The hardware and software implementation of the adaptive platform for an onboard spacecraft control system

I N Kartsan¹, A O Zhukov²,³,⁴, A O Platonov¹ and S V Efremova¹

¹Reshetnev Siberian State University of Science and Technology, 31, Krasnoyarsky Rabochy Av., Krasnoyarsk, 660037, Russia
²Institute of Astronomy of the Russian Academy of Sciences, 48, street Pyatnitskaya, Moscow, 119017, Russia
³Shternberg State Astronomical Institute of Lomonosov Moscow State University, 13, Universitetsky pr., Moscow, 119234, Russia
⁴Russian Technological University, 78, Vernadskogo Av., Moscow, 119454, Russia

E-mail: kartsan2003@mail.ru, aozhukov@mail.ru

Abstract. In this paper the authors discuss hardware and software implementation solutions for an adaptable platform used in onboard spacecraft electronic equipment complexes. The proposed solutions ensure the reduction of time required for designing and manufacturing the spacecraft, increase of the export potential of communication and other services by enabling the adaptation of the onboard equipment to new protocols and technologies in the fields of communication, data processing, and control in the process of the spacecraft’s operation in orbit and expanding its hardware and software unification.

1. Introduction

Currently, there is a clear trend in designing complex intelligent distributed executive and data management robotic systems and technologies for terrestrial and aerospace applications. Due to the complexity of such systems, there are several ways to improve their energy consumption and accuracy characteristics, along with the ability to function in certain conditions. Thus, we can see a definite trend in the development of integrated systems for terrestrial and aerospace applications, the design and purpose of which require the presence of a module which enables monitoring the current state of the entire system as well as the possibility of bidirectional data exchange and management of system components.

Spacecraft and ground control stations are complex systems, which enhance the chances of failures in separate software modules responsible for data management and processing. This may lead to the loss of valuable data and signalize not only significant financial losses, but also a reduction in the efficiency of the intended use of these systems. All this predetermines the increased reliability requirements for the entire software system [1-3].

By applying a methodology of multiversion formation of fault-tolerant software, it is possible not only to provide a higher level of reliability, but to achieve higher levels of fault-tolerance and survivability of the control systems and data processing systems. This methodology is based on software redundancy, the introduction of which enables significantly increasing the level of reliability,
survivability, and performance stability of the whole software complex of the spacecraft control system, regardless of many environmental factors [4-6].

A large number of software modules, their additional redundant versions, as well as restrictions on the cost of their manufacturing and the maximum time requirements to the execution of individual modules, set the designer the task of choosing an effective architecture for both the ground segment of the spacecraft control software and the onboard software.

Currently, the scope of application is constantly expanding and the complexity of existing space software systems is increasing. Therefore, these software systems are becoming increasingly vulnerable to errors or low quality of programs and can be even be damaged. This has led to the development and application of methods, standards, and means of automating industrial software engineering, ensuring the production of complex software architectures of space systems with specified high quality indicators and real limitations on the use of development resources [7-8]

One of the most promising and already proven methods for ensuring the high reliability and fault-tolerance of software is the introduction of software redundancy, i.e. duplication of software components. However, the simple duplication of components as in the case of hardware redundancy is inadmissible as, unlike hardware, software defects have an internal nature. When duplicating, the existing errors will be copied. This is why when introducing software redundancy, it is supposed that the appearance of failure in functionally equivalent modules (versions) on the same input data occurs in different points of execution.

2. Adaptive platform software algorithm

The following algorithm enables developing multiversion software for spacecraft from available or potentially possible elements in order to obtain a highly reliable system. The choice of software composition is made considering the limitations on the attributes calculated additively. With the help of the proposed algorithm it is also possible to form multiversion software of computer systems, communication systems, control and data processing systems, etc. [4].

We shall consider multiversion software, which consists of separate software modules implemented in the form of a number of multiversions. The failure of any of the modules leads to failure of the entire software system. Each of the modules can be implemented by \( u_{i(l)} \) methods characterized by different values of attributes, such as reliability, cost, resource intensity, etc. It is necessary to define the software variant (i.e. to choose the method of implementation of each module), which delivers an extremum of the target function \( P \) and provides successful solution of all tasks set before the software; and the costs should not exceed the specified limit.

Let us consider exclusively the principally possible variants of software in order to make it easier to demonstrate the application of the proposed approach to the formation of multiversion software. The mathematical model of this problem has the following form: to determine the variant \( v_0 \), which delivers the maximum of the target function

\[
P(v) = \prod_{i=1}^{n} P_i(u_{i(l)})
\]

subject to restrictions:

\[
g_a(v) = \sum_{i=1}^{n} g_{a_i}(u_{i(l)}) \leq g_a^*, \quad (a = 1, \ldots, q)
\]

\[
g_a(v) = \sum_{i=1}^{n} g_{a_i}(u_{i(l)}) \geq g_a^*, \quad (a = q + 1, \ldots, Q)
\]

\[
v \in V, u_{i(l)} \in U_i, (i = 1, \ldots, n)
\]
where \( U_i = \{u_{i(l_1)}, \ldots, u_{i(l_2)}, \ldots, u_{i(l_m)}\} \) (\( i = 1, \ldots, n \)) is the set of all the multiversions that can be used in the i-module, the number of multiversions in the set of \( U_i \) is equal to \( m_i \);

\[
V = \prod_{i=1}^{n} U_i
\]

\( v \) is the current version of the system,

\( P_i(u_{i(l)}) \) is the reliability (probability of failure-free operation at a given time interval) of the \( l_i \)-th type i-th module multiversion;

\( g_a(u_{i(l)}) \) is the value of the a-th attribute for the multiversion of the \( l_i \)-th type of i-th module, which is limited;

\( g_a(v) \) is the sum of the values of the a-th attribute used for all software;

\( g_n^a \) is the limitation on the a-th attribute for the entire software as a whole.

One of the possible approaches to solving this problem is the following algorithm.

**Figure 1.** Sorting algorithm for versions of multiversion software modules by attribute limitations.
The sorting of each of the attributes is performed in a designated order. For this purpose, having considered every criterion for each of the n modules, let us order all \( m_i \) types of multiversions in ascending order, according to the values of the considered attribute.

The sum of the first values of the ordered types of the multiversions represents the minimum resources required to form multiversion software. Obviously, this sum should be less than the limit on the current attribute \( g_{a}^{*} \), as a necessary condition for the existence of allowable solutions.

The next step is to define the “tolerance” for each module using the formula

\[
\Delta g_{ai} = g_{a}^{*} - g_{als},
\]

where \( g_{als} \) is the sum of elements of the first column.

All multiversions of the i-module that exceed the value of “tolerance” are discarded and are no longer considered.

Having considered all the attributes, we perform the following iteration, i.e. move to step 1. The algorithm finishes its work when the multiversions are no longer eliminated.

It may happen that all the multiversions will be discarded, in which case it is necessary to execute one or both of the following procedures:

- Expand the restrictions on the software.
- Use less resource-intensive multiversions in modules.

Calculations end when the sorting is completed, i.e. when a solution is found, or when all multiversions are discarded. The solution is the result obtained at the last iteration.

Multiversion software, the composition of which was selected with the help of the proposed algorithm, consists of three consecutively executed modules. These modules solve, respectively, three tasks within the framework of the general problem of analysis and placement of communication equipment sets of the onboard mobile communication complex:

- Analysis and selection of communication equipment components.
- Assessment of parameters of influence of communication equipment components.
- Optimal location of communication equipment components on board the mobile communication complex taking into account their mutual influence.

The failure of one of the modules leads to failure of the entire software system

3. Conclusion
The sorting algorithm for module versions based on restrictions by attributes calculated additively allows to form the composition of fundamentally possible multiversion software for control and processing systems and, in particular, for communication satellites.

Acknowledgements
The reported study was funded by RFBR, project number 19-37-90076.

References
[1] Kartsan I N, Efremova S V, Khrapunova V V and Tolstopiatov M I 2018 IOP Conference Series: Materials Science and Engineering 450 (2) 022015
[2] Kartsan I N, Tyapkin V N, Dmitriev D D, Goncharov A E and Zelenkov P V 2015 IOP Conference Series: Materials Science and Engineering 94 (1) 012010
[3] Tyapkin V N, Kartsan I N, Dmitriev D D and Efremova S V 2017 International Siberian Conference on Control and Communications, SIBCON 2017 - Proceedings 2017 7998452
[4] Kudymov V I, Brezitskaya V V, Zelenkov P V, Kartsan I N and Malanina Yu N 2018 *IOP Conference Series: Materials Science and Engineering* 450 (5) 052009

[5] Semenenko M G, Kniazeva I V, Beckel L S et al 2019 *IOP Conference Series: Materials Science and Engineering* 537(3) 032095

[6] Saramud M V, Zelenkov P V, Kovalev I V et al 2016 *IOP Conference Series: Materials Science and Engineering* 155 (1) 012015

[7] Kartsan I N 2018 *Research journal of pharmaceutical biological and chemical sciences* 9(5) 2357-67

[8] Kartsan I N, Tyapkin V N, Dmitriev D D et al 2016 *IOP Conference Series: Materials Science and Engineering* 155 (1) 012018