UDC 004.054

B. I. Moroz, Doctor of Engineering Science,
Head of the Information System and Technology
Department of the University
of Customs and Finance
A. O. Holtvianskyi, Postgraduate Student
of the Information System and Technology
Department, University of Customs and Finance

DEFINITIONS AND CRITERIA FAILOVER RESILIENCY SOFTWARE TESTING OF SOFTWARE

Sets out the need to define criteria for failover software as one of the key elements ensuring the quality of software and their influence on the process of software development. The analysis of well-known models of reliability of software. Considered the principles and characteristics of models to ensure reliable program and fault tolerance criteria. Based on the conducted analysis also put forward their own criteria of fault tolerance, which should be considered in the development and testing of software systems regardless of their production and purpose. Considered the possibility of developing fault tolerant software, and some aspects that have a direct impact on the cost of development and maintenance of reliability programs and software systems.

Key words: *fault tolerance; criteria; software quality; software; development; reliability.*

Визначено критерії відмовостійкості програмного забезпечення як одного з основних елементів забезпечення якості програм і програмних комплексів та їх вплив на процес розробки програмного забезпечення. Проаналізовано загальновідомі моделі забезпечення надійності програмного забезпечення. Розглянуто принципи та характеристики моделей надійності програмного забезпечення та їхні критерії відмовостійкості. На підставі проведеного аналізу також висунуто свої критері відмовостійкості, які слід зважати під час розробки та тестування програмних комплексів незалежно від їх виробничого і цільового призначення. Розглянуто можливість розробки відмовостійкого програмного забезпечення, а також деякі аспекти, які безпосередньо випливають на вартість розробки і забезпечення надійності програм і програмних комплексів.

Ключові слова: відмовостійкість; критерій; програмне забезпечення; якість; розробка; надійність.

Problem formulation. Software fall-over protection is the complex system of the not formalized criteria the main objective of which is the provision of the acceptable level of reliability and quality of the software.

The main problem [1–3] consists in the necessity to formalize the generalized criteria of fall-over protection in order to obtain the software fall-over protection evaluation complex criteria.

Analysis of recent researches and publications. In modern world, the main role in the information processing plays software. One of the fundamental problems of such software development is the problem of guarantee of its long-term usage, failure and fault-free operation [1].

The main objective of fall-over protection achievement is the development of reliable systems. In a broader sense, the fall-free protection is the ability of the system to ensure sufficient level of user maintenance notwithstanding any factors and exposures [2].

Year after year information systems penetrate into various spheres of society, correspondingly, provision of such systems fall-free protection becomes more and more priority problem [3].

However, determination of fall-free protection indices is specific and sophisticated problem [4], and such problem shall be considered on a case-by-case basis.

In order to define weak points of the system it is required to provide the appropriate software fall-free protection testing.

Purpose of the article. To define main fall-free protection criteria that shall be considered at the software development and to consider main models of the software reliability.

Main material. Software reliability models may be divided into such groups as discrete and static. The main difference between these groups consists in the fact that the discrete models test the software at first, searching for bug, and then correct them, and the static models do not consider the period of program bugs introduction [3].

Let us briefly run through the main models of the software reliability.

One of the main models of the software reliability is the model of Jelinski-Moranda, developed in 1972, that is one of the first models of the software reliability. The main principles, on which this model is based, are:

– the number of bugs in code is fixed and known;

– fault handling in case of bug introduction is carried out instantaneously anddoes not lead to introduction of new bugs into the software at its testing;

– intervals between the failures is independent.

The software reliability function, obtained with the help of this model, may be expressed as:

$$Rm(t) = \exp(-(N-m)ct), \tag{1}$$

where $R(t)$ – software reliability function;

m – number of failures, detected within the period of testing;

c – proportionality factor.

The next reliability model is the reliability model of Mills, the reliability provision of which is based on equal level of system accidental faults and accidental artificial faults detection by the tester. The fact is that before the testing process start the program and its modules are filled with certain number of known, artificial faults.

The initial number of faults (N) is estimated with the help of the formula:

$$N = \frac{s \times n}{v}, \tag{2}$$

where s – total number of artificial faults;

n – number of accidental faults;

V – number of detected artificial faults.

In Bernoulli reliability model, the program start is performed with only two results "true" or "false". And the probability of obtaining of the program start incorrect results out of the total number of starts may be expressed with the help of binominal distribution:

$$B(p,n,k) = C(n,k) \times p^k \times (1-p)^{(n-k)}, \tag{3}$$

where $C(n, k)$ – binomial coefficient; p – probability of event; n – number of tests.

One of the main software reliability models is intuitive model. This model of the software reliability provision organizes independent testing of the program by different programmers. During such tests there is carried out the faults fixation. And at the evaluation of the faults remaining number, the test results of different programmer are collected and compared.

Correspondingly, the evaluation of the initial number of program faults is defined with the help of the following formula:

$$N = \frac{n_1 \times n_2}{n_{12}}, \tag{4}$$

where N – evaluation of the initial number of faults.

Specific task performance evaluation is required and important at the software development. Just with their help we may estimate the satisfaction or dissatisfaction of the set problems and requirements.

The analyzed sources [1–4] have no complete algorithm of the software fall-free protection criteria complex determination.

The authors believe that at the software development there should be set up the following generalized criteria and requirements of reliability, raising them to extremum and omitting secondary criteria:

1. Program component importance;
2. Program component failure probability;
3. Total duration of continuous operation;
4. Total duration of downtime;
5. Failure probability;
6. Time of the system recovery upon failure and its reconfiguration;
7. Failure rate;
8. Time between failures.

Conclusions and further researches directions. From our point of view, in systems that deal with information processing the problem of the software fall-free protection provision and determination of their criteria shall be set maximally clear, taking into consideration the application domain.

At such approach, one should not discard the fact that almost all reliability criteria describe internal faults and incorrect user input data sets countereffortsfforts, but in such a case there is not considered the external, intentional interference into the software operation.

There shall be paid special attention to the reliability criteria development at software coding. However, the fall-free protection provision is the operation that requires redundancy.

Perspectives of further studies are connected with:

1) development of the software fall-free protection testing method, based on the proposed criteria;

2) optimization of the software fall-free protection testing methods.

List of sources used:

1. Липаев В. В. Тестирование компонентов и комплексов программ : учебник / Липаев В. В. – М. ; Берлин : Директ–Медиа, 2015. – 528 с.

2. Прокопенко А. В. Синтез систем реального времени с гарантированной доступностью программно-информационных ресурсов : монография / А. В. Прокопенко, М. А. Русаков, Р. Ю. Царев ; Сиб. федер. ун-т, Ин-т космич. и информ. технологий. – Красноярск : СФУ, 2013. – 91 с.

3. Dubrova E. Fault Tolerant Design: An Introduction / Dubrova E. ; Department of Microelectronics and Information Technology, Rolal Institute of Technology. – Stockholm, Sweeden, 2008. – 147 p.

4. Кузнецов А. С. Многоэтапный анализ архитектурной надежности и синтез отказоустойчивости программного обеспечения сложных систем / Кузнецов А. С., Ченцов С. В., Царев Р. Ю. ; М-во образования и науки Рос. Федерации, Сиб. федерал. ун-т. – Красноярск : СФУ, 2013. – 142 с.