International Journal of Mathematical, Engineering and Management Sciences
Vol. 6, No. 4, 1044-1054, 2021
https://doi.org/10.33889/IJMEMS.2021.6.4.061

Application of Software Reliability Model for Safety Assessment of E/E/PE Safety-Related Software

Shinji Inoue
Faculty of Informatics,
Kansai University, Osaka, Japan.
Corresponding author: ino@kansai-u.ac.jp

Takaji Fujiwara
SRATECH Laboratory, Hyogo, Japan.
E-mail: fujiwara.takaji@nifty.com

Shigeru Yamada
Graduate School of Engineering,
Tottori University, Tottori, Japan.
E-mail: yamada@tottori-u.ac.jp

(Received on March 9, 2021; Accepted on May 15, 2021)

Abstract
Quantitative and analytical safety assessment methods of E/E/PE safety-related software systems based on the SIL defined by IEC 61508 have been proposed. IEC 61508 does not provide us with quantitative and analytical methods for safety assessment of the software. Our methods give us quantitative information on safety measures for deciding the safety integrity level and testing time duration for achieving certain safety integrity level of E/E/PE software, respectively. Our stochastic modeling approaches are based on software reliability modeling and software reliability assessment techniques. Numerical examples for our methods have been shown for explaining how to use our software safety assessment approaches conforming IEC 61508.

Keywords- IEC 61508, PFD, PFH, E/E/PE systems, SIL, Software safety assessment, Software reliability model.

1. Introduction

Functional safety means maintaining certain safety level by the functional aspects. In other words, the purpose of functional safety is the transition to or maintain specified safety state by automatic protection function. This is definitely different from the notion of essential safety, which means maintaining the certain safety state by eliminating the factors disturbing the safety fundamentally. Especially for the functional safety, it expands the opportunities of applying electrical/electronic/programmable electronic safety-related systems (abbreviated as E/E/PE systems) for realizing the functional safety in several practical fields along with the advancement of information processing technologies. For examples, the pre-crash safety system and the air-bag system of an automobile are the ones operated by the E/E/PE systems. Recently, international standards have been issued for respective industrial fields, such as ISO 26262 for the functional safety of automotive and IEC 61511 for process industry. Especially, IEC 61508 (IEC 61508, 2010) is known as the related international basic standard for the E/E/PE systems. In IEC 61508, it is required to analyze the safety integrity level (abbreviated as SIL) quantitatively for the hardware harmful event of the system by following the random occurrences of the hardware failures. That is, for the hardware of the systems, it is required to assess their safety based on the probability-based mathematical approach. It is worth mentioning that their safety is measured based on the target

failure measures.

Regarding software failures of the system, IEC 61508 treats it as the systematic or deterministic failure. Then, the SIL for the software of E/E/PE systems is decided by following software development techniques applied in the software development process. That is, software development managers need to apply certain development techniques for achieving certain safety integrity level. Treating software failures as the systematic failure must be agreed because software failures occur inevitably due to the existence of software bugs, which have been introduced within the development process and the failure occurrence mechanism is essentially different from the random failure occurrence mechanism for the hardware. However, for embedded software, like safety-related software, it is better to treat the software failures and harmful event occurrences as the randomly occurred failures or harmful events because the execution frequency of program paths or software modules in embedded system has variation due to a lot of type of inputs to the software in operation, i.e., we cannot easily obtain the information on when certain program paths or modules having software bugs are executed. In that situation, it is possible to say that the time-intervals of software failures or harmful events occurrences are treated as random variables and a probability-based analytical model for describing the time-varying random software failure occurrences must be useful for quantitative and analytical safety assessment of the software in E/E/PE systems.

This paper proposes probability model-based methods for conducting quantitative software safety assessment with the time-varying uncertainty of dangerous software failure-occurrences of the E/E/PE systems by applying techniques in reliability modeling for software. Concretely, approximated methods for estimating the target failure measures are proposed. These measures are used for deciding the achievement of SIL and supporting estimation of the time-duration needed to achieve certain level of SIL defined by IEC 61508. We would stress that proposing the estimation methods for the target failure measures for software SIL assessment by applying a software reliability model is our main purpose in this paper, not proposing new software reliability models. Further, numerical examples are shown for explaining how to use our approaches by applying software fault counting data.

2. Literature Review

In recent years, several methods for calculating such measures for deciding the level of safety integrity of hardware in the E/E/PE systems have been proposed for enabling us to execute safety assessment quantitatively based on the SIL by considering several types of the specific internal system structures. Misumi and Sato (1999) proposed calculating methods for the target failure measures by considering harmful risk event occurrence mechanism. Kato and Satoh (2000) defined new safety demand modes for the hardware and proposed methods for calculating the target failure measures for the new safety demand modes. Ghadhab et al. (2019), discussed the usage of dynamic fault tree for safety analysis of vehicle guidance systems based on ISO 26262.

As mentioned above, there are a lot of attentions on quantitative assessment methods for the hardware based on SIL. On the other hand, it rarely gets the chance to read papers discussing quantitative safety assessment for the software because IEC 61508 does not require to conduct it for software. Recently, (Fujiwara et al., 2011) discussed the needfulness of quantitative SIL assessment for the software and proposed fundamentals for the SIL-based software safety assessment.

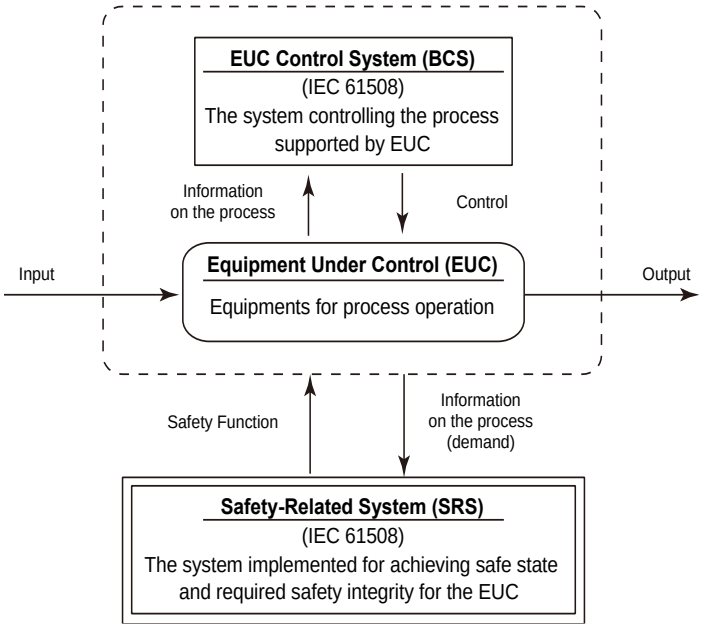

Figure 1. Basic configuration of the whole system including E/E/PE system.

Gu (2011) discussed a supporting approach for estimating software SIL of an embedded system by considering the failure-occurrence behavior between hardware and software interaction. However, this paper has not incorporated the time-varying uncertainty of the software failures. And the remaining problem is that it has not been proposed a method for estimating time-duration needed to test the software for conforming the certain level of SIL. This problem should be very important for management of designing and checking the safety level of the systems.

3. SIL and Target Failure Measures

One of the main features of IEC 61508 is to require conducting safety assessment based on the SIL. The SIL represents the safety level divided into four levels for the systems. As the first, Figure 1 represents the configuration of basic whole system. From Figure 1, the E/E/PE system is incrementally assembled to the basic system consisted of EUC and BCS. The EUC and BCS basically take the role of intended functions. And the E/E/PE system takes the safety role of the whole system finally when the basic system consisted of EUC and BCS fails the safety function and then the safety demand is issued from the basic system consisted by EUC and BCS. Therefore, the safety assessment is needed to consider not only the state of the systems but also the frequency of the safety demands. The SIL is defined by the two operation modes depending on the frequency of the safety demands to the systems in operation because the system's state and the safety demands requested to the system (see Figure 1). The values for each level are measured by the target failure measures. The details on SIL can be checked in the related basic standard and books.

If the safety demands are requested not greater than 1 per year, the low-demand mode should be applied for the assessment based on the SIL and the safety integrity is measured based on the average probability of failure on demand of the safety function (abbreviated as PFD). For examples, if the system is needed to achieve SIL 1 in this mode, we need to conform $10^{-2} \leq \text{PFD} \leq 10^{-1}$. On the other hand, the average frequency of a dangerous failure of the safety function [1/h]

(abbreviated as PFH) is used for the safety assessment of the system if the safety demands frequency is greater than 1 per year, i.e., the system is operated as the high demand or continuous mode in operation. For examples, if the system is needed to achieve SIL 1 in this mode, we need to conform $10^{-6} \leq \text{PFH}\ (1/\text{hour}) \leq 10^{-5}$.

PFD is given by dividing the down time-duration by the total operation time-duration of E/E/PE system. That is, the PFD is essentially equivalent to the quantified measure of the unavailability of the system and can be treated as a probability. Especially for this paper, we approximately give the basic notion for estimating this safety measure as

$$\text{PFD} = 1 - \text{Software Reliabilty}. \tag{1}$$

Essentially, a high availability is obtained from the high reliability in case of the constant down time-duration for each repair. For the high frequency of safety demands or considering the situation that the system receives the safety demands continuously, we need to use the safety measures of PFH. Therefore, the PFH is essentially same as the software hazard rate for the system. Ignored the down time for repair, this safety measure is approximately estimated as:

$$\text{PFH} = \frac{1}{\text{MTBF}}, \tag{2}$$

in which the MTBF is the abbreviation of the mean time between software failures (Yamada, 2014). It notes that the unit of PFH is 1/hour.

4. Software Reliability Model

The software reliability in Eq. (1) and the software hazard rate or MTBF in Eq. (2) can be estimated by applying software reliability models (Pham, 2000; Pham, 2007; Yamada, 2011; Yamada, 2014). We briefly introduce software reliability modeling and software reliability assessment measures related to our approaches in this paper because proposing new software reliability models is not our purpose of this paper. The books (Pham, 2000; Pham, 2007; Yamada, 2011; Yamada, 2014) discuss the details on software reliability assessment models and mathematical models in software reliability.

Among the software reliability models, nonhomogeneous Poisson process models are widely applied in practical. Letting $\{S(t), t \geq 0\}$ be the cumulative number of software bugs detected during $(0, t]$, the stochastic behavior of $N(t)$ follows that

$$\Pr\{S(t) = n\} = \frac{\{Z(t)\}^n}{n!} \exp[-Z(t)] \quad (n = 0, 1, 2, \cdots), \tag{3}$$

where $Z(t)$ is the expectation of $S(t)$. After considering the mathematical structure of the expectation and then obtaining the specific stochastic behavior of stochastic law in Eq. (3) along with software reliability data observed, some useful assessment measures, such as a reliability function and an instantaneous MTBF can be obtained. The reliability function, $R(x|t)$, is the time-varying function representing the failure-free operation within the time-interval from t to $t + x$ and is formulated that

$$R(x|t) \equiv \Pr\{S(t + x) - S(t) = 0\} = \exp[-\{Z(t + x) - Z(t)\}]. \tag{4}$$

And the instantaneous MTBF is also calculated by

$$MTBF_I(t) = \frac{1}{z(t)}, \tag{5}$$

where $z(t) = dZ(t)/dt$ and is the intensity function.

The aims and objectives of our research work is to propose estimation methods for the target failure measures based on the existing notion of reliability modeling and assessment techniques for software system. However, more specific analytical expressions are needed for estimating the PFD and PFH, respectively, by considering the dangerous failure rate and the difference of executing environment for the software because the software reliability models describe the software failure occurrence phenomenon including both of the safe and dangerous software failure occurrences. The next section gives the discussions on how to incorporates these two factors in estimating the target failure measures.

5. Target Failure Measures

As the first step, the dangerous software failure occurrence rate, DFR $(0 < \text{DFR} \leq 1)$, is incorporated for calculating the target failure measures by applying software reliability models. As for describing the difference of executing environment, an environment function, $EF(x)$, is incorporated for enabling us to transform the arbitrary operation time-duration $(t, t + x]$ into the testing time-duration. The environmental function represents the difference of the executing environment between testing and operation.

Incorporating the two factors, the dangerous software failure occurrence rate and the environment function into the basic frameworks in Eqs. (1) and (2), we propose an approximate analytical expression for the PFD, which is the target failure measure for the low-demand operation mode:

$$PFD(x|t) = 1 - \exp[-\text{DFR} \cdot \{Z\big(t + EF(x)\big) - Z(t)\}], \tag{6}$$

which is derived based on the basic notion of PFD in Eq. (1). It is worth mentioning that Eq. (6) represents the PFD at certain elapsed operation time x under the condition that the testing have been executed up to time t or the testing has been terminated at time t. Given the certain elapsed operation time x, Eq. (6) is a time varying function of t essentially. Regarding the PFH, which is the target failure measure for the high demand or continuous operation mode,

$$PFH(t) = \frac{\text{DFR}}{EF^{-1}(MTBF_I(t))}, \tag{7}$$

by following the basic notion in Eq. (2) and by incorporating the dangerous software failure occurrence rate and the environment function. In Eq. (7), $EF^{-1}(\cdot)$ is the inverse function of the environment function. That is, the denominator of the right-hand side in Eq. (7) represents

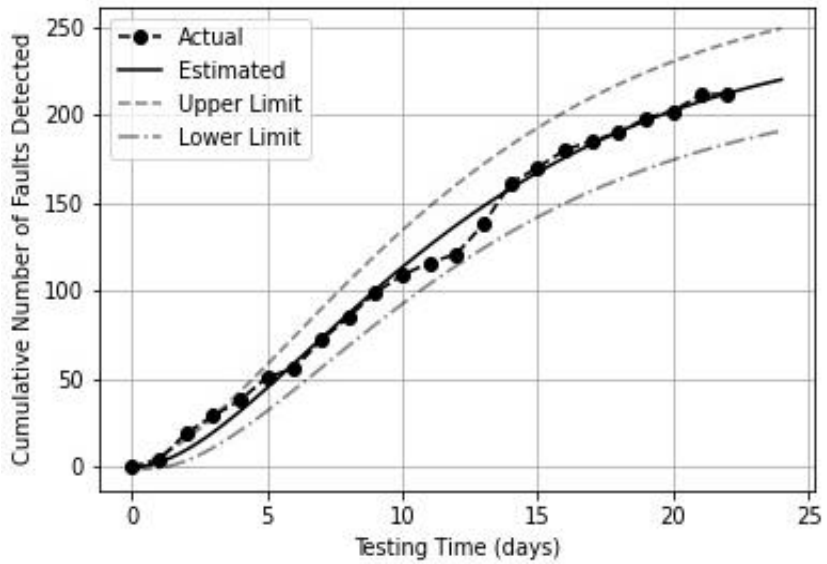

Figure 2. Estimated delayed S-shaped software reliability model with 95% confidence limits.

transforming the MTBF for testing into the MTBF for operation by using the environmental function.

6. Application Examples

Application examples in this section are for showing estimating procedures for the target failure measures and for showing SIL-based assessment. We apply the following software fault counting data consisted of 22 data pairs: (t_k, y_k) $(k = 1, 2, \cdots, 22;\ t_{22} = 22\ \text{(days)}, y_{22} = 212)$ (Yamada, 2011), where y_k is the total number of software bugs detected during $(0, t_k]$. This data has not been collected from testing for E/E/PE system software. The purpose of this section is just to show how to apply our approaches from software reliability data.

Applying a suitable nonhomogeneous Poisson process model which describes well the observed data is the first step in our approaches. As one of the examples, a delayed S-shaped software reliability model (Pham, 2007; Yamada, 2011), which is one of the models applied often in practical software reliability assessment. In this case, this model must be suitable for applying the data above because the software reliability data shows a S-shaped software reliability growth curve. The mathematical formulation of this model is $Z(t) = \alpha[1 - (1 + \beta t)\exp[-\beta t]]$, in which α is the initial total number of bugs in software and β is the rate of fault detection. The parameters have been estimated by the method of maximum likelihood as $\hat{\alpha} = 248.68$ and $\hat{\beta} = 0.1543$, respectively, where $\hat{\alpha}$ and $\hat{\beta}$ are the estimated values of the parameters α and β, respectively. Just for information, Figure 2 shows the estimated behavior of the delayed S-shaped software reliability model with 95% confidence limits. We have confirmed statistically that the estimated software reliability model in Figure 2 fits to the actual data with 1% level of significance by Kolmogorov-Smirnov goodness-of-fit testing.

Now it moves to the next step for estimating the target failure measures based on the estimated reliability model shown in Figure 2. We set DFR=0.01 just for showing numerical examples.

Further, as the simplest form, we set the environment function as $EF(x) = \mathrm{EC} \times x$, where EC is the coefficient for the difference of executing environment between the testing and operation. This paper sets x in the environment function as 1000 days and EC as 0.01, respectively, for just showing the numerical examples.

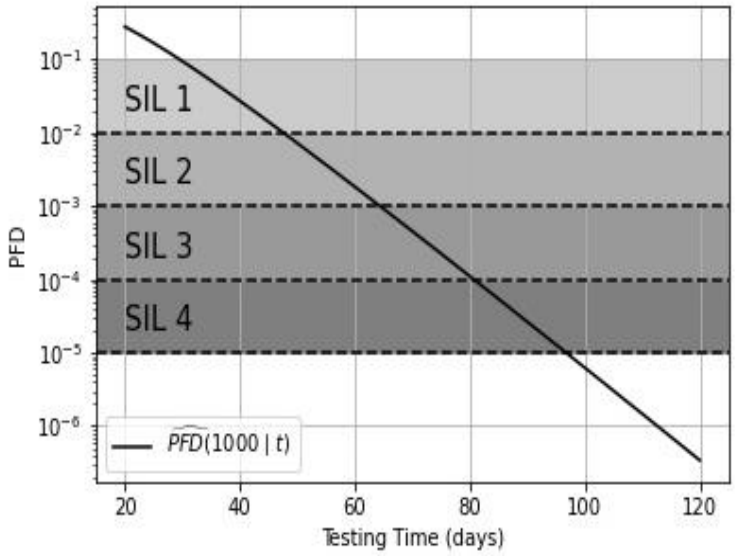

Figure 3. Estimated target failure measures of PFD.

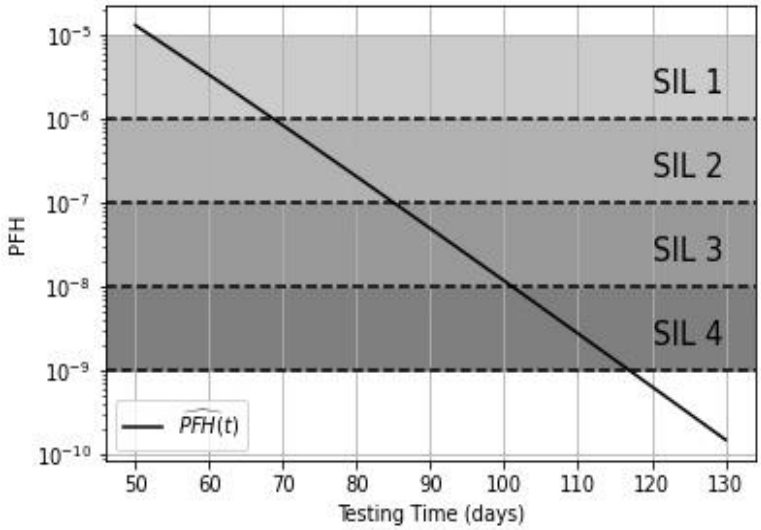

Figure 4. Estimated target failure measures of PFH.

In this situation, 1000 days in operation is equivalent to the 10 days in testing. It is worth mentioning that the values of DFR and EC and the mathematical structure of environmental function must be carefully considered along with software failure data observed and practical situations. Actually,

the value of EC is needed to give experimentally based on the difference of coverage maturity ratio for the possible software paths. And the designed operation time-duration is better to be set according to the proof-testing interval for examples. Now we consider the situation that the designed operation period is 1000 days in operation as the requirement. Then, the following equation:

$$PFD(1000|t) = 1 - \exp[-0.01\{Z(t + 0.01 \times 1000) - Z(t)\} \quad (8)$$

is obtained for approximately estimating PFD from Eq. (6). Eq. (8) enables us to estimate how much the testing time-duration is needed for conforming certain level of safety integrity for PFD. Figure 3 depicts the behavior of $\widehat{PFD}(1000|t)$ in Eq. (8) along with the testing time t and with the region for each level of safety integrity. The horizontal axis in Figure 3 follows the logarithmic axis.

Table 1. Estimated testing time required at least for conforming SIL.

SIL	Testing Time Required (days)	
	PFD	PFH
4	more than 80.68	more than 101.1
3	more than 64.36	more than 85.10
2	more than 47.54	more than 68.80
1	more than 29.54	more than 52.07

From Figure 3, the software development manager can see that how much the testing time-duration is needed for conforming certain level of SIL. Further, the second column of Table 1 shows the estimated testing time required at least for conforming each level of safety integrity. For example, about 8 days additional testing is required for conforming SIL 1 because the PFD at the test termination time of the data have not achieved yet the region of SIL 1. Regarding the PFH,

$$PFH(t) = \frac{\text{DFR}}{MTBF_I(t)/\text{EC}} = \frac{0.01}{100 \times MTBF_I(t)}. \quad (9)$$

Figure 4 shows the behavior of $\widehat{PFH}(t)$ in Eq. (9), in which we assume that this software has been designed as the high demand or continuous operation mode. And the third column of Table 1 shows the estimated testing time required at least for achieving certain level of safety integrity for PFH. From Figure 4 and Table 1, about 31 days of additional testing are needed to conform at least SIL 1 for this operation mode. Further, as shown in Figures 3 and 4, this operation mode requires longer testing time than the low demand operation mode for achieving each level of safety integrity.

7. Discussions

The previous section has provided numerical illustrations for the safety assessment methods by applying actual software reliability data based on proposed methods. To the best of our knowledge, this paper firstly discussed quantitative software safety assessment methods for conforming IEC 61508. And existing software reliability modeling and assessment technologies have been efficiently utilized in developing our mathematical models for estimating the targe failure measures, such as PFD and PFH, in this paper. Therefore, some other existing reliability models (Pham, 2000; Pham, 2007; Yamada, 2011; Yamada, 2014) can be applied in our assessment scheme. Further, combining our research work and the safety assessment method for the whole

E/E/PE hardware system is expected to yield more useful assessment technologies for the E/E/PE systems. The contribution, implications for practice, and limitation and future research directions are mentioned below.

7.1 Contributions

As mentioned, the probability-based quantitative safety assessment for the system is required for only hardware system by IEC 61508 based on the SIL. However, this paper has pointed out the needfulness of such kinds of assessment methodologies for the software of the systems. Research contributions of this paper can be listed as follows:

- Discussed the needfulness of probability-based quantitative safety assessment for E/E/PE system software.
- Proposed basic framework for estimating target failure measures.
- Proposed probability-based mathematical models for estimating quantitative measures for deciding the level of SIL by efficiently using existing mathematical assessment of software reliability and by considering the software executing environment between testing and operation.
- Generated useful information on software development management, e.g., the achieved SIL for the developed E/E/PE system software, testing time duration required to achieve certain SIL of E/E/PE software, respectively.

7.2 Implications for Practice

In development of the software of E/E/PE system, IEC 61508 instructs applying recommended software development technologies, such as CASE tools and formal methods, for achieving certain level of SIL. However, it is doubtful whether the developed software system of E/E/PE system really achieves the required level of safety integrity. This implies there exists the needfulness of the quantitative safety assessment method following an objective way for the software. This paper provides quantitative software safety assessment methods for conforming IEC 61508 under the technical background of reliability modeling and quantitative assessment technologies of software systems. The key procedures for conducting software safety assessment proposed in this paper is summarized as follows:

Step (1): Collecting software failure time interval data or software faults counting data with the rate of dangerous software failure occurrences for deciding the value of DFR.
Step (2): Choosing a software reliability model fitting well to the collected software reliability data and estimating the value of parameters in the applied software reliability model. It is better to conduct checking the goodness-of-fit of the model applied to data observed.
Step (3): Setting designed operation time-duration (e.g., the proof-testing time interval).
Step (4): Estimating the software reliability and instantaneous MTBF based on the estimated reliability model for the software system.
Step (5): Setting the value of EC experimentally based on the difference of coverage maturity ratio for the possible software paths.
Step (6): Introducing the above information to Eqs. (6) and (7).
Step (7): Confirming achieved current level of SIL and testing time duration required to achieve certain SIL of E/E/PE software by checking the time-dependent behaviors PFD and PFH.

Step (1)-Step (4) is well known procedure for conduct usual software reliability analysis in practice

and it is worth mentioning that some research institutes distribute software reliability assessment tools packaged Step (1)-Step (4) by online. Therefore, it is expected that our proposed approaches support software safety assessment conforming IEC 61508 without any difficulty.

7.3 Limitations and Future Research Directions

The limitations of this research works are summarized as follows:

- The estimation methods for the target failure measures are approximated methods. Especially, the PFD in Eq. (1) does not directly consider the frequency of the safety demand to the E/E/PE system.
- The verification of mathematical structure in the environment function, $EF(x)$, have not conducted yet by using practical software reliability data. This paper used the simplest form for the environment function.

By following the limitations listed above, the further studies are needed in the future. That is, more investigations on our approaches with appropriate values of EC and DFR are needed by using software failure and dangerous failure data collected from testing for the E/E/PE system software and more investigations on the suitable mathematical structure of the environment function are also needed for improving the accuracy of safety assessment. Considering the time-dependent dangerous failure occurrence mechanism between software and hardware interactions in E/E/PE system by incorporating our software safety assessment approach is expected to develop more practical and useful methodologies for the safety assessment of the system.

8. Conclusions

Estimation methods for the software target failure measures, which are the basic measures for deciding the level of SIL for the E/E/PE system have been discussed in this paper. The key point of this paper is that our approaches enable us to obtain some useful information for supporting software development management and for software safety assessment based on the SIL defined by IEC 61508 for E/E/PE system. Further, by applying our approaches, software developing manager could obtain the information on how much testing time-duration is needed for achieving certain level of safety integrity along with the designed operation mode defined by IEC 61508.

Conflict of Interest

The authors confirm that there is no conflict of interest to declare for this publication.

Acknowledgements

The authors would like to thank the editor and anonymous reviewers for their comments that help improve the quality of this work. This research was partially supported by the JSPS KAKENHI (C), Grant No. 19K04144.

References

Fujiwara, T., Kimura, M., Satoh, Y., & Yamada, S. (2011). A method of calculating safety integrity level for IEC 61508 conformity software. In *2011 IEEE 17th Pacific Rim International Symposium on Dependable Computing* (pp. 296-301). IEEE. Pasadena, CA, USA.

Ghadhab, M., Junges, S., Katoen, J.P., Kuntz, M., & Volk, M. (2019). Safety analysis for vehicle guidance systems with dynamic fault trees. *Reliability Engineering and System Safety*, *186*, 37-50.

Gu, T. (2011). A novel approach supporting evaluation of software safety integrity level on embedded systems. In *The 5th International Conference on New Trends in Information Science and Service Science* (Vol. 1, pp. 140-145). IEEE. Macao, China.

IEC 61508-2-10. Functional safety of electrical / electronic / programmable electronic safety-related systems.

Kato, E., & Sato, Y. (2000). Safety integrity level model for IEC 61508-Examination of modes of operation. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E83-A(5), 863-865.

Misumi, Y., & Sato, Y. (1999). Estimation of average hazardous-event-frequency for allocation of safety-integrity levels. *Reliability Engineering and System Safety*, *66*(2), 135-144.

Pham, H. (2000). *Software reliability*. Springer Verlag, Singapore.

Pham, H. (2007). *System software reliability*. Springer Verlag, London.

Yamada, S. (2011). *Elements of software reliability-modeling approach*. Kyoritsu-Shuppan, Tokyo.

Yamada, S. (2014). *Software reliability modeling: fundamentals and applications*. Vol. 5, Springer, Tokyo.