Program Tools for Estimation of the Green Software Reliability

Dmitry Maevsky*, *Elena* Maevskaya, *Ludmila* Shapa and *Dmitry* Stetsyuk

Odessa National Polytechnic University, 65044 Boulevard of Shevchenko, 1, Ukraine

Abstract. The paper deals with a review of the current software of estimation of the green software reliability. The paper demonstrates that any software is capable of becoming a “green” one if it is at the high reliability level, which has to be continuously measured and controlled. In order to implement these actions, the paper offers the results of reviewing the most frequently used software performing the reliability estimation. On the basis of the analysis of this software characteristics a taxonomic system of classification features describing its (software) functionality and the ease of presentation of the estimation results with the help of this software is offered. By using the offered taxonomy a system of classification of software reliability estimation tools was created. The conclusions of the current trends of research in the area of reliability estimation and green software development were made.

1 Introduction

At present any software is considered to be a green one if the electric energy is saved in using this type of software. It is really so. However as to the critical-purpose software systems the saving factor is just one (and not the most important) of the requirements of greenness. On the first place in the critical-purpose software systems there is the factor of reliability and resilience. Indeed, the defects containing in a software can destabilize the performance of many critical infrastructures such as nuclear power, transport, communications, health service. Herein the material and energy resources required to reconstruct the mentioned critical infrastructures or to eliminate the consequences of accidents occurring at them can be many times greater than the cost of electrical energy, which is saved in the process of their usage.

So, the provision of stated indicators of Green Software reliability is of vital importance for creating the qualitative and safe software systems. At present there exist about sixty various software reliability grows models (SRGM) to calculate these values. To obtain the numerical values of the reliability indicators these models uses time series of data of the identified defects. The indicators themselves are found by doing the complicated math and using the nonlinear programming unit. The manual calculation of these coefficients takes a lot of time, and this fact became an impulsive cause for creation of a large number (more than twenty) of program tools for software reliability estimation. Each of these software tools as a rule permits for estimating the reliability according to not one but a significant number of SPMs. The availability of the variety of program tools makes some difficulties for a developer in choosing the best and most suitable SRGM of the amount of implemented in it (program tool). The complexity of the task is aggravated by the two factors. Firstly, as a rule the software reliability theory is unknown to developers. Secondly, there is no single all-purpose SRGM, which accurately determines the reliability indicators in all cases. In choosing the model the indicators classification system offered for the first time in [1] and described in detail in monograph [2] can be useful for developers. However, the classification of software reliability estimation program tools is absent at the moment. That is why its development is timely and topical.

2 The review of reliability estimation program tools

The review of program tools found in the available literary sources containing the information, the amount of which allows to understand their functionality and user interface, is given below.

SRMP (Software Reliability Modeling Programs) [3] – a program tool for reliability estimation created by Bev Littlewood in 1988. It includes the following reliability models: Goel-Okumoto, Musa-Okumoto, Duane as well as Littlewood-Werall quadratic and linear models. SRMP presents the results of estimation in the form of tables and graphs. The estimation accuracy indicators are formed in a separate way. SRMP accepts only time between failure as input data. The results provided in graphic form with some statistics.

SMERF [3, 4] (Statistical Modelling and Estimation of Reliability Functions for Software) – a tool developed by William Farr in 1988. The reliability estimation according to such SRMs as Littlewood-Werall quadratic and linear models, Musa’s basic and logarithmic models, geometric one, Jelinsky-Moranda, Yamada, Schneidewind, Brooks and Motley’s binomial and

* Corresponding author: Dmitry.A.Maevsky@gmail.com

Poisson's models, generalized Poisson model is implemented in it.

RGA (Reliability Growth Analysis and Repairable System Analysis)– a program tool created in 1992 by ReliaSoft Corporation and Dr. Larry Crow. According to [5], RGA includes full support for traditional reliability growth analysis using the applicable models – Crow-AMSAA (NHPP), Duane, Gompertz, Modified Gompertz, Lloyd-Lipow or Logistic – for a variety of developmental data types – time-to-failure, discrete (success/failure) and reliability data. As described in [5], RGA offers all of the major reliability growth models, plus advanced analysis methods that are not available in another program tools. RGA also provides opportunities for fielded repairable system analysis, including a Design of Reliability Test utility and a method for analyzing the system's reliability behaviour over time in order to calculate optimum overhaul times and other metrics of interest. The latest version of the program tool was published in March 2016.

CASRE (Computer-Aided Software Reliability Estimation Tool) [3, 6] – a tool, which was implemented in 1993 by Allen P. Nikora. It includes the following reliability models: Brooks-Motley binominal model, Schneidewind, Jelinsky-Moranda, Littlewood-Verrall, Musa-Okumoto, basic and logarithmic Musa models. CASRE provides the developed automated support of software reliability modelling. The main advantage of CASRE is its various reliability estimations under a paradigm which linearly combines the component models. This tool features an enhanced graphical user-interface which greatly facilitates the process of software reliability estimation.

ROBUST (Reliability of Basic and Ultra-reliable Software sysTems) [7, 8] – a program tool created by Y. K. Malaiya, Jason Denton and Naixin Li in 1995. It supports four SRGMs: Muas, Musa-Okumoto, Duane and S-shaped. It gives the possibility to take into account the metrics containing the information of a program, crew of developers, etc. ROBUST provides a wide range of support for traditional software reliability growth models. It is capable of working with failure data in either a cumulative time to failure or failure interval format.

CARATS (Computer-Aided Reliability Assessment Tool for Software Based on Object-Oriented Design) [9] – a tool created by Chien-Chia Chen, Chu-Ti Lin, Hen-Hsen Huang and Shih-Wei Huang. The following SRGMs are implemented: Yamada, Musa-Okumoto, Goel-Okumoto, Duane. There is a chance to compare SRGMs with the help of maximum likelihood method and the one of least deviation. CARATS can use both traditional SRGMs and neural-network methods to assess software reliability.

Refis (A New Statistical Software Reliability Tool) [10] – a program tool developed by M. A. A. Boon, E. Brandt, Corro Ramos, A. Di Bucchianico and R. Henzen in 2006. It supports the following SRGMs: Littlewood-Verall, geometrical one, Musa, Jelinsky-Moranda, Muas-Okumoto, Goel-Okumoto, Schick and Wolverton, Schneidewind, S-shaped, Duane.

SRATS (Software Reliability Assessment Tool on Spreadsheet) [11] – a tool, which was created in 2005. SRATS functions as a Microsoft Excel add-in, that is why it cannot be used as an independent tool. The latest version came into existence in 2012.

SafeMan [12] – a program tool developed by Takaji Fujiwara and Mitsuhiro Kimura in 2014. Only two RSGMs have been implemented – Musa model and S-shaped model. This tool has function of automatic selection of the optimum SRGM.

3 Classification features formation

The presented classification features aimed at the software reliability estimation by program tools are to reflect two factors, which are important for users: the ease of a program tool usage and its functionality. The first factor is stipulated by the interface characteristics of a program tool, and the second – by mathematical apparatus implemented in it.

The authors offer to attribute the method of source data statement and the one of result presentation to the interface features. Let us consider them in detail.

The ease (and frequently the use) is identified with the help of a method of source data statement. E.g. if a user has got the cumulative series of distribution of the failures detected in the course of time, and a program tool is able only to fix the moments of their detection at the input then such a program tool cannot be used.

That is why the author offer to use a type of source data, which is appropriate for this particular program tool, as the **first classification feature**. We distinguish such types of source data presentation:

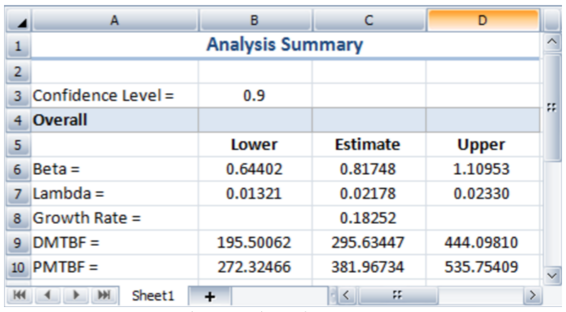
– TF (Time of Failure) – the astronomical time of the failure emergence;

– TBF (Time Between Failures) –time interval between two successive failures;

– CF (Cumulative Failure) –the cumulative amount of failures;

– FI (Failures experienced in a time Interval) – a number of failures detected at the predetermined time interval.

The second feature of Software reliability estimation tools classification is a method of results presentation of their run. The method of presentation is of great importance since it is the method itself that determines the ease for a user, who deal with a Software tool. The coefficients of applied reliability models do not provide a user with any useful information. On the basis of models coefficients, the main reliability indicators - failure rate, MTBF and hazard function - can be only identified. These indicators provide the users, who are not informed enough in the theory of Software reliability, with little data as well. The users of the Software reliability estimation tools are to be either the specialists in the program debug area or the ones in the Software engineering field. For them the results of run of Software reliability estimation tools should be presented in the familiar terms "a number of failures at a time" and "the law of changes of the number of failures over time". As the analysis carried out by the authors has shown the majority of the current Software reliability estimation tools provides the user with the two mentioned types of results – the main indicators of reliability and the ones of the number of failures (Fig. 1).



	A	B	C	D
1	Analysis Summary			
2				
3	Confidence Level =	0.9		
4	**Overall**			
5		Lower	Estimate	Upper
6	Beta =	0.64402	0.81748	1.10953
7	Lambda =	0.01321	0.02178	0.02330
8	Growth Rate =		0.18252	
9	DMTBF =	195.50062	295.63447	444.09810
10	PMTBF =	272.32466	381.96734	535.75409

Sheet1

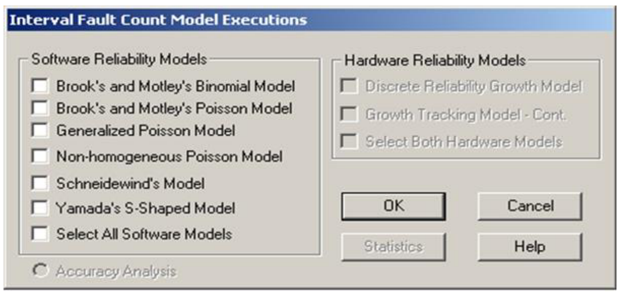
Fig. 1. The results of program RGA run

There are two types of presentation of the indicators of the number of failures. The first type is a table in which the times and the number of failures in Software over times are given. The second type are the graphs displaying the data of the table. The graphic presentation of results is more demonstrative but the tabular form is more functional. Using the table data one can accurately identify the number of failures in software and evaluate the rate of its changes. We distinguish three features demonstrating the presentation of the results of Software tools run:

– R –the results are the indicators of reliability;
– T –the results are presented in a tabular form;
– G –the results are given in a graphical form.

In each case the simultaneous usage of several features is possible.

The third classification feature displays the functionality of Software reliability estimation tools. At present the functionality is indicated on the basis of the SRGM list implemented in this model. However, a mere listing of models cannot be informative for a user in principle. The users of Software reliability estimation tools are not specialized in the theory of Software reliability. They do not know the peculiarities of existing models and the rules of their usage. The ideal case is when a user does not generally suspect about the existence of the models. In this case the model itself is not of any importance for him (her) to obtain some result. The only thing a user need is to obtain a reliable result. However, all the considered Software reliability estimation tools are implied to be utilized by a user for the manual choice of one or several models from the list (Fig. 2).



Interval Fault Count Model Executions

Software Reliability Models
☐ Brook's and Motley's Binomial Model
☐ Brook's and Motley's Poisson Model
☐ Generalized Poisson Model
☐ Non-homogeneous Poisson Model
☐ Schneidewind's Model
☐ Yamada's S-Shaped Model
☐ Select All Software Models

Hardware Reliability Models
☐ Discrete Reliability Growth Model
☐ Growth Tracking Model - Cont.
☐ Select Both Hardware Models

Fig. 2. The window of choice of the reliability model in program SMERFS

Taking into account the fact that the different set of models are implemented in different Software reliability estimation tools the choice of the necessary tool is the impossible task for an unexperienced user today. This task can be carried out only with the help of a specialist in this field or at random.

In the given paper the authors offer to change the approach to the choice of Software reliability estimation tool. The users are not informed in reliability models area but they know the peculiarities of their own Software, the conditions of its development and testing. However, each of the reliability models is created taking account the definite assumptions, which are generally determined by testing conditions. E.g. in the majority of models an assumption is taken that in correcting a failure the new failures are not included in Software. That is why the choice of a particular reliability model in every particular case is made on the basis of comparison of assumptions of a model and the conditions of testing. On the other hand, a set of models implemented in each Software reliability estimation tool, creates a set of assumptions, with the help of which it meets this Software tool. So it becomes possible to choose a Software tool on the basis of comparing the testing conditions and a set of assumptions.

In the fullest form the assumptions of the majority of RSGMs have been analysed and generalized in the paper [13]. In this paper the authors offer a matrix of RSGM assumptions in the lines of which all models assumptions are grouped and in the columns – the models, which meet these assumptions. All assumptions are united in the four groups:

– I –take into account the way of detecting and correcting the failures;

– II –display the models classification according to [1];

– III –take into account the way of distributing the detected failures in time;

– IV –the assumptions, which are inherent only to some individual models.

Each of the groups in its turn is divided into subgroups. The complete list of all subgroups with the detailed explanations one can find in the above mentioned paper. The matrix of matches between assumptions and reliability models is presented ibidem.

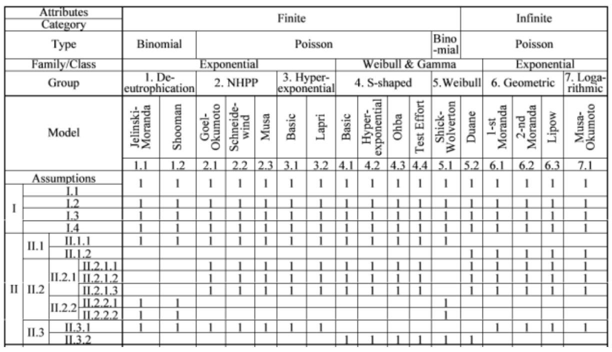
Based on all said above as the third functional feature of Software reliability estimation tools the authors offer to use the assumptions of models implemented in them (tools) according to the matrix of assumptions.

The authors of article [13] propose the following sequence of steps for software reliability evaluation using assumption matrix:

1. Analysis of software requirements and software engineering process.
2. Obtaining of the information about assumptions usable to the developed software.
3. SRGM choice using assumption matrix.
4. Determination of SRGM input parameters and analysis of their complexing opportunities.
5. SRGM parameters fitting.

6. Calculation of software reliability measures and statistical analysis of obtained data.

Fig. 3 shows a fragment of the assumptions matrix (according to [13], Table 1).



Attributes / Category			Finite												Infinite				
Type			Binomial		Poisson									Bino -mial	Poisson				
Family/Class			Exponential							Weibull & Gamma						Exponential			
Group			1. De-eutrophication		2. NHPP			3. Hyper-exponential		4. S-shaped				5.Weibull		6. Geometric			7. Loga-rithmic
Model			Jelinski-Moranda	Shooman	Goel-Okumoto	Schneide-wind	Musa	Basic	Lapri	Basic	Hyper-exponential	Ohba	Test Effort	Shick-Wolverton	Duane	1-st Moranda	2-nd Moranda	Lipow	Musa-Okumoto
			1.1	1.2	2.1	2.2	2.3	3.1	3.2	4.1	4.2	4.3	4.4	5.1	5.2	6.1	6.2	6.3	7.1
Assumptions																			
I		I.1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
		I.2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
		I.3	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
		I.4	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
II	II.1	II.1.1	1	1	1	1	1	1	1	1	1	1	1	1					
		II.1.2													1	1	1	1	1
	II.2	II.2.1 II.2.1.1			1	1	1	1	1	1	1	1	1		1	1	1	1	1
		II.2.1.2			1	1	1	1	1	1	1	1	1		1	1	1	1	1
		II.2.1.3			1	1	1	1	1	1	1	1	1		1	1	1	1	1
		II.2.2 II.2.2.1	1	1										1					
		II.2.2.2	1	1										1					
	II.3	II.3.1	1	1	1	1	1	1	1							1	1	1	1
		II.3.2								1	1	1	1	1	1				

Fig. 3. Fragment of Assumption Matrix

The forth classification feature, which also indicates a Software tool functionality, is to perform the prediction of reliability indicators changes in the subsequent periods of time. Broadly speaking the reliability indicators prediction is not a complicated mathematical problem. The point is that a reliability model is finally a mathematical formula determining the law of changes of the number of failures over time.

The formula contains the coefficients identified on the basis of the time series of failure detection. That is why one should indicate the model coefficients only once and then, substituting different time values into the formula, identify the predicted indicators of reliability. However, taking into account the fact that a final user is badly (or not at all) informed in the reliability models area a Software tool is to take the task of calculation of the predicted indicators. We are able to distinguish two values of the feature:

– E – a Software tool performs only the reliability estimation;

– P – a Software tool performs both the estimation and prediction of reliability.

We can present the classification features (taxons) system graphically (Fig. 4).

The group of interface features determines two aspects in the system. First, it can be called a traditional ease when using a Software tool.

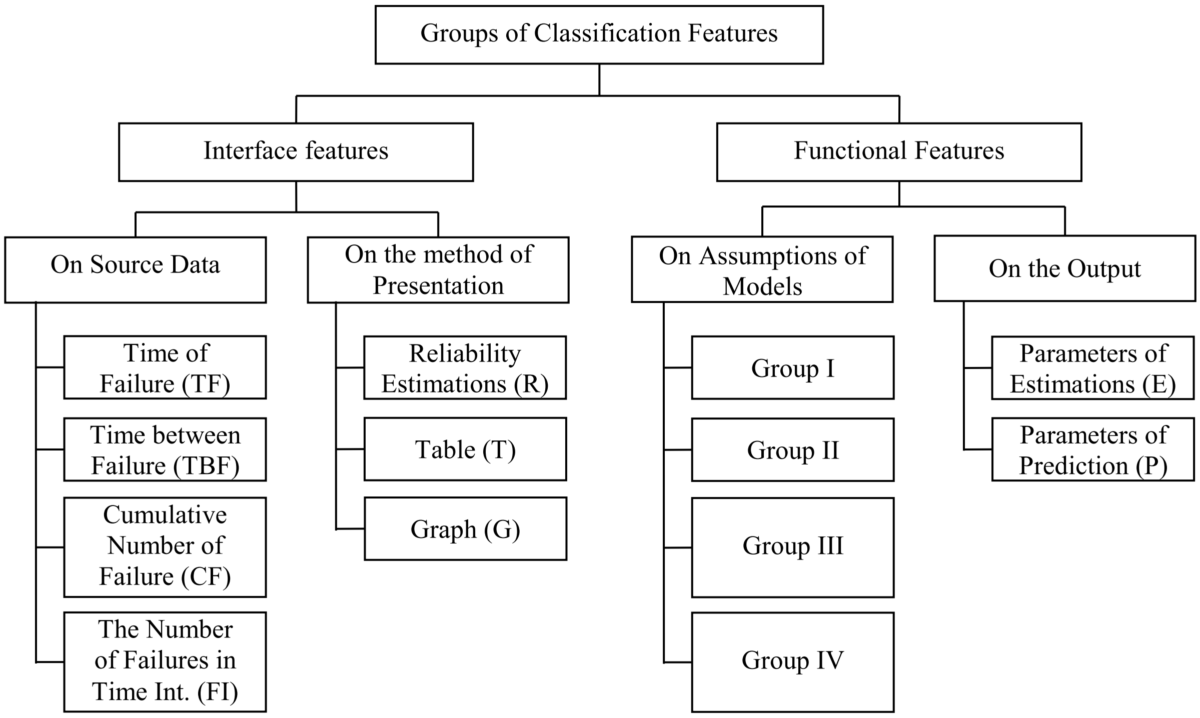

Groups of Classification Features
Interface features
Functional Features
On Source Data
On the method of Presentation
On Assumptions of Models
On the Output
Time of Failure (TF)
Time between Failure (TBF)
Cumulative Number of Failure (CF)
The Number of Failures in Time Int. (FI)
Reliability Estimations (R)
Table (T)
Graph (G)
Group I
Group II
Group III
Group IV
Parameters of Estimations (E)
Parameters of Prediction (P)

Fig. 4. Classification Features of Software Reliability Estimations Tools

However, the way of setting the source data, which can be also attributed to the source data, which can be also attributed to the interface features, determines finally a reliability model used That is why we can say that in the developed system of classification the interface features determine one more – methodological aspect of usage of Software reliability estimation tools.

The group of functional features identifies the potential of a Software tool to estimate the reliability. The usage of simple assumptions of models understandable for a user allows to significantly ease the choice of a suitable Software tool of all the diversity of the current Software tools.

On the basis of the offered classification features the classification of Software reliability estimation tools described above has been carried out. The classification is given in table 1 at the previous page. The distinguished features have been presented in the columns of the table according to the hierarchy levels shown in Fig.4.

Table 1. Classification of Software Reliability Estimation Tools

Software Tool	Interface Features						Functional Features					
	Source Data				Results		Assumptions of Models				Prediction	
	TF	TBF	C	FI	G	T	I	II	III	IV	E	P
SRMP		v			v	v	I	1, 2.1, 3	2, 5	2.1, 5.2	v	
SMERFS	v	v			v	v	I	II	1, 2, 4, 5, 6, 7	1.1, 2.1, 2.3, 4.1, 5.2, 7.1	v	v
RGA			v	v	v	v	I	1.2, 2.1, 3.2	5	5.2	v	
CASRE	v	v	v	v	v	v	I	1, 2, 3.1	1, 2, 7	1.1, 2.1, 7.1	v	v
ROBUST	v	v					I	1, 2.2, 3	2, 4, 5, 7	2.3, 4.1, 5.2, 7.1	v	v
CARATS	v	v			v		I	1, 2.2, 3	2, 4, 5, 7	2.1, 4.1, 5.2, 7.1	v	v
Refis		v	v		v	v	I	II	1, 2, 4, 5, 6, 7	1.1, 2, 4.1, 5, 6.1, 7.1	v	
SRATS			v	v	v	v	I	1, 2.1, 3	2, 4, 7	2.1, 4.1, 7.1	v	
SafeMan	v		v		v	v	I	1.1, 2.2, 3	2, 4	2.3, 4.1	v	

Conclusion

The paper offers an hierarchic group of classification features – taxons, which represent the Software reliability estimation tools and their main characteristics in the most complete and correct way. Both interface and functional (performance) characteristics of such kind of Software tools have been clearly highlighted in the system. The usage of the developed taxonometric features allows to create a classification of the current Software tools. The developed classification system is able to facilitate a user's efforts in choosing a software reliability estimation tool necessary for him (her) among the variety of tools. The advantage of the classification system offered is as follows: it can be employed by users, who are not specialists in the field of Software reliability theory. An ordinary user does not know and he (she) must not, which reliability models are realized in this or that software tool, and understand the peculiarities of these models operation. He (she) does not need to know such "terrible" specific terms as "Вейбулл's distribution", «Software Reliability Growth Models», «Maximum Likelihood Estimation» and others. Furthermore a user is even unaware of the existence of Software reliability models itself. On the basis of the system offered a user is able to make a choice of the most suitable Software tool, having only understandable information available for him (her). The information only reflects the conditions of Software testing and creating as well as a defect detection and correction method. The usage of the reliability estimation software tool classification demonstrated in the article will allow to make the reliability theory closer to everyday life and give an opportunity to specialists to develop more reliable and stable Software..

References

1. J.D. Musa, K. Okumoto, Electronic Systems Effectiveness and Life Cycle Costing, NATO ASI Series, **3,** 395, (1983)
2. M.R. Lyu *Handbook of Software Reliability Engineer-ing* (New-York: McGraw-Hill, 1996)
3. G.E. Stark *International Symposium on Software Reliability Engineering*, 90, (1991)
4. D.R. Wallace, *26th Annual NASA Software Engineering Workshop*, 147, (2001)
5. L. Crow, ReliaSoft, (http://www.reliasoft.com/rga/)
6. M.R. Lyu, A.P. Nikora, *Fifth International Workshop on Computer-Aided Software Engineering*, 264, (1992)
7. Y. Malaiya, J. Denton, N. Li, (http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.5.5397)
8. Li, N. *Sixth International Symposium on Software Reliability Engineering*, 375, (1995)
9. C.C. Chen, C.T. Lin, H.H. Huang, S.W. Huang, C.Y. Huang, *IEEE Region 10 Conference TENCON 2006,* 1, (2006)
10. M.A.A. Boon, E. Brandt, I.C. Ramos, (http://www.refis.nl)
11. H. Okamura, T. Dohi, Software Reliability Engineering (ISSRE), 100, (2013)
12. T. Fujiwara, M. Kimur, JSEA, **7**, 396, (2014)
13. V.S. Kharchenko, O.M. Tarasyuk, V.V. Sklyar, V.Y. Dubnitsky, *International Computer Software and Applications (COMPSAC 2002)*, 541, (2002)