A Novel Safety Metric SM_{EP} for Performance Distribution Analysis in Software System

R. Selvarani and R. Bharathi

Abstract Focusing on safety attributes becomes an essential practice towards the safety critical software system (SCSS) development. The system should be error free for a perfect decision-making and subsequent operations. This paper presents an analysis on error propagation in the modules through a novel safety metric known as SM_{EP}, which can be characterized depending on the performance rate of the working module. We propose a framework for the analysis of occurrence of error in various modules and the intensity of it is quantified through probabilistic model and universal generating function technique.

Keywords Safety critical • Error propagation • UGF • Performance • Modules

1 Introduction

SCSS are pervasive in the medical field and it has to be designed with maximum care. The dependability requirement is an important criterion in these systems. To reduce the probability of losses, appropriate failure analysis practices have to be used to validate the safety of the critical system [1].

To achieve error free scenario of SCSS is very difficult, although the system has been well tested, used, and documented. If one part of a system fails, this can affect other parts and in worst case results in partial or even total system failure. To avoid such incidents, research on failure analysis is of high importance. Failure analysis is the proven approach for mitigating the system hazards and failure modes and consequently determines which of those are influenced by or affected by software or

R. Selvarani
Computer Science and Engineering, Alliance University, Banglore, India
e-mail: selvarani.r@alliance.edu.in

R. Bharathi (✉)
Information Science and Engineering, PESIT-BSC, Banglore, India
e-mail: rbharathi@pes.edu

A. Hosseinian-Far et al. (eds.), *Strategic Engineering for Cloud Computing and Big Data Analytics*, DOI 10.1007/978-3-319-52491-7_3

lack of software [1]. A failure of a safety critical system can be defined as "the non performance or incapability of the system or a component of the system to perform or meet the expectation for a specific time under stated environmental conditions."

The error propagation probability is a condition that once an error occurs in a system module, it might propagate to other modules and thereby cascades the error to the system output [2]. The error propagation analysis is a vital activity for the efficient and robust designing of safety critical software system.

Error propagation between software modules is a quantitative factor that reflects on the reliability of a safety critical software product. In general, the SCSS is considered between two major states, perfect functioning and failure state. Here we are considering several intermittent states between the two major states for the failure analysis. Hence these systems can be termed as *Multistate Systems* (MS) in our research. The reliability of a MS can be defined as a measure of the capability of the system to execute required performance level [3].

The presence of an error [4] in a software module might trigger an error in other modules of the system that are interconnected. Identifying error propagation in a software system is an important task during the development activity. Propagation analysis may be used to identify the critical modules in a system, and to determine how other modules are affected in the presence of errors. This concept will aid in system testing and debugging through generating required test cases that will stimulate fault activation in the identified critical modules and facilitate error detection [5].

The errors under consideration might be due to faulty design, which could result in errors and data errors due to wrong data, late data, or early data. The impact of error propagation across modules can be assessed by analyzing the error propagation process and arrive at a general expression to estimate the performance distribution of each module using computational intelligence because of its complexity and randomness [6]. As per IEC 61508, it is necessary to see that the design and performance of critical systems is safety enough to meet tolerable risk targets, taking into account of all failure sources including systematic hardware and software faults and random faults [7].

The reliability and performance of a multistate safety critical system can be computed by using Universal Generating Function (UGF) technique [8]. The UGF technique is based on probability theory to assess and express models through polynomial functions. The UGF technique applied for failure analysis in safety critical systems in this paper is adapted by following the procedure given by Levitin et al. [8, 9].

Hence the error propagation analysis provides the base for the reliability evaluation, since the occurrence of error propagation across the modules has a significant effect on the system behavior during critical states.

The paper is structured as follows: Sect. 2 describes background and Sect. 3 describes the proposed approach through a framework. The analysis of error propagation and failure of a SCSS is depicted in Sect. 4. Conclusion and looking beyond the area of this research are discussed in Sect. 5.

2 Background

According to Avizienis et al. [4], a *failure* is an event that occurs when the delivered service no longer complies with the expected service of the system. An *error* is an incorrect internal state that is liable to the occurrence of a failure or another error. However all errors may not reach the system's output to cause a failure. A *fault* is active when it results in an error otherwise it is said to be inactive. Nevertheless, not all faults lead to an error, and not all errors lead to a failure.

2.1 Error Propagation

Error propagation (EP) can be defined as a condition where a failure of a component may potentially impact other system components or the environment through interactions, such as communication of bad data, no data, and early/late data [10]. Such failures represent hazards that if not handled properly can result in potential damage. The interaction topology and hierarchical structure of the system architecture model provide the information on how the erroneous behavior of each system component interacts through error propagation.

Morozov et al. [5] have used probabilistic error propagation techniques for diagnosing the system. Henceforth it aids in tracing back the path of error propagation path to the error-origin. Moreover this diagnosis helps in error localization procedure, testing, and debugging [5].

The influence of error propagation in the overall system reliability has been demonstrated in [11]. With the help of UML artifacts, the system architectural information is used to find the probability of error propagation across system components [11]. Since they have used UML artifacts, their model can be used to predict reliability in the early phases of system development.

Hiller et al. [12] have initiated a new concept called "Error Permeability" through software modules, as well as a set of associated measures namely error exposure, relative permeability, and non-weighted relative permeability. They found these measures are helpful in assessing the vulnerable modules, which are exposed to error propagation.

A bottom-up approach is considered to estimate the reliability for component-based system in [2]. Foremost, the reliability of system component was assessed. Based on the architectural information, the system reliability was estimated taking into the account of error propagation probability. The system analysis was carried out through the failure model by considering only data errors across components. Authors in [2] have concluded that error propagation is a significant characteristic of each system component and defined as the probability that a component propagates the erroneous inputs to the generated output. Their approach can be used in the early prediction of system reliability.

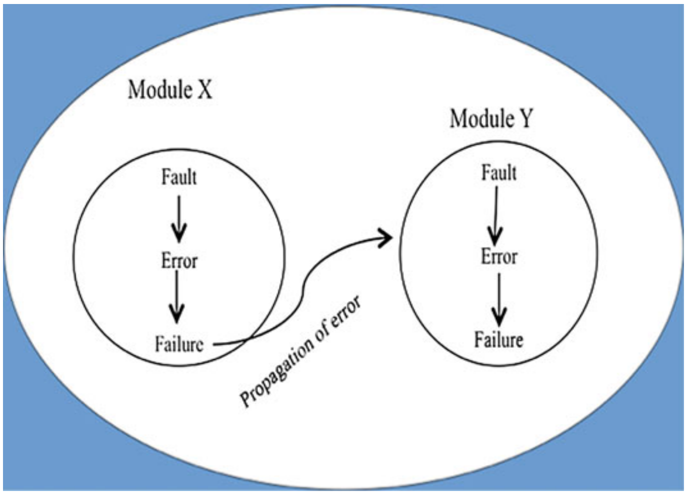

Fig. 1 Inter-modular error propagation

An error happens when there is an activation of a fault [4]. An error occurs in a module when there is a fault in the module and henceforth it cannot directly cause an error in other modules. Relatedly an error in a module can lead to its failure only within that module. The reason for the module error is either due to the activation of fault in the same module or due to deviated input service from other modules. A module failure is defined as the deviation of the performance from its accepted output behavior. If the failed module is the output interface module of the system then its failure is considered as a system failure [13]. System failures are defined based on its boundary. A system failure is said to occur when error propagates outside the system. Figure 1 depicts intermodular error propagation. Module X influences module Y.

In safety critical system, certain factors are considered crucial which signifies the safety of a system and such critical attributes should be consistently monitored throughout the lifecycle of the system [14]. This work focuses on analyzing error propagation in safety critical software systems. In this approach, we use a methodology called universal generating function (UGF) to quantify the performance distribution of a multistate safety critical software system [3] and subsequently introduce a new metric called Safety Metric (SM_{EP}).

2.2 MSS and Universal Generating Function

The UGF technique also called as u function is a mathematical tool introduced by Ushakov [15] and Levitin [8] expanded and proved that UGF is an effective technique for assessing the performance of real-world systems, in specific

Multistate Systems. In general all traditional reliability models perceived system as binary state systems, states being a perfect functionality and a complete failure. In reality, each system has different performance levels and various failure modes affecting the system performance [3]. Such systems are termed as Multistate Systems (MS).

Let us assume a MS composed of n modules. In order to assess the reliability of a MS, it is necessary to analyze the characteristic of each module present in the system. A system module *'m'* can have different performance rates and represented by a finite set $q_{m,}$ such that $q_m = \{q_{m1}, q_{m2}, \ldots .q_{mi\ldots.}q_{mk_m}\}$ [16], where q_{mi} is the performance rate of module m in the ith state and $q_i = \{1, 2, \ldots .k_m\}$. The performance rate $Q_m(t)$ of module 'm', at time $t \geq 0$ is a random variable that takes its value from q_m: $Q_m(t) \in q_m$.

Let the ordered set $p_m = \{p_{m1}, p_{m2}, \ldots .p_{mi}, \ldots p_{mj_m}\}$ associate the probability of each state with performance rate of the system module m, where $p_{mi} = Pr\{Q_m = q_{mi}\}$.

The mapping $q_{mi} \rightarrow p_{mi}$ is called the probability mass function (pmf) [17].

The random performance [18] of each module m defined as polynomials can be termed as module's UGF *($u_m(z)$)*

$$u_m(z) = \sum_{i=0}^{k} P_{mi} z^{q_{mi}}, \; where\, m = 1, 2 \ldots n. \tag{1}$$

Similarly the performance rates of all 'm' system modules have to be determined. At each instant $t \geq 0$, all the system modules have their performance rates corresponding to their states. The UGF for the MS denoted as *"($U_S(Z)$)"* can be arrived, by determining the modules interconnection through system architecture. The random performance of the system as a whole at an instant $t \geq 0$ is dependent on the performance state of its modules. The UGF technique specifies an algebraic procedure to calculate the performance distribution of the entire MS, denoted as *$U_S(z)$,*

$$U_s(z) = f\{u_{m1}(z), u_{m2}(z), \ldots, u_{mn}(z)\}, \tag{2}$$

$$U_s(z) = \nabla_{\phi}\{u_{m1}(z), u_{m2}(z), \ldots, u_{mn}(z)\} \tag{3}$$

where ∇ is the composition operator and ϕ is the system structure function. In order to assess the performance distribution of the complete system with the arbitrary structure function ϕ, a composition operator ∇ is used across individual *u function* of m system modules [17].

$U_S(z)$ is a U function representation of performance distribution of the whole MS software system. The composition operator ∇ determines the U function of the whole system by exercising numerical operations on the individual u functions of the system modules. The structure function $\phi(\cdot)$ in composition operator ∇

expresses the complete performance rate of the system consisting of different modules in terms of individual performance rates of modules. The structure function $\phi(\cdot)$ depends upon the system architecture and nature of interaction among system modules.

Reliability is nothing but continuity of expected service [4] and it is well known that, it can be quantitatively measured as failures over time. The UGF technique can be used for estimating software reliability of the system as a whole consisting of *n* modules. Each of the modules performs a sub-function and the combined execution of all modules performs a major function [17].

An assumption while using the UGF technique is that the system modules are mutually independent of their performance.

3 Proposed Approach

Error Propagation (EP) is defined as the condition where an error (or failure) propagates across system modules [19]. Our approach focuses on quantifying the propagation of error between modules in safety critical software system.

The analysis proposed in this research contains four different stages and explained through a framework. The framework, as shown in Fig. 2, is based on bottom-up approach in assessing the performance distribution of SCSS To start with, we have arrived at the performance distribution of each system module ($\mathrm{PD}_{\mathrm{MOD}}$) using *U function*.

The probability of error propagation in a module **($\mathbf{PD}_{\mathbf{MOD}}$ + $\mathbf{SM}_{\mathbf{EP}}$)** is quantified in the second step. As third step, the performance distribution of subsystems **($\mathbf{PD}_{\mathbf{SS}}$ + $\mathbf{SM}_{\mathbf{EP}}$)** is arrived through composition operator. As the final step the failure prediction is achieved through recursive operations for quantifying the error propagation throughout the system **($\mathbf{PD}_{\mathbf{SYS}}$ + $\mathbf{SM}_{\mathbf{EP}}$).**

During software development, this framework would be helpful to demonstrate the probability of error propagation to identify the error prone areas.

4 Error Propagation and Failure Analysis

The error propagation and failure analysis model is a conceptual framework for analyzing the occurrence of error propagation in SCSS. The system considered is broken down into subsystem, and each subsystem in turn is subdivided into modules called elements.

A module is an atomic structure, which performs definite function(s) of a complex system.

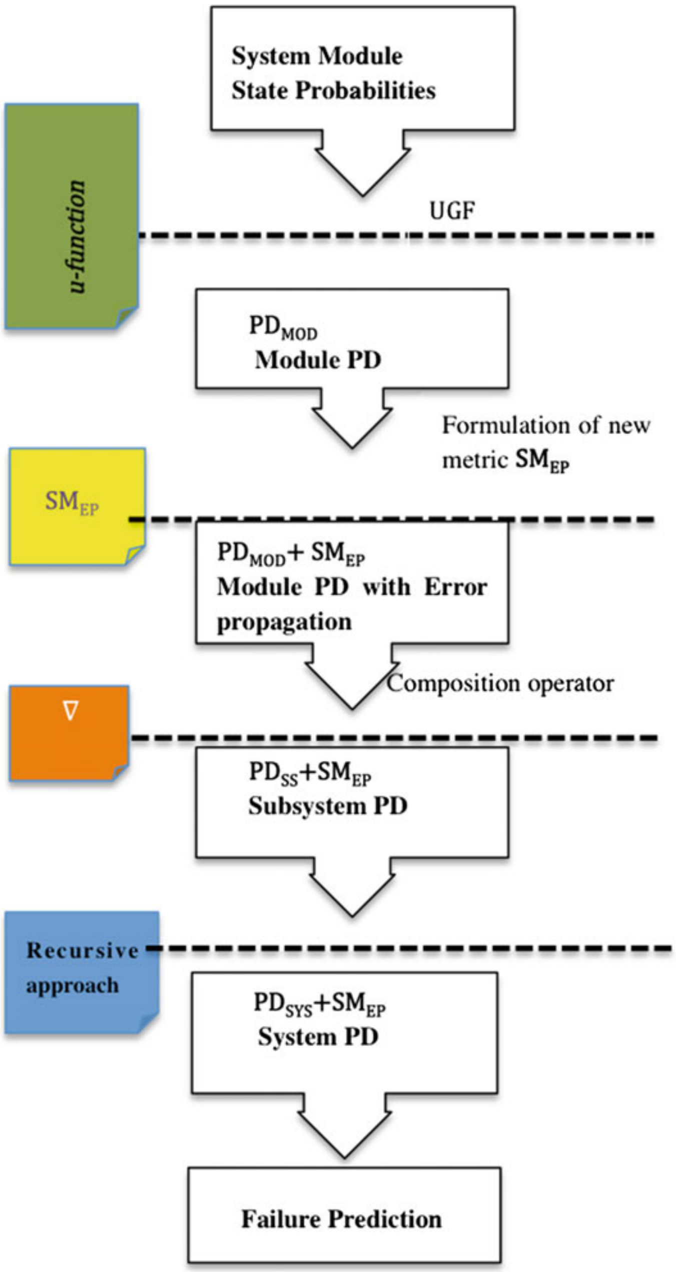
System Module State Probabilities
UGF
u-function
PD_{MOD} Module PD
Formulation of new metric SM_{EP}
SM_{EP}
$PD_{MOD}+ SM_{EP}$ Module PD with Error propagation
Composition operator
∇
$PD_{SS}+SM_{EP}$ Subsystem PD
Recursive approach
$PD_{SYS}+SM_{EP}$ System PD
Failure Prediction

Fig. 2 Error propagation and failure analysis

4.1 Performance Distribution of System Module

The performance rate of a module can be measured in terms of levels of failure.

Let us assume that the performance rate of a module *m* with 0% failure is q_{m1}, 10% failure is q_{m2}, 30% failure is q_{m3}, 50% failure is q_{m4}, and 100% failure is q_{m5}.

The state of each module m can be represented by a discrete random variable Q_m that takes value from the set,

$$Q_m = \{q_{m1}, q_{m2}, q_{m3}, q_{m4}, q_{m5}\}$$

The random performance of a module varies from perfect functioning state to complete failure state.

The probabilities associated with different states (performance rates) of a module *m* at time *t* can be represented by the set, $P_m = \{p_{m1}, p_{m2}, p_{m3}, p_{m4}, p_{m5}\}$, where $P_{mh} = Pr\{Q_m = q_{mh}\}$.

The module's states is the composition of the group of mutually exclusive events,

$$\sum_{h=1}^{5} P_{mh} = 1 \tag{5}$$

The performance distribution of a module *m* (pmf of discrete random variable G) can be defined as

$$u_m(z) = \sum_{h=1}^{5} P_{mh} z^{q_{mh}} \tag{6}$$

The performance distribution of any pair of system modules *l* and *m*, connected in series or parallel [18], can be determined by,

$$u_l(z)\nabla u_m(z) = \sum_{h=1}^{5} P_{lh} z^{q_{ih}} \nabla \sum_{h=1}^{5} P_{mh} z^{q_{mh}} \tag{7}$$

The composition operator ∇ determines the *u function* for two modules based on whether they are connected in parallel or series using the structure function ø. The equation arrived in Eq. (7) quantifies the performance distribution of combination of modules. Levitin et al. in [9] have demonstrated the determination of performance distribution when the modules are connected in series or parallel. A failure in a module may potentially impact other modules through error propagation.

4.2 Formulation of Safety Metric SM$_{EP}$

The probability of occurrence of EP in a module can be defined by introducing a new state in that module [9]. Assuming that the state 0 of each module corresponds to the EP originated from this module [8]. The Eq. (6) can be rewritten as,

$$u_m(z)_{ep} = P_{m0} z^{q_{m0}} + \sum_{h=1}^{5} P_{mh} z^{q_{mh}} \tag{8}$$

$$u_m(z)_{ep} = P_{m0} z^{q_{m0}} + u_m(z) \tag{9}$$

where p_{m0} is the probability state for error propagation and q_{m0} is the performance of the module at state 0.

$u_m(z)$ represents all states except the state of error propagation as represented in Eq. (6).

The performance distribution of a module m at state 0 is the state of error propagation which is given by $P_{m0} z^{q_{m0}}$ and is termed as safety metric SM$_{EP}$. This metric depends upon the probability of the module performance with respect to EP, whether a failed module can cause EP or not and whether a module can be infuenced by EP or not. The safety metric SM$_{EP}$ of each module will carry a weightage based on the probability of propagating error. If the module does not propagate any error, the corresponding state probability should be equated to zero [9].

$$p_{m0} = 0 \tag{10}$$

By substituting Eq. (10) in Eq. (8), the SM$_{EP.}$ is quantified as zero. Therefore Eq. (8) becomes,

$$u_m(z)_{ep} = \sum_{h=1}^{5} p_{mh} z^{q_{mh}} \tag{11}$$

The module that does not have error propagation property or state is given by $u_m(z)_{ep} = u_m(z)$, the *u function* in Eq. (11) is reduced to Eq. (6).

If a failed module causes error propagation, then the performance of the module in that state of error propagation is

$$q_{m0} = \alpha \tag{12}$$

where the value of α can be of any random performance q_{m1} or q_{m2} or q_{m3} or q_{m4} or q_{m5}. When any operational module m that will not fail due to error propagation can be represented by conditional pmf [9],

$$u_m(z)_{ep} = \sum_{h=1}^{5} \frac{p_{mh}}{1 - p_{m0}} z^{q_{mh}} \tag{13}$$

Because the module can be in any one of the five states as defined in Eq. (6). The safety metric SM_{EP} depends on the performance of each module in the multistate system. This metric helps to estimate the probability of EP to hazardous modules of the system and identify modules that should be protected with fault detection and fault recovery mechanisms.

Based on the above considerations, the conditional u functions of each system module have to be estimated. Depending upon the subsystem architecture, the u function of each subsystem can be quantified by applying the composition operator $\nabla_{\o}$. Then the recursive approach is used to obtain the entire *u function* of safety critical software system, which will be elaborated in the subsequent work.

5 Conclusions

This approach proposes a new framework to analyze the failure of multistate safety critical software with respect to error propagation and arrive at a new metric called safety metric SM_{EP}. This proposed new metric will be the key finding for the failure analysis of real-time safety critical system. This metric has the application in finding the failure probability of each module, the migration of error propagation from modular level to subsystem and then to system level and the process of identifying the most critical module in the whole safety critical software system and the impact of error propagation in the performance of SCSS. Our future work will continue by applying the safety metric SM_{EP} in relevant real-time SCSS for its failure analysis.

References

1. Sundararajan, A., & Selvarani, R. (2012). Case study of failure analysis techniques for safety critical systems. In *Advances in Computer Science, Engineering & Applications* (pp. 367–377). Heidelberg: Springer.
2. Cortellessa, V., & Grassi V. (2007). A modeling approach to analyze the impact of error propagation on reliability of component-based systems. In *International Symposium on Component-Based Software Engineering*. Heidelberg: Springer.
3. Levitin, G. (2008). A universal generating function in the analysis of multi-state systems. In *Handbook of performability engineering* (pp. 447–464). London: Springer.
4. Avizienis, A., et al. (2004). Basic concepts and taxonomy of dependable and secure computing. *IEEE Transactions on Dependable and Secure Computing 1.1*, 11–33.
5. Morozov, A., & Janschek, K. (2014). Probabilistic error propagation model for mechatronic systems. *Mechatronics, 24*(8), 1189–1202.
6. Eberhart, R. C., & Shi, Y. (2007). Chapter nine—Computational intelligence implementations. In *Computational intelligence* (pp. 373–388). Burlington: Morgan Kaufmann. ISBN 9781558607590.
7. IEC 61508. (2005). Functional Safety of Electrical/Electronic/Programmable Electronic Safety-related Systems. International Electro technical Commission (IEC 61508).
8. Levitin, G. (2007). Block diagram method for analyzing multi-state systems with uncovered failures. *Reliability Engineering & System Safety, 92*(6), 727–734.

9. Levitin, G., & Xing, L. (2010). Reliability and performance of multi-state systems with propagated failures having selective effect. *Reliability Engineering & System Safety, 95*(6), 655–661.
10. Feiler, P. H., Goodenough, J. B., Gurfinkel, A., Weinstock, C. B., Wrage, L. (2012). Reliability validation and improvement framework. Carnegie-Mellon Univ Pittsburgh Pa Software Engineering Inst.
11. Popic, P., et al. (2005). Error propagation in the reliability analysis of component based systems. In *16th IEEE International Symposium on Software Reliability Engineering (ISSRE'05)*. IEEE.
12. Hiller, M., Jhumka, A., & Suri, N. (2001). An approach for analysing the propagation of data errors in software. In *International Conference on Dependable Systems and Networks, 2001, DSN 2001*. IEEE.
13. Mohamed, A., & Zulkernine M. (2010). Failure type-aware reliability assessment with component failure dependency. In *2010 Fourth International Conference on Secure Software Integration and Reliability Improvement (SSIRI)*. IEEE.
14. Vinisha, F. A., & Selvarani, R. (2012). Study of architectural design patterns in concurrence with analysis of design pattern in safety critical systems. In *Advances in Computer Science, Engineering & Applications* (pp. 393–402). Heidelberg: Springer.
15. Ushakov, I. A. (1986). A universal generating function. *Soviet Journal of Computer and Systems Sciences, 24*(5), 118–129.
16. Levitin, G., & Lisnianski, A. (2003). Multi-state system reliability: Assessment, optimization and applications.
17. Levitin, G. (2005). *The universal generating function in reliability analysis and optimization*. London: Springer.
18. Levitin, G., Zhang, T., & Xie, M. (2006). State probability of a series-parallel repairable system with two-types of failure states. *International Journal of Systems Science, 37*(14), 1011–1020.
19. Sarshar, S. (2011). Analysis of Error Propagation Between Software Processes. *Nuclear Power-System Simulations and Operation*, 69.