УДК 004.4

INTELLECTUALIZED SYSTEM OF ANALYSIS OF THE QUALITY OF SOFTWARE SYSTEMS

Valerii Krutko[1)] , Iryna Spivak[2)], Svitlana Krepych[3)], Yurii Dzyga[4)]
West Ukrainian National University
[1)]master; [2-3)] associate professor; [4)] phd. student

I. Problem statement

The importance of high-quality software in the success of a product is widely acknowledged. Despite the fact that bugs and mistakes in applications are something we all are quite aware about, some of them can cause severe repercussions, impacting security, reliability, and user satisfaction [1-3]. Hence, ensuring the high quality of a software product is a primary objective to be reached during the development stage. In terms of cost reduction and time efficiency in development, early identification of mistakes is crucial. In iterative development, the mistakes fixing complexity increases over time, elevating the cost, and reducing the probability of being fixed correctly (without introducing new mistakes). It is necessary to develop a model for determining the general indicator of the reliability of the software system, taking into account its complexity [3-6].

II. Purpose

Currently, various reliability models are used to address the issue of evaluating the reliability of software systems. However, these models are mostly adaptations of hardware system reliability models with minimal modifications, which have simplifications and assumptions that significantly limits their applicability to actual software products. This paper aims to suggest a method for assessing software reliability that considers the occurrence of failures rather than mistakes, as well as the complexity and structure of the software product.

III. Main part

A flowchart of the approach to assessing the reliability of software systems based on a graph model of method dependence is shown in figure 1.

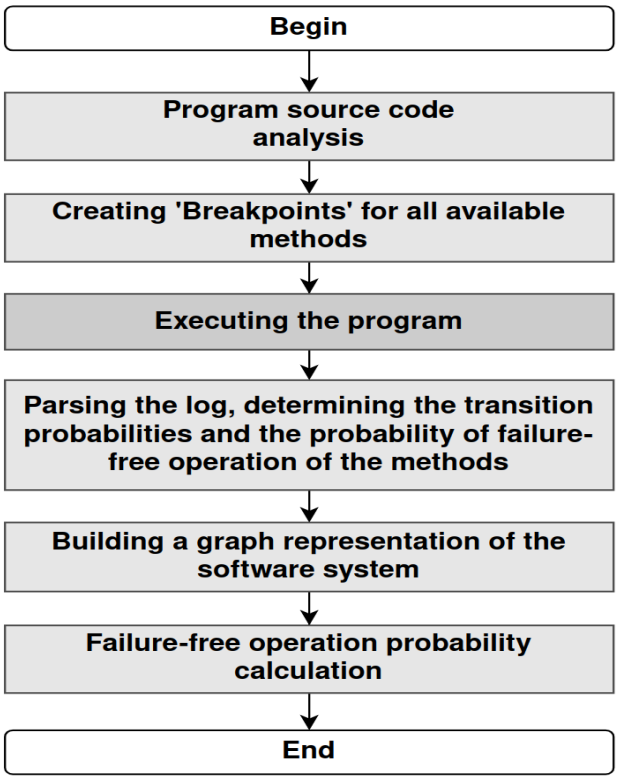

Figure1 – Suggested approach action sequence

Software will be implemented in the form of plug-in for the integrated development environment IntelliJ IDEA from JetBrains. The application itself does not have a program interface, as all functions are implemented using the IDE' context menu. The program code of the plugin is written in the Kotlin programming language, since the API of the IntelliJ IDEA environment is written in this language.

As can be seen from Figure 1, the log is a key element of the analysis because it is used to build a graph model of the software system, as well as to fill in the matrices required for reliability calculation.

Despite the fact that we have access to the method hierarchy via IDE API, this information is not enough to build a graph. This is because this hierarchy of methods does not contain any informatoin about conditional statements and various checks that may be present in the program code. The task of estimating transition probabilities based on code analysis is extremely non-trivial and cannot be solved algorithm. Within the

suggested approach, transition probabilities are calculated statistically as the ratio of the number of child method invocations to the number of invocations of the parent method based on information from log.

It is convenient to represent the structure of the designed software in the form of a use case diagram. Figure 2 shows the capabilities of the system, as well as the actions that the user can perform using the graphical user interface.

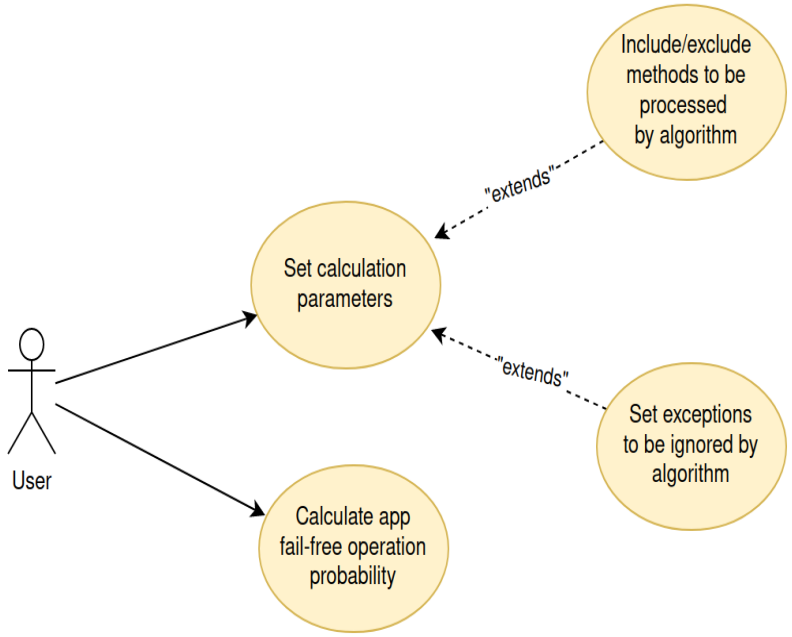

Figure 2 – Use case diagram

An ability to manually include and exclude methods from processing allows to fine-tune which parts of the software will be analyzed. In addition, there are applications that have not one, but many entry points, for example, web applications written using the SpringFramework, where each individual method of the controller is executed asynchronously and may not interact with others, so this approach will allow to perform the analysis of the reliability of individual modules [7,8].

Also, as it shown in Figure 2, plugin can be configured to ignore some exceptions. This allows you to ignore exceptions that are not the result of any mistake, but caused by incorrect input data, temporary unavailability of external services, etc.

Conclusions

This paper investigates an approach to assessing the reliability of software systems based on a graph model of method dependencies. This approach includes elements of structural analysis of application source code, followed by building a map of the relationships between methods, as well as determining stochastic indicators of their reliability. Due to the fact that this approach takes into account the architecture of the program under study and operates with actual reliability indicators of methods, it should provide higher accuracy than systems that predict the reliability of software systems in accordance with various probability distribution functions.

References

1. ISO/IEC 25010:2011, Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models, Standard, International Organization for Standardization, 2011.
2. Крепич С.Я. Моделювання та забезпечення функціональної придатності статичних систем методами аналізу інтервальних даних/ дисертація на здобуття ступ. к.т.н. – Тернопіль, 2016. 166с
3. V. V. Vyshnivskyi, V. V. Vasylenko, M. P. Hnidenko, Osnovy nadiinosti ta diahnostyky informatsiinykh system. Navch.posibn.[Fundamentals of reliability and diagnostics of information systems. Study guide.], volume 188, FOP Huliaieva V. M., 2020.
4. Крепич С.Я. Програмний комплекс оцінювання функціональної придатності пристроїв при заданих допустимих значеннях вихідних характеристик та допусків на параметри їх елементів. Сучасні комп‘ютерні інформаційні технології: Матеріали V Всеукраїнської школи-семінару молодих вчених і студентів АСІТ‘2015. – Тернопіль: ТНЕУ, 2015. – С. 23-5.
5. I. Spivak, S. Krepych, S. Budenchuk. "Methods and means of expert evaluation of software systems on the basis of interval data analysis", in Proceedings of 14th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering, TCSET ’18, LvivSlavske, Ukraine, 2018.
6. Крепич С.Я., Співак І.Я. Оцінювання часової складності застосування методу Монте-Карло та інтервального аналізу даних для встановлення функціональної придатності РЕК. Сучасні комп’ютерні інформаційні технології: Матеріали III Всеукраїнської школи-семінару молодих вчених і студентів АСІТ’2013. – Тернопіль: Економічна думка, 2013. – С.36-37.
7. Стахів П.Г., Дивак М.П., Крепич С.Я. Синтез радіо-електронних кіл при заданизх обмеженях на вихідні характеристики та за умов заданих допусків на параметри елементів. Вимірювальна та обчислювальна техніка в технологічних процсах. Вип.3. 2014, с.39-47
8. I.Spivak, S.Krepych, R.Krepych, A.Bayurskii, "Construction of a criterion for assessing the level of objectivity of experts based on a modified interval expert appraisal method", in IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (PIC S&T),2019, pp.311-314