’s information criterion (AIC) was used to compare likelihood function maximization. This is applied to maximize the Kullback–Leibler level between the probability distribution of the model and the data [42].

\[
AIC = -2 \log L + 2m
\]
\[
L = \prod_{i=1}^{n} \frac{(m(t_i) - m(t_{i-1}))^{y_i-y_{i-1}} e^{-(m(t_i)-m(t_{i-1}))}}{(y_i-y_{i-1})!}
\]
\[
\log L = \sum_{i=1}^{n} \left\{ (y_i - y_{i-1}) \ln(m(t_i) - m(t_{i-1})) - (m(t_i) - m(t_{i-1})) - \ln((y_i - y_{i-1})!) \right\}
\] (16)

Seventh, the predicted relative variation (PRV) is the standard deviation of the prediction bias and is defined as [43]

\[
PRV = \sqrt{\frac{\sum_{i=1}^{n} (y_i - \hat{m}(t_i) - \text{Bias})^2}{n-1}}
\] (17)

Here, the bias is \[ \frac{\sum_{i=1}^{n} [\hat{m}(t_i) - y_i]}{n} \].

The root mean square prediction error (RMSPE) can estimate the closeness with which the model predicts the observation [44]:

\[
RMSPE = \sqrt{\text{Variance}^2 + \text{Bias}^2}
\] (18)

Ninth, the mean error of prediction (MEOP) sums the absolute value of the deviation between the actual data and the estimated curve and is defined as [38]

\[
MEOP = \frac{\sum_{i=1}^{n} |\hat{m}(t_i) - y_i|}{n - m + 1}
\] (19)

Tenth, the Theil statistic (TS) is the average percentage of deviation over all periods with regard to the actual values. The closer the Theil statistic is to zero, the better the prediction capability of the model. This is defined as [45]

\[
TS = 100 \times \sqrt{\frac{\sum_{i=1}^{n} (y_i - \hat{m}(t_i))^2}{\sum_{i=1}^{n} y_i^2}} \% 
\] (20)

Eleventh, it takes into account the tradeoff between the uncertainty in the model and the number of parameters in the model by slightly increasing the penalty each time parameters are added to the model when the sample is considerably small [46].

\[
PC = \left( n - m \right) \log \left( \frac{\sum_{i=1}^{n} (\hat{m}(t_i) - y_i)^2}{n} \right) + m \left( \frac{n - 1}{n - m} \right)
\] (21)

Based on the above criteria, we compared the proposed model with the existing NHPP software reliability model. When Adj_R^2 is closer to 1, and the other 10 criteria are closer to 0, it indicates a better fit. Using R and MATLAB, the parameters of each model were estimated through the LSE method, and the goodness of fit is calculated to compare the superiority. This is a method of estimating parameters through the difference
between the model in Table 1 and the actual number of failures in Table 2, and follows 
$LSE = \sum_{t=1}^{n}(y_t - m(t))^2$ [47].

### 3.3. Results of Dataset 1

Table 3 shows the estimated values for the parameters of each model obtained using 
dataset 1. Each parameter of the proposed model is represented by $\hat{a} = 80.0907$, $\hat{b} = 0.07231$, $\hat{c} = 15.9288$, and $\hat{h} = 9.8182$. Figure 2 shows the result of calculating the estimated value 
of $m(t)$ at each time point based on the cumulative number of failures at each time point 
in dataset 1 and each model equation. The black dotted line represents the actual data, 
and the dark red solid line represents the predicted failure value at each time point of the 
proposed model. Compared with other models, it shows the predicted value closest to the 
actual value.

Table 3. Parameter estimation of model from dataset 1.

| No. | Model   | Estimation              |
|-----|---------|-------------------------|
| 1   | GO      | $\hat{a} = 191.3881$, $\hat{b} = 0.0483$ |
| 2   | HDOG    | $\hat{a} = 191.3880$, $\hat{b} = 0.04832$, $\hat{c} = 1.3929$ |
| 3   | DS      | $\hat{a} = 92.0916$, $\hat{b} = 0.3034$ |
| 4   | IS      | $\hat{a} = 88.9815$, $\hat{b} = 0.3274$, $\hat{p} = 3.9383$ |
| 5   | ZFR     | $\hat{a} = 14.6285$, $\hat{b} = 0.21179$, $\hat{c} = 15.1085$, $\hat{p} = 0.2085$ |
| 6   | YE      | $\hat{a} = 212.1517$, $\hat{b} = 0.0276$, $\hat{c} = 3.3321$ |
| 7   | YR      | $\hat{a} = 101.8036$, $\hat{b} = 0.5271$, $\hat{c} = 0.0000332$, $\hat{p} = 0.0000684$ |
| 8   | YID1    | $\hat{a} = 181.0676$, $\hat{b} = 0.05131$, $\hat{c} = 0.00114$ |
| 9   | YID2    | $\hat{a} = 140.9842$, $\hat{b} = 0.06636$, $\hat{c} = 0.0126$ |
| 10  | PZ      | $\hat{a} = 27.3845$, $\hat{b} = 0.41396$, $\hat{c} = 63.7781$ |
| 11  | PNZ     | $\hat{a} = 27.3845$, $\hat{b} = 0.0218$, $\hat{c} = 0.0000332$, $\hat{p} = 0.0000684$ |
| 12  | TP      | $\hat{a} = 114.6551$, $\hat{b} = 0.38022$, $\hat{c} = 0.6721$, $\hat{p} = 0.000333$ |
| 13  | KSRGM   | $\hat{a} = 3.2070$, $\hat{b} = 8.76921$ |
| 14  | RMD     | $\hat{a} = 78.5350$, $\hat{b} = 0.21915$ |
| 15  | IFD     | $\hat{a} = 7.6597$, $\hat{b} = 0.84452$, $\hat{d} = 0.00171$ |
| 16  | Vtub    | $\hat{a} = 1.8954$, $\hat{b} = 0.70887$, $\hat{c} = 6.7593$ |
| 17  | TC      | $\hat{a} = 0.1736$, $\hat{b} = 1.33331$, $\hat{c} = 11.5457$ |
| 18  | 3P      | $\hat{a} = 1.4640$, $\hat{b} = 0.3299$, $\hat{p} = 0.6008$ |
| 19  | DP1     | $\hat{a} = 46197.046$, $\hat{b} = 0.00451$ |
| 20  | DP2     | $\hat{a} = 3.7402$, $\hat{b} = 30.01197$ |
| 21  | DPF     | $\hat{a} = 80.3065$, $\hat{b} = 0.06122$ |
| 22  | Proposed model | $\hat{a} = 80.0907$, $\hat{b} = 0.07231$ |
Table 4 shows the results of calculating the criteria of each model using the parameters obtained through dataset 1. As a result, the values of MSE, MAE, PRR, PP, PRV, RMSPE, MEOP, TS, PC of the proposed model show the smallest values of 9.9274, 3.2140, 0.0647, 0.0577, 2.6866, 2.6870, 2.8569, 4.6682, and 13.0594, respectively. The AIC shows the second smallest value at 73.4605. In addition, Adj_\text{R}^2 is 0.9821, which is the closest to 1. The DPF model shows the highest value with AIC = 73.2850, and the second highest result for the other criteria. The model with the third highest criterion is the Vtub model.

Table 4. Comparison of all criteria from dataset 1.

| No. | Model     | MSE    | MAE    | Adj_\text{R}^2 | PRR | PP    | AIC  | PRV | RMSPE | MEOP | TS    | PC   |
|-----|-----------|--------|--------|----------------|-----|-------|------|-----|-------|------|-------|------|
| 1   | GO        | 21.1918| 4.4966 | 0.9627        | 0.4187 | 0.2758 | 80.5308 | 4.3889 | 4.3892 | 0.8782 | 7.6254 | 16.5565 |
| 2   | HDOG      | 23.5465| 4.9962 | 0.9581        | 0.4187 | 0.2758 | 82.5308 | 4.3892 | 4.3892 | 4.4966 | 7.6254 | 16.5875 |
| 3   | DS        | 22.4994| 3.8718 | 0.9604        | 9.4838 | 0.7292 | 92.7861 | 4.4119 | 4.5135 | 3.5198 | 7.8572 | 16.8538 |
| 4   | IS        | 16.8528| 3.8882 | 0.9700        | 1.4452 | 0.3769 | 80.9153 | 3.6786 | 3.7104 | 3.4994 | 6.4512 | 15.0824 |
| 5   | ZFR       | 24.2415| 5.0835 | 0.9540        | 1.2042 | 0.3441 | 86.8823 | 3.6068 | 3.6338 | 4.9744 | 6.3174 | 18.4484 |
| 6   | YE        | 26.5773| 5.6309 | 0.9519        | 0.4182 | 0.2753 | 84.5796 | 4.3963 | 4.3965 | 5.0052 | 7.6380 | 16.9984 |
| 7   | YR        | 33.2210| 5.1600 | 0.9398        | 23.6437| 0.9460 | 103.5558| 4.6865 | 4.8967 | 5.8687 | 8.3395 | 17.8909 |
| 8   | YID1      | 23.6037| 4.9679 | 0.9579        | 0.4140 | 0.2764 | 82.4537 | 4.3944 | 4.3946 | 4.4711 | 7.6347 | 16.5984 |
| 9   | YID2      | 23.8374| 5.0079 | 0.9575        | 0.4070 | 0.2771 | 82.5743 | 4.4162 | 4.4163 | 4.5071 | 7.6724 | 16.6427 |
| 10  | PZ        | 217.3085| 17.6277| 0.5983        | 0.5816 | 1.2389 | 84.1256 | 4.7893 | 5.1243 | 15.4923 | 20.4300 | 24.8035 |
| 11  | PNZ       | 4000.777| 68.0533| -6.2441       | 2.6223 | 10.4004| 142.6741| 25.7721| 52.1779| 60.9182| 93.7126| 37.0531|
| 12  | TP        | 31.0301| 7.0496 | 0.9387        | 1.6004 | 0.3946 | 89.3412 | 3.7102 | 3.7518 | 5.8747 | 6.5247 | 21.7987 |
| 13  | KSRGM     | 26.0975| 5.4865 | 0.9527        | 1.5345 | 0.4198 | 88.8357 | 4.3440 | 4.3556 | 4.8769 | 7.5688 | 16.9255 |
| 14  | RMD       | 21.9129| 5.0225 | 0.9604        | 1.1967 | 0.3659 | 84.6918 | 3.9784 | 3.9099 | 4.4644 | 6.9355 | 16.2264 |
| 15  | IFD       | 29.9616| 5.5723 | 0.9467        | 0.653  | 0.3039 | 87.6572 | 4.9344 | 4.9498 | 5.0151 | 8.6017 | 17.6717 |
| 16  | Vtub      | 18.9012| 4.8111 | 0.9650        | 0.7591 | 0.2784 | 82.2273 | 3.4511 | 3.4667 | 4.2097 | 6.0252 | 16.2579 |
| 17  | TC        | 26.6474| 5.8343 | 0.9507        | 1.8283 | 0.4312 | 89.1447 | 4.0938 | 4.1159 | 5.1050 | 7.1541 | 17.4601 |
| 18  | TP        | 21.7357| 5.0238 | 0.9599        | 1.4395 | 0.3767 | 84.9059 | 3.6859 | 3.7164 | 4.9358 | 6.4613 | 16.7470 |
| 19  | TP        | 361.825| 19.1467| 0.3631        | 448.095| 3.2745 | 174.8811| 14.9748| 17.8944| 17.4061| 31.5087| 30.7442|
| 20  | TP        | 113.5394| 11.4623| 0.7945        | 0.5996 | 1.0133 | 109.7792| 9.0867 | 9.0870 | 10.1888| 15.7870| 22.8067|
| 21  | TP        | 9.9490 | 3.2278 | 0.9819        | 0.0689 | 0.0606 | 73.2850 | 2.6894 | 2.6899 | 2.8692 | 4.6732 | 13.0680|
| 22  | Proposed  | 9.9274 | 3.2140 | 0.9821        | 0.0647 | 0.0577 | 73.4605 | 2.6866 | 2.6870 | 2.8569 | 4.6682 | 13.0594|

Figure 2. Prediction of all models for dataset 1.
3.4. Results of Dataset 2

Table 5 shows the estimated values for the parameters of each model obtained using dataset 2. Each parameter of the proposed model is represented as $\hat{a} = 79.1444$, $\hat{b} = 0.2001$, $\hat{c} = 72.3208$, and $\hat{h} = 9.3327$. Figure 3 shows the results of calculating the estimated value of $m(t)$ for each point in time based on the cumulative number of failures at each time point in dataset 2 and each model equation. Here, the black dotted line represents the actual data, whereas the dark red solid line represents the predicted failure value at each time point of the proposed model. Compared with the other models, the predicted value is closest to the actual value.

Table 6 shows the results of calculating the criteria of each model using the parameters obtained through dataset 2. As a result, the values of MSE, MAE, PRR, PP, AIC, PRV, RMSPE, MEOP, TS, and PC of the proposed model are 18.9722, 4.3544, 0.1615, 0.1482, 92.2155, 3.7139, 3.7145, 3.8706, 6.3751, and 15.6500, respectively, which show the smallest criteria. $\text{Adj}_R^2$ is 0.9723, which is the closest to 1. The model with the second highest criterion is DPF, and Vtub is the third best-fitting model.

Table 5. Parameter estimation of model from dataset 2.

| No. | Model  | Estimation                      |
|-----|--------|---------------------------------|
| 1   | GO     | $\hat{a} = 518.0607$, $\hat{b} = 0.0156$ |
| 2   | HDOG   | $\hat{a} = 518.0607$, $\hat{b} = 0.01561$, $\hat{c} = 15.4479$ |
| 3   | DS     | $\hat{a} = 105.5701$, $\hat{b} = 0.2548$ |
| 4   | IS     | $\hat{a} = 4.3416$, $\hat{b} = 0.3506$, $\hat{c} = 67.6487$, $\hat{\beta} = 0.00128$, $\hat{\gamma} = 88.6105$ |
| 5   | ZFR    | $\hat{a} = 583.6809$, $\hat{\beta} = 0.00224$, $\hat{\gamma} = 88.6105$ |
| 6   | YE     | $\hat{a} = 2.1725$, $\hat{b} = 0.7383$, $\hat{\beta} = 0.000359$, $\hat{N} = 94.4688$, $\hat{\epsilon} = 59.8790$ |
| 7   | YR     | $\hat{a} = 21.4202$, $\hat{b} = 0.2758$, $\hat{\alpha} = 0.0000000161$, $\hat{\beta} = 4.1249$, $\hat{\gamma} = 0.0000000161$, $\hat{\delta} = 0.0000168$, $\hat{m_0} = 0.000213$ |
| 8   | YID1   | $\hat{a} = 79.1444$, $\hat{b} = 0.1928$, $\hat{c} = 72.3208$, $\hat{h} = 9.3327$ |
| 9   | YID2   | $\hat{a} = 92.1555$, $\hat{b} = 3.7145$, $\hat{c} = 6.3751$, $\hat{h} = 15.6500$ |
3.5. Results of Dataset 3

Table 7 shows the estimated values for the parameters of each model obtained using dataset 3. Each parameter of the proposed model is represented through $\hat{a} = 194.7684$, $\hat{b} = 0.3062$, $\hat{c} = 307.0805$, and $\hat{h} = 135.5641$. Figure 4 shows the results of calculating the
The Vtub model shows the most suitable criteria of 0.0094, 0.0090, 4.6356, 4.6357, and 2.5315 predicted value closest to the actual value. Parameter estimation of model from dataset 3. Table 7.

In the process of calculating the AIC value, the denominator is 0, so the AIC calculation cannot be performed. If there is no difference between the value at a specific point in time and the next point in time, the calculation is no longer being applied. In the process of calculating the AIC value, DPF, KSRGM, and the newly proposed model, a value indicating t = 14 is shown, indicating that the calculation is no longer being applied. In the process of calculating the AIC value, if there is no difference between the value at a specific point in time and the next point in time, the denominator is 0, so the AIC calculation cannot be performed.

Table 8 shows the results of calculating the criteria of each model using the parameters obtained through dataset 3. As a result, MSE and PC of the proposed model show the smallest values of 26.8047 and 24.5551, respectively, and Adj_R^2 shows the closest value to 1 at 0.9765. In addition, MAE, PRR, PP, PRV, RMSPE, MEOP, and TS are 4.9209, 0.0096, 0.0092, 4.6668, 4.6668, 4.5694, and 2.5484, respectively, showing the second smallest values.

Figure 4 shows the estimated failure values at each time point using the developed models. The Vtub model shows the most suitable criteria of 4.9195 and 4.5682 with MAE and MEOP. However, in calculating the AIC of the Vtub model, DPF, KSRGM, and the newly proposed model, a value indicating t = 14 is shown, indicating that the calculation is no longer being applied. In the process of calculating the AIC value, if there is no difference between the value at a specific point in time and the next point in time, the denominator is 0, so the AIC calculation can be performed.

Table 7. Parameter estimation of model from dataset 3.

| No. | Model | Estimation |
|-----|-------|------------|
| 1   | GO    | \( \hat{\alpha} = 197.387, \hat{b} = 0.399 \) |
| 2   | HDOG  | \( \hat{\alpha} = 197.3858, \hat{b} = 0.3985, \hat{\gamma} = 0.00088 \) |
| 3   | DS    | \( \hat{\alpha} = 192.528, \hat{b} = 0.882 \) |
| 4   | IS    | \( \hat{\alpha} = 197.354, \hat{b} = 0.399, \hat{\beta} = 0.000001 \) |
| 5   | ZFR   | \( \hat{\alpha} = 198.0864, \hat{b} = 0.0038, \hat{\alpha} = 1545.538, \hat{\beta} = 3.7206, \hat{c} = 603.6647, \hat{p} = 4.7236 \) |
| 6   | YE    | \( \hat{\alpha} = 248.808, \hat{\gamma} = 0.00797 \) |
| 7   | YR    | \( \hat{\alpha} = 206.0833, \hat{\alpha} = 1.0937 \) |
| 8   | YID1  | \( \hat{\alpha} = 183.4522, \hat{b} = 0.4620, \hat{\alpha} = 0.0066 \) |
| 9   | YID2  | \( \hat{\alpha} = 182.934, \hat{b} = 0.464, \hat{\alpha} = 0.0071 \) |
| 10  | PZ    | \( \hat{\alpha} = 195.990, \hat{b} = 0.3987, \hat{\alpha} = 0000, \hat{\alpha} = 1.390 \) |
| 11  | PNZ   | \( \hat{\alpha} = 183.125, \hat{b} = 0.463 \) |
| 12  | TP    | \( \hat{\alpha} = 0.007, \hat{\beta} = 0.0001 \) |
| 13  | KSRGM | \( \hat{\alpha} = 0.6853, \hat{b} = 0.0256 \) |
| 14  | RMD   | \( \hat{\alpha} = 35.9711, \hat{\beta} = 208.4596 \) |
| 15  | IFD   | \( \hat{\alpha} = 0.774, \hat{\alpha} = 0.0189 \) |
| 16  | Vtub  | \( \hat{\alpha} = 1.2170, \hat{b} = 2.9515, \hat{\alpha} = 0.0595 \) |
| 17  | TC    | \( \hat{\alpha} = 0.033, \hat{b} = 0.774, \hat{\alpha} = 181.0 \) |
| 18  | 3P    | \( \hat{\alpha} = 0.0307, \hat{b} = 0.2038, \hat{\beta} = 0.000581 \) |
| 19  | DP1   | \( \hat{\alpha} = 0.1105, \hat{\beta} = 3.1046 \) |
| 20  | DP2   | \( \hat{\alpha} = 0.0290, \hat{\gamma} = 6.0596 \) |
| 21  | DPF   | \( \hat{\alpha} = 194.766, \hat{\beta} = 0.304 \) |
| 22  | Proposed model | \( \hat{\alpha} = 307.0805, \hat{\beta} = 135.5641 \) |
Figure 4. Prediction of all models for dataset 3.

Table 8. Comparison of all criteria from dataset 3.
4. Optimal Release Time

When releasing software, it is very important that find the optimal release time. In order to find that, we need to find a time that minimizes the cost. We apply \( m(t) \) proposed in Section 2 to the cost model to find the optimal time point between time to market and the minimum cost. The optimal time is suggested based on the cost model that reflects the software installation cost, software test cost, operation cost, software removal cost, and risk cost when the software failure occurs. Figure 5 describes the software field environment from the software installation of the software cost model. The expected software cost model follows Equation (22) [30,31].

\[
C(T) = C_0 + C_1 T + C_2 m(T) + C_3 (1 - R(x|T))
\]  

(22)

where \( C_0 \) is the installation cost for system testing, \( C_1 \) is the system test cost per unit time, \( C_2 \) is the error removal cost per unit time during the test phase, and \( C_3 \) is the penalty cost owing to a system failure. In addition, \( x \) represents the time the software was used. In addition, in the cost model equation, \( R(x|T) \) follows (23) [32,33].

\[
R(x|T) = e^{-|m(t)+x−m(t)|}
\]  

(23)

Figure 5. System cost model structure.

In this section, we propose a cost model using dataset 1 based on the proposed software reliability model and find the optimal time point between time to market and the minimum cost by changing the cost coefficients from \( C_0 \) to \( C_3 \).

4.1. Results of the Optimal Release Time

For the parameters of the cost model, \( a, b, c \), and \( h \) calculated through numerical examples described in Section 3 were used. The cost coefficient of the cost model aims to find the optimal release time with the lowest cost by finding the optimal value through the changes in several values. The baseline value of the cost coefficient is as follows:

\[
C_0 = 500, \ C_1 = 20, \ C_2 = 50, \ C_3 = 5000, \ x = 6
\]

Here, baseline denotes to the reference value for confirming the change of the cost coefficient. The total cost value obtains as a reference value is 4888.856, and the optimal release time \( T \) at this time is 18.3. Table 9 changes the cost coefficient of each reference value, checks the minimum cost \( C(T) \) and optimal release time \( T^* \), and then checks the changing trend to find the most optimal release time \( T^* \). When \( x = 2 \), the smallest total cost value obtains 4886.985 at \( T^* = 18.2 \). When \( x = 4 \), the smallest total cost value shows 4888.735 at \( T^* = 18.3 \). When \( x = 6 \), the smallest total cost value shows 4888.856 at \( T^* = 18.3 \). When \( x \) is 8 and 10, the smallest total cost value shows 4888.863 at \( T^* = 18.3 \).
Table 9. Optimal release time of expected total cost according to baseline.

| Base | \( x = 2 \) | \( x = 4 \) | \( x = 6 \) | \( x = 8 \) | \( x = 10 \) |
|------|-------------|-------------|-------------|-------------|-------------|
| \( T^* \) | \( C(T) \) | \( T^* \) | \( C(T) \) | \( T^* \) | \( C(T) \) | \( T^* \) | \( C(T) \) |
| 500  | 18.2        | 4886.985    | 18.3        | 4888.735    | 18.3        | 4888.856    | 18.3        | 4888.863    |
| 300  | 18.2        | 4886.985    | 18.3        | 4888.735    | 18.3        | 4888.856    | 18.3        | 4888.863    |
| 700  | 18.2        | 5086.985    | 18.3        | 5088.735    | 18.3        | 5088.856    | 18.3        | 5088.863    |

Here, \( C_0 \) is the setup cost, and as the value increases, the cost, which is directly proportional, increases as well; thus, the lower the setup cost is, the lower the cost. Table 10 compares the changes when the coefficients of are 300, 500, and 700. It is found that the higher the value is, the higher the total cost value, whereas the optimal time does not change. Therefore, it appears that \( C_0 \) does not help determine the optimal release point. However, because the setup cost for a system stabilization is required, the appropriate \( C_0 \) cost coefficient is set to 500. Figure 6 shows a graph of the results according to the change in \( C_0 \).

Table 10. Optimal release time of expected total cost according to \( C_0 \).

| \( C_0 \) | \( x = 2 \) | \( x = 4 \) | \( x = 6 \) | \( x = 8 \) | \( x = 10 \) |
|----------|-------------|-------------|-------------|-------------|-------------|
| \( T^* \) | \( C(T) \) | \( T^* \) | \( C(T) \) | \( T^* \) | \( C(T) \) | \( T^* \) | \( C(T) \) |
| 300      | 18.2        | 4686.985    | 18.3        | 4688.735    | 18.3        | 4688.856    | 18.3        | 4688.863    |
| 500      | 18.2        | 4886.985    | 18.3        | 4888.735    | 18.3        | 4888.856    | 18.3        | 4888.863    |
| 700      | 18.2        | 5086.985    | 18.3        | 5088.735    | 18.3        | 5088.856    | 18.3        | 5088.863    |

Figure 6. Optimal release time of total cost according to \( C_0 \).

Table 11 compares the changes when the coefficients of \( C_1 \) are 10, 20, and 30. The results show that when \( C_1 \) is 10, the total cost is the minimum at approximately 18.9 to 19.0, and when \( C_1 \) is 20, the minimum value is at 18.2 to 18.3, and when it is 30, the total cost shows the minimum value at approximately 17.8 to 17.9. As the cost coefficient \( C_1 \) increases, the optimal release time is gradually pushed back. Figure 7 shows a graph of the results according to the changes in \( C_1 \).
Table 11. Optimal release time of expected total cost according to $C_1$.

| $C_1$ | $x=2$ | $x=4$ | $x=6$ | $x=8$ | $x=10$ |
|-------|-------|-------|-------|-------|-------|
|       | $T^*$ | $C(T)$ | $T^*$ | $C(T)$ | $T^*$ | $C(T)$ | $T^*$ | $C(T)$ | $T^*$ | $C(T)$ |
| 10    | 18.9  | 4702.044 | 19.0  | 4702.818 | 19.0  | 4702.864 | 19.0  | 4702.866 | 19.0  | 4702.866 |
| 20    | 18.2  | 4886.985 | 18.3  | 4888.735 | 18.3  | 4888.856 | 18.3  | 4888.863 | 18.3  | 4888.863 |
| 30    | 17.8  | 5066.820 | 17.9  | 5069.652 | 17.9  | 5069.861 | 17.9  | 5069.873 | 17.9  | 5069.874 |

Figure 7. Optimal release time of total cost according to $C_1$.

Table 12 compares the changes when the coefficients of $C_2$ are 30, 40, 50, and 60. It can be seen that the cost coefficient $C_2$ does not change from 18.2 to 18.3 at the optimal release time as the value changes. Figure 8 shows a graph of the results according to the change in $C_2$.

Table 13 compares the changes when the coefficients of $C_3$ are 5000, 7000, 10,000, and 15,000. The results show that when $C_3$ is 5000, the total cost is the minimum at approximately 18.2 to 18.3; when it is 7000, it shows the minimum value at 18.5 to 18.6; when it is 10,000, the total cost shows the minimum value at approximately 18.9 to 19.0; and when it is 15,000, the total cost shows the minimum value at approximately 19.2 to 19.3. This indicates that the optimal release time gradually increases as the cost coefficient $C_3$ increases. Figure 9 shows a graph of the results according to the changes in $C_3$.

Table 12. Optimal release time of expected total cost according to $C_2$.

| $C_2$ | $x=2$ | $x=4$ | $x=6$ | $x=8$ | $x=10$ |
|-------|-------|-------|-------|-------|-------|
|       | $T^*$ | $C(T)$ | $T^*$ | $C(T)$ | $T^*$ | $C(T)$ | $T^*$ | $C(T)$ | $T^*$ | $C(T)$ |
| 30    | 18.2  | 3285.254 | 18.3  | 3286.995 | 18.3  | 3287.116 | 18.3  | 3287.123 | 18.3  | 3287.123 |
| 40    | 18.2  | 4086.120 | 18.3  | 4087.865 | 18.3  | 4087.986 | 18.3  | 4087.993 | 18.3  | 4087.993 |
| 50    | 18.2  | 4886.985 | 18.3  | 4888.735 | 18.3  | 4888.856 | 18.3  | 4888.863 | 18.3  | 4888.863 |
| 60    | 18.2  | 5687.851 | 18.3  | 5689.604 | 18.3  | 5689.726 | 18.3  | 5689.733 | 18.3  | 5689.733 |
Figure 8. Optimal release time of the total cost according to $C_2$.

Table 13. Optimal release time of expected total cost according to $C_3$.

| $C_3$ | $x=2$ | $x=4$ | $x=6$ | $x=8$ | $x=10$ |
|-------|-------|-------|-------|-------|-------|
|       | $T^*$ | $C(T)$ | $T^*$ | $C(T)$ | $T^*$ | $C(T)$ | $T^*$ | $C(T)$ | $T^*$ | $C(T)$ |
| 5000  | 18.2  | 4886.985 | 18.3  | 4888.735 | 18.3  | 4888.856 | 18.3  | 4888.863 | 18.3  | 4888.863 |
| 7000  | 18.5  | 4893.210 | 18.6  | 4894.846 | 18.6  | 4894.958 | 18.6  | 4894.964 | 18.6  | 4894.964 |
| 10,000| 18.9  | 4899.648 | 19.0  | 4901.186 | 19.0  | 4901.277 | 19.0  | 4901.281 | 19.0  | 4901.282 |
| 15,000| 19.2  | 4906.802 | 19.3  | 4908.208 | 19.3  | 4908.297 | 19.3  | 4908.301 | 19.3  | 4908.301 |

Figure 9. Optimal release time of the total cost according to $C_3$. 

```plaintext
\[
C_3 = 3x
\]
```
4.2. Results of Variation in Cost Model for Changes in Parameter

In this section, we check whether the optimal release time is affected by the change in the cost model according to the change in the parameters of the proposed model. The parameters \(a\), \(b\), \(c\), and \(h\) of the proposed model are set at \(-20\%\), \(-10\%\), \(0\%\), \(10\%\), and \(20\%\), respectively, in 10% increments, and the coefficient of the cost model is fixed at the baseline value in Section 4.1. Thus, the minimum cost value is calculated depending on changes in the parameters, and it derives appropriate release time. In Table 14, \(0\%\) is the same as the value suggested in Table 9 by substituting the parameter estimates described in Section 3 and the coefficient values of the cost model proposed in Section 4.

Table 14. Optimal release time of cost according to parameter change.

| \(T^*\) | \(C(T)\) | \(T^*\) | \(C(T)\) | \(T^*\) | \(C(T)\) | \(T^*\) | \(C(T)\) | \(T^*\) | \(C(T)\) |
|---|---|---|---|---|---|---|---|---|---|
| \(-20\%\) | \(-10\%\) | \(0\%\) | \(10\%\) | \(20\%\) |
| \(a\) | 20.4 | 4129.849 | 19.2 | 4507.887 | 18.3 | 4888.856 | 17.6 | 5272.215 | 16.8 | 5657.258 |
| \(b\) | 22.6 | 4979.801 | 20.0 | 4929.426 | 18.3 | 4888.856 | 16.8 | 4855.545 | 15.4 | 4827.546 |
| \(c\) | 16.4 | 4847.806 | 17.4 | 4869.089 | 18.3 | 4888.856 | 19.2 | 4907.319 | 20.0 | 4924.648 |
| \(h\) | 18.6 | 4892.931 | 18.4 | 4890.757 | 18.3 | 4888.856 | 18.2 | 4887.130 | 18.2 | 4885.583 |

From Table 14 and Figures 10–13, the value of the cost model \(C(T)\) increases as the change in parameter \(a\) increases, whereas the optimal release time \(T^*\) decreases. As the values of parameters \(b\) and \(h\) increase, the cost model \(C(T)\) increases, and the release time \(T^*\) is shown to decrease; in addition, it is found that the change in parameter \(h\) had a very slight effect on the optimal release time compared to parameter \(b\). As the value of parameter \(c\) increases, the cost model \(C(T)\) and release time \(T^*\) increase together. Based on this, it is found that parameter \(a\) had a very large minimum width of the cost model compared with the changes of the other parameters, and parameter \(b\) had the greatest influence on determining the optimal release time.

![Figure 10](image-url)  
Figure 10. Optimal release time of cost according to \(a\).
Figure 11. Optimal release time of cost according to $b$.

Figure 12. Optimal release time of cost according to $c$. 
When the test cost was increased, the release time gradually increased, as did the overall cost; therefore, the optimal release time can be achieved when the change in the cost coefficient was suggested, and the total cost was analyzed accordingly. When the test cost was increased, the release time gradually increased, as did the overall cost; therefore, the optimal release time can be achieved when \( C_1 \) is 20. In the proposed model, fault detection rate \( b \) was found to be the most important parameter for determining the optimal release time.

In the past, studies were conducted by assuming independence in the case of software failures; however, in a real environment, the software execution environment is extremely diverse and complex. Therefore, it is necessary to develop a model that assumes a dependent failure occurrence and propose a model that considers the actual operating environment. We plan to conduct a study using machine learning and deep learning for the proposed software-dependent failure occurrence in the future work.

**Author Contributions:** Conceptualization, H.P.; Funding acquisition, I.H.C., K.Y.S.; Software, Y.S.K.; Writing—original draft, Y.S.K.; Writing—review and editing, K.Y.S., I.H.C. and H.P. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2018R1D1A1B07045734, NRF-2021R1F1A1048592, and NRF-2021R1I1A1A01059842).

**Institutional Review Board Statement:** Not required.

**Informed Consent Statement:** Not applicable.
Data Availability Statement: Data available in a publicly accessible repository.

Acknowledgments: This research was supported by the National Research Foundation of Korea. We are pleased to thank the Editor and the Referees for their useful suggestions.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Wu, Y.C.; Wu, Y.J.; Wu, S.M. An outlook of a future smart city in Taiwan from post–internet of things to artificial intelligence internet of things. In Smart Cities: Issues and Challenges; Elsevier: Amsterdam, The Netherlands, 2019; pp. 263–282.

2. Goel, A.L.; Okumoto, K. Time-dependent error-detection rate model for software reliability and other performance measures.
   IEEE Trans. Reliab. 1979, 28, 206–211. [CrossRef]

3. Hossain, S.A.; Dahiyi, R.C. Estimating the parameters of a non-homogeneous Poisson-process model for software reliability.
   IEEE Trans. Reliab. 1993, 42, 604–612. [CrossRef]

4. Yamada, S.; Ohba, M.; Osaki, S. S-shaped reliability growth modeling for software fault detection.
   IEEE Trans. Reliab. 1983, 32, 475–484. [CrossRef]

5. Ohba, M. Inflection S-shaped software reliability growth model. In Stochastic Models in Reliability Theory; Osaki, S., Hatoyama, Y., Eds.; Springer: Berlin, Germany, 1994; pp. 144–162.

6. Zhang, X.M.; Teng, X.L.; Pham, H. Considering fault removal efficiency in software reliability assessment.
   IEEE Trans. Syst. Man Cybern. Part A-Syst. Hum. 2003, 33, 114–120. [CrossRef]

7. Yamada, S.; Ohtera, H.; Narihisa, H. Software Reliability Growth Models with Testing-Effort.
   IEEE Trans. Reliab. 1986, 35, 19–23. [CrossRef]

8. Yamada, S.; Tokuno, K.; Osaki, S. Imperfect debugging models with fault introduction rate for software reliability assessment.
   Int. J. Syst. Sci. 1992, 23, 2241–2252. [CrossRef]

9. Pham, H.; Zhang, X. An NHPP software reliability models and its comparison.
   Int. J. Reliab. Qual. Saf. Eng. 1997, 4, 269–282. [CrossRef]

10. Pham, H.; Nordmann, L.; Zhang, X. A general imperfect software debugging model with S-shaped fault detection rate.
    IEEE Trans. Reliab. 1999, 48, 169–175. [CrossRef]

11. Teng, X.; Pham, H. A new methodology for predicting software reliability in the random field environments.
    IEEE Trans. Reliab. 2006, 55, 458–468. [CrossRef]

12. Kapur, P.K.; Pham, H.; Anand, S.; Yadav, K. A unified approach for developing software reliability growth models in the presence of imperfect debugging and error generation.
    IEEE Trans. Reliab. 2011, 60, 331–340. [CrossRef]

13. Roy, P.; Mahapatra, G.S.; Dey, K.N. An NHPP software reliability growth model with imperfect debugging and error generation.
    Int. J. Reliab. Qual. Saf. Eng. 2014, 21, 1–3. [CrossRef]

14. Pham, H. System Software Reliability; Springer: London, UK, 2006.

15. Pham, H. A new software reliability model with Vtub-Shaped fault detection rate and the uncertainty of operating environments.
    Optimization 2014, 63, 1481–1490. [CrossRef]

16. Chang, I.H.; Pham, H.; Lee, S.W.; Song, K.Y. A testing-coverage software reliability model with the uncertainty of operation environments.
    Int. J. Syst. Sci.-Oper. Logist. 2014, 1, 220–227.

17. Song, K.Y.; Chang, I.H.; Pham, H. A Three-parameter fault-detection software reliability model with the uncertainty of operating environments.
    J. Syst. Sci. Syst. Eng. 2017, 26, 121–132. [CrossRef]

18. Ramasamy, S.; Lakshmanan, I. Machine learning approach for software reliability growth modeling with infinite testing effort function.
    Math. Probl. Eng. 2017, 8040346. [CrossRef]

19. Kim, Y.S.; Chang, I.H.; Lee, D.H. Non-Parametric Software Reliability Model Using Deep Neural Network and NHPP Software Reliability Growth Model Comparison.
    J. Korean. Data Anal. Soc. 2020, 22, 2371–2382. [CrossRef]

20. Begum, M.; Hafiz, S.B.; Islam, J.; Hossain, M.J. Long-term Software Fault Prediction with Robust Prediction Interval Analysis via Refined Artificial Neural Network (RANN) Approach.
    Eng. Lett. 2021, 29, 1158–1171.

21. Zhu, J.; Gong, Z.; Sun, Y.; Dou, Z. Chaotic neural network model for SMISs reliability prediction based on interdependent network SMISs reliability prediction by chaotic neural network.
    Qual. Reliab. Eng. Int. 2021, 37, 717–742. [CrossRef]

22. Sahu, K.; Alzahrani, F.A.; Srivastava, R.K.; Kumar, R. Evaluating the Impact of Prediction Techniques: Software Reliability Perspective.
    CMC-Comput. Mat. Contin. 2021, 67, 1471–1488. [CrossRef]

23. Ogundoyin, S.O.; Kamil, I.A. A Fuzzy-AHP based prioritization of trust criteria in fog computing services.
    Appl. Soft. Comput. 2020, 97, 106789. [CrossRef]

24. Rafi, S.; Akbar, M.A.; Yu, W.; Alsanad, A.; Gumaei, A.; Sarwar, M.U. Exploration of DevOps testing process capabilities: An ISM and fuzzy TOPSIS analysis.
    Appl. Soft Comput. 2021, 108377. [CrossRef]

25. Li, Q.; Pham, H. Modeling Software Fault-Detection and Fault-Correction Processes by Considering the Dependencies between Fault Amounts.
    Appl. Sci. 2021, 11, 6998. [CrossRef]

26. Son, H.I.; Kwon, K.R.; Kim, J.O. Reliability Analysis of Power System with Dependent Failure.
    J. Korean. Inst. Illum. Electr. Install. Eng. 2011, 25, 62–68.
27. Pan, Z.; Nonaka, Y. Importance analysis for the systems with common cause failures. Reliab. Eng. Syst. Saf. 1995, 50, 297–300. [CrossRef]
28. Pham, L.; Pham, H. Software Reliability Models with Time Dependent Hazard Rate Based on Bayesian Approach. IEEE Trans. Syst. Man Cybern. Part A-Syst. Hum. 2000, 30, 25–35. [CrossRef]
29. Lee, D.H.; Chang, I.H.; Pham, H. Software reliability model with dependent failures and SPRT. Mathematics 2020, 8, 1366. [CrossRef]
30. Kim, H.C. The Property of Learning effect based on Delayed Software S-Shaped Reliability Model using Finite NHPP Software Cost Model. Indian J. Sci. Technol. 2015, 8, 1–7. [CrossRef]
31. Yang, B.; Xie, M. A study of operational and testing reliability in software reliability analysis. Reliab. Eng. Syst. Saf. 2000, 70, 323–329. [CrossRef]
32. Pham, L.; Pham, H. Software Reliability Models with Time Dependent Hazard Rate Based on Bayesian Approach. IEEE Trans. Syst. Man Cybern. Part A-Syst. Hum. 2000, 30, 25–35. [CrossRef]
33. Lee, D.H.; Chang, I.H.; Pham, H. Software reliability model with dependent failures and SPRT. Mathematics 2020, 8, 1366. [CrossRef]
34. Song, K.Y.; Chang, I.H. A Sensitivity Analysis of a New NHPP Software Reliability Model with the Generalized Exponential Fault Detection Rate Function Considering the Uncertainty of Operating Environments. J. Korean. Data Anal. Soc. 2020, 22, 473–482. [CrossRef]
35. Li, X.; Xie, M.; Ng, S.H. Sensitivity analysis of release time of software reliability models incorporating testing effort with multiple change-points. Appl. Math. Model. 2010, 34, 3560–3570. [CrossRef]
36. Stringfellow, C.; Andrews, A.A. An empirical method for selecting software reliability growth models. Empir. Softw. Eng. 2002, 7, 319–343. [CrossRef]
37. Inoue, S.; Yamada, S. Discrete software reliability assessment with discretized NHPP models. Comput. Math. Appl. 2006, 51, 161–170. [CrossRef]
38. Anjum, M.; Haque, M.A.; Ahmad, N. Analysis and ranking of software reliability models based on weighted criteria value. Int. J. Inf. Technol. Comput. Sci. 2013, 2, 1–14. [CrossRef]
39. Jeske, D.R.; Zhang, X. Some successful approaches to software reliability modeling in industry. J. Syst. Softw. 2005, 74, 85–99. [CrossRef]
40. Iqbal, J. Software reliability growth models: A comparison of linear and exponential fault content functions for study of imperfect debugging situations. Cogent Eng. 2017, 4, 1286739. [CrossRef]
41. Zhao, J.; Liu, H.W.; Cui, G.; Yang, X.Z. Software reliability growth model with change-point and environmental function. J. Syst. Softw. 2006, 79, 1578–1587. [CrossRef]
42. Akaike, H. A new look at statistical model identification. IEEE Trans. Autom. Control 1974, 19, 716–719. [CrossRef]
43. Pilai, K.; Nair, V.S. A model for software development effort and cost estimation. IEEE Trans. Softw. Eng. 1997, 23, 485–497. [CrossRef]
44. Peng, R.; Li, Y.F.; Zhang, W.J.; Hu, Q.P. Testing effort dependent software reliability model for imperfect debugging process considering both detection and correction. Reliab. Eng. Syst. Saf. 2014, 126, 37–43. [CrossRef]
45. Sharma, K.; Garg, R.; Nagpal, C.K.; Garg, R.K. Selection of optimal software reliability growth models using a distance based approach. IEEE Trans. Reliab. 2010, 59, 266–276. [CrossRef]
46. Pham, H. On estimating the number of deaths related to Covid-19. Mathematics 2020, 8, 655. [CrossRef]
47. Wang, L.; Hu, Q.; Liu, J. Software reliability growth modeling and analysis with dual fault detection and correction processes. IIE Trans. 2016, 48, 359–370. [CrossRef]