Reliability Engineering and System Safety 145 (2016) 366–372

Contents lists available at ScienceDirect

Reliability Engineering and System Safety

journal homepage: www.elsevier.com/locate/ress

Proof-testing strategies induced by dangerous detected failures of safety-instrumented systems

Yiliu Liu*, Marvin Rausand

Department of Production and Quality Engineering, Norwegian University of Science and Technology, NO 7491 Trondheim, Norway

ARTICLE INFO

Available online 21 July 2015

Keywords:
Safety-instrumented system
Proof test
Insert test
Dangerous detected failure
Dangerous undetected failure

ABSTRACT

Some dangerous failures of safety-instrumented systems (SISs) are detected almost immediately by diagnostic self-testing as dangerous detected (DD) failures, whereas other dangerous failures can only be detected by proof-testing, and are therefore called dangerous undetected (DU) failures. Some items may have a DU- and a DD-failure at the same time. After the repair of a DD-failure is completed, the maintenance team has two options: to perform an *insert* proof test for DU-failure or not. If an insert proof test is performed, it is necessary to decide whether the next scheduled proof test should be postponed or performed at the scheduled time. This paper analyzes the effects of different testing strategies on the safety performance of a single channel of a SIS. The safety performance is analyzed by Petri nets and by approximation formulas and the results obtained by the two approaches are compared. It is shown that insert testing improves the safety performance of the channel, but the feasibility and cost of the strategy may be a hindrance to recommend insert testing.

1. Introduction

Safety-instrumented systems (SISs) are widely used in many industries (e.g., process, nuclear, oil and gas industry) to prevent hazardous events and to mitigate the consequences of such events [2,3]. International standard [2] uses safety integrity, the probability of a SIS satisfactorily performing the specified safety instrumented function under all the stated conditions within a stated period of time as a performance measure.

A SIS has at least three subsystems: sensor, logic solver and final element subsystems. A sensor subsystem (with one or more sensors) detects possible undesired event and send signals to the logic solver subsystem (with one or more logic solvers), which can interpret these signals and decides which actions should be taken. The final element subsystem also can have one more elements that take prescribed actions to prevent harm to plants, processes or machineries, namely, equipments under control (EUCs) [16]. Each subsystem may have one or more channels, which can independently perform a safety function.

In terms of safety integrity assessment of SISs, many researches have been carried out, e.g. for measuring the effects of system architectures [6,14,20], effects of testing strategies [11,19], different demand modes and associated measures [1,7,9,13], and varying modeling methods [2,5]. More information about research achievements and directions of SISs can be found in [15,16].

In the literature, dangerous failures is regarded to occur once a SIS has no capability to response to a demand, which can be an event or a condition [16]. After such failures, the SIS will fall into a dangerous fault state. In fact, most modern SISs have built-in diagnostic self-testing capabilities that can detect many dangerous failures almost immediately such that a repair action can be initiated. These dangerous failures are called dangerous detected (DD) failures. It should be noted that diagnostic tests seldom discover all dangerous failures/faults, and the percentage of faults that can be revealed is called as diagnostic coverage (DC). Dangerous failures that are not detected by diagnostic testing are called dangerous undetected (DU) failures and are only revealed in proof tests that are carried out at regular intervals (e.g., once per year).

The mean time from a DD-failure occurs until the function is restored, MTTR, is usually rather short (e.g., 5–8 h), and DD-failures will therefore not be a main contributor to the safety unavailability of a SIS that is operated in low-demand mode (i.e., where demands for the safety function do not occur more often than once per year). The average probability of failure on demand (PFD_{avg}) in a (long) period is always used as the performance measure of a SIS/SIS subsystem/SIS channel [2,3,10].

For some channels, DD-failures can be repaired on-line while the process is running as normal during the repair. In most cases, however, the EUC has to be brought to a safe state (most often stopped) during the repair of the DD-failure. For some channels, DD- and DU-failures can be present at the same time and repairing a DD-failure does not guarantee that a DU-failure is not remaining in the channel. In some cases, it may be possible to proof-test for a DU-failure as part of the repair of the DD-failure.

* Corresponding author.
E-mail address: yiliu.liu@ntnu.no (Y. Liu).

http://dx.doi.org/10.1016/j.ress.2015.06.016
0951-8320/

Such proof tests can be regarded as *insert tests* between two scheduled tests, such that the total number of proof tests in a certain time period will increase. This means that the average length of the proof test interval will be reduced. Because the length of the proof test interval has a significant influence on the unavailability of a SIS [17], the new proof tests induced by DD-failures should also have influence. Thus, the objective of this paper is to model the relationship between such proof tests induced by DD-failures and SIS performance, and to study the effects of these tests.

The remainder of the paper is organized as follows: Section 2 presents the possible follow-up test strategies of a single channel SIS after a DD-failure is revealed. Next, the modeling approach is briefly introduced, and Petri net models for different strategies are studied in Section 3. The effects of different test strategies on the SIS availability performance are analyzed in Section 4. And then, general approximation formulas are proposed for more complex systems. Finally, Section 6 presents conclusions and research perspectives.

2. Testing strategies induced by DD-failures

First, we study a simple SIS subsystem with only one channel. When a DD-failure in this system is detected, the maintenance team can repair the SIS channel in a short time, and then they have three options for testing the SIS for DU-failures:

- *Strategy* I: Do not perform any insert proof test for DU-failures.
- *Strategy* II: Perform an insert proof test for DU-failures, while keeping the proof-testing schedule unchanged.
- *Strategy* III: Perform an insert proof test for DU-failures, and change the proof-testing schedule (by postponing the subsequent proof test).

To illustrate the difference between strategies II and III, consider a solenoid valve that is scheduled to be proof-tested each April and assume that a DD-failure occurs in September. If strategy II is applied, a proof test for DU-failures is initiated immediately after having repaired the DD-failure, and the next proof test is still carried out the next April. If, on the other hand, strategy III is applied, the next proof test is postponed till next September keeping the same interval between two proof tests.

We use the long-term average probability of failure on demand, PFD_{avg}, to measure the safety unavailability of the SIS. DU-failures are always the main contributor to the PFD_{avg} because they may put the SIS in an unavailable state for a long time until a proof test is carried out. Fig. 1 illustrates possible shapes of the probability of failure on demand, PFD(t), as a function of the time t ,when test strategies II and III are applied, respectively.

In Fig. 1, $t_1, t_2, \ldots$ denote the times when DD-failures occur, and τ is the test interval. It is shown in Fig. 1(a) that the predefined proof-testing schedule is kept unchanged under strategy II, and each proof test can reduce the value of PFD(t) to 0. Fig. 1(b) for strategy III illustrates that the time to the next proof test is recounted after an insert test induced by a DD-failure.

3. Petri net analysis

Petri nets are used in this paper to model the different testing strategies. Petri nets have been adapted to SIS reliability analysis [5,7,16] especially for testing strategies of SISs [10,12], and is also a recommended modeling approach in IEC 61508 [2] and ISO 12489 [8].

The international standard IEC 62551 [4] defines the terminology of Petri nets in dependability analysis. Places (shown as circles in Fig. 2) and transitions (shown as bars) are two basic elements, which are connected with directed arcs. Tokens are illustrated as bullets to

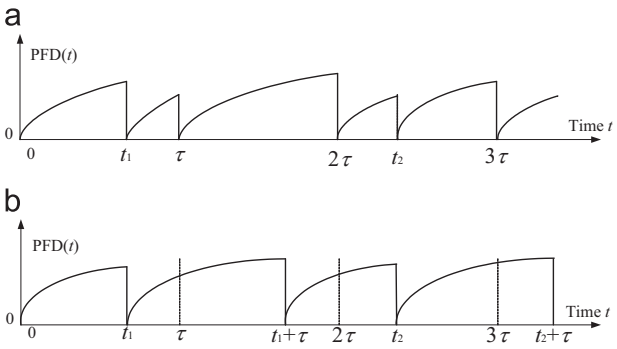

Fig. 1. PFD for test strategies II (a) and III (b) as a function of time t, adopted from [12].

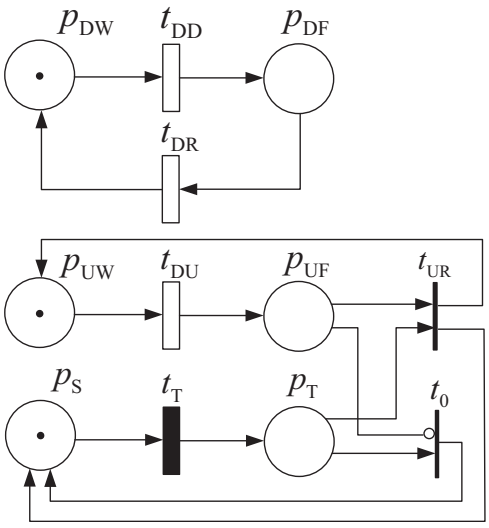

Fig. 2. Petri net model for test strategy I.

express the movable resources in the system and reside in the places. For each arc, a multiplicity is assigned to denote the token delivering capacity of the arc. The distribution of tokens in the places is regarded as a marking, and each marking represents a system state.

When all input places to a transition have at least as many tokens as the multiplicities of the associate arcs to the transition, the transition is enabled. And then, the transition can be fired to change the distribution of tokens in places. A firing time (delay from enabled to fired) can be assigned to each transition. In IEC 62551 [4], a thin bar is used to represent an immediate transition (zero firing time), a blank bar is for a transition with exponential firing time, and a filled thick bar is for the transition with constant firing time.

In addition, an inhibitor arc (shown as a small circle at the end of an arc) is sometimes used to prevent a transition from being enabled. Such a special arc enables its output transition when there is no token in the associate place. More details for Petri nets can be found in IEC 62551 [4].

Petri net models in IEC 61508 [2] have in addition predicates and assertions, which are defined in [2] and [18] as

- a predicate (identified by "?" or "??") is a formula to control the enabling condition of a transition;
- an assertion (identified by "!" or "!!") is a formula used to update one variable when the transition is fired.

Such interpretations are also helpful in some ordinary Petri net, and may make the model more compact and understandable.