Selection of Software Testing Method by Using ARAS Method

DARJAN M. KARABAŠEVIĆ, University Business Academy in Novi Sad, Faculty of Applied Management, Economics and Finance, Belgrade
MLAĐAN V. MAKSIMOVIĆ, University Business Academy in Novi Sad Faculty of Applied Management, Economics and Finance, Belgrade
DRAGIŠA M. STANUJKIĆ, University of Belgrade, Technical Faculty in Bor, Bor
GORAN B. JOCIĆ, University Business Academy in Novi Sad, Faculty of Applied Management, Economics and Finance, Belgrade
DUŠAN P. RAJČEVIĆ, University Business Academy in Novi Sad, Faculty of Applied Management, Economics and Finance, Belgrade

Review paper
UDC: 004.4
DOI: 10.5937/tehnika1805724K

The accelerated development of informatics and computing in the last decades has led software to become present in all segments of the society. When the presence and value of software began to grow, software testing has become a very important activity in software development. Software has become indispensable in the economy, education, healthcare, communications, the media, politics, etc. Software errors can cause huge pecuniary damages, as well as non-material losses (reputation, honour ...) so they must be eliminated as early as possible. It is important to recognize the importance of software testing as a basic phase in the software development cycle. Testing helps in reducing the risk of product failure and ensures that the product meets business and technical requirements. Therefore, the main goal of this paper is to present an approach based on multiple-criteria decision-making methods in the selection of software testing method. For the selection of alternatives, in our case software testing methods, Additive Ratio Assessment (ARAS) method is applied. Applicability, usability and efficacy of the proposed approach is demonstrated on conducted illustrative example of selection of software testing method.

Key words: *software, software testing methods, MCDM, ARAS*

1. INTRODUCTION

The accelerated development of informatics and computing in the last decades has led software to become present in all segments of the society. When the presence and value of software began to grow, software testing has become a very important activity in software development. Software has become indispensable in the economy, education, healthcare, communications, the media, politics, etc. Software errors can cause huge pecuniary damages, as well as non-material losses (reputation, honour ...) so they must be eliminated as early as possible. It is important to recognize the importance of software testing as a basic phase in the software development cycle. Testing helps in reducing the risk of product failure and ensures that the product meets business and technical requirements.

Today, there are a large number of software vendors in the world. Each of them is trying to make software without any defects. However, there is practically no software without defects. Errors that occur during operation are normal and everyday thing and are not directly related to software development. Errors occur in all areas of work and are caused by both human and machine factors. System or software errors

Autrhor's address: Darjan Karabašević, University Business Academy in Novi Sad, Faculty of Applied Management, Economics and Finance, Belgrade, Jevrejska 24

e-mail: darjan.karabasevic@mef.edu.rs

Paper received: 17.07.2018.

Paper accepted: 12.09.2018.

can occur when the environment changes (eg. software is running on new hardware), source code, or in interaction with other software. Institute of Electrical and Electronics Engineers (IEEE) provided the following definition of the occurrence of the error: „A human being can make a mistake that becomes a defect in the code, software, system, or document. If the defect in the code is executed, the system will not perform its task or will perform it incorrectly, which will lead to a software error“ [1].

Due to the strong competition in the software market, in order to be competitive, manufacturers must deliver new software at short intervals. This leaves them insufficient time to test software, so mistakes in software are more common. When there is a flaw in the software, the manufacturer must react quickly and do something about it. It has been proven that if a flaw in the software remain undetected for a long time, its removal will cost more. The costs of correcting the error at the analysis stage are incomparably lower than the cost of correcting the same error in later stages of development or delivery [2].

If the error in software is detected at a later stage of development or after delivery, it can be corrected in two ways, either by modifying part of the existing software, or by creating completely new software. If the flaw is caused by poorly designed software, eliminating it by altering part of the code may cause new, possibly bigger problems. It's sometimes more cost-effective to re-develop the entire software than repair an existing one. Bertolino [3] states that „testing is essential activity in software engineering“. Also the testing in „simplest terms, it amounts to observing the execution of software system to validate whether it behaves as intends and identify potential malfunctions“.

Burnstein [4] defines software testing as a process and as an activity: -Software testing is a process that demonstrates that defects do not exist in the software developed for the given application. - Software testing is an activity that ensures the necessary level of confidence in the fact that the program or system executes what it was supposed to do, based on a set of requests that the user specified.

Lazić [5] states that software testing consists of dynamic program verification activities based on the final set of tests, selected in a convenient way from an infinite set of possible ways to execute a program, and according to the specified expected behaviour of the software in a developed application, i.e. required software quality. The problem of software testing is also complex because of the astronomically large number of scenarios of using the software product and the conditions in which the program is found during use.

Based on the above stated, the aim of the paper is to present an approach based on multiple-criteria decision-making methods in the selection of software testing method.

For the selection of alternatives, in our case software testing methods, Additive Ratio Assessment (ARAS) method is applied. Applicability, usability and efficacy of the proposed approach is demonstrated on conducted illustrative example of selection of software testing method.

Therefore, the rest of the paper is structured as follows: in Section 2 some basic definitions related to software testing methods and multiple-criteria decision-making are given. In section 3 Additive Ratio Assessment (ARAS) method is presented, whereas in Section 4 an illustrative example is conducted. Finally, conclusions are presented in final section.

2. THE THEORETICAL BACKGROUND

Software Testing Methods are ways that the software testing team will test the system. For example, it is necessary to define whether software it will be tested manually or will write a script that automatically tests the system, whether only new functionalities will be tested, or tests on all previously performed functionalities will be repeated. The test method is a test strategy that tells the test team how to test it.

Depending on whether the software testing is performed only on the basis of the output or the internal structure of the software is included, the tests are classified by IEEE as [6]:

- Black Box (functional) testing. Reveals errors based on results (exits) when executing the program. Black box testing neglects the internal path of processing within the software code;
- White Box (structural) design testing. Examines the code structure in order to identify errors. The term „White“ is given to highlight the difference between this method and Black Box testing.

In many cases, both concepts are used to test complex software solutions. Sometimes the so-called Gray Box testing is applied, some modules are tested with Black box, and others with the White Box method.

In addition to the above two methods, Boštjančič et al. [7] additionally suggest that the software testing methodology should be expanded by adding the following phases:

- Debugging. Debugging involves the removal of syntax and logical errors during application development.
- Testing Correctness. Testing Correctness includes Black Box testing and White Box testing.

- Performance testing. Performance testing includes finding and troubleshooting problems that degrade software performance.
- Reliability testing. Reliability testing determines the probability of functioning of the software without error and without failure.
- Security testing. The purpose of security testing is to identify and correct errors that potentially jeopardize security, and validate the effectiveness of protection measures.

By testing the software, it is first determined how much the software performs the job for which it is intended, and then how it behaves in different exploitation conditions.

Basically, all the definitions of program testing and software testing methods are the tendency to answer the question: does the program behave in the way it is required? Therefore, the key concept in software testing is its specification, because by definitions the above checks are relying on it specification.

Karabasevic et al. [8] emphasizes that "when making decisions, decision makers are often faced with the problem of selecting the optimal alternative from a set of available alternatives, sometimes with the presence of possible limitations.

Relevant approach to making decisions and the adoption of sustainable solutions is provided by the Multiple-criteria Decision Making Methods (MCDM). Also, Stanujkic et al. [9] describe multiple-criteria decision making as „he process of selecting one from a set of available alternatives, or ranking alternatives, based on a set of criteria of usually different significance".

Therefore, methods of multi-criteria decision-making are applied to problems when it is necessary to make the decision to choose one of several potential solutions for a problem. This implies the process of choosing one of the more feasible alternative solutions for which certain goals are set.

By the time and having in mind the extremely dynamic development of the area of multi-criteria decision making, many MCDM methods are proposed. Karabasevic et al. [10] especially highlights some prominent methods such as: SAW or the WS; AHP, TODIM; TOPSIS; PROMETHEE; COPRAS and ELECTRE, as well as some newly-developed methods such as: ARAS; MULTIMOORA; SWARA; WASPAS; EDAS and so on.

Therefore, the extremely dynamic development of the field of operational research, i.e. field of multi-criteria decision-making methods, shows that the application of multi-criteria decision-making methods is a good choice in solving problems and adopting sustainable solutions.

3. METHODOLOGY

Additive Ratio ASsessment (ARAS) method is developed by Zavadskas and Turskis [11]. Therefore for the ARAS method can be said, although is newly-developed method, that ARAS method is very effective and easy to use when it comes to solving MCDM problems. Until now ARAS method has been applied in numerous cases to solve various decision-making problems, such as: subcontractor selection [12], personnel selection [13-15], ranking of companies according to the indicators of corporate social responsibility [16], oil and gas well drilling projects evaluation [17], sustainable building assessment/certification [18] and so on.

Problem solving procedure using the ARAS method, similar to other MCDM methods, begins by forming the decision matrix and determining the weights of the criteria.

A further procedure to solve the problems of multi-criteria decision-making by applying ARAS method can be accurately expressed using the following steps [14-15]:

Step 1. Determining optimal performance for each criterion. In this step, the domain expert/decision maker determines the optimal performance rating for each criterion. If the domain expert/decision maker has no preferences, the optimal performance ratings can be determined as follows:

$$x_{0j} = \max_i x_{ij} , \quad (1)$$

where x_{0j} is the optimal performance rating in relation to the *j*-th criterion.

Step 2. Calculate a normalized decision matrix $R = [r_{ij}]$. Normalized performance ratings are calculated by using the following formula:

$$r_{ij} = \frac{x_{ij}}{\sum_{i=0}^{m} x_{ij}} , \quad (2)$$

where r_{ij} is the normalized performance rating of the *i*-th alternative in relation to the *j*-th criterion.

Step 3: Calculate a weighted normalized decision matrix $V = [v_{ij}]$. The weighted normalized performance ratings are calculated by using the following formula:

$$v_{ij} = w_j \cdot r_{ij} , \quad (3)$$

where v_{ij} is the weighted normalized performance rating of the *i*-th alternative in relation to the *j*-th criterion.

Step 4. Calculate the overall performance index for each alternative. The overall performance index S_i for each alternative can be calculated as the sum of the weighted normalized performance ratings using the following formula:

$$S_i = \sum_{j=1}^{n} v_{ij} \quad . \tag{4}$$

Step 5. Calculate the degree of utility for each alternative. Degree of utility can be calculated by applying the following formula:

$$Q_i = \frac{S_i}{S_0} \quad , \tag{5}$$

where Q_i is the degree of the utility of the i-th alternative, and S_0 is the overall performance index of the optimal alternative, which is usually 1.

Step 6. Rank the alternatives and/or select the most efficient one. The considered alternatives are ranked by ascending Q_i, i.e. the alternatives with the higher values of Q_i have a higher priority (rank) and the alternative with the largest value of Q_i is the best-placed one.

4. AN ILLUSTRATIVE EXAMPLE

In order to present applicability, usability and efficacy of the proposed approach, in this section an illustrative example is shown. Three domain experts have evaluated the four alternatives i.e. testing methods: A_1 - automated unit testing – these kind of testing requires minimal human intervention [19; 20]; A_2 - incremental testing top-down - strategy where the top components of the application are tested first [19; 21]; A_3 - incremental testing bottom-up - strategy where the terminal-bottom components of the application are tested first [19; 21] and A_4 - usability test – testing and evaluation of the interfaces with real users [19; 21].

For the evaluation of the alternatives total of 6 criteria were used: C_1 - data-flow [22]; C_2 control-flow [22]; C_3 – effectiveness; C_4 - implementation evaluation; C_5 – single person management – it can be tested by single person and C_6 – overall opinion. Attributed weights for all of the evaluation criteria is the same i.e. same importance (0.16).

The overall ratings of the evaluated alternatives are determined as a geometric mean of the ratings obtained from the domain experts/decision makers.

Table 1 shows the average ratings of the evaluated alternatives, also in Table 1 are shown optimal performance ratings in row A_0 obtained by applying Eq. (1).

Table 1. The average ratings of the alternatives obtained by 3 domain experts/decision-makers

	C_1	C_2	C_3	C_4	C_5	C_6
A_0	5.00	4.22	4.31	4.00	4.64	4.64
A_1	3.30	3.00	3.30	2.71	3.00	3.17
A_2	2.52	4.22	3.91	3.63	3.30	2.62
A_3	3.63	3.63	4.31	3.63	3.30	4.31
A_4	5.00	4.00	4.31	4.00	4.64	4.64

Table 2 shows the normalized ratings, determined by applying Eq. (2), and weights of the criteria.

Table 2. The normalized decision-making matrix

	C_1	C_2	C_3	C_4	C_5	C_6
w_i	0.16	0.16	0.16	0.16	0.16	0.16
A_0	0.17	0.16	0.16	0.15	0.16	0.16
A_1	0.13	0.22	0.19	0.20	0.17	0.14
A_2	0.19	0.19	0.21	0.20	0.17	0.22
A_3	0.26	0.21	0.21	0.22	0.25	0.24
A_4	0.17	0.16	0.16	0.15	0.16	0.16

Table 3 shows the overall performance of the evaluated alternatives obtained by using Eq. (4) and (5).

Table 3. The overall results of the ranked alternatives

	S_i	Q_i	Rank
A_0	0.23		
A_1	0.16	0.69	4
A_2	0.18	0.76	3
A_3	0.20	0.85	2
A_4	0.23	0.99	1

According to Table 3, the alternative A_4 has the highest overall importance and therefore the best results in terms of the evaluated criteria.

5. CONSLUSION

Software testing is a very important process and activity, it can be said that software testing is one activity in which huge resources are invested, which indicates that this is one of the most important activities during his life cycle. Software testing is carried out both during development, initial realization and later in exploitation and maintenance phases. Testing methods and, above all, the selection of an adequate method is significant, especially from the aspect of the final product and outcome. In this paper, an illustrative example was successfully conducted aimed at indicating that multi-criteria decision making can be successfully applied for the selection of methods for software testing. An approach based on the ARAS method proved to be very easy to apply, also, should not be ignored the overall efficiency and the usability of the proposed approach. Based on the stances of 3 domain experts /

decision makers, the alternative designated as A_4 is best rated in terms of evaluated criteria. The proposed approach can easily be extended with additional criteria if necessary, also, other alternatives may be considered if necessary. Recommendations for future research can be focused primarily on the application of other multi-criteria decision-making methods, such as EDAS, CODAS and so on. Also, instead of the same weights of evaluation criteria, for determining the weights of the criteria could be applied methods such as PIPRECIA, SWARA, R-SWARA, KEMIRA and so on.

REFERENCES

[1] Galin D. Software quality assurance: from theory to implementation, Pearson Education, 2004.

[2] Zhivich M, and Cunningham, R. K. The real cost of software errors. *IEEE Security and Privacy*, Vol., 7., No. 2., pp. 87-90, 2009.

[3] Bertolino A. Software testing research: Achievements, challenges, dreams, In 2007 Future of Software Engineering, *IEEE Computer Society*, pp. 85-103, 2007.

[4] Burnstein I. *Practical software testing: a process-oriented approach*, Springer Science & Business Media, pp. 61-95, 2006.

[5] Lazić, L. Testiranje softvera kao mera zaštite korisnika softvera, Naučno stručno savetovanje: Zloupotreba informacionih tehnologija i zaštita, Ziteh.p. 337, 2010.

[6] Henard C, Papadakis M, Harman M, Jia Y, & Le Traon Y. Comparing white-box and black-box test prioritization, *In Software Engineering (ICSE), 2016 IEEE/ACM 38th International Conference* on, pp. 523-534., 2016.

[7] Boštjančič S, Stojanović M, and Nedeljković R. Metodologija testiranja softverskog proizvoda, *Zbornik radova TELFOR*, pp. 577-580., 2007.

[8] Karabašević D, Stanujkić D, Đorđević B, and Stanujkić A. The weighted sum preferred levels of performances approach to solving problems in human resources management, *Serbian Journal of Management*, Vol.13, No.1, pp. 145-156, 2018.

[9] Stanujkić D, Đorđević B, and Đorđević M. Comparative analysis of some prominent MCDM methods: A case of ranking Serbian banks, *Serbian Journal of Management*, Vol.8. No.2, pp. 213-241., 2013.

[10] Karabasevic D, Zavadskas E. K, Stanujkic D, Popovic G, and Brzakovic M. An approach to personnel selection in the IT industry based on the EDAS method, *Transformations in Business and Economcis*, Vol.17. No. 2, pp. 54-65, 2018.

[11] Zavadskas E. K, and Turskis Z. A new additive ratio assessment (ARAS) method in multicriteria decision-making, *Technological and Economic Development of Economy*, Vol.16. No. 2, pp. 159-172., 2010.

[12] Koçak S, Kazaz A, and Ulubeyli S. Subcontractor selection with additive ratio assessment method, *Journal of Construction Engineering*, Vol.1. No.1, pp. 18-32, 2018.

[13] Karabasevic D, Zavadskas E. K, Turskis Z, and Stanujkic D. The framework for the selection of personnel based on the SWARA and ARAS methods under uncertainties, *Informatica*, Vol. 27. No.1, pp. 49-65, 2016.

[14] Karabasevic D, Stanujkić D, and Urošević S. The MCDM Model for Personnel Selection Based on SWARA and ARAS Methods, *Management*, Vol. 77, pp. 43-52, 2015.

[15] Stanujkic, D, Djordjevic B, and Karabasevic D. Selection of candidates in the process of recruitment and selection of personnel based on the SWARA and ARAS methods, *Quaestus - Multidisciplinary Research Journal*, Vol. 7, pp. 53-64, 2015.

[16] Karabasevic D, Paunkovic J, and Stanujkic D. Ranking of companies according to the indicators of corporate social responsibility based on SWARA and ARAS methods, *Serbian journal of Management*, Vol. 11. No.1, pp. 43-53, 2016.

[17] Dahooie J. H, Zavadskas E. K, Abolhasani M, Vanaki A, and Turskis Z. A Novel Approach for Evaluation of Projects Using an Interval–Valued Fuzzy Additive Ratio Assessment (ARAS) Method: A Case Study of Oil and Gas Well Drilling Projects, *Symmetry*, Vol.10. No.2, pp. 45. 2018.

[18] Medineckiene M, Zavadskas E. K, Björk F, and Turskis Z. Multi-criteria decision-making system for sustainable building assessment/certification, *Archives of Civil and Mechanical Engineering*, Vol.15. No.1, pp. 11-18, 2015.

[19] Hernández-Ledesma G, Ramos E. G, Fernández-y-Fernández C. A, Aguilar-Cisneros J. A, Rosas-Sumano J. J, and Morales-Ignacio L. A. Selection of Best Software Engineering Practices: A Multi-Criteria Decision Making Approach, *Research in Computing Science*, Vol. 136, pp. 47-60, 2017.

[20] Beck K, and Gamma E. *Extreme programming explained: embrace change*, Addison-wesley professional, 2000.

[21] Myers G. J, Sandler C, Badgett T, and Thomas T. M. *The Art of Software Testing, Business Data Processing*: a Wiley Series, 2004.

[22]Vilkomir S. A, and Bowen J. P. Formalization of software testing criteria using the Z notation, *In Computer Software and Applications Conference, 2001. COMPSAC 2001. 25th Annual International*, pp. 351-356, IEEE, 2001.

REZIME

IZBOR METODA TESTIRANJA SOFTVERA PRIMENOM ARAS METODE

Ubrzani razvoj informatike i računarstva u poslednoj dekadi, doveo je do toga da softver postaje prisutan u svim segmentima društva. Kako je prisutnost i vrednost softvera počela da raste, testiranje softvera postaje izuzetno značajna aktivnost u oblasti razvoja softvera. Softver je postao neophodan u ekonomiji, obrazovanju, zdravstvu, komunikacijama, medijima, politici itd. Greške na softveru mogu da prouzrokuju ogromnu materijalnu štetu, kao i nematerijalne gubitke (reputacija, čast…), tako da se iste moraju otkloniti što pre. Važno je prepoznati važnost testiranja softvera kao osnovne faze u ciklusu razvoja softvera. Testiranje pomaže u smanjenju rizika od neuspeha proizvoda i osigurava da proizvod ispunjava sve poslovne i tehničke zahteve. Stoga, glavni cilj ovog rada je da predstavi pristup zasnovan na metodama višekriterijumskog odlučivanja u izboru metoda za testiranje softvera. Za izbor alternativa, u našem slučaju metoda testiranja softvera, ARAS metoda je primenjena. Primenljivost, upotrebljivost i efikasnost predloženog pristupa je demonstrirana u sprovedenom ilustrativnom primeru izbora metoda testiranja softvera.

Ključne reči: *Softver, metode testiranja softvera, VKO, ARAS*