Similarity Based Fuzzy Logic Modeling for the Software Reliability Assessment

Svitlana Yaremchuk[1], Vyacheslav Kharchenko [2]

[1] *Danube Institute of National University "Odessa Maritime Academy", 9, Fanagoriyskaya St., Izmail, 68600, Ukraine*
[2] *National Aerospace University "KhAI", 17, Chkalov St. Kharkiv, 61070, Ukraine*

Abstract
The approach to the assessment of the software reliability is described, which is based on Fuzzy logic modeling using data set of the similar software systems. The suggested Fuzzy Inference Reliability Model is based on the dependencies between the metric estimates of the complexity of modules and the number of faults. The model allows predicting the number of faults in any module of the newly developed system to perform the effective verification. The automatic extraction of fuzzy logical rules from the data set of the similar system and the import of rules into MATLAB is proposed. The automation of the modeling process would reduce the time and the effort to process large sets of experimental data. It is proposed to develop the model by increasing the number of input variables taking into account the factors of the development process affecting its reliability: the qualification of developers, the level of maturity of the development process, the degree of code reuse.

Keywords
Big data, similarity, software reliability, assessment, fuzzy logic modeling

1. Introduction

Assurance of reliability, security and safety of software systems is one of the key challenges in IT development and implementation. The development of modern software systems (SWS) is a complex multi-stage iteration process. The reliability of the system is formed at each stage and depends on a variety of internal features and external factors. First of all, internal features are complexity, reusing of software components and so on. Important external factors are developers' qualification, development processes management level and so on. These properties are evaluated by different quantitative and qualitative indicators. Diapasons of assessment are fuzzy and overlapping. A great number of fuzzy estimates are not always possible to take into account by using analytical software reliability models. However, this can be done using the techniques of soft computing [1].

1.1. Motivation

The development of soft computing model requires experimental data, that allows to determine the dependence of SWS reliability on internal features and external factors. We suggest using the data from previously developed SWS that have similar properties and similar reliability. We called such systems similar software systems, and described them in the works [2, 3, 4].

During the development and verification process of SWS enormous arrays of information on their properties are accumulated. Experimental data, any information about SWS reliability are considered as Big Data of Software Reliability (BDSWR). This valuable and available information resource of

COLINS-2022: 6th International Conference on Computational Linguistics and Intelligent Systems, May 12–13, 2022, Gliwice, Poland
EMAIL: s.yaremchuck@csn.khai.edu (S. Yaremchuk); kharchenko@csn.khai.edu (V. Kharchenko)
ORCID: 0000-0002-6736-1471 (S. Yaremchuk); 0000-0001-5352-077X (V. Kharchenko)

software reliability contains the experimental data about the actual reliability of the earlier developed SWS. BDSWR include the development artifacts and data about their reliability, which include software requirements and the revealed errors in them, the charts and descriptions of the architecture and faults and vulnerabilities in them, the source code and the faults description in it, time series of failures, different metric estimates, etc. Similar software systems can be searched in BDSWR arrays. However, according to the overview of modern works it is concluded that BDSWR have not been systematically analyzed and used for the SWS reliability assessment.

The research motivation consists in developing the technique of SWS reliability evaluation on the base of BDSWR analysis, finding similar systems and soft computing model development and application.

1.2. State of art

Similar entities are widely used in big data analysis for mirror Web sites detection, for filtering of the similar materials on the news websites, for ratings determination of different products, for determination of similar sets of goods or services in on-line trade, for plagiarism detection in different artifacts, including SWS. Two programs can be similar at the level of purpose, algorithm, or implementation [2-4].

The widely applied soft computing techniques are listed below such as: Fuzzy Inference System (FIS), Artificial Neural networks (NN) and Adaptive Neuro Fuzzy Inference System (ANFIS), Evolutionary Algorithms (EA), K-Nearest Neighbour (K-NN), Support Vector Machines (SVM), Probabilistic Reasoning (PR) or Probabilistic Logic (PL), Evolutionary Computation (EC), Genetic Algorithms (GA), Chaos Theory (CT), Hybrid Model and some other [1]. According to the overview of modern works, the most demanded soft computing techniques are fuzzy modeling techniques, which allow to create input and output variable dependence models based on fuzzy rules.

In the work [5] the authors estimated the reliability of Component Based Software System (CBSS) by the FIS with two different numbers of membership functions. After compression of the output reliability values for different input sets, the authors made the analysis. The analysis showed that FIS model with five membership functions provides better results as per three memberships function. CBSS reliability estimation was performed in only four factors that is Reusability, Operational profile, Component dependency and Application complexity. The model does not take into account important factors that affect reliability - this is the qualification of developers and the level of the development process maturity. But CBSS reliability is affected by more other factors like Availability, Performance, Fault density, Software quality, together with functionality, Capability, Install ability Usability, Serviceability, Capability, Install ability and Maintainability.

In the paper [6] the authors research aspect oriented software systems that provide the new methods for the separation of concerns multiple module configuration or intervention and automatic integration of them with a system. In this paper a method using fuzzy logic to measure software reliability based on the above aspects is presented. As the input variables the authors used the following metrics based on aspects: Concern Diffusion over Operations, Degree of Scattering across Classes, Degree of Scattering across Methods, Lines of Concern Code. The authors used the exponential membership functions of input and output model variables, the centroid defuzzification, which returns the center of area under the curve, and 81 rules.

There are such shortcomings of this model. Software tools are not developed to measure the metrics of concern. Experts need to manually select the concern code.

In the work [7] the author makes a point that the utilization of Software Reliability Growth Models (SRGM) plays a major role in monitoring progress, accurately predicting the number of faults in the software during both development and testing processes; defines the release date of a software product, helps in allocating resources and estimating the cost for software maintenance. This leads to achieving the required reliability level of a software product. The author investigated the use of fuzzy logic on building SRGM to estimate the expected software faults during testing process. A fuzzy nonlinear regression models were developed for predicting the accumulated faults of software engineering applications. The proposed fuzzy model consists of a collection of linear sub-models, based on the Takagi-Sugeno technique and attached efficiently using fuzzy membership functions to represent the expected software faults as a function of historical measured faults. A data set provided by John Musa of bell telephone laboratories (i.e., real time control, military and operating system applications) was used to show the potential of using fuzzy logic in solving the software reliability modeling problem. The developed models provided high performance modeling capabilities. In this case, it takes time and financial investment to further verify and achieve the required reliability. If this is not possible, the developers state the lack of reliability of the system and plan significant costs for an extensive support service.

In the paper [8] the authors defined a methodology to estimate the reliability on the basis of Product Metrics, Project Management Metrics, Process Metrics and Process Metrics. These are the following metrics: Maturity (Number of successful version released and Level of CMM), Fault Tolerance, Exception Handling, Recoverability, Reliability Compliance, Personal Satisfaction (Recommendation and Popularity) and Efficiency. The metrics were quantified on the basis of survey Fuzzy Reliability Survey Questionnaire (FRSQ). The authors notified the questionnaire system FRSQ as a very efficient one in evaluating the performance since every parameter is considered precisely. The proposed evaluation method is very simple and effective.

In our opinion, if the data for building a model are collected using a questionnaire survey of ordinary users, these data will depend on subjective perception and insufficient qualification of users in issues of assessing reliability. Reliability estimates from different users may be inconsistent because reliability is highly dependent on the system usage profile.

The study [9] has highlighted the weaknesses of earlier software reliability prediction efforts, and subsequently proposed a structured framework that may overcome the inadequacies of earlier studies and quantifies the reliability on the basis of the requirement and design phase measures before the coding starts. After describing all the phases of the framework the authors formulated the salient features of the framework.

The proposed framework evaluates reliability at the requirements and architecture stages of the system. However, the greatest number of faults is contained in the components of the newly created source code. The cost of verifying the code is comparable to the cost of developing it. In order to predict verification costs, it is necessary to predict the number of faults in the code.

The performed analysis shows that statistical methods and analytical models of the reliability assessment are applicable for accurate numerical data: the number of faults, time series of failures, etc.

However, at the SWS code development stage we have numeric data with fuzzy overlapping boundaries. Modules, code components can have the same or similar values of their property estimates.

Different numbers of faults can be detected in these modules. In this case, it is problematic to use analytical models, but it makes sense to evaluate the probabilities of fault content in modules depending on the estimates of their properties using metrics. For such tasks, the most often used methods are artificial neural networks methods and fuzzy inference models.

Artificial neural networks require parameter settings, during the overtraining they lose previously received information, and they do not explain why a certain value was obtained at the output. Artificial neural network is a "black box." These methods are used at the early stages of data analysis, when the data have not been studied enough; the direction and results of the analysis are not known in advance.

Fuzzy infection model requires the fuzzification of variables and the development of logical rules [10]. In the case of fuzzy modeling of simple tasks with a small number of variables, few rules are set by the model developer based on his expertise.

In the case of SWS reliability modeling, many logical rules must be used. In this case, the knowledge of the model developer is not enough. To retrieve rules, you need the similar SWS data.

Fuzzy inference models allow to visualize simulation results very well. This greatly facilitates their understanding and analysis. The model allows the adding and accumulating of the variables and rules. These factors determine the feasibility of using fuzzy simulations to assessment SWS reliability using the MATLAB [11].

The paper [12] presents software reliability apportionment using fuzzy logic. The proposed methodology attempts to allocate target reliability of the system to its modules. For calculation of reliability, the first step is to aggregate the opinion of all team members about each module. The second step is to calculate the proportionality factor of the software system. The authors defuzzified the Fuzzy Proportionality Factor by using defuzzification formula. After obtaining crisp values, the authors have to calculate weightage of each module. Based on weightage, reliability of each module is calculated. Operational Profile is one of the main parameters for making effective testing and improved reliability by testing most used functions in the first phase and lesser used functions in the next phase. Compared with the previous result, the reliability has been improved. It helps to allocate reliability to all modules before the actual system is built. Also it considers Operational Profile as a parameter in proportionality factor, which helps to allocate reliability to most of the used modules, therefore making reliability apportionment beneficial for every module.

The authors [12] evaluated the reliability of the four modules. However, modern SWS consist of hundreds and thousands of various modules. The difficulty of using the proposed methodology lies in the heavy labor intensity and is time-consuming for summarizing the opinion of all team members about each module.

In the paper [13] a new mathematical model is proposed to guesstimate the reliability of Component-based Software (CBS) using series and parallel reliability models approach. The output of the proposed model is compared with the outputs of soft computing techniques PSO and Fuzzy logic to compare the best value of reliability. The result shows that Fuzzy logic is more compatible for predicting reliability as compared to PSO. It is observed that the proposed reliability model has a lower error rate in predicting CBSE reliability as compared to reliability prediction utilizing fuzzy logic and PSO.

However, the proposed model only takes into account few factors. To improve the model, new factors can be added. However, this will increase the complexity by forming a large number of parameter combinations.

The paper [14] discusses fuzzy software reliability growth model under imperfect debugging environment. Log-logistic testing effort function (LLTEF) is incorporated into software reliability growth model (SRGM). It is shown that the proposed SRGM with LLTEF gives reasonable prediction of software reliability under the fuzzy model. Further, under optimal release policy, software cost based on reliability criteria is calculated under the fuzzy condition. The difficulties of using this model are caused by uncertainties for trapezoidal fuzzy numbers that are associated with the calculated parameters of the SRGM proposed due to human behavior. There is a different level of uncertainties to compute various assessment of software reliability.

The authors [15] discussed the different classification schemes of existing soft computing techniques. In classification schemes of existing soft computing techniques Neuro-Fuzzy System is present. This led us to the idea of combining Data Mining and Fuzzy-Logic methods into a hybrid method that would combine the strengths of both methods.

The article [16] a model of a fuzzy hierarchical system for assessing the security profile has been developed, which sets a set of assessment criteria and the sequence of their use. The proposed hierarchical model makes it possible to present the assessment process in an explicit form and implement the process of checking the criteria, indicating the degree of confidence of the expert in the relevance of the assessment criteria. It is this circumstance that is the disadvantage of this model. The surveys of the expert require a lot of resources.

The authors [17] emphasize again that an order to develop a reliability model using fuzzy logic, access to reliable expert knowledge is necessary.

Analysis of this work led us to such a thought. You can reduce the cost of using expert knowledge and eliminate its lack of reliability if the fuzzy logic rules are automatically extracted from the experimental data set.

The authors of the reviewed works do not use similar systems to build fuzzy reliability models, and or not use these models to predict the reliability of newly created systems.

The purpose of the research are the following:

- to build the Fuzzy Inference Reliability Model (FIRM) for the similar software system. To achieve this goal, it is necessary to:
- define input/output linguistic variables and select an algorithm for the model;
- draw up logical rules of the model;
- visualize and analyze the obtained results;

- to offer the approach of predictive estimation of reliability of a new system based on the FIRM and data of a similar system.

The paper is structured as follows: section II describes FIRM; section III describes FIRM based approach to software reliability assessment; section IV contains conclusions.

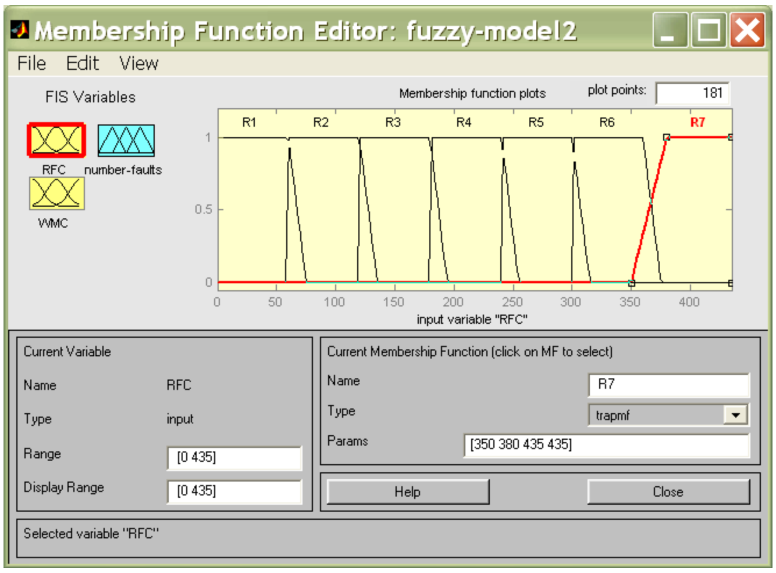
2. Fuzzy inference reliability model

Data sets from storage [18] contain the name of the module, its metric estimates (on metrics WMC, DIT, NOC and others), and the number of faults (bug). The general characteristics of this system are as follows. It is an open-source academic project with 412 thousand code lines, 885 components and 625 faults. The expediency of using these experimental data consists of their availability and sufficient volume.

2.1. Linguistic variables and model algorithm

Data sets from storage [18] contain the name of the module, its metric estimates (on metrics WMC, DIT, NOC and others), and the number of faults (bug). The general characteristics of this system are as follows. It is an open-source academic project with 412 thousand code lines, 885 components and 625 faults. The expediency of using these experimental data consists of their availability and sufficient volume.

The statistical analysis of the data set showed that the highest correlations with the number of faults in the module belong to metric estimates of the structural (RFC, Response for Class) and functional (WMC, Weighted Methods per Class) complexity. Therefore, we defined RFC and WMC estimates as model input variables. The parameters of the model input variables are shown in Figure 1.



plot points: 181

Current Variable

Name	RFC
Type	input
Range	[0 435]
Display Range	[0 435]

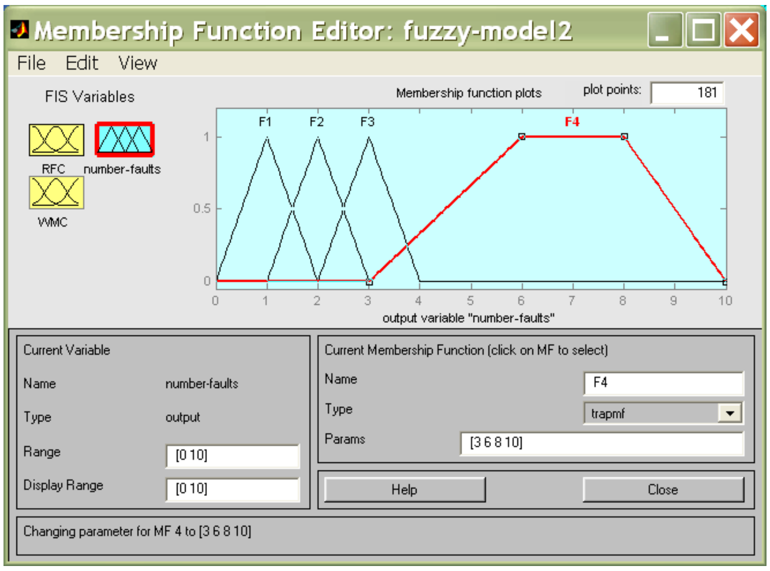
Current Membership Function (click on MF to select)

Name	R7
Type	trapmf
Params	[350 380 435 435]

Selected variable "RFC"

Figure 1: Model input variable parameters - RFC estimates

The model output variable is the number of faults in the module. The parameters of the model output variable are shown in Figure 2.


Membership Function Editor: fuzzy-model2
File Edit View
FIS Variables
RFC
number-faults
WMC
Membership function plots
plot points: 181
F1
F2
F3
F4
output variable "number-faults"

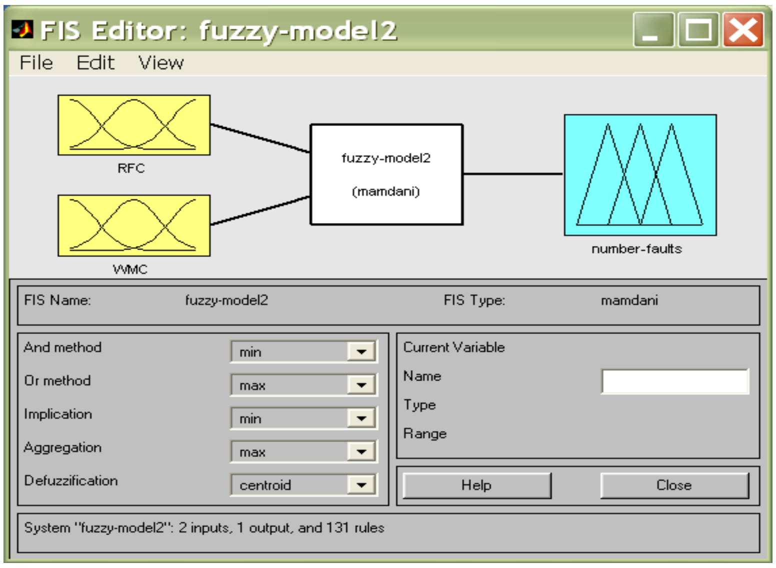
Current Variable
Name: number-faults
Type: output
Range: [0 10]
Display Range: [0 10]

Current Membership Function (click on MF to select)
Name: F4
Type: trapmf
Params: [3 6 8 10]

Changing parameter for MF 4 to [3 6 8 10]

Figure 2: Parameters of the output variable

A review of sources [5-10] showed that for fuzzy modelling of software reliability, the Mamdani algorithm described in [1] was most often used. We also used the Mamdani algorithm to build the model. Model parameters are shown in Figure 3.



FIS Name: fuzzy-model2
FIS Type: mamdani

And method: min
Or method: max
Implication: min
Aggregation: max
Defuzzification: centroid

Current Variable
Name
Type
Range

System "fuzzy-model2": 2 inputs, 1 output, and 131 rules

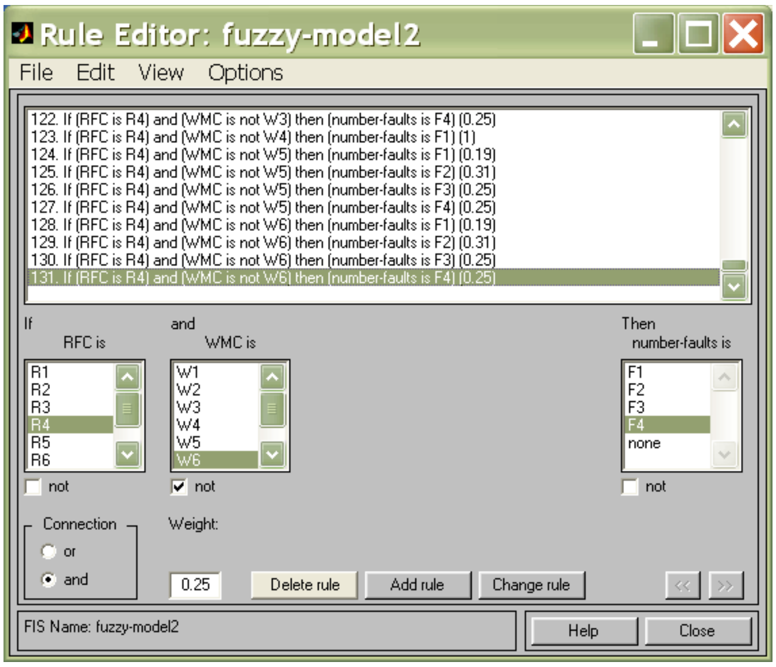
Figure 3: Model parameters

2.2. Fuzzy If-Then rules

The proposed model has the following characteristic. To construct logical rules that define the dependencies between the input and output variables of the model, we did not use expert estimates. We extracted logical rules from experimental data of the similar system using filtering, grouping and aggregation operations in MS Excel.

We used the following principle to construct logical rules. Monotonically growing metric estimates of RFC and WMC were divided into 7 intervals (phases) with an equal number of modules in each interval. This is shown in Figure 2. The input variables for RFC are denoted as R1-R7, WMC as W1-W7. The number of faults was divided into 4 intervals and designated as F1-F4. This is shown in Figure 3.

Next, logical rules were defined as all possible combinations of input and output variables with different weights. Weights are represented as probabilities of the number of faults in the module. As a result, a base of 131 rules was formed. The final part of the rules is shown in Figure 4.



```
122. If (RFC is R4) and (WMC is not W3) then (number-faults is F4) (0.25)
123. If (RFC is R4) and (WMC is not W4) then (number-faults is F1) (1)
124. If (RFC is R4) and (WMC is not W5) then (number-faults is F1) (0.19)
125. If (RFC is R4) and (WMC is not W5) then (number-faults is F2) (0.31)
126. If (RFC is R4) and (WMC is not W5) then (number-faults is F3) (0.25)
127. If (RFC is R4) and (WMC is not W5) then (number-faults is F4) (0.25)
128. If (RFC is R4) and (WMC is not W6) then (number-faults is F1) (0.19)
129. If (RFC is R4) and (WMC is not W6) then (number-faults is F2) (0.31)
130. If (RFC is R4) and (WMC is not W6) then (number-faults is F3) (0.25)
131. If (RFC is R4) and (WMC is not W6) then (number-faults is F4) (0.25)
```

Rule Editor: fuzzy-model2

If RFC is: R1, R2, R3, R4 (selected), R5, R6 — not [ ]

and WMC is: W1, W2, W3, W4, W5, W6 (selected) — not [✓]

Then number-faults is: F1, F2, F3, F4 (selected), none — not [ ]

Connection: or [✓ and]

Weight: 0.25

FIS Name: fuzzy-model2

Figure 4: Logical Rule Build Window

The FIRM build technique consists of the following steps:

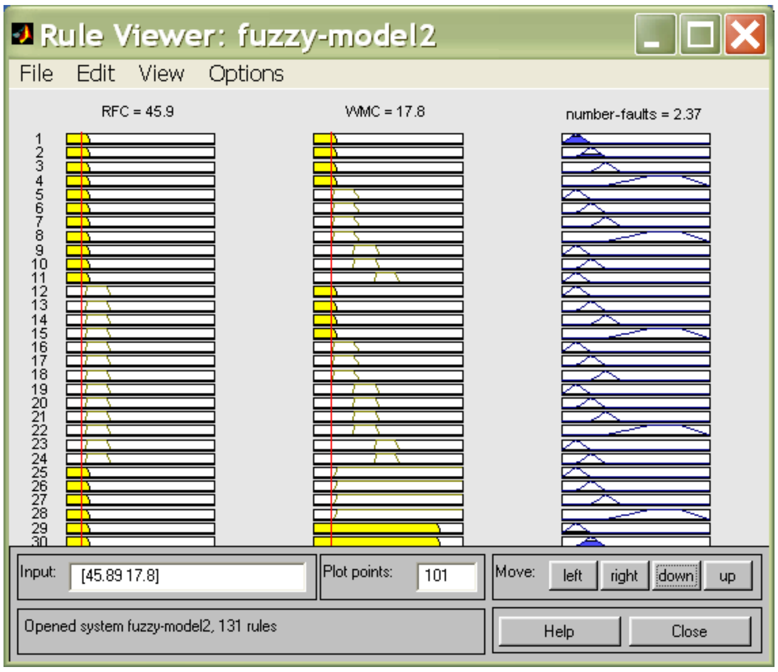
- Step 1. Determining the number and parameters of input and output variables with their subsequent fuzzification;
- Step 2. Defining the parameters of the algorithm used (Aggregation, Activation and Accumulation operation types);
- Step 3. Generation and entry of the rule base for fuzzy output;
- Step 4. Defuzzification of the output variable (Defuzzification).

After the construction of the FIRM, the following results were obtained.

2.3. Visualization and analysis of the obtained results

The visualization of results using diagrams allows to analyze in detail the dependencies of the number of faults on the metric estimates of modules over the entire range of their values.

The Rule Viewer window of the MATLAB allows to determine the most probable number of faults for any module of the SWS being developed. Figure 5 shows the case of practical use of the model. This figure shows that in a module with metric estimates of RFC≈46 and WMC≈18 the most probable number of faults is about 2 (2.37). This knowledge can be used by SWS testing and quality specialists.



Input: [45.89 17.8]

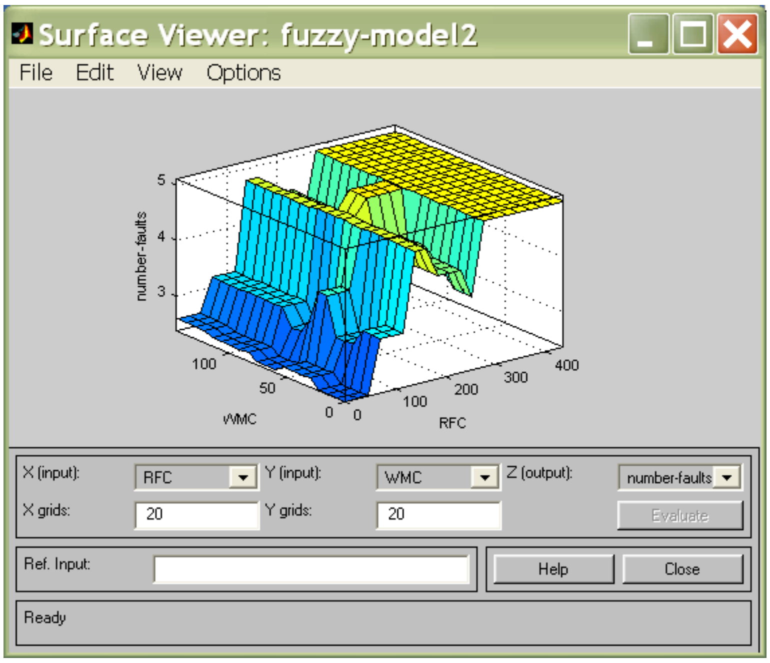
Plot points: 101

Move:

Opened system fuzzy-model2, 131 rules

Figure 5: Case of practical use of the model

Figure 6 shows the 3D surface of the model. The surface clearly represents the reliability of the SWS modules in the form of a multi-stage three-dimensional figure. Each step of the figure shows the dynamic of faults occurrence in the modules. The figure as a whole illustrates the tendency of the number of faults to grow as metric estimates increase. The blue plane shows the fastest increase in the number of faults from about 1 to 3.



X (input): RFC

Y (input): WMC

Z (output): number-faults

X grids: 20

Y grids: 20

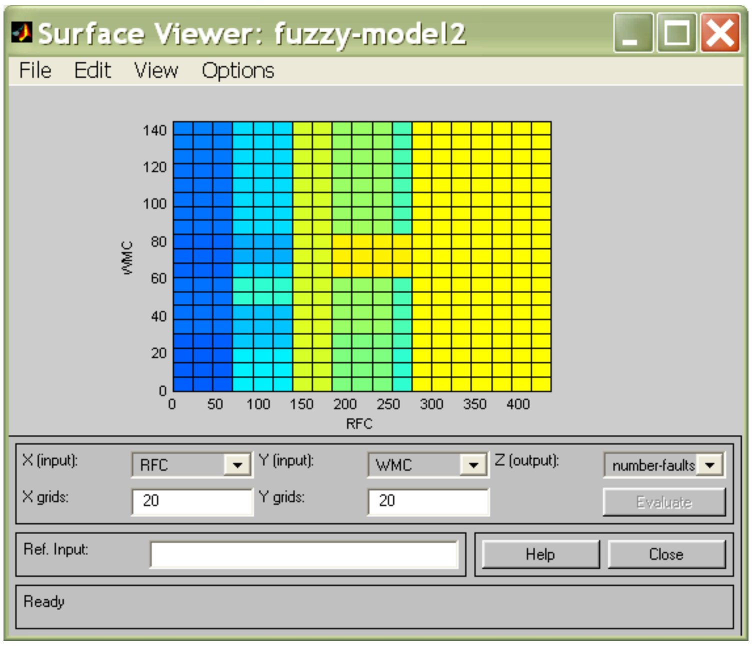
Ref. Input:

Ready

Figure 6: 3D Model Surface

The blue plane shows a less rapid increase in the number of faults from about 3 to 5. The green plane shows a decrease in the number of faults compared to the blue plane from 5 to 4. The yellow plane shows the largest number of faults from 5 and above.

Figure 7 shows the model as a flat chess chart. The chess chart shows multi-colored areas with numerical estimates of the properties of the SWS modules. The most defective modules are in the yellow area. These are RFC estimates from 150 to 200 for any WMC estimates. These are RFC estimates from 200 to 250 with WMC estimates from 60 to 80. These are RFC estimates above 300 for any WMC estimates.



X (input): RFC
Y (input): WMC
Z (output): number-faults
X grids: 20
Y grids: 20

Figure 7: A model in the form of a chess chart

The practical use of the proposed FIRM in software engineering is as follows. FIRM allows you to predict the most likely number of faults in a software module with different metric estimates.

The resulting surface and chess chart directs the efforts of specialists to the most defective modules for more efficient verification. Each module from the yellow area contains more faults than the other areas. Such modules need to be verified first, and then we are to test the modules from the green, blue and blue areas.

It is necessary to automate input processing with a software agent for improving the accuracy and reducing the simulation costs. The agent will allow you to easily change the number and gradations of input linguistic variables for a comprehensive study of the model. The agent will also automatically extract logical rules from the dataset and generate them in the desired format for import into the MATLAB. In our opinion, the accuracy of the simulation should be assessed after a comprehensive study of the model using an agent. These activities will achieve high accuracy in modeling the reliability of software modules based on a fuzzy model of similar SWS to accurately predict the reliability of newly developed systems. In the process of building a FIRM, you can do without subjective and expensive expert assessments. Rules can be automatically retrieved from sufficient data using the target software agent. The agent can be developed by software company specialists.

The restriction of the proposed model is a small number of input variables. However, this restriction can be removed by adding variables that affect the reliability of the modules. The MATLAB allows to use up to ten input variables.

In the work [4], we received confirmation of the assumption that similar systems have similar reliability using statistical methods. We believe that with the help of FIRM it is possible to confirm the similarity of reliability for such systems by fuzzy methods. To do this, you need to build and visualize surfaces and chess charts for similar systems. Their comparison will confirm the assumption of the reliability similarity of similar systems.

2.4. Discussion

When building a FIRM, you must specify logical rules that define the values of the output variables for all the intervals (phases) of the input variables. However, experimental data are incomplete for natural reasons. Modules with some metric estimates in the system may simply be missing. Incomplete experimental data can be supplemented in two ways:

- firstly, by use of expert knowledge;
- secondly, joining of data sets of two or more similar systems.

To supplement incomplete data, we used the third way when building the FIRM. This allowed us to increase the number of fuzzy rules to more fully describe the dependencies of the input and output variables of the model. It should be noted that the number of FIRM input variables and their gradations remain debatable issues.

When building the FIRM, we discovered interesting characteristics and made an assumption. As the number of faults in modules increases, it is necessary to increase the number of phases for input variables. However, this assumption requires theoretical justification and experimental verification.

3. The proposed approach based on the FIRM and known assessments of a similar system

We call software systems that have similar properties as similar ones, and assume that similar software systems have similar reliability. The principles of similarity of systems described in the work [4]. In this work, the reliability of similar software systems was investigated. Analysis showed that 79% of the systems had a high and medium similarity of reliability. The obtained results allowed us to draw a conclusion about the similarity of reliability for similar systems. This allows the reliability estimates of a similar software system to be used to predict the reliability of a new system.

Developers can use predictive reliability estimates to plan inspection, refactoring, and testing processes. These activities will increase the reliability of new software systems while reducing the verifying cost.

In our opinion, it is advisable to start developing FIRM by the modeling of the dependence of the reliability on the structural and functional complexity of modules. This is the dependence between the number of faults in the modules and the metric complexity estimates of the modules. For modeling, we suggest using the data sets of the similar system that were developed and verified earlier. Such data is generated automatically during the coding and the testing processes. This data is stored in version control systems and bug tracking system. This data is freely available on global resources.

The method of the predictive assessment of the reliability of the new system based on the fuzzy model and data of the similar system consists of the following stages:

1. Specialists develop modules of the new SWS. Metric estimates of modules are known, the number of faults in modules is not known. It is necessary to predict the number of faults in order to plan the testing time and resources. To predict the number of faults, it is necessary to find a similar system with a known number of faults.
2. Experts extract data sets of different systems from the repository [18]. The software agent for search of similar system is described in the work [2]. By means of a software agent, experts process datasets and find the similar SWS. Since the similar SWS was developed and verified earlier, the number of faults in the modules is known. The data set of a similar system contains the number of detected faults in the modules and metric estimates of the modules. This data is sufficient to build a FIRM.
3. Experts use metric estimates of modules of a similar system as input variables of the model, and number faults in a module as output variable. Experts describe linguistic variables, choose an algorithm, compose the logical rules of the model and form a FIRM.
4. Experts use FIRM to predict the number of faults in the modules of the new SWS.

4. Conclusions

The following results have been presented.

- The authors of the reviewed works do not use similar systems to build fuzzy reliability models or do not use these models to predict the reliability of newly created systems.
- The FIRM build technique is described.

- The Fuzzy Inference Reliability Model has been developed to predict the number of faults in the modules of the developed system, which will increase the effectiveness of verification.
- The method of predictive estimation of reliability of a new system based on the fuzzy model and data of a similar system is described.
- Computer experiments have shown the possibility of applying the developed model in practice.

It is possible to automatically extract fuzzy logical rules from experimental data, and submit the rules electronically for later import into the MATLAB system. Automating the rule extraction process will reduce the time and cost for the simulation process.

The development of FIRM consists in increasing the number of input variables of the model. Additional input variables can include various factors that affect reliability: the qualifications of developers, the level of maturity of the development process, the degree of reuse of native and third-party development code, and others. The study of the properties of the model and its development determine the directions of further research.

5. Acknowledgements

This research was supported has been supported by the project STARC (Methodology of SusTAinable Development and InfoRmation Technologies of Green Computing and Communication) funded by Department of Education and Science of Ukraine. Besides, research has been supported by the ECHO project which has received funding from the European Union's Horizon 2020 research and innovation program under the grant agreement no 830943.

6. References

[1] Zadeh, Lotfi A., «Fuzzy Logic, Neural Networks, and Soft Computing», Communications of the ACM, Vol. 37 No. 3, (1994) 77—84

[2] Kharchenko, V., Yaremchuck, S. Technology Oriented Assessment of Software Reliability: Big Data Based Search of Similar Programs. Conference Proceedings of the 13th International Conference ICTERI' 2017, Kyiv, Ukraine, 2017, Vol. 1844, pp. 686-698. http://ceur-ws.org/Vol-1844/10000686.pdf

[3] S. Yaremchuk, V. Kharchenko. Technology oriented assessment of software reliability: big data based search of similar programs. Conference Proceedings of 2018 IEEE 9th International Conference on Dependable Systems, Services and Technologies DESSERT'2018. Ukraine, Kyiv, 2018, pp. 485-490. doi: 10.1109/DESSERT.2018.8409182.

[4] Yaremchuk, S., Kharchenko, V., Gorbenko, A. Search of similar programs using code metrics and Big Data based assessment of software reliability. The chapter 10 of the book Applications of Big Data Analytics: Trends, Issues, and Challenges. Springer International Publishing, Switzerland, pages 185-211, 2018. doi:10.1007/978-3-319-76472-6.

[5] Gopal Prasad Jaiswa], Ram Nivas Giri. A Fuzzy Inference Model for Reliability Estimation of Component Based Software System. International Journal of Computer Science Trends and Technology (IJCST), Vol. 3 Issue 3, (2015) 177-183. http://www.ijcstjournal.org/volume-3/issue-3/IJCST-V3I3P31.pdf

[6] Mohammad Zavvar, Farhad Ramezani. A Method Based on Fuzzy System for Assessing the Reliability of Software Based Aspects. Advances in Science and Technology Research Journal, Volume 9, No. 27, (2015) 143–148. doi:10.1016/j.procs.2015.08.418.

[7] Sultan Aljahdali. Development of Software Reliability Growth Models for Industrial Applications Using Fuzzy Logic. Journal of Computer Science 7 (10), (2011) 1574-1580. doi:10.3844/jcssp.2011.1574.1580.

[8] Feroza Haque, Sanjeev Bansal. Fuzzy Based Software Reliability Estimation. International Journal of Computer Science and Technology, IJCST Vol. 3, Issue 2, (2012) 835-838. http://ijcst.com/vol32/5/feroza.pdf

[9] Syed Wajahat Abbas Rizvi, Vivek Kumar Singh, Raees Ahmad Khan. Fuzzy Logic based Software Reliability Quantification Framework: Early Stage Perspective (FL SRQF). Processing Twelfth International Multi-Conference on Information-2016 (IMCIP-2016), Procedia Computer Science 89, (2016) 359 – 368. doi: 10.1016/j.procs.2016.06.083.

[10] Andrzej Piegat. Fuzzy Modeling and Control. Heidelberg, Physica Verlag, Springer Verlag Company, (2001) p. 704/ https://link.springer.com/book/10.1007/978-3-7908-1824-6.

[11] Brian Hahn and Dan Valentine. Essential MATLAB for Engineers and Scientists (Sixth Edition). Elsevier Ltd., Oxford, UK, (2017) p. 411/ https://www.amazon.com/Essential-MATLAB-Engineers-Scientists-Brian/dp/0081008775.

[12] Amrita and Dilip Kumar Yadav. Software Reliability Apportionment using Fuzzy Logic. Indian Journal of Science and Technology, Vol 9(45), (2016) 1-8. doi: 10.17485/ijst/2016/v9i45/106347.

[13] Chander Diwaker, Pradeep Tomar, Arun Solanki, Anand Nayyar, Nz Jhanjhi, Azween Abdullah, and Mahadevan Supramaniam. A New Model for Predicting Component-Based Software Reliability Using Soft Computing. IEEE ACCESS (2019) 147191-147203. doi: 10.1109/ IEEEACCESS.2019.2946862.

[14] S. Rani, N. Ahmad. Analysis of Fuzzy Software Reliability Growth Model and Optimal Release Policy with Log-logistic Testing Effort under Imperfect Debugging. IJCSNS International Journal of Computer Science and Network Security, VOL.19 No.7, 2019. http://paper.ijcsns.org/07_book/201907/20190722.pdf

[15] Parmanand Kushwah, Ramnaresh Sharma. Soft Computing Techniques for Software Reliability. International Journal of Scientific Engineering and Research (IJSER). Volume 9 Issue 7, (2021) 1-14. https://www.ijser.in/archives/v9i7/SE21715194206.pdf

[16] I. Shelechov, N. Barchenko, V. Kalchenko, V. Obodiak. A hierarchical fuzzy quality assessment of complex security information systems. RADIO ELECTRONIC AND COMPUTER SYSTEMS, 2020, № 4 (96). P 106-115. https://essuir.sumdu.edu.ua/handle/123456789/83451

[17] Andrzej Żyluk, Konrad Kuźma, Norbert Grzesik, Mariusz Zieja, and Justyna Tomaszewska. Fuzzy Logic in Aircraft Onboard Systems Reliability Evaluation—A New Approach. (2021) 7913. doi: 10.3390/s21237913.

[18] Tera-PROMISE Home, 2018. http://openscience.us/repo/defect/ck