Potapov I.V.

ISSUES OF SOFTWARE SYSTEMS DEPENDABILITY

The article addresses the key problems, tasks and solutions within the system analysis methodology of software systems dependability. The author presents opinions regarding the solution of software systems dependability problems. The problems in question are classified and organized.

Keywords: *dependability, software tool, software system.*

Introduction

As early as the end of the last century there was a visible trend of significant redistribution of the global IT market in favor of software. As of today, computer technology is used everywhere. Due to that fact software systems users and developers alike are interested in the development of quality software products compliant with current global standards. Let us not dwell upon the general matters while noting one important detail: in those cases when higher quality requirements are imposed on computer-assisted systems, such properties as software dependability and safety are often involved. The article specifically deals with the dependability as it is understood in engineering, and not other properties defined with similar terms like precision, authenticity, plausibility, etc.

System analysis methods are widely used in software systems development for description, analysis and design. This method can be used in the software systems dependability studies. Problem definition is one of the key stages of system analysis [1]. The author tentatively summarizes the problems and tasks the users and developers of dependable software systems face. The constraints of a journal publication call for a certain transformation of the information acquired in the process of problem definition and originally represented in a "knot" of problems. This transformation can be considered as an adaptation of the input data supplied by the involved parties. System analysis literature recommends classifying and structuring problems, which is what this article aims to do. Referenced are the fundamental works that reflect major ideas and trends that characterize the problems of software systems dependability. Many problems and tasks that today's researchers, developers and users face are fundamental to this field of knowledge, therefore the authors and publications referenced in this article, including the very earliest ones, can quite be considered as the foundation for problems definition. This article also aims to classify and analyze publications on software systems dependability.

Problems definition

Experts often point out the fact that software systems (e.g. in [2]) are complex as it is understood by system theory and system analysis researchers. That fact largely defines the key problems and tasks of dependable software systems development, most notably the design, debugging and errors diagnostics, individual components and whole system fault detection. In [3], the complexity is considered to be one of the key factors that define software system undependability. In [4], the authors point out the correlation between the complexity and

the software system quality, while managing those factors represents an additional dependability problem. According to the authors of [5], the problems of the software systems dependability theory rely on the matters relating to complex software systems dependability research.

One of the important software systems dependability problems is the software errors research [3, 5, 6], as their presence is a key reasons of software systems failures. The very definition of the terms "error" and "failure" of a software system is a problem of its own.

In [2], the author analyses the concepts of program correctness and program errors. The problems may include the development of correctness and absence of errors evaluation method. Apart stand the problems of precise definition of the error for their reliable identification and cause isolation, as well as the development of methods for their detection, as the presence of errors is largely determined by the user's expectation and is not always strictly formalized [2, 3, 7]. Taking user's expectation into consideration in the design of software systems [3] can be regarded as a significant part of the dependability agenda. The same book dwells upon the problems related to the introduction of errors into software and their manifestation in the form of software system failure. [2] points out the importance of classification, identification of characteristics and error statistics analysis, as well as statistical determination of the total number of errors in a program. This problem along with a number of solutions is discussed in [8, 9]. The authors share their experience of methodology development and practical execution of software errors classification and statistical processing. Organizing data collection for error analysis and classification is a significant problem as well [7, 9].

This aspect should also include the matters of program code analysis for identification of statistical characteristics that would allow for a tentative evaluation of software system properties including dependability [4]. It can be assumed that using appropriate structural model generation methods would allow revealing the patterns of program errors introduction.

Identifying what is a random value, random process, etc. is one of important problems that define the applicability of conventional methods of the dependability theory. It should be noted that software errors themselves are not random values. Experts point out that random events are manifestations of errors occurring during the software systems operation [2, 8, 10, 11]. The random nature of the events is explained by the fact that in order for the error to manifest itself, a specific situation must occur (a combination of a certain condition of the system and an input action). As such combinations are fairly rare and hardly predictable the moment of software error can be deemed a random value. In other words, within the software dependability problem field it is possible to consider the characteristics of input action flows.

Among the error problems are the matters related to the requirement to reduce the number of software system design errors [2, 10]. In [8], the author stresses the requirement to prevent errors and manage the dependability during the design, development and modification of software systems, as well as to use functional failure protection systems during the operation. [7] defines and specifies the problems of software system dependability requirements justification, as well as organization of system design with the required level of dependability. The paper also sets forth a detailed classification of factors that define the software system dependability problematics.

A number of issues of software system functional safety that are studied using the same methods are an important part of the dependability problematics. Ony of the key differences that requires a terminological delimitation of these concepts consists in the fact that an analysis of software system operation does not include all failures as in the dependability theory, but only those that entail safety-specific consequences [10]. Paper [7] details the dependability of software measured in terms of losses incurred by the user due to design and operation flaws. The same paper defines an important part of the dependability problematics, the requirement to take into consideration the software system operational environment.

One of the central problems of software system dependability is the significant difference between a computer program and a technical device. Dependability requirements are defined by experts who are guided by conventional concepts of the dependability theory and the current standard developed for engineering systems. At the same time, software systems differ from technical system so significantly that conventional approaches may turn out to be completely or partially inapplicable. For example, in [8] it is clearly stated that the research of software system dependability should primarily focus on real-time systems. The author notes the impossibility to calculate software system dependability using conventional methods (that statement is somewhat attenuated in the sense that the conventional approaches are limitedly applicable only to real-time systems). The difficulties of analytical calculation of software system dependability and functional safety are also noted in [10]. That allows extending the problematics indicating the requirement to define appropriate concepts, dependability indicators and approaches to analytical, statistical and experimental studies of software system dependability. At the same time, it is highly important to clearly understand the differences between software systems and other types of technical equipment in terms of dependability. Those are detailed in several fundamental papers, of which we should note [3, 5, 8, 10, 12-14]. Those differences largely define and compliment the respective aspects of the problematics. Special methods of software system dependability studies must be developed. A particular attention should be given to the design stage, definitions and terms application criteria must be made clear. For instance, it is very important [3, 5, 12] to define the terms "error", "dependability»", etc.

In [2, 10], the authors look at two varieties of software systems, the first of which is characterized by the fact that the developers can give less attention to the matters of dependability, as the quality of the software system mostly matters only to the developers themselves. The author points

out that the matter of dependability is mostly relevant to software systems of the second type that are more similar to other industrial products.

In [2, 8, 10], technical systems dependability indicators are covered. It is suggested to regard time to failure and availability factor as usable for the purpose of software system analysis. The classification of models for software system dependability indicators calculation, as well as a discussion of the related problems is given in [5]. For example, there is a well-known problem of software system dependability indicators identification that is due to the fact that some important quality characteristics (number of software errors, etc.) are not perfectly suited for dependability evaluation. This part of the problematics includes the requirement to take into consideration the specific features of software as compared to other engineering products. In [12], for instance some dependability indicators are covered that take into consideration the specificity of software system failure.

There are several ways to evaluate the dependability of software as part of information systems [7, 14]. On the one hand, software is an extremely important component and its dependability must be evaluated separately. On the other hand, when information systems are considered, it is assumed that computer systems and software failures are interconnected, which means that the dependability of such systems must be evaluated as a whole.

Software systems failures may occur upon introduction of data that was not taken into consideration during debugging and testing [2, 6]. There is a problem of failure recording during software system dependability testing that is due to their relatively rare occurrence, the high labor cost of software testing and significant expenditures, especially in case of functional safety research [7, 8, 10].

It may become necessary to ensure classification of failures and faults according to recovery time, as that affects the software system malfunctions registration. In order to improve dependability it is required to properly organize the procedure of system recovery after failures. In some cases subject to time parameters the recovery allows transforming failures into less severe faults. Those ideas and concepts are covered in [2, 10]. Those papers, beside the classification of failures and faults, specify and detail software system recovery and utilization of various types of redundancy for that purpose.

The matters of applicability of redundancy in software, as well as software system data and operating processes have been discussed by many authors, namely [2, 4, 13]. In most cases, the definitions subject matter terms complies with dependability theory conventions. The main focus is on software (algorithmic), information and time redundancy. In this context, we must deal with the traditional issues of optimization problem definition [2], i.e. the problematics are extended with the management of redundant resources management. At the same time, it is not completely clear how to quantify the software system dependability subject to expenditures. Based on [4], it can be suggested to consider this problem along with a group of questions regarding the collection and analysis of statistical information. The opinion regarding the insufficient efficiency of redundancy is also part of the problematics [13].

In [14] and papers referred in it, the authors, while examining the matters of classifications and causes of software errors (which in turn is part of the software system dependability problematics), touch upon another important issue, the influence of the human factor on software systems dependability or rather the problem of evaluating this influence and the properties of the programmer (or team of programmers). Some of the related issues are covered in [3, 5, 9, 12, 15].

The requirement to develop models of the studied systems and processes due to their complexity becomes an increasingly important part of software systems dependability problematics. A significant role is played by the tolerances and assumptions [2] used in the development of mathematical models. Note that time as a parameter of a mathematical model does not have the meaning typical to dependability studies of other engineering systems [5], excluding perhaps real-time software systems [2, 8, 10]. Among the disadvantages that complement the problematics of software system dependability models [8] notes, that dependability prediction models designed using conventional methods may be usable only if the number of failures is high, while attempts to apply them to well debugged and highly dependable software systems do not yield acceptable results. In literature, we can find a great number of mathematical models developed for software dependability research that are classified by different criteria [6, 11, 13].

Related matters of software system quality constitute an important part of the problematics. Current standards in this area that define dependability as a quality characteristic can be considered obsolete. Experts turn to more recent versions of international standards or gently avoid those matters. In any case, the matters of quality are a topic apart. As quality analysis is not the focus of this paper, below will be covered a number more general issues that may be relevant to further research.

Experts regard dependability as one of the quality characteristics of software systems [8, 10, 14]. Both quality and dependability problematics may include matters related to quality evaluation and management. In [4], software system quality is associated to some aspects of structural complexity. Special complexity factors of software systems and components are analyzed. In [9], dependability problematics includes the analysis of the structural features of software. The problem of software system complexity evaluation and correlation between complexity and dependability is described in [5], where, among other things, the authors point out the dependence of the errors on not the size of program code, but on the software system structure, rule out the possibility of generalizing the error research results even in case of constant development environment and suggest an individual approach to each particular newly developed software system. In [14], certain qualities are associated with program usability, which corresponds with the above part of the software system dependability and quality problematics, where software errors and functional failures are considered in the context of user expectations.

While finalizing the problematics we need to return to the above question of improvement of the standards of software system dependability and quality [8, 10]. Probably, the solution to those problems is closely related to the general dependability issues of the standard, as described in [16]. The main problems in software system dependability are the standardization of terminology [10] (especially the basic concepts of failure, operational condition, etc.), formalization of dependability and quality indicators, improvement of design and maintenance processes. Standardization will not be covered here, as it is a major topic of a separate paper.

Discussion and conclusions

The development of the problematics is only the beginning of a problem domain analysis. At this stage it is too early to make the final conclusions and take decisions, but we can make some assumptions regarding the further efforts of the researchers in this area of knowledge.

For instance, it can be said that a comprehensive study and solution of the problems described above requires the use of system analysis methods. That statement is based on the following. First, there are difficulties in applying conventional formal methods due to obvious problems of formalization of the processes in question, awkwardness of object descriptions, multicriteriality of the decision-making tasks, etc. Second, the specific tasks resulting from the decomposition of the general task have less uncertainty and can probably be solved subject to their further integration for the purpose of obtaining system-wide results.

It can be noted that the simple classification and arrangement of problems given in this paper are not the only ones possible. The practice of system analysis research involves considering various ways of problematics elaboration. One of the solutions rather than all the others would probably allow implementing the above method in obtaining the comprehensive solution.

Examining almost each individual problem covered in this paper may yield a disappointing conclusion that a universal solution does not exist. That becomes clear when, for example, it is required to take the decision to study the dependability of a software system as a component of a hardware and software system or an independent object. Many of the other key matters have the same decision-making problems. Improvements can probably be achieved by developing an independent software system dependability theory and examination of the totality of problems by means of system analysis.

Software system quality research can also hold the solution to the dependability problems. On the one hand, quality is characterized by dependability indicators, on the other hand, dependable operation is possible only if the quality of design, implementation and maintenance is high. The researcher should pay a special attention to the human factor, as software systems development is a creative process that is not easily formalized.

In conclusion we should note that this area of research requires an independent theory that is not based on the conventional technology dependability theory. That is due to the special features of software systems that complicate the study of their dependability by means of individual components dependability examination. It should be added that the system analysis methodology involves experts in various areas of knowledge participating in the study and solution of tasks. That is simplified by the publication of papers that, among other things, summarize and systemize the knowledge that promotes further research in this area. Perhaps that would help find the solution to the engineering tasks referred to in [17] on page 4.

References

1. **Peregudov F.I., Tarasenko F.P.** Introduction to system analysis: 3rd edition. Tomsk: Publishing house NTL, 2001. 396 p.

2. **Lipaiev V.V.** Software design: Moscow. Vysshaya Shkola, 1990. 303 p.

3. **Myers G.** Software reliability: translation from English. Moscow. Mir, 1980. 360 p.

4. **Shtrik A.A., Osovetsky L.G., Messikh I.G.** Structured design of reliable embedded computer software. Leningrad: Mashinostroenie. Leningrad branch, 1989. 296 p.

5. **Palchiun B.P., Yusupov R.M.** Software reliability evaluation. Saint-Petersburg: Nauka, 1994. 84 p.

6. **Polonnikov R.I., Nikandrov A.V.** Methods of software reliability indicators evaluation. Saint-Petersburg: Politekhnika, 1992. 78 p.

7. **Shurakov V.V.** Reliability of data processing systems software: 2nd edition, revised and enlarged. Moscow. Finansy i statistika, 1987. 272 p.

8. **Lipaiev V.V.** Software reliability: Moscow. Sinteg, 1998. 232 p.

9. **Thayer T., Lipov M., Nelson E.** Software reliability: translation from English. Moscow. Mir, 1981. 323 p.

10. **Lipaiev V.V.** Reliability and functional safety of real-time software systems. Moscow. ZAO Svetlitsa, 2013. 192 p.

11. **Cherkesov G.N.** Reliability hardware and software systems: Saint-Petersburg: Piter, 2005. 480 p.

12. **Karpovsky E.Ya., Chizhov S.A.** Software products reliability. Kiev: Tekhnika, 1990. 160 p.

13. **Smagin V.A., Dorokhov A.N.** Introduction to the theory of software reliability: Saint-Petersburg: Baltic State Engineering University, 2009. 304 p.

14. **Shubinsky I.B.** Functional reliability of information systems. Analysis methods. Moscow. Dependability Journal, 2012. 296 p.

15. **Abramova N.A.** On some of the myths regarding the quality assessment of software. Dependability, 2004. Issue 1. P. 38–63.

16. **Netes V.A., Tarasyev Yu.I., Shper V.L.** Topical issues of reliability terminology standardization. Dependability. 2014. Issue 2. P. 116 – 119.

17. **Ushakov I.A., Shubinsky I.B.** Pressing problems of engineering dependability (Who are we? Where are we coming from? Where are we going?). Dependability. 2012. Issue 2. P. 3 – 4.