UDC 681.2: 621.3.082.1: 531.75.08

Kozlyuk I.O.
orcid.org/0000-0001-8239-8937
Kovalenko Yu.B.
orcid.org/0000-0002-6714-4258

FUNCTIONAL BASES OF THE SOFTWARE DEVELOPMENT AND OPERATION IN AVIONICS

National Aviation University

avia_ira@ukr.net
yleejulee22@gmail.com

Introduction

Many flight control systems in civil aviation have been successfully developed with the help of special equipment and software. The basic principles of computer software and hardware for avionics require detailed research as, in particular, theoretical and methodological principles of software development and practical application.

Substantiation of the basic principles of software in avionics is evolving as research is conducted and experience is gained in automation and software support architecture. Specificity in the field of structural analysis defines the term architecture, which describes the typology of methods, their functionality and the relationship of the components of the system being modeled.

Functional principles of software development and operation determine the analysis and development of various solutions for improvement and innovation of aviation products and aviation enterprises in particular. These principles are formed on the ideas of integration, which is part of a comprehensive analysis of business processes.

Thus, the relevance of the article is the study of the avionics system, which has a modular multilevel structure. At the same time, classical approaches to the development of aircraft operation systems are inefficient, which requires the development of a specific methodology that takes into account the subject area under study. This is due to the need to integrate software for the operation of civil aircraft with third-party software and hardware systems used for flight control.

The following practically significant constructions and components are of great importance in this context: the software architecture and the subsequent V-shaped model, as well as the unity and integrity between the structural components.

Literature analysis and problem statement

In [1, 2] the results of research on the development of a system for supporting air traffic control solutions are presented. Such system requires an iterative design process. A separate design subsystem performs functions aimed at standard platforms that can be used in several types of applications simultaneously. In our study, we use the sharing of multiple applications and the reuse of computing resources, which primarily affects the practical and research part of our research.

In [3] the results of the study of certification of processing systems in avionics and some issues of integration activities and analysis of software programs are presented. This allowed us to formulate requirements to a visualization platform that supports several sections of a real-time operating system (RTOS) on a multi-core processor used in avionics.

In [4], the results of the study of the compliance with the requirements based on the search for evolutionary calculations (CEC) are presented. It is stated that requirements monitoring is a multi-purpose search task, due to the purpose of each requirement

to one or more software elements (code elements, API documentation and comments). This allows us to justify the purpose of a non-dominated sorting genetic algorithm (NSGA-II).

In [5] the results of the study of the architecture of AECS Airbus A-380, Boeing 787, Sukhoi Superjet 100 and MC-21 are presented. It is stated that these systems are designed with a similar design and contain central data processing and transmission. Interfaces of ARINC 825, ARINC 664, and ARINC 429 are used for data exchange between them. However, subsequent upgrades of such systems are limited due to low flexibility outside the network distribution. Therefore, it can be concluded that it is appropriate to conduct a research on the functional principles of software development and operation in avionics, taking into account the urgent issues of system reliability, data transmission, noise reduction, weight reduction and dimensions of the cable network.

With the development of integrated modular avionics (IMA), dynamic reconfiguration not only provides benefits in terms of resource utilization and aircraft configuration, but is also used as an effective failure management tool. In [6-7] the results of the study of the dynamic reconfiguration processes and its elements in avionics are presented. The systems of the Petra network, which is widely used as a modeling tool, have been studied. This method of validation of dynamic IMA reconfiguration is preferable to start designing the system.

Among the promising standards for the development and further operation of software in aviation and, in general, in the aerospace industry there is the SPACE standard. The main purpose of this standard is to ensure the compatibility of different types of software, data transmission and information management to solve specific problems in the aerospace industry [8]. The issues related to the optimization of this software development require further study, which should be held taking into account current trends in the development of information technology.

However, the issues related to the creation of software packages for the management of certain processes in civil aviation (including flight control) are insufficiently studied. It is so due to the specifics of this industry and the trend according to which the dominant place in the market of IT technologies belongs to foreign companies [9, 10].

In practice, aerospace equipment suppliers are interested in using multi-core processors (MCPs) in their systems. Multiple cores integrated into a single device allow combining multiple functions on a single processor and on the same hardware. The development and use of the most sophisticated equipment by the aviation industry to perform important software functions of aircraft raises new issues of safety and certification [11].

The problem of determining the functional principles of software development and operation in avionics is based on fundamental elements of different practices in the software solutions application.

In particular, the system of application of dynamic reconfiguration and other modular designs also require the use of functional and generalized principles in operation, which will avoid many shortcomings and malfunctions, and so on.

The purpose and objectives of the study

The purpose of the study is to substantiate the functional principles of software systems, including the structure of the optimal cycle of software development for civil aviation flight control.

Achieving the above-mentioned goal involves solving the following tasks:

- To identify the main components of the software development process, including the functional aspects of the development of applied software solutions for the operation of aircraft in the field of civil aviation;
- To characterize the elements of the system approach application to the software and hardware functionality in avionics.

Identification of the main components of the software development process and functional aspects of the development of applied software solutions in civil aviation

In recent years (2018-2020), a number of architectures and standards for the development of IMA that use the requirements of the 6532 ARINC specification have emerged [15]. The ARINC specification 653 defines at a high-level an example of software for the IMA architecture. The widespread acceptance and support of ARINC 653 is also evident in the Future Airborne Capability Environment (FACE) of the United States military avionics programs. These and other IMA standards place new demands on the software architecture, especially the implementation of RTOS (Real-Time Operating System) provided by the COTS (Commercial off-the-shelf) provider [15].

The development of avionics software is based on the basic standard RTCA/DO-178B [18]. Despite its separation from the real of software development, this standard describes the entire development process and makes requirements for such software. To develop critically important avionics systems, it is necessary to ensure the minimum error possibility (for the highest level in avionics, the probability of failure is set) [19, 3], as well as to minimize the cost of developing and correcting the code. Due to the system complexity and its relationship with complex hardware (or other software), in the field of avionics a V-shaped model of development should be used, taking into account the key components in creating software (Fig. 1).

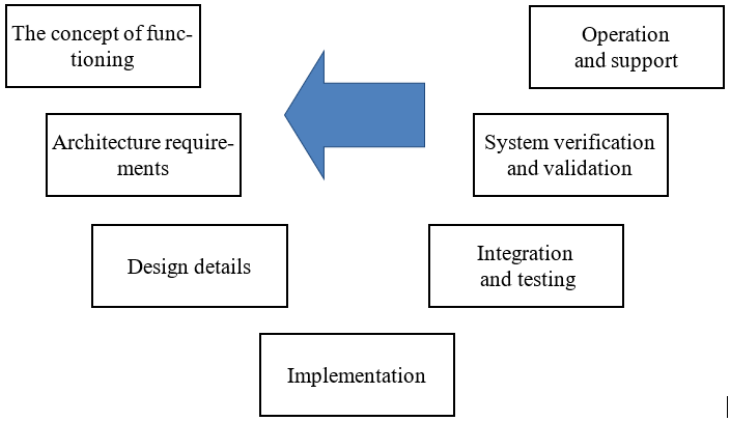

Fig. 1. V-shaped model of software development for aircraft operation

First of all, the V-shaped model allows to provide synchronization of all participants of the project on each iteration, and also gives the chance to use already developed data and methodology. Secondly, it can be adapted to any project and does not depend on the types of organizations and projects (Fig. 1). Thirdly, the V-model allows to divide the development into separate components, each of which will include the necessary actions, instructions, recommendations and a detailed explanation of the activity. This is especially important for the multi-iterative cycle of development and avionics software testing, because it allows, in fact, to divide the software development into separate subcycles. Usually the V-model is generalized into a spiral model of development (Fig. 2).

The model allows to estimate risks at each stage of development. In addition, it also optimally distributes the specialists' workload, when there is lack of time and resources, into short periods of time (Iteration Packages synchronized with the V-shaped model in each Baseline). Modern solutions for control and monitoring of aircraft operation systems (Flight Control System) is a complicated software and hardware complex, the work of which is generally unknown to any of the employees and developers. Aviation software has nuances and typical solutions that should be emphasized [18].

Note that based on the identified relationships between the source data for the design of the primary structure of functions and

parameters that characterize the final structure of functions, as well as expressions:

$$F: (n, k_{OM}, z) \to K_{A3}, \quad (1)$$

$$F: (F1, k_{BM}, z) \to n, \quad (2)$$

$$F: (n, L) \to m. \quad (3)$$

We can conclude that the optimization of the structure of functions is to minimize each of the parameters that characterize it.

In order to estimate the changes in the structure of functions and compare them between each other, we introduce the target optimization function y:

$$y = \frac{\Delta}{\Delta n} * n + \frac{\Delta}{\Delta K_{A3}} K_{A3} + \frac{\Delta}{\Delta m} * m. \quad (4)$$

Expression (4) allows us to determine the optimal configuration of the BWC, provided that all optimization parameters have equal priorities. However, in practice, in each case, the priorities between the parameters may be different. In order for the developer to be able to carry out optimization taking into account the priority tasks of designing the BWW, we introduce the weights a, b, c, then expression (4) will take the form:

$$y = a * \frac{\Delta \max}{\Delta n} * n + b * \frac{\Delta \max}{\Delta K_{A3}} K_{A3} + c * \frac{\Delta \max}{\Delta m} * m, \quad (5)$$

where:

$$\Delta n = nmin_{max} \quad (6)$$

$$\Delta K_{A3} = \Delta K_{A3} A3_{min_{max}} \quad (7)$$

$$\Delta m = mmin_{max} \quad (8)$$

and Δmax = the largest value among Δn, ΔKAZ, Δm.

The values of the weights a, b, c are determined expertly from the range from 0 to 1. The optimal configuration of the BWW will correspond to the minimum value of y.

To control each stage and for further possibility of certification, the process is divided into different levels, each of which corresponds to its own document, for which a backlog is created, which controls it (report). As a result, each stage of development, all errors and corrections are classified and documented. Repeating each iteration of the development of the development of the probability of error is reduced. These documents are also created on the basis of internal standards of the developer and the requirements of the customer.

Components of the process of software development for the operation of wind turbines are presented in the form of a structural UML-diagram (Unified Modeling Language - unified modeling language) (Fig. 3).

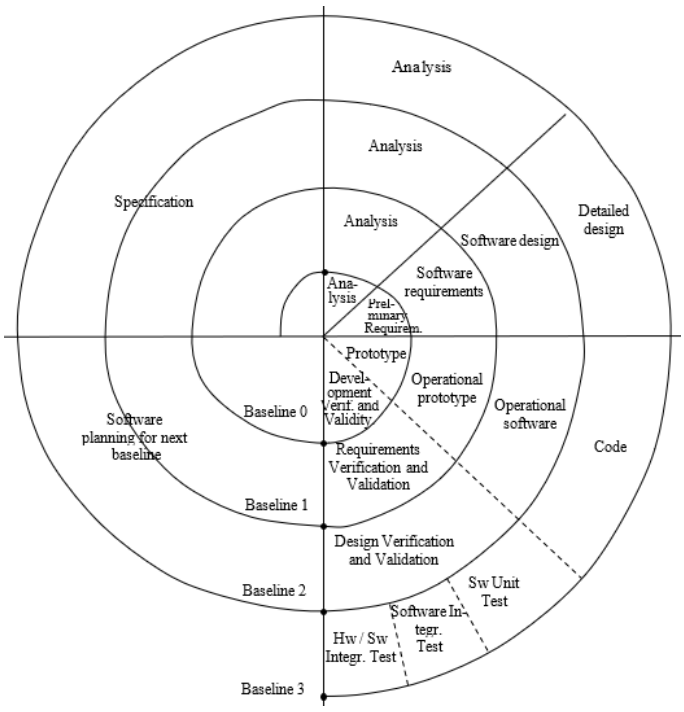

Fig. 2. Generalization of the V-model into a spiral model of software development and testing for the aircraft operation

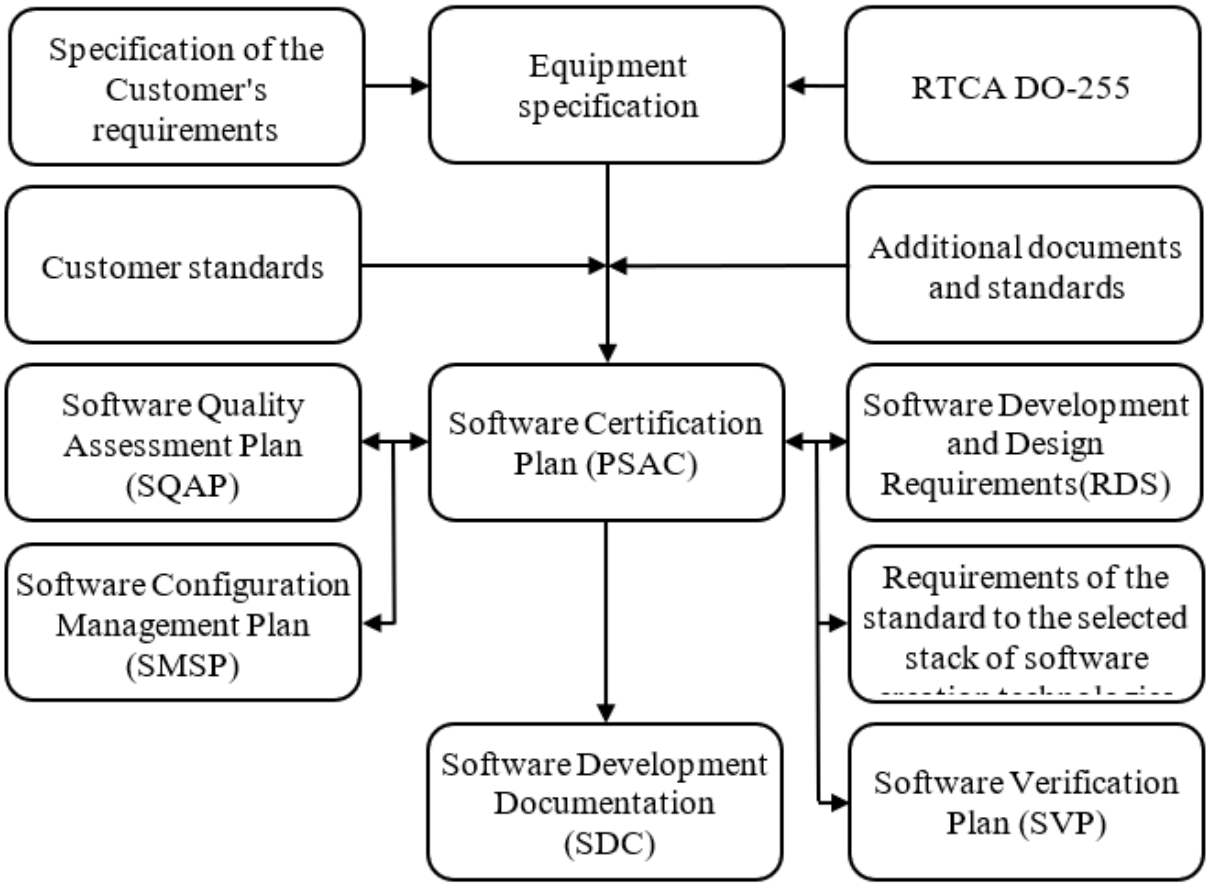

Fig. 3. Structural UML-diagram of the relationship of documents and requirements in the process of software development for aircraft operation.

To control each stage and for further certification, the process is divided into different levels. Each level corresponds to its own document. For each document a backlog is created, which controls it. As a result, each stage of development, all errors and corrections are classified and documented. Due to repeating the iteration of the development each time the probability of error is reduced. These documents are also created on the basis of the developer internal standards and the customer requirements.

The components of the aircraft software development process are presented in the form of a structural UML-diagram (Unified Modeling Language) (Fig. 3).

As can be seen from Fig. 3, the software customer is at the top of the process. Therefore, the first step is to analyze the customer's requirements and determine the basic system functionality. The general concept, software scheme, and technical details of the equipment used are based on it. That is, the creation of primary system specifications (Equipment Specification) and requirements (System Requirements).

Most of the available tools do not support automatic link tracking updates. One of the problems in software maintenance is the automatic support of tracking requirements. The process of generating requirements tracking is time consuming and error prone [4]. An option to overcome these difficulties may be the practice of certain companies to accumulate history of changes from previous service experience.

When the base on which the system will be formed is determined, the Software Development Plan and Qualification Plan (plan for Software Aspects of Certification) are approved. Despite the fact that the main thing for the customer is to get ready-made software, a parallel process is the hardware development, which cannot be ignored, in particular, because the development of software in avionics is very closely related to the hardware. Although most software is embedded code, but it is highly dependent on the layout of the systems.

Systematic approach application to the functionality of software and hardware in avionics.

During latest decade, original equipment manufacturers have considered the use of systems based on COTS (Commercial Orbital Transportation Services). At the same time, there is a tendency to move away from the systems development based on integrated architectures, where each individual subsystem performs a special function. There is a direction for common computing platforms that can be used in multiple types of applications and, in some cases, can run multiple applications at the same time. This approach, known as integrated modular avionics or IMA, results in fewer subsystems that take up less space, reduce cost, and power consumption. A number of software developers for civil aviation face some challenges in its optimization based on the use of IMA principles. Therefore, the study of theoretical and practical aspects of software development for flight control in civil aviation is relevant and significant in modern conditions and involves solving a range of technical problems.

Modern means of informatization make it possible to automate certain procedures that are performed as part of the process of organization and management of civil aviation flights. Management and control of operations, according to some authors [9], is based on the implementation of adaptive multi-hypothetical algorithms, which are evaluated on the basis of the Bayesian method. The above algorithms provide monitoring of the trajectory in a three-dimensional coordinate system.

Let's pay attention to the software formation taking into account the risk-oriented approach used in the aircraft operation, based on the following prerequisites [12]:

- Identification of threats (list of dangerous events) and factors.
- Risk assessment.
- Identification of ways to reduce risks (by the method of level management).
- Assessment of economic costs to reduce risks to an acceptable level.
- Creation of a posteriori (active) and a priori (proactive) expert knowledge base for risk assessment in order to make proactively corrective decisions aimed at reducing risks.
- Preventive (proactive) danger forecasting.
- Risks identification using the concept of "threat" and "danger" as a possible proactive prediction of the system in possible dangerous situations by type of threat.

Based on the above, we can conclude that the author proposes to design automated flight control systems based on the risk tolerance criteria. However, the difficulty of using such an approach is that the software development process must meet the criteria of the RUP (rational unified programming) and XP (extreme programming) approaches. And, in addition, should take into account at the initial stage of determining the test cases on the basis of which the software will be created ("development through testing" program creating approach) [12].

Software efficiency is estimated by such important criteria as modularity, ease of integration, ease of operation and adaptability of the system. Software standardization determines the manufacturability of the PNP procedure. Plug and Play technology was primarily an industry standard for automatic processing and operation of software, which simplified the operation and interaction of programs and their components [6].

However, in case of errors in IMA systems, response mechanisms, such as component failure, etc., will be the first to respond, as in dynamic reconfiguration (Fig. 4).

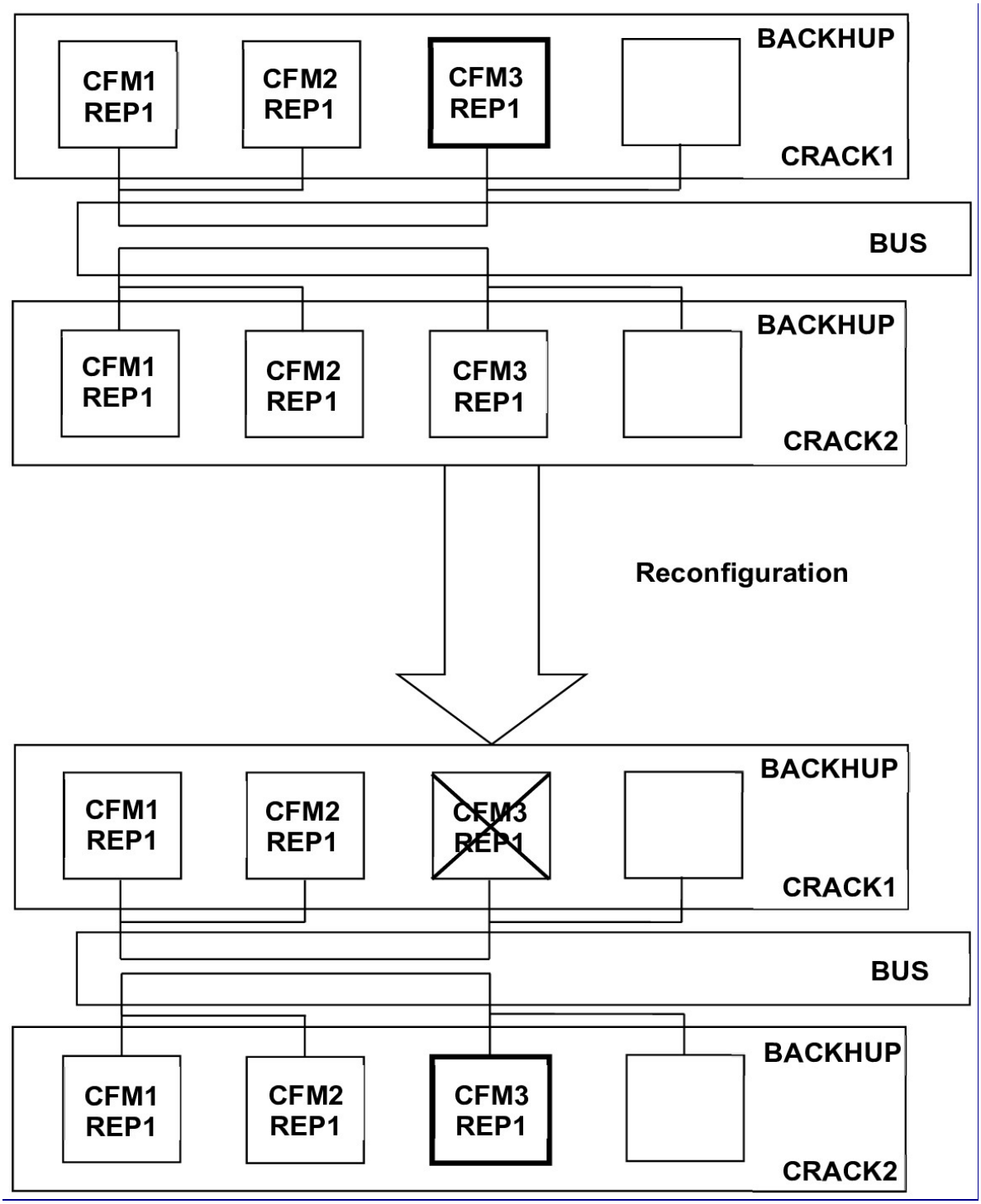

Fig. 4. IMA reconfiguration peculiarities

A reconfiguration peculiarities.

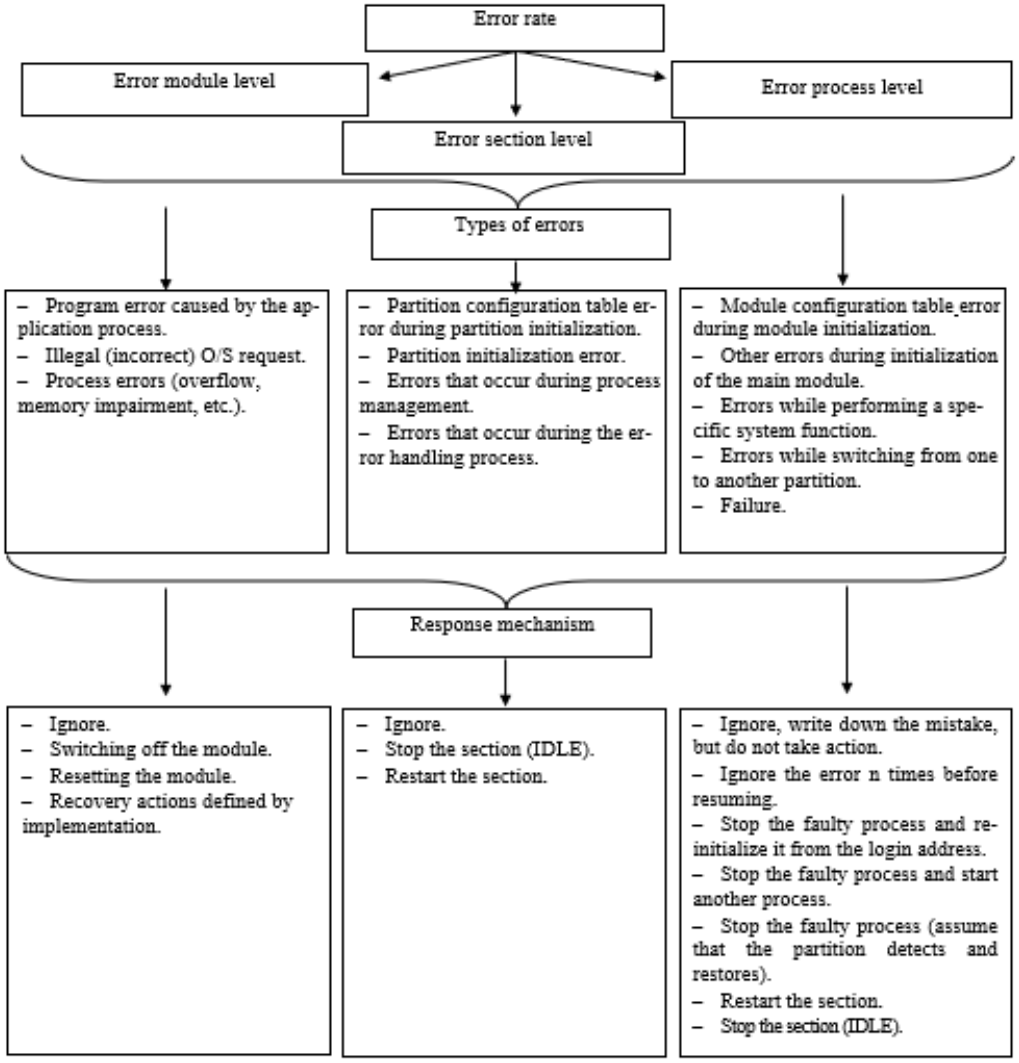

Fig. 5. Classification of errors and response mechanisms

Based on ARINC 653 [13] the errors in IMA and response mechanisms are classified as shown in Figure 5.

If the error cannot be resolved by response mechanisms, it causes a dynamic reconfiguration. In such cases, the reconfiguration is referred to as software, because the hardware failure becomes inevitable [14].

Dynamic IMA reconfiguration can change tasks depending on requirements and effectively recover from failure [2]. This makes the system more flexible, reduces equipment redundancy and the cost of unscheduled maintenance. In addition, the human factor involvement increases the complexity of dynamic reconfiguration [7].

Significant attention needs to be paid to the key criteria of remote control of the aircraft operation process as the main technological components of the integrated modular avionics development (IMA), namely:

- Laptops at the flight control point must be connected to the equipment status identification systems by graphic display.
- The software structure must have a user-friendly interface.
- The software module is needed to identify the status of the aircraft maintenance system.
- The aircraft systems control module should be present in an emergency situation.
- All system components (clusters) must have a modular structure [16].

As for the IMA reconfiguration mechanism, it includes statistical and dynamic reconfigurations. When the system is initialized, applications are loaded into the target module [17]. In the future, the operation reduces the need for maintenance and provides the possibility of replacing the module. The configurator allows you to detect faults, alert and report about them. At the same time, there is a connection between the detection of similarities and the occurrence of errors of this type [7].

Software for the operation of civil aircraft must also be integrated with third-party software used for flight control. According to the authors, the interaction between complex programs (for example, title displays, map display systems and weather radars) should be provided. In particular, it is proposed to use the technology of multi-core processors to facilitate such a variety of programs as:

- Input/output loading.
- Running multiple, different operating environments (such as Linux or other non-COTS operating systems) using virtualization.

Which allows to meet the increasingly demanding needs of these applications in terms of power and processing speed [1].

However, there are separate architectural criteria for the creation of software in the field of civil aviation [2], namely:

- IMA architecture should allow multiple programs to share the same computing resources (multithreading) to reduce the number of subsystems, which will allow more efficient use of system resources and leave room for further expansion.
- The IMA architecture must isolate the application not only from the base bus architecture but also from the base hardware architecture. This practice increases the mobility of applications between different platforms and allows the introduction of new equipment to replace outdated architectures.
- Maximum use (IMA architecture should allow reuse of obsolete code). This practice reduces development time by giving the developer a method of redistributing existing applications without major changes.
- Reducing the cost of change (IMA architecture should reduce the cost of change, as it facilitates reuse, and by separating the components of the platform that run on a single processor, it simplifies the impact analysis, reducing the cost of reuse testing) [2].

The application of the concept of distributed IMA (DIMA) is a promising trend in the development of avionics. This is especially true for the practical application of applied software solutions. The modules in separate units are installed throughout the aircraft near the sensors and actuators. This approach allows to reduce the weight and size of the cable network, and reduces the noise of transmitted data. That in turn reduces the development time of further modernization and increases the reliability of the system [5].

At the heart of the software design process are the fundamental principles that are present throughout the development process, and the main one is "dissimilarity". This principle determines that each part of the aircraft operation control system must be implemented by different groups of people on different hardware components and using different software. First of all, of such software designs as: development tools and programming languages. Thus, the system is divided into software and hardware-independent parts, and the development process is controlled by different groups of people for different tasks and at different levels in accordance with Fig. 3 structures.

Currently, the main approach to the design and development of on-board systems of civil aircraft is the approach of integrated modular avionics. According to this approach, specialized controllers are replaced by general-purpose processor modules, which provide independent operation of different aviation systems, their own wires of each aviation subsystem are replaced by virtual connections within a switched network infrastructure based on technologies such as AFDX (Avionics Full DupleX Switched Ethernet) and CAN (Controller area network). This reduces unreasonable duplication of hardware, which leads to unacceptable levels of power consumption and complexity of the

onboard equipment system. On the other hand, this approach greatly complicates the process of software and hardware development, sets new challenges in the design and integration of software and hardware. With the implementation of the IMA approach in the complex of onboard equipment of the aircraft, a new subsystem appears. It provides a hardware platform for the operation of software of other onboard systems. This subsystem is called the IMA platform and codenamed ATA-42. The team responsible for designing, configuring and verifying the IMA platform is usually called the system integration team, as its task is not only to develop a stand-alone subsystem, but also to coordinate the needs of all users of the platform and ultimately integrate the entire software and hardware components that use IMA platform.

In the context of designing complex software and hardware systems, such as the IMA platform, the main core is the architecture of the complex, around which the requirements for the system as a whole and its individual components, design trade-offs, analysis and verification, etc., are built. Therefore, it is not surprising that architectural models that describe the components of the system and the relationship between them, become the basis for the formation of new technologies and tools for design automation. They allow to describe different aspects of architecture in a single formalized model, which can be processed by different tools to check the internal consistency of the architecture, meet the system's various requirements, automate design decisions, generate configuration data/files, source code and more.

The result of the initial design of certain components of software development processes is a system model, usually performed in Matlab/Simulink, Labview environments. Based on the model, documents are created that regulate which hardware should be used and what connections they should have with each other. At least the result of this stage is the creation of two documents: the definition of hardware and hardware-software components.

Next is the stage of the engineering process of preparation, assembly of boards and finished modules (Control Electronic, etc.), i.e. directly installation, wiring of the necessary microcontrollers, chips, peripherals, for which the necessary software will be written. To interact with the hardware, there must be drivers and interaction layers (framework layer), on the basis of which the application must be built. Avionics software development should begin with the creation of appropriate drivers/functionalities, or make changes to the software library, based on the document HSI (Hardware-Software Interface).

Accordingly, the Universal Framework Library includes a variety of drivers for devices certified for use in aviation, as well as certified standard functions and procedures. This is a matter of principle, because, for example, the usual strcmp function cannot be used directly, it must be rewritten according to standards and be certified. A set of such certified standard functions, prototypes, templates serves as a basis in the Framework layer. This attitude is especially critical for secure mathematical operations (fast detailed division for integer processors, module, root, as an example), and for working with memory. All algorithms are different from STL.

For use on all sorts of devices, the Framework should include driver sets with the DrvHigh <-> DrvLow structure. Here, in the DrvHigh package you need to place all sorts of interfaces for device drivers (Flash, Eeprom memory, digital-to-analog, analog-to-digital converters, real-time clocks, interrupts, CAN, ARINC, LAN chips, etc.). In turn, each of these driver interfaces can use one or more drivers for a specific device (a memory chip, converter, etc.).

Once the hardware is installed and configured, you should create a Software Requirement Document that describes the functions of the software. This is the document on the basis of which the software (application) should be developed. According to this approach, the avionics software developer does not see the full picture of what he is actually creating the program for, but operates with the

necessary requirements and the architecture that he must create. The requirements for the developer are the following documents:

- Software Design Standard – a standard that defines the overall style of the program and the approach to creating architecture.
- Programming Standard – a programming standard that determines what is allowed to write in code and in what style.
- Software Requirements Document – software requirements, documented and divided into different Baseline and iteration package within them (high-level specification) [19, 3].

Thus, the process of developing a system for aircraft operation should take place from 2-3 large iterations (baselines) to 18-20 for large and complex systems (including more than 40 – for Framework complexes). Software and hardware units require separate certification. Moreover, each unit must have its own certificate, which will properly affect the system approach used in the work of software and hardware of automated systems.

The main components of the software development process for the civil aircraft operation are based on a multilevel system, which is divided into software and hardware-independent parts.

In addition, the development process is controlled by different groups of people for different (not always directly dependent) tasks and at different levels of creation and implementation of the software product and its components. Therefore, the classical approaches to the development of aircraft operation system are inefficient, require further development and application of appropriate software techniques in avionics.

Thus, avionics systems now are complexes of software and hardware, so the methods and approaches developed in the field of design and analysis of avionics and software systems should enrich each other.

For this reason, experience in the use of formal methods of verification of complex software and hardware systems, such as operating systems and microprocessors, has quickly mastered another area – the development of tools for design and integration of avionics systems, as many problems in this new area can be solved based on modeling and verification technologies.

Review of the results of the study of the functional principles of development and subsequent operation of software in aviation

The obtained results of research in the field of software development and operation are substantiated by a careful analysis of the introduction and operation of software products in the field of aviation. Unfortunately, there are some limitations to the study of the functional foundations of software development and operation in the field of aviation. They are justified primarily by the problems of the practical nature of software development and operation, namely the financing of software and hardware of automated systems. Prospects for the study of the functional principles of development and operation of software in avionics are justified by the fact that when analyzing the activities of the business entity, each aspect can be given enough attention without being distracted by its relationship with other categories.

Conclusions

1. Software development and operation in the field of aviation have own specifics, which is based on relevant standards.

Ensuring the continuity and integration of software in the process of its development and testing involves the use of a wide range of technologies.

Based on the use of this model both the development of software solutions is held and relevant documents are created. The documentation is a description of the main blocks of the program, the order of integration of modules, system architecture, and other formalities that reflect the results of creating source code.

The main components of the process of creating software for the operation of aircraft in the field of civil aviation are based on a multilevel system, which is divided into software and hardware-independent parts. In addition, the development process is controlled

by different groups of people for different (not always directly dependent) tasks and at different levels of creation and implementation of the software product and its components. Therefore, the classical approaches to the development of aircraft operation system are inefficient, require further development and application of appropriate software techniques in avionics.

The process of developing a system for aircraft operation should take place from 2-3 large iterations (baselines) to 18-20 for large and complex systems (including more than 40 – for Framework complexes). Software and hardware units require separate certification. Each unit must have its own certificate.

2. The considered methodical approaches to software development in the field of civil aircraft operation allowed to determine the V-shaped model of this process and to substantiate its main components. It is recommended to fragment the software development process, namely: the stages of design development in UML or system modeling, description of functionality and algorithm for solving the implemented requirements for the tester. The algorithm is implemented according to the black box model: a description of the input and the expected response. In addition, we note the stages of the process of direct code generation, the process of processing and compiling software code to the state of absence of errors and the process of running and integrating software on the simulator. As for the introduction of results into the version control system and reports, this is the final and important stage of development.

Thus, for the design of modern avionics systems, a set of tools called MASIW provides both a common platform for the design and analysis of architectural models, and a specialized solution for a specific domain of avionics systems. It supports the creation, editing, and management of AADL models in both text and graphics formats. MASIW also provides various functions for the analysis and synthesis of AADL models, simplifies the solution of a number of tasks related to the development of aviation systems. It allows you to conveniently and clearly create and edit models of such systems in AADL, as well as analyze such models for compliance with various requirements related to both the structure and behavior of the model (calculate various temporal characteristics, predict the behavior of the simulated system in various situations, including non-standard behavior of components and failures within the system). In addition, it facilitates the design of architecture through the implementation of a number of algorithms for model synthesis. This allows, in particular, to distribute the tasks on the computing units so that each task was allocated enough processor time, and generate an onboard network model and network resource allocation scheme according to the needs of system components.

The design of modern avionics systems (MASIW tool) is constantly evolving – this development allows for cooperation, close cooperation with customers, potential users and with the international community of developers of open standards and open tools for support for the development, integration and verification of responsible systems based on the use of modeling tools.

Literature

1. *Hayley J., Reynolds R., Lokhande K., Kuffner M., Yenson S.* Human-Systems Integration and Air Traffic Control. Lincoln laboratory journal, 2012. – Vol. 19, №. 1. – P. 34-49.

2. *Parkinson P., Kinnan L.* Safety-Critical Software Development for Integrated Modular Avionics. – Wind River, 2015. – Vol. 11. – No.2. [Internet Resource] / Access mode: https://events.windriver.com/wrcd01/-wrcm/2015/02/Safety-Critical-Software-Development-for-Integrated-Modular-Avionics-White-Paper-1.pdf.

3. *Tiedeman H., Parkinson, P.* "Experiences of Civil Certification of Multi-Core Processing Systems in Commercial and Military Avionics, Integration Activities, and Analysis," SAE Int. J. Adv. & Curr. Work. in Mobility 1 (2): 419-428, 2019.

4. *Adnane Ghannem, Mohamed Salah Hamdi, Marouane Kessentini, Hany H. Ammar.* Search-based requirements traceability recovery: A multi-objective

approach, Proc. IEEE Congress on Evolutionary Computation (CEC), 2017. – p. 1183-1190.

5. *ES Neretin* et al 2019 J. Phys.: Conf. Ser. 1353 012005.

6. *Chuanwen Lin* et al 2020 J. Phys .: Conf. Ser. 1544 012171.

7. *Jiang Z., Zhao T., Wang S., Ju H.* New Model-Based Analysis Method with Multiple Constraints for Integrated Modular Avionics Dynamic Reconfiguration Process. Processes, 2020. – 574 p.

8. *Rozhdestvenskaya K.N.* Temporary analysis of the control system in the data processing network. Information and control systems, 2019. – № 1. – P. 32-39.

9. *Bondar D.S., Prokhorov A.V.* Analiz pokazateley nadezhnosti aerodromnykh sistem upravleniya vozdushnym dvizheniyem. Nauchnyy vestnik MGTU GA, 2016. – №5. [Internet Resource] / Access mode: https://cyberleninka.ru/article/n/analiz-pokazateley-nadezhnosti-aerodromnyh-sistem-upravleniya-vozdushnym-dvizheniem.

10. *Vysotsky A.V., Korshets A.A., Lymar R.V., Makarov S.A., Martynov A.A.* Automation of processes of collection, processing and display of information on the air situation at the command and control point in the management of flights of state aviation. Collection of scientific works of Kharkiv National University of the Air Force, 2018. – №3 (57). – P. 103-109.

11. J. *Athavale, R. Mariani, M. Paulitsch*. "Flight Safety Certification Implications for Complex Multi-Core Processor Based Avionics Systems." IEEE International Reliability Physics Symposium (IRPS), Monterey, 2019. – P. 1-6.

12. *Yevdokimov V.* Integrated safety management system for aviation activities based on ICAO standards and recommended practice. Journal of Science, Practice, Economics, 2013. – №2 (45). [Internet Resource] / Access mode: https://cyberleninka.ru/article/n/integrirovannaya-sistema-upravleniya-bezopasnostyu-aviatsionnoy-deyatelnosti-na-osnove-standartov-i-rekomendovannoy-praktiki-ikao.

13. Aeronautical Radio. Avionics Application Software Standard Interface; ARINC653: Annapolis, 2010.

14. *Montano G., McDermid J.* Human Involvement in Dynamic Reconfiguration of Integrated Modular Avionics, Avionics. In Proceedings of the 27th Digital Avionics Systems Conference, St. Paul, 26–30 October 2008; IEEE: Piscataway, 2008.

15. ARINC Specification 653. Avionics Application Software Standard Interface. [Internet Resource] / Access mode: https://www.sae.org/standards/content/arinc653p3a-1/.

16. *Il'yenko C.C.* Avtomatizatsiya, distantsionnoye upravleniye i nadezhnost 'svetosignal'noy sistemy sovremennykh aerodromov grazhdanskoy aviatsii. Naukoyemkiye tekhnologii, 2016. – № 2 (30). – P. 211-215.

17. Committee, AE ARINC 664 Aircraft Data Networks, Part7: Avionics Full Duplex Switched Ethernet (AFDX) Network; Aeronautical Radio, Inc.: Annapolis, 2005.

18. DO-178B Software Considerations in Airborne Systems and Equipment Certification. Dept. of Measurement and Information Systems. [Internet Resource] / Access mode: https://inf.mit.bme.hu/sites/default/files/materials/taxonomy/term/445/13/13_CES_DO-178B.pdf.

19. RTCA DO-255 / EUROCAE ED-96. Requirements Specification for Avionics Computer Resource (ACR). [Internet Resource] / Access mode: https: // standards. globalspec. com / std / 1968378 / RTCA% 20DO-255.

Козлюк І.О., Коваленко Ю.Б.

ФУНКЦІОНАЛЬНІ ЗАСАДИ РОЗРОБКИ ТА ЕКСПЛУАТАЦІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ В АВІОНІЦІ

У ході дослідження проведено аналіз функціональних засад та підходів щодо визначення основних особливостей програмного забезпечення, яке використовується для експлуатації повітряних суден. Охарактеризовано теоретичні аспекти побудови архітектури програмних рішень у сфері цивільної авіації на засадах сучасних підходів розробки систем автоматизації. Розглянуто основні нормативні вимоги до організаційного та технологічного забезпечення процесу розробки програмних рішень у сфері цивільної авіації. Визначено особливості V-подібної моделі розробки програмного забезпечення щодо експлуатації повітряних суден. Документація до програмного забезпечення повинна складатися з опису основних блоків програми, порядку інтеграції модулів, архітектуру системи та інші формальності, які відображають результати створення вихідного програмного коду. На основі описаного підходу до розробки програмного комплексу для експлуатації повітряних суден сформовано бачення та критерії циклу розробки інформаційних рішень у сфері цивільної авіації, що належним чином впливає на розвиток галузі. В залежності від функціональних характеристик отриманих програмних рішень складається фінальна документація, яка також містить результати його тестування в умовах передпромислової експлуатації. Сучасні системи авіоніки мають модульну багаторівневу структуру. При цьому класичні підходи до розробки систем експлуатації повітряних суден є малоефективними, що потребує розробки специфічної методики, яка враховує досліджувану предметну сферу. Це пояснюється необхідністю інтеграції програмного забезпечення для експлуатації суден цивільної авіації зі сторонніми програмно-апаратними комплексами, що використовуються для контролю за польотами.

Ключові слова: *програмне забезпечення, цивільні авіація, підтримка програмного забезпечення, авіоніка, архітектура програмного забезпечення.*

Kozlyuk I.O., Kovalenko Yu.B.

FUNCTIONAL BASES OF THE SOFTWARE DEVELOPMENT AND OPERATION IN AVIONICS

The study analyzes the functional principles and approaches to determine the main features of the software used for the aircraft operation. Theoretical aspects of creating the software architecture in the field of civil aviation based on modern approaches to the automation systems development are described. The main regulatory requirements for organizational and technological support of the process of developing software solutions in the field of civil aviation are considered. The peculiarities of the V-shaped model of aircraft operation software development are determined. The documentation for the software should consist of a description of the main blocks of the program, the order of modules integration, system architecture and other formalities that reflect the results of the source code. Based on the described approach to the development of a software package for the aircraft operation, the vision and criteria of the cycle of development of information solutions in the field of civil aviation is created. Depending on the functional characteristics of the obtained software solutions, the final documentation is compiled, which also contains the results of its testing under the conditions of pre-industrial operation. Modern avionics systems have a modular multilevel structure. At the same time, the classical approaches to the development of aircraft operation systems are inefficient, which requires the development of a specific methodology that takes into account

the studied subject area. This is due to the need to integrate software for the operation of civil aircraft with third-party software and hardware used for flight control.

Keywords: *software, civil aviation, software support, avionics, software architecture.*

Козлюк И.А., Коваленко Ю.Б.

ФУНКЦИОНАЛЬНЫЕ ОСНОВЫ РАЗРАБОТКИ И ЭКСПЛУАТАЦИИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ В АВИОНИКЕ

В ходе исследования проведен анализ базовых функциональных основ и подходов к определению основных особенностей программного обеспечения, используемое для эксплуатации воздушных суден. Охарактеризованы теоретические аспекты построения архитектуры программных решений в области гражданской авиации на основе современных подходов разработки систем автоматизации. Рассмотрены основные нормативные требования к организационному и технологическому обеспечению процесса разработки программных решений в сфере гражданской авиации. Определены особенности V-образной модели разработки программного обеспечения эксплуатации воздушных суден. Документация к программному обеспечению должна составляться из описания основных блоков программы, порядка интеграции модулей, архитектуры системы и других формальностей, которые отображают результаты создания исходящего программного кода. На основе описанного подхода к разработке программного комплекса для эксплуатации воздушных суден сформировано видение и критерии цикла разработки информационных решений в сфере гражданской авиации, что определенным образом влияет на развитие отрасли. В зависимости от функциональных характеристик полученных программных решений составляется финальная документация, которая также содержит результаты его тестирования в условиях предпромышленной эксплуатации. Современные системы авионики имеют модульную многоуровневую структуру. При этом классические подходы к разработке систем эксплуатации воздушных суден являются малоэффективными, что требует разработки специфической методики, учитывающей исследуемую предметную сферу. Это поясняется необходимостью интеграции программного обеспечения для эксплуатации суден гражданской авиации со сторонними программно-аппаратными комплексами, используемыми для контроля за полетами.

Ключевые слова: *программное обеспечение, гражданская авиация, поддержка программного обеспечения, авионика, архитектура программного обеспечения.*