Journal of Theoretical and Applied Information Technology
31st January 2021. Vol.99. No 2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

SPECIFIC FEATURES, REQUIREMENTS, ATTRIBUTES AND METRICS OF SAFETY OF SOFTWARE FOR SPACE PURPOSES

[1]E.E. ISMAIL, [1,2]N.K. UTELIEVA, [1]R.Z. MULAEV, [1]T.S. AUELBEKOV

[1]NJSC "Gumarbek Daukeev Almaty University of Power engineering and Telecommunications" Almaty, Republic of Kazakhstan, e.ismail@aues.kz , nurshatu7@mail.ru, rus.mulaev@mail.ru, temirlan261096@gmail.com
[2]Al-Farabi Kazakh National University, Almaty, Republic of Kazakhstan, nurshatu7@mail.ru

ABSTRACT

The purpose of the work is to analyze the security features of software for space purposes (SWSP), justify security requirements, determine the main attributes, indicators and metrics of security characteristics. An analysis of the features and requirements for SWSP safety is made. The main objects and threats to SWSP safety have been identified. An attribute model of SWSP safety is proposed, the main attributes are identified and a set of metrics is proposed for a quantitative assessment of the safety level. The results obtained create the prerequisites for the formalization and objective solution of the problem of quantitative assessment of the safety level of SWSP.

Keywords: *Software For Space Purposes Safety, Features, Requirements, Model, Attributes, Metrics*

1. INTRODUCTION

In modern space systems and complexes, software is an important and integral part. The software, as a rule, controls, works or interacts with equipment, electronics, tools, the malfunctioning of which (failure) can lead to human victims, accidents or other types of dangers. Many examples can be cited of the loss of expensive rocket and space technology or catastrophic situations as a result of insufficient attention to the safety of software for rocket and space technique.

It should be noted, the safety - this property of the system. The software itself can not harm a person, material damage to the system or the environment. However, the software, which is one of the main components of the system, can only contribute to its safety. That is, software safety contributes to the safety of the system as a whole.

In this regard, the safety of space systems software is becoming a vital issue and an urgent task. The solution to this problem requires new approaches to the development, analysis, verification and testing of software for space purposes (SWSP), taking into account the priority of the safety factor.

The growing complexity and, as a result, the vulnerability of hardware-software complex and software for accidental and unforeseen negative impacts make SWSP safety a critical issue in the space industry.

«The theoretical, methodological provisions, models and standards developed to date do not fully reflect the problems of software safety analysis and assessment. There is no unified methodological approach to assessing quantitative indicators of SWSP safety.

The multi-connectivity and consistency of the software safety problem leads to the need to develop formal approaches to defining software security concepts, as well as formal methods for analyzing its safety properties. However, at present these issues have not yet been sufficiently worked out. »

Careful specification, assessment and enhancement of the safety of space systems and their software is a key factor in increasing their effectiveness. This can be achieved by highlighting, defining and providing the required characteristics of safety properties and attributes using standardized and formalized methods.

The series of modern software quality standards ISO/IEC 25000 (SQuaRE) defines the concept and general methodology for describing and evaluating the quality of general-purpose software products. However, they do not define specific indicators, metrics, measurement methods that would allow us to evaluate the established quality characteristics, especially for critical

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

software. In this connection, it becomes necessary to establish their own quality models for specific software, taking into account their features, metrics and methods for measuring the priority attributes of quality characteristics, such as security.

The purpose of this article is to analyze SWSP safety features, justify SWSP safety requirements and determine the main attributes, indicators and metrics of software safety characteristics in accordance with the requirements of the software quality assessment methodology of international standards ISO/IEC 25000.

2. THE CONCEPT, MODEL, OBJECTS AND THREATS TO SOFTWARE SAFETY

In a broad sense, the concept of safety is generally defined as the absence of unacceptable risk associated with the possibility of harm and (or) damage. In the narrow sense, as applied to software, software safety can be defined as the property of a software product to keep in time within the established limits the values of the admissible risk of harm and (or) damage under given conditions of use.

In the ISO/IEC 9126-1 standard, the characteristic "Safety" was defined as "The capability of the software product to achieve acceptable levels of risk of harm to people, business, software, property or the environment in a specified context of use" [1].

The ISO/IEC 25010 standard, adopted instead of the ISO/IEC 9126-1 standard, instead of the "safety" characteristic of a software product introduces the characteristic "freedom from risk" - degree to which a product or system mitigates the potential risk to economic status, human life, health, or the environment [2].

Comparison of these definitions shows that the characteristics "Safety" and "freedom from risk" are almost equivalent.

In [3], software safety is considered as a property of this software to function without manifesting various negative consequences for a particular computer system.

Also, in some sources [4], the safety of a computer system (CS) is considered in two aspects:

1) software and data security in a computer system;

2) the safety (reliability, dependability) of the functions of a computer system or the functional component of safety.

In this regard, for the second component of safety associated with the proper functioning of a computer system, the concept of "functional safety" is introduced, which is defined as follows:

- software functional safety - the ability of the software to achieve an acceptable level of risk of failure situations and loss of performance due to unintentional, accidental defects in specific application conditions [5];

- functional safety of software - the ability of software to achieve an acceptable level of risk to human health, property or the environment in a given application context [6]

- functional safety - the ability to exclude or minimize for users, other systems and the environment, dangerous (including catastrophic) consequences in case of failures for users, other systems and the environment [7].

Analysis and summarizing of various definitions of software safety used in the literature, and also taking into account the fact that the concept of "risk" is used as a generally accepted safety measure, which is a function of time, it is proposed to consider "software safety" as the degree of software's ability to save over time in permissible limits, the level of risk of harm and (or) damage to a certain level (category) in the face of identified safety threats in specified conditions of use for a certain period of operation. Moreover, for each type of risk there may be defined acceptable risk, risk category and combination of hazardous events.

Safety as a system property includes a safety object, safety threats and their consequences, safety methods and means, safety indicators and measures, the relationship and relationships between which are described by a safety model. A software safety model is understood as a structured set of interconnected objects, entities, threats to the safety of characteristics and relations between them, which form the basis for the specification of quality requirements and quality assessment of a software product.

Software safety is associated with protection against fault situations and loss of system performance due to unintentional, accidental defects and fault of the software, or not correctly executed of the software, data corruption, hardware faults and failures, the negative impact of the environment in which the program operates, etc.

Software safety objects are affected by various unintentional destabilizing threats (risk factors), which can be divided into internal ones related to defects in software development and disruptions to its technological processes, and external ones caused by the environment in which this product operates.

Internal sources of software safety threats

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

include:

- errors in determining safety requirements;
- errors in defining safety functions;
- errors in the analysis of dangers and risks;
- errors in determining the category of criticality, policies and safety standards;
- errors in determining the requirements for methods and means of ensuring safety;
- defects and errors in determining the functions, conditions and parameters of the external environment in which software (system) should be applied;
- design errors for safety functions;
- programming errors and defects in program texts and data descriptions, as well as in the initial and resulting documentation for software components;
- insufficient effectiveness of the methods and means used to ensure the safety of functioning and restoration of the system's operability in the conditions of random and deliberate negative influences from the external environment.

External sources of safety threats that pose safety threats to the functioning of the software during its operation usually include:

- deliberate, negative influences of subjects with the aim of distorting software modules, data and documents critical to safety functions;
- failure or inadequate performance of safety critical functions;
- critical errors and unauthorized actions of operational, administrative and maintenance personnel during the operation of the system and software;
- false positives, multi-task blocking of safety functions;
- distortion in the telecommunication channels of critical information coming from external sources and transmitted to consumers;
- invalid or unreliable values and changes in the characteristics of the measured signals and data from the external environment;
- failure (temporary disruption of the software), blocking in multitasking mode, failure of critical data conversion functions;
- critical failures and failures in hardware and software tools interacting with software;
- viruses, interference, crashes and failures affecting software safety, distributed through telecommunication channels.

External sources of safety threats that pose safety threats to the functioning of the software during its operation usually include:

- deliberate, negative influences of subjects with the aim of distorting software modules, data and documents critical to safety functions;
- failure or inadequate performance of safety critical functions;
- critical errors and unauthorized actions of operational, administrative and maintenance personnel during the operation of the system and software;
- false positives and blocking safety functions in multitasking mode;
- distortion in the telecommunication channels of critical information coming from external sources and transmitted to consumers;
- invalid or unreliable values and changes in the characteristics of the measured signals and data from the external environment;
- failure (temporary disruption of the PS), blocking in multitasking mode, failure of critical data conversion functions;
- critical failures and failures in hardware and software tools interacting with software;
- viruses, interference, crashes and failures affecting software security, distributed through telecommunication channels.

Based on the analysis of threats and potential dangers for software safety, the following main safety objects can be distinguished:

- dangers (threats), defects and failures that can cause a situation critical to safety;
- safety requirements;
- safety-critical functions;
- design solutions for architecture and safety functions;
- design solutions for defects, errors, failures and failure tolerance;
- software modules, components that implement safety functions;
- software modules, components that generate data used for decision-making in safety-critical systems;
- software modules, components that monitor the state of hardware components, the malfunctioning of which can lead to an unacceptable risk of a dangerous situation;
- management and software development processes;
- verification and testing processes;
- risk management processes;
- defects and failures of the hardware and software environment interacting with the software, which can lead to inadequate program execution, adversely affect safety functions, and contribute to a dangerous situation;
- staff errors.

An inseparable link should be taken as the basis for the study of safety issues of the safety

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

control system: safety object - threats to it. A large number or uncertainty of safety objects determines a huge number of factors, conditions that constitute or may constitute a safety threat.

3. FEATURES AND GENERAL REQUIREMENTS FOR THE SAFETY OF SWSP

Under the software for space purposes hereinafter refers to software designed for use in space systems and various objects of space technology.

By belonging to the objects of space technology, the following main types of PSKN can be distinguished:

- for airborne control systems for manned and automatic spacecraft;
- for airborne computing systems of launch vehicles, space tug;
- for technical and launch complexes, ground-based automated spacecraft control complexes, ground-based equipment and structures;
- for payloads;
- for experiments and simulations.

The affiliation SWSP of space technique objects defines specific requirements, for example, the safety requirements for the software of the spacecraft control system, of launch vehicles, space tug and payloads differ significantly from the safety requirements SWSP for experiments and modeling.

The software, which is part of the on-board systems, traditionally has high requirements for reliability and safety.

Features and requirements for SWSP depend on the level of criticality of the functions it implements for the safety of the system of which it is a part.

The criticality category or software safety level is determined by the severity of the consequences of abnormal functioning in the following areas: “threat to human health and life - environmental damage - material losses”, taking into account the likelihood of their occurrence.

ECSS-Q-ST-80C [7] defines 4 critical categories of space-based software based on the severity of the consequences of system failures. The table shows the main features of the software criticality categories and recommendations for assigning them to these categories.

In accordance with the ECSS-Q-ST-80C standard, critical software includes software that supports the critical safety or reliability function, which in the event of improper or inadvertent operation can lead to catastrophic or critical consequences [8]. The same standard establishes that critical software includes software of categories A, B or C (Table 1).

The NASA Standard “Software Safety Guidebook” [9] defines “Safety-Critical Software” as software (software systems) whose failure or improper operation can lead to catastrophic or critical consequences (loss of life, threat to their lives, significant damage to property or the environment). Sometimes the same term refers to software that is developed in accordance with special standards designed for mission-critical areas.

Table 1 - Software For Space Purposes Criticality Categories

Criticality category	The main characteristic of the category of criticality	Recommendations on referring to the category of criticality
A	Software that if not executed, or if not correctly executed, or whose anomalous behaviour can cause or contribute to a system failure resulting in: - *Catastrophic consequences* (death of people, threat to their life, destruction, loss of property, equipment);	*Category A* should include software that is part of the elements of the system that perform the basic functions of ensuring or maintaining safety, prevent the occurrence of failure situations with catastrophic consequences, or mitigate the consequences of these events.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

B	Software that if not executed, or if not correctly executed, or whose anomalous behaviour can cause or contribute to a system failure resulting in: - *Critical consequences* (damage not threatening the life of people, significant damage to equipment, harmful effects on the environment);	*Category B* should include software that perform additional functions to ensure or maintain safety functions, prevent failure situations with critical consequences, or mitigate the consequences of these events, and monitor the operability of category A.
C	Software that if not executed, or if not correctly executed, or whose anomalous behaviour can cause or contribute to a system failure resulting in: - *Major consequences* (moderate decrease in the ability of the controlled object or the ability of personnel to cope with adverse situations);	*Category C* should include software that performs auxiliary safety functions, prevents failure situations with significant consequences, or mitigates the consequences of these events, monitors or records the status of system elements, and determines their safety status.
D	Software that if not executed, or if not correctly executed, or whose anomalous behaviour can cause or contribute to a system failure resulting in: - *Minor or Negligible consequences* (a slight decrease in the safety of the control object and requires personnel actions that are feasible within their capabilities).	*Category D* should include software whose abnormal behavior can cause (or contribute) to the occurrence of a malfunction of the system, leading to an insignificant failure situation.

One of the main requirements for SWSP critical to safety is the requirement of a safe failure, which means that any possible failure of the SWSP should not cause a dangerous situation (there is a potential risk of a situation with a threat of damage).

One of the main safety requirements for SWSP is the requirement of safe failure, which means that any possible failure of a safety control system should not cause a dangerous situation (the presence of a potential risk of a situation with a threat of damage).

The criticality category (or safety level) of the SWSP determines the scope of requirements related to safety and risk reduction. The higher the criticality category of the SWSP, the greater the scope of additional safety requirements. In this case, the number of safety requirements may exceed the number of functional requirements for SWSP. It is well known that the implementation of additional requirements leads to a significant increase in the complexity and amount of program code.

For example, the NASA standard [9] establishes three basic levels of critical software safety: “full”, “moderate”, “minimum”.

The “full” safety level includes software of the systems and subsystems with a high level of criticality of hazards (for example, catastrophic or critical with a high probability), which can lead to software failures in a very short period of time (insufficient to respond and prevent a dangerous situation) . For this category of software, maximum efforts and resources are required to ensure safety [9].

The “moderate” safety level includes software of systems and subsystems with limited severity of the consequences of potential danger (for example, for dangers of a critical level with low probability or moderate level of criticality), which can lead to software failures, or with the time of occurrence of a dangerous situation, sufficient to respond to it, prevent failure and initiate hazard management [9].

The “minimum” safety level includes software of systems with a low level of potential danger (for example, for dangers of a moderate criticality level with a low probability), which can result in software failures, or in which hazards are controlled by non-software tools [9].

The software safety category (level) it is customary to establish based on the risk

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

assessment, customer requirements, adopted safety policies and resource limits for safety. This is a rather complicated task, for the solution of which it is necessary:

- identify and analyze the possible dangers that a software product may create;
- assess the severity levels of hazards (NASA classification: catastrophic, critical, moderate, insignificant);
- assess the potential for hazards;
- assess the risks of hazards;
- assess the degree to which the software used should reduce the risk of a hazard (mitigate the risk of danger).

Depending on the adopted safety category, software sets requirements for safety functions, necessary measures, processes, checks and tests, as well as requirements related to verification, validation, their independence and levels of evidence.

4. ATTRIBUTES AND METRICS OF THE SWSP SAFETY

Direct measurement of software safety is not possible. To assess the safety of SWSP and establish compliance of safety indicators with the requirements of technical specifications and specifications, it is necessary to establish a set of measured indicators, special metrics and ensure their measurement with the required objectivity, accuracy and reliability. One approach to assessing SWSP safety is to evaluate the relevant safety-related attributes set in the software quality model in the international standard ISO/IEC 25010.

Software safety in accordance with the ISO/IEC 25010 standard is evaluated by the «Freedom from risk» characteristic, which is part of the quality-in-use characteristics that characterize the quality of the software during normal operation and the effectiveness of satisfying user needs.

To assess the “Freedom from Risk” characteristic, standard subcharacteristics (attributes) [2] are used, which allow obtaining a generalized assessment of the risk of harm to users, economic and environmental risks, these are:

- the degree of reduction of economic risk (economic risk mitigation);
- the degree of risk reduction for the health and safety of people (health and safety risk mitigation);
- the degree of reduction of environmental risks (environmental risk mitigation).

However, an analysis of the safety risks of critical software shows that a violation of the safety of its operation can also lead to damage to other software and data (program files, databases), the use of which by the system or other applications can produce unexpected results, the consequences of which may be critical for the safety of the system in whole. For example, software and data used to make decisions on ensuring the safety of the system, as well as those that could have a negative impact on safety functions, etc.).

Therefore, to evaluate SWSP safety, it is proposed to use another additional subcharacteristic (attribute) - the degree of reduction of the potential risk of data and software corruption in the intended use contexts.

Also, in order to simplify the assessment of the degree of risk reduction for the health and safety of people, it is proposed to divide this attribute into two: the degree of risk reduction for the health and safety of operating personnel and the degree of risk reduction for the safety of people who could potentially suffer from the use of the system.

In view of the foregoing, an attributive model of the “Freedom from Risk” characteristic of the SWSP presented in Figure 1 is proposed.

The proposed model of the “Freedom from Risk” characteristic can be modified depending on the criticality category of the SWSP, especially safety threats in specific application conditions, by deriving or introducing additional attributes and changing requirements for them.

Journal of Theoretical and Applied Information Technology
31st January 2021. Vol.99. No 2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

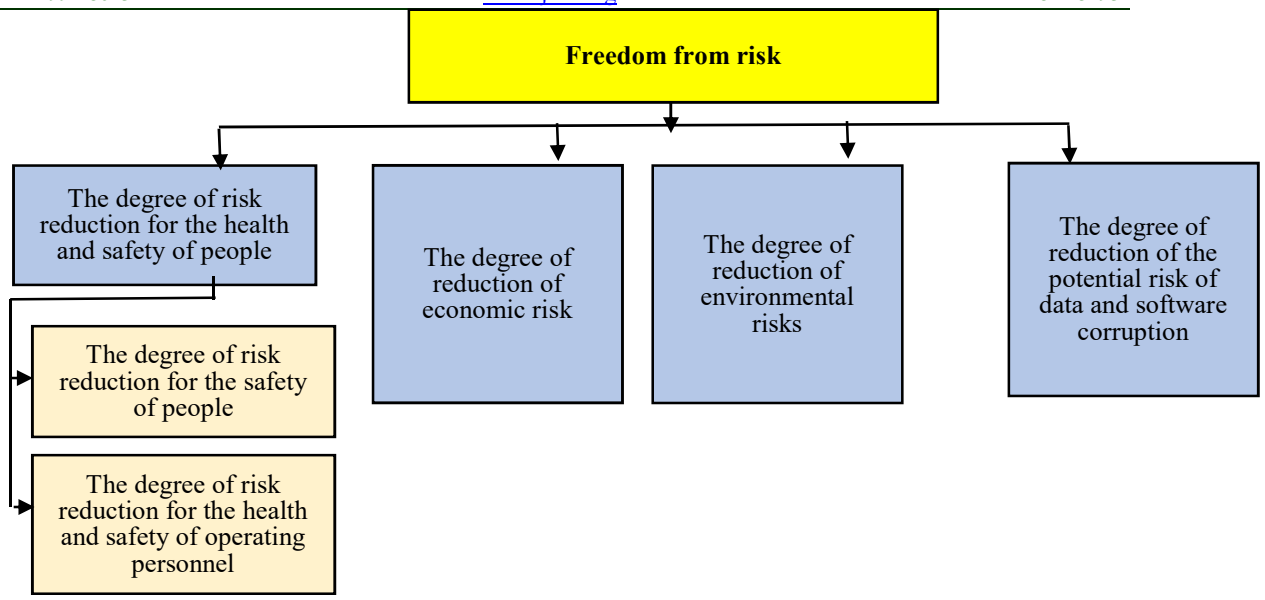

Figure 1- Attributive model of the characteristic "Freedom from risk" of SWSP quality

As a numerical measure of software SWSP assessment, it is advisable to use a universally accepted universal safety indicator - risk, which is a function of the probability of a dangerous event and the severity of its consequences over a specific period of time.

In relation to software safety, risk is a function of the probability of an unwanted event occurring in the software (security violation, fault, failure) and the potential severity of the consequences and uncertainties resulting from this. Therefore, in order to assess software safety risk, it is necessary to first identify and record dangerous events and potential damage from it.

In the general case, the risk *R(t)* over time *t* associated with some dangerous event is defined as the product of the probability of this event *P(t)* with the potential damage *C*, the consequences arising from this event

$$R(t) = P(t) \cdot C. \quad (1)$$

It is well known that it is impossible to directly measure risk. Therefore, for a quantitative risk assessment, the concept of the frequency of risk realization is usually used, which is determined by the ratio of the number of negative consequences of a specific hazard to their possible number over a certain period of time.

$$R(t) = N(t)/\cdot Q(f), \quad (2)$$

where *N(t)* - is a quantitative indicator of the frequency of adverse events over time *t*; *Q(f)* - is the number of risk objects exposed to a particular risk factor *f*.

For the quantitative assessment of SWSP safety attributes (indicators) defined in the model in Figure 1, it is necessary to establish appropriate metrics, which is understood as a numerical measure that allows you to quantify certain properties of a software product. The metric is also characterized by the scale and the measurement method used.

It is advisable to establish SWSP safety metrics and methods for measuring them, taking into account its features and application conditions, as well as the established criticality category.

Also, when choosing software safety attribute evaluation metrics, you must be guided by the following well-known rules:

- the metric should have the same meaning for both the customer and the developer;
- the metric should be objective, and its definition is unambiguous;
- the metric should provide the ability to track the trend of change;
- the metric should provide the ability to automate the assessment process.

Based on the analysis of the SWSP features for the quantitative assessment of safety attributes (indicators) in accordance with the above model, the metrics and methods for measuring them are presented in Table 2. These metrics comply with the requirements of the international standard ISO / IEC 25010.

When using the SWSP safety metrics in Table 2, it is necessary to ensure the objectivity and reproducibility of measurements. The results of the assessment of safety indicators should have accuracy and certainty, which are sufficient for comparison with the requirements of specifications.

5. CONCLUSION

In accordance with the set goal of the work, the following main results were obtained, which determine its scientific novelty:

- a new approach to the analysis and assessment of SWSP safety is proposed based on the generalization of the classical theory of information technology safety and the methodology for assessing the quality of software, recommended by the family of international standards ISO / IEC 25000;

- the definition of the concept of "software safety" is proposed on the basis of generalization of the definitions of information technology safety theory and the concept of functional safety of the system;

- taking into account the features of the SWSP, the main safety objects and sources of safety threats were identified, and an attributive model of SWSP safety was built;

- for a quantitative assessment of the safety of the SWSP, a set of measured indicators, numerical measures and measurement methods are proposed.

The proposed approach to the analysis and assessment of the safety of the SWSP allows one to obtain an integral quantitative assessment of the safety level of the SWSP taking into account the category of its criticality and specific conditions of use.

To evaluate the integral indicator of the SWSP safety level, a linear functional can be used, the components of which are normalized values of attributes and metrics with corresponding weighting factors. The choice of weights depends on the characteristics of a particular SWSP and the conditions for its use.

The results obtained create the preconditions for the development of formal methods of analysis and assessment of software safety.

REFERENCES

[1] ISO/IEC 9126-1:2001. Software engineering – Software product quality – Part 1: Quality model. – 2001. – 32 p. (en)

[2] ISO/IEC 25010:2011 Systems and software engineering. Systems and software Quality Requirements and Evaluation (SQuaRE). System and software quality models.- (http://www.iso.org/iso/home/search.htm?qt=ISO%2FIEC+25010%3A2011+&sort=rel&type=simple&published=on) (en)

[3] Kazarin O.V. Bezopasnost' programmnogo obespecheniya komp'yuternykh sistem.- M.: MGUL, 2003.- 212 s. (ru)

[4] Kulikov S. S. Modeli bezopasnosti komp'yuternykh sistem: ucheb. posobiye [Elektronnyy resurs]. – Voronezh: FGBOU VPO «Voronezhskiy gosudarstvennyy tekhnicheskiy universitet», 2015, 179 s. (ru)

[5] Lipayev V.V. Funktsional'naya bezopasnost' programmnykh sredstv //Jet Info, 2004. -№ 8 (135). - s. 1-28. (ru)

[6] SOU-N NKAU 0058: 2009 Galuzeva systema upravlinnja jakistju. Vymogy do funkcional'noi' bezpeky programnogo zabezpechennja programno-tehnichnyh kompleksiv krytychnogo pryznachennja. Nastanova nacional'nogo kosmichnogo agentstva Ukrai'ny [Tekst] / Harchenko V. s. (Nauk. Kerivnyk rozrobky). - 2009. - 57 s. (ukr)

[7] ECSS-Q-80C-2003 Space Product Assurance: Software Product Assurance.- (https://ecss.nl/standard/ecss-q-80b-software-product-assurance-10-october-2003/) (en)

[8] ECSS-Q-ST-80C-2009 Space Product Assurance: Software Product Assurance.- (https://ecss.nl/standard/ecss-q-st-80c-rev-1-software-product-assurance-15-february-2017/)

[9] NASA-GB-8719.13 NASA Software Safety Guidebook.- (https://standards.nasa.gov/standard/nasa/nasa-gb-871913) (en)

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table 2 - SWSP safety Security Metrics

Name of the metric	Purpose of the metric	Used method	Measured quantities, formulas for calculating elements	Interpretation of the measured value of the indicator
The degree of reduction of economic risk (economic risk mitigation)	Assessment of the degree of reduction of the potential risk of economic damage in the intended contexts of use	Collection and processing of statistics on fatal software failures, their distribution according to the level of severity of economic damage	Degree of mitigation of risk $X = (A_i * C_i / B) / R_{iac}$ A_i - the number of cases of economic damage of the *i*-th category; B - total number of software usage situations; C_i – allowable size of economic damage to the *i*-th category; R_{iac} – acceptable risk of economic damage of the *i*-th category.	0 <= X <= n The closer to 0, the better.
The degree of risk reduction for the health and safety of operating personnel (for SWSP criticality category - A)	Assessment of the degree of reduction of potential risks to the health and safety of operating personnel in the intended contexts of using the system.	Collection and processing of statistics on catastrophic software failures, their distribution according to the severity level of harm to the health and safety of users	Degree of mitigation of risk $X = (A_i * C_i / B) / R_{iac}$ A_i - the number of users exposed to danger; B - total number of users; C_i – allowable size of damage of the *i*-th category; R_{iac} – acceptable risk of harm to the health and safety of people of the *i*-th category.	0 <= X <= n The closer to 0, the better.
The degree of risk reduction for the safety of people who could potentially suffer from the use of the system (for SWSP criticality category – A)	Assessment of the degree of reduction of the potential risk to the safety of people who could potentially suffer from using the system	Collection and processing of statistics on catastrophic software failures, their distribution according to the level of severity of harm to human health and safety	Degree of mitigation of risk $X = (A_i * C_i / B) / R_{iac}$ A_i - the number of people exposed to danger; B - total number of people potentially covered by the system; C_i – allowable size of damage of the i-th category; R_{iac} – acceptable risk of harm to the health and safety of people of the *i*-th category.	0 <= X <= n The closer to 0, the better.
The degree of reduction of the potential risk of data and software corruption	Assessment of degree of reduction of potential risk of software and data damage	Collection and processing of statistics on software failures, with the consequences of damage to other software and data	Degree of mitigation of risk $X = (A * C / B) / R_{ac}$ A - the number of cases of damage to other software and data when using SWSP; B - total number of SWSP usage situations; C – the average size of the damage caused by harm to other software and data; R_{ac} – acceptable risk of damage from harm to other software and data.	0 <= X <= n The closer to 0, the better.