Journal of Engineering and Applied Sciences 13 (23): 10080-10085, 2018
ISSN: 1816-949X

Development of a Software System to Ensure the Reliability and Fault Tolerance in Information Systems

[1]Askar Boranbayev, [2]Seilkhan Boranbayev, [2]Assel Nurusheva and [2]Kuanysh Yersakhanov
[1]Department of Computer Science, Nazarbayev University, Astana, Kazakhstan
[2]Department of Information Systems, L.N. Gumilyov Eurasian National University,
Astana, Kazakhstan

Abstract: The use of information systems in various fields shows that there are situations in which there were failures due to defects in the software which in turn led to great damage. This is especially, dangerous for critical systems (satellite management, complex administrative systems, banking systems, technological and information systems, etc.). This study is devoted to ensuring the reliability of information systems. An approach based on risk assessment and neutralization is used to increase the reliability of information systems. This approach allows for early risk assessment of the software development process and determines the most effective mitigation strategies.

Key words: Method, reliability, software, application, design, banking system

INTRODUCTION

Currently, there are various methods for creating reliable Information Systems (IS). Various models of IS reliability have been created. However, the problem of IS reliability is far from being solved. Big issues of systems are that the reliability of them is very low (Lim *et al*., 2017). The main reason for this is that each IS is unique. The assessment of IS reliability is usually obtained from test results. Progress in improving the reliability of software is not as high as in the field of hardware. This is a consequence of the fact that the software is less standardized and much more complicated than the hardware. Some Software Tools (ST) are so, complex that when evaluating reliability, its individual modules are examined. One of the important factors affecting reliability is the number of errors in the software. Errors in the development stage lead to a decrease in the software reliability. There are areas in which it is impossible to test and determine the reliability of software on real objects (for example, nuclear reactors, aviation and space equipment, etc.). In these cases, test benches are usually created. The probability of operational state in a certain time interval, the intensity and the average error occurrence time are usually used to assess the reliability of software tools.

The possibility of software failures is a risk in any software system. The occurrence of a single software failure can lead to disastrous consequences in many areas where critical systems with complex designs (such as nuclear power plants or the management of transport systems) are used. For this moment, the reliability of design and production of mechanical and electromechanical components has risen to a high level nad most of the failures appearing in hardware-software hybrid systems are software failures. In this regard, it is important to perform a risk analysis in all areas of design and implementation of software. With the proliferation of software and hardware-software systems, risk analysis became necessary everywhere, even in projects of small scale. It is necessary not only to determine the risk but also to take measures to prevent or identify potential failures in the software (Vucovich *et al*., 2007).

This study considers an approach based on risk assessment and neutralization to increase the reliability of information systems. The approach is based on the adaptation of the Red and Green methods to assess the risks of information systems (Lough *et al*., 2006a, b). It is shown that the red method can be used in the early stages of the software development process to identify and analyze the risk represented by potential software errors. For the first time, the red method was used in the field of electromechanical design. The green method is a tool that helps in reducing risk by using risk assessment and mitigation strategies based on historical data (Krus, 2012). This study shows that the red and green methods can be adapted and used in the field of information systems to provide the risk assessment based on its functionality.

Corresponding Author: Askar Boranbayev, Department of Computer Science, Nazarbayev University, Astana, Kazakhstan

MATERIALS AND METHODS

Methods red and green: Red provides an important information about the product, showing the exact status of the risk of the test product (Lough *et al*., 2009). This method provides valuable historical information on the likelihood of occurrence and the consequences of specific product risks. In order to perform the analysis, it is necessary to collect data that contains information about components, functions, failure modes and the severity of registered failures.

To obtain the final result about the product risk, three matrices must be filled with important data. One of these matrices is the Function-Component matrix (EC) which has information about the functions and components of a test product. In case of information systems, we may consider modules as components of the product. Next, there is a matrix called the Component-Failure (CF). Here is the information about failures and components of the test product. The two matrices are multiplied together Eq. 1 to get the resulting Function-Failure matrix (EF) that forms the failure knowledge base (Vucovich *et al*., 2007):

$$EF = EC \times CF$$

$$EC_{k,i} = \begin{cases} 1 & \text{where component i} \\ & \text{has been used} \\ \text{to provide functionality k} & \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

$$CF_{i,j} = \text{n where component i has experienced n failures of type j}$$

$$EF_{k,j} = \text{p where functionality k is associated p times with failure mode j}$$

To calculate the likelihood of occurrence of a specific failure, data from the Function-Failure matrix (EF) is used. The CF' matrix contains information about the failure severity. This matrix contains records of failure severity on a 5-point scale (where 1 means that the failure is insignificant and 5 means that the risk is high). If no failures are registered, the value 0 is used (Vucovich *et al*., 2007).

Red provides two methods for calculating the likelihood of failure (L1-prod and L2) and two methods for calculating the severity level of the failure (C1 and C2) (Lough *et al*., 2006a, b; AlKazimi *et al*., 2015).

The consequence mappings are indicated in Eq. 2 for C1 and in Eq. 3 for C2. The formula for C2 depends on the value of h which is the relevant number of function-failure combinations for this function (the number of products in $EC_{k,r} \times CF_{r,j}$ that are different from zero) (Vucovich *et al*., 2007):

$$C_{k,j} = \max_{r}\left(EC_{k,r} \times CF'_{r,j}\right) \quad (2)$$

$$C_{k,j} = \text{int}\left(\frac{1}{h}\sum_{r} EC_{k,r} \times CF'_{r,j}\right) \quad (3)$$

As a result, filling in the fields of EC, CF and CF' matrices, we can get 4 mappings: L1, L2, C1, C2. The mapping data allows us to create a chart, based on which we can distinguish three types of risk: low (green color), medium (yellow color) and high (red color) risk.

The green method, as mentioned earlier, helps in reducing risk, using risk mitigation strategies of collected historical data on risk reduction. This method can help the developer in reducing or eliminating unplanned failures that may arise in software development. Like the red method, green collects data on risk mitigation strategies into matrices and multiplies them together to form links between failure modes and strategies. To effectively use green method, the knowledge base must exist (Krus and Grantham, 2013). In study of Alkazimi *et al.* (2015) the green method was used as a tool to select suitable risk mitigation strategy to prevent prospective failures in upstream industry.

The knowledge base consists of matrices: a Failure-Parameter matrix (FP) and a Parameter-Strategy matrix (PS) (Krus, 2012). Another matrix is the mitigation Strategy likelihood Consequence change matrix (SC), contains information about the changes in the likelihood and consequences caused by the mitigation strategy. The matrices FP and PS are multiplied together, eventually creating a Failure mode mitigation Strategy matrix (FS) that contains information about mitigation strategies for certain failure modes and the frequency of their occurrences. Using the FS matrix, we can calculate the popularity of the mitigation strategy and by using SC matrix and original likelihood and consequence, we can calculate new values for likelihood and consequence of the risk (Krus and Grantham, 2013).

The green knowledge base should contain as much historical data as possible in order to provide the most accurate ratings. To fill this knowledge base, you need information from bug reports which contains information on risk mitigation.

The first stage of the knowledge base construction is population of the FP matrix. This matrix contains the failure modes and parameters that correspond to them. Recording of parameters that correspond to the mitigation strategy in the PS matrix is the second stage of construction. Lastly, information on changes in likelihood and consequences is collected for each strategy and recorded in the SC matrix (Krus and Grantham, 2013).

Table 1: Strategies

Strategy	Popularity	New consequence	New likelihood
Condition material	3	5	<1
Condition part	4	5	<1
Convert material	4	4.8	0.96
Convert part	3	5	0.98
Decrease power assist	1	5	<1
Import lubricant	1	5	<1
Increase controls	2	<5	<1
Increase flow	1	5	<1
Position part	1	5	<1
Remove part	1	<5	<1
Separate contaminant	1	5	<1
Shape part	5	5	<1
Stabilize flow	1	5	<1
Stabilize process	1	5	<1

When filling the FP matrix, first, you need to determine the failure mode specified in the report. If this is not explicitly specified, the report context can be used to determine the failure mode. Then, the parameters that caused or are associated with the failure mode are determined. The design and environment parameters used in the failure mode models can be those parameters. This matrix is a binary matrix, only records whether the parameter was associated with the failure mode nad not how often this occurred. Thus, in the FP matrix, you must enter 1 for each combination of failure mode and parameter found nad 0 is filled in for those that do not have this relationship (Krus and Grantham, 2013).

When filling out the PS matrix, first, you need to define a mitigation strategy chosen from the mitigation strategy taxonomy. Next, you need to define a parameter that has been changed by the strategy. After defining the mitigation strategy and particular parameter, the intersection of this strategy and parameter is incremented by one in the PS matrix (Krus and Grantham, 2013).

Changes in the likelihood and consequences of the risk that are caused by the strategy are in the SC matrix. It is used to calculate the new values of risk likelihood and consequences. This matrix uses the mitigation strategy, taken from the taxonomy of the risk reduction strategy, in the form of rows nad the likelihood and consequences changes in the form of columns. For this matrix, the change to the likelihood and the consequence of risk is recorded for the mitigation strategy determined when the PS matrix is filled. These values represent percent changes (from 0-100%) for each mitigation strategy (Krus and Grantham, 2013).

The most effective strategies are identified from the various strategies for each failure with medium or high level of risk. The final table indicates the data on the popularity, new consequence and new likelihood (Table 1) (Krus and Grantham, 2013).

Ensuring reliability and fault tolerance of information systems: The red method was used in engineering design. In this study, we show how the red method can be used in the field of information system reliability assessment.

Table 2: Severity of the failure

Name	Description
S1(blocker)	The most serious error. It is impossible to research with the system. Such errors must be corrected urgently
S2 (critical)	A critical error in which a certain part of the system does not research This problem must be solved in order to continue to research with the basic functions of the system
S3 (major)	This type of error in which something is working incorrectly but not particularly dangerous
S4 (minor)	Usually, minor errors do not disrupt the operation of the system. A problem might be in the user interface
S5 (trivial)	An error that does not pose a threat to the system, usually a problem of third-party library or service

Table 3: Failure priorities

Name	Description
P1 (high)	It requires a timely solution for the problem, since, this error is critical for the project
P2 (medium)	Not being a particularly dangerous threat to the system in any case, a solution to this problem is required.
P3 (low)	Not critical and does not require a quick solution but the error must be fixed

When developing an information system, especially if it is a complex system, there is always a possibility that an error will occur. To describe the situation and the sequence of actions that led to the incorrect operation of the system, according to the main fields of bug/defect report of pro testing-software testing website a document called "bug report" is created. In each bug report such attributes as severity and priority are indicated. Severity indicates how serious the failure is and how it can affect the software performance. Priority indicates the priority for each task to fix the problem. Thus, the first in the queue is a failure with the highest priority. The severity of the failure is usually identified by 5 types (Table 2). The priority of the failure is usually identified by 3 types (Table 3).

Thus, the data that is presented in the bug report can be used to determine the reliability of information systems using the red method. The CWE website lists 25 common programming errors (Anonymous, 2008, 2011). The following fields were listed for each error:

- Weakness prevalence
- Remediation cost
- Attack frequency
- Consequences
- Ease of detection
- Attacker awareness

In addition, for each error, prevention and mitigation measures are indicated. For example, we will view the problem #18 (Use of potentially dangerous function)

Table 4: Using a potentially dangerous function

Fields	Function
Weakness prevalence	High
Remediation cost	Medium
Attack frequency	Rarely
Consequences	Data loss, code execution
Ease of detection	Easy
Attacker awareness	High

Table 5: Bug report

Short description	Search on the main page of the Dream Book does not work correctly
Project	http://www.ameno.ru/
Application component	Search engine in the Dream Book
Version number	0.001
Importance	
S1 blocker	S3 Major
S2 critical	
S3 major	
S4 minor	
S5 trivial	
Priority	
P1 high	Filled in by the manager
P2 medium	
P3 low	
Status	New
Researcher	Alexey Bulat
Appointed	Developer name
Steps	Open the main page of the site: http://www.ameno.ru/ At the bottom of the page we find the section: dream dictionary (Russian language) Enter a search word, for example “night” Click “Search”
Actual result	The request failed validation
Expected result	The search went well, the description of the required dream is shown correctly

listed on the site (Table 4): the following recommendations were given as prevention and mitigation.

Prevention and mitigations

Build and compilation, implementation: Identify a list of prohibited API functions and prohibit developers from using these functions, providing safer alternatives. In some cases, automatic code analysis tools or the compiler can be instructed to spot use of prohibited functions, such as the “banned.h” include file from Microsoft’s SDL.

Using the above information, it is possible to fill out the red method tables to get a summary chart that will reveal the medium and high risks that must be mitigated. For identified risks, the green method mitigation strategies can be used. For a more detailed example, we will view one (Table 5) of the bug reports according to the bug report of pro testing-software testing website and specify which data can be used for the red method. From this bug report these elements can be used:

- Component: the search engine in the dream book
- Function: search
- Failure: request failed validation
- Importance: S3 major

5	0	0	1	0	0
4	0	0	0	0	0
3	0	0	0	0	0
2	0	0	0	0	0
1	0	0	0	0	0
Likelihood/consequence	1	2	3	4	5

Fig. 1: Chart for displaying C1L2

5	0	0	1	0	0
4	0	0	0	0	0
3	0	0	0	0	0
2	0	0	0	0	0
1	0	0	0	0	0
Likelihood/consequence	1	2	3	4	5

Fig. 2: Chart for displaying C1L1

Accordingly, it is necessary to fill 3 matrices in red method on the basis of which we will get the final result. For example:

For the EC matrix (Function-Component), we specify the function search, the component search engine in the Dream Book. For the CF matrix (Component-Failure), we specify the component search engine in the Dream Book and failure the request failed validation. For the CF’ matrix (severity of failure-component), we specify the component search engine in the Dream Book and the severity of the failure S3 (3 on a 5-point scale).

Multiplying the matrices EC and CF, we get the matrix EF (Function-Failure) which is used to calculate L1, L2 (likelihood). The EC and CF’ matrices will be used to calculate C1 and C2 (consequences). As a result, you can fill in the chart with L1, L2, C1 and C2:

- EC×CF = EF = 1
- CF’ = 3
- L1, L2 = int (5×1/1) = 5
- C1 = 3
- C2 = int (1/1×3) 3

Thus, since, we fill the matrix with only one component, the chart will have this form (Fig. 1 and 2): thus, 1 element is in the red zone nad requires a quick correction.

To automate the calculation of reliability and fault tolerance of Information systems a special prototype based on red and green methods was developed. This prototype may have changes in the future. Loading of initial data into the prototype software is done in two ways: manually and using Excel.

Manually; Step 1: When you select the "Manually" option, a new page opens in which you must enter the number and names of information system components. The same actions must be repeated for functions and failures. Additionally when filling in the number and names of failures, it is also, necessary to enter a level of risk on a 5-point scale: 1-5 (here 1-insignificant, 5-dangerous).

Step 2: The prototype present 3 tables based on the input data:

- Component and function
- Function and failure
- Function and severity of risk (for failures)

At the beginning, these tables are shown to the user to verify that the application has filled the data correctly. If an error is found in the presented tables, then the user has the opportunity to correct or add data by one of the correction methods, for example, go back to the page for filling in components, functions and failures (step 1) or change it by clicking the desired cell. If no errors are found, then the user proceeds to the next page.

Step 3: The new page presents 4 new tables obtained using the special algorithms and data from the previous tables. Here, the user needs to choose 2 out of 4 tables which he considers more suitable for his system. After making a choice, the user moves to a new page.

Step 4: The new page presents the summary chart which shows risks (their number and groups) that are in one of three zones:

- Green: insignificant
- Yellow: not dangerous but must be fixed
- Red: dangerous

Next, the user can save the resulting chart on his computer.

Load data using Excel; Step 1: The user needs to load the Excel file with the tables that are filled in according to the instructions. If it is not possible to collect data into tables due to the large amount of data that exists in the Excel file, the data will be sorted by the certain algorithm. Then the next processes are similar to steps 2-4 described in the "Manually" option.

RESULTS AND DISCUSSION

Thus, the prototype software solves the following tasks:

Forecasting of potential software failures; Risk assessment: It is determined when and how often failures occur based on mentioned methods. It is also, possible to identify failures in the early development stages of subsequent products. The information on the frequency and levels of the severity of failures proceeds to the software system. This allows user to reduce the severity level of risks or completely eliminate them.

Sorting the bug reports: The bug reports describe the situations and actions that led to the incorrect operation of the tested object. The reasons and expected result are indicated. Some bug reports are large in size and therefore, in order to assess the risks, you need to have a well-sorted bug report. The prototype software allows users to sort the file (bug report) by certain algorithm, so that, it can be used to assess the risk. This makes it easier to load data into the software.

Use of mitigation strategies: To prevent (mitigate) failures, it is required to have a database with mitigation strategies. To build such a database, information from bug reports is used. Also, it is possible to find a suitable (close) strategy in created database. Mitigation strategies can improve the reliability of certain parts of the software product or the whole product. Most of these strategies are created on the basis of expert knowledge. The software system allows you to determine which strategies should be used to reduce the risks of the information system.

Getting a detailed bug report: The software allows you to review and evaluate the risk of the whole product or each risk individually. It allows you to assess the reliability of the product, improve the product quality in the future, find suitable strategies for mitigating failures.

CONCLUSION

In this study, an approach for assessing risks and ensuring the reliability of information system was considered and the prototype software based on the red and green methods was presented. Risk assessment plays an important role in ensuring the reliability of the information system, especially if it provides automation of the operation of critical systems. Having a good tool for risk assessment can prevent threats that can negatively affect the operation of the system. Modern tools for risk assessment require expert knowledge in order to predict possible failures, the likelihood of their occurrence and consequences. The data used for risk assessment should be well catalogued and correctly used, as incorrect filling of information can lead to a great damage and losses. The software system provides information in a format that is understandable to any user. The end result is information about the information system risks. Further, based on the received information on risks, the software system generates data on strategies for certain failures and allows you to choose the best way to eliminate them. The proposed approach can be used in further development of the research presented by Boranbayev *et al.* (2014, 2015).

REFERENCES

AlKazimi, M.A., H. Altabbakh, S. Murray and K. Grantham, 2015. Evaluating generated risk event effect neutralization as a new mitigation strategy tool in the upstream industry. Procedia Manuf., 3: 1374-1378.

Anonymous, 2008. [Main fields bug/defect report]. Protest ES Ltd, Russia. (In Russian) https://translate.google.com/translate?hl=en&sl=ru&u=http://www.protesting.ru/testing/bugstructure.html&prev=search

Anonymous, 2011. CWE/SANSTop 25 most dangerous software errors. MITRE, McLean, Virginia, USA. http://cwe.mitre.org/top25/

Boranbayev, S., S. Altayev and A. Boranbayev, 2015. Applying the method of diverse redundancy in cloud based systems for increasing reliability. Proceedings of the 12th International Conference on Information Technology-New Generations (ITNG), April 13-15, 2015, IEEE, Las Vegas, Nevada, ISBN: 978-1-4799- 8828-0, pp: 796-799.

Boranbayev, S., S. Altayev, A. Boranbayev and A. Nurbekov, 2014. Mathematical model for optimal designing of reliable information systems. Proceedings of the 8th IEEE International Conference on Application of Information and Communication Technologies (AICT), October 15-17, 2014, IEEE, Astana, Kazakhstan, ISBN:978-1-4799-4120-9, pp: 1-5.

Krus, D. and K. Grantham, 2013. Failure prevention through the cataloging of successful risk mitigation strategies. J. Fail. Anal. Prev., 13: 712-721.

Krus, D.A., 2012. The risk mitigation strategy taxonomy and generated risk event effect neutralization method. Ph.D Thesis, Missouri University of Science and Technology, Rolla, Missouri.

Lim, B., C. Yeon-Choon, L. Choel, C. Sungguk and J. Byungkook, 2017. An advanced voice recording and control system with fault-tolerance for high reliability and stability. J. Eng. Appl. Sci., 12: 2855-2860.

Lough, K.G., R. Stone and I.Y. Tumer, 2009. The risk in early design method. J. Eng. Des., 20: 155-173.

Lough, K.G., R.B. Stone and I. Tumer, 2006a. Prescribing and implementing the Risk in Early Design (RED) method. Proceedings of the ASME 2006 18th International Conference on Design Engineering Technical Conferences and Computers and Information in Engineering, September 10-13, 2006, American Society of Mechanical Engineers, Philadelphia, Pennsylvania, USA., pp: 431-439.

Lough, K.G., R.B. Stone and I. Tumer, 2006b. The risk in Early Design (RED) method: Likelihood and consequence formulations. Proceedings of the 2006b ASME 32nd International Conference on Design Engineering Technical Conferences and Computers and Information in Engineering, September 10-13, 2006, American Society of Mechanical Engineers, Philadelphia, Pennsylvania, USA., pp: 1119-1129.

Vucovich, J.P., R.B. Stone, X. Liu and I.Y. Tumer, 2007. Risk assessment in early software design based on the software function-failure design method. Proceedings of the 31st Annual International Conference on Computer Software and Applications COMPSAC Vol. 1, July 24-27, 2007, IEEE, Beijing, China, pp: 405-412.