

# THE PRACTICAL APPROACH TO THE RELIABILITY ANALYSIS OF THE SOFTWARE ARCHITECTURE OF A COMPLEX COMPANY CONTROL SYSTEM

**I V Kovalev, P V Zelenkov, and S Ognerubov**

Siberian State Aerospace University named after Academician M.F.Reshetnev  
31 "KrasnoyarskiyRabochiy" prospect, Krasnoyarsk, 660037, Russia.

E-mail: zelenkov@sibsau.ru

**Abstract.** The practical aspects of the implementation of reliability analysis of the architecture of a complex control system of a company are considered in this article. The comparative analysis for two variants of software architecture using different factors is presented, the relations between the reliability characteristics and the amount of system architecture components and their connections with each other are defined.

## I. Introduction

Software architecture as a research area is wide [1-3]. There are not enough technical and management manuals for the software architecture control for software companies. The wide architecture of a complex company control system (CCCS), which is a member of the group of software systems which operate in real-time and have a very complex (intricate) communication complex (including software versions) is considered as an example in [4, 5]. The following architecture levels are chosen to describe a software system [6]:

- processes;
- functions;
- primitives;
- data;
- structures;
- messages.

To measure the amount of changes which are generated by the subsystem module failures additional data is required. Using that data and the information about the architecture, the system processes and the module changes as a result of a failure it is possible to measure and increase the reliability of a software architecture [7-9].

In the software reliability model [10], one architecture level matches the subsystem's processes and additional architecture levels match the functions, the primitives, the data, the structures and the messages for these processes. Different types of component are placed on different levels and connected using their communication and calling services. The architecture matches the generalized scheme which is presented as an example in [11].

In the given architecture the processes interact through messages using functions and primitives. The functions use the data and the structures. The processes, the functions, the primitives, the data, the structures and the messages are located on different architecture levels in the model. In this architecture, all components of the same type use patterns and the same connection for every component of a given type.

## II. The initial parameters of the architecture levels components

The values of the parameters for the presented model depend on the type of the considered real-time software architecture, the amount of the architecture levels, the components and their relations. Different architectures are considered in order to show how the amount of components



and other parameters of a software system influence the reliability. In that case, the amount of functions, primitives and data change that shows the change in the amount of components of an architecture of a software system.

The unconditional and conditional failure probabilities are different for different amounts of components. The unconditional failure probability depends on the amount of code in the components. For example, it can be proportionate to the number of lines of code in the components or can be dependent on the ratio of the amount of changed lines to the amount of all lines in the program. The unconditional failure probability can be estimated as the rate of the component recoveries independent from the recoveries of other components. The conditional failure probability depends on the amount of code in the components and the relations between the components. For example, the conditional failure probability can be higher for a component which depends on a higher amount of other components (under the condition of a static dependence between components) or is connected to a higher amount of other components (under the condition of a dynamic dependence between components). The conditional failure probability can be estimated as the rate of the component recoveries which are generated by other components recoveries.

Two variants of architecture are presented for comparison. Table I and table II show the parameters used in that example [12].

### III. The comparison of architecture variants result analysis

The comparison analysis for two variants of telecommunication software architecture of a complex company control system is done using the dialog system of the architecture estimation [13].

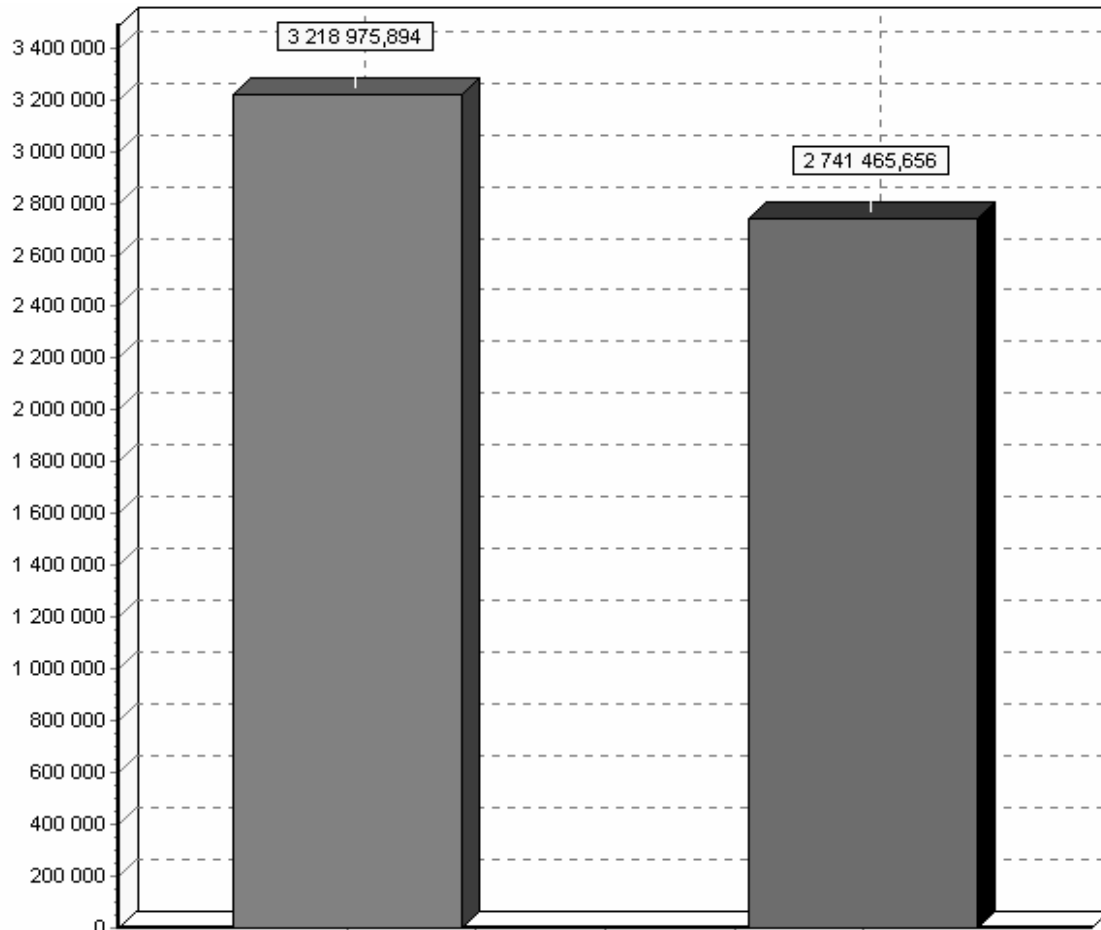
**Table 1.** The values of the parameters of the architecture variant A.

The architecture levels	The amount of components	The component failure probability	The conditional failure probability on different levels	The access, analysis and recovery time	The percentage of component usage
The processes	7	0.1	0.0	5	100
The functions	399	0.2	0.8	7	80
The primitives	138	0.05	0.4	12	90
The data	202	0.12	0.5	12	25
The structures	49	0.05	0.2	11	40
The messages	34	0.05	0.1	14	60

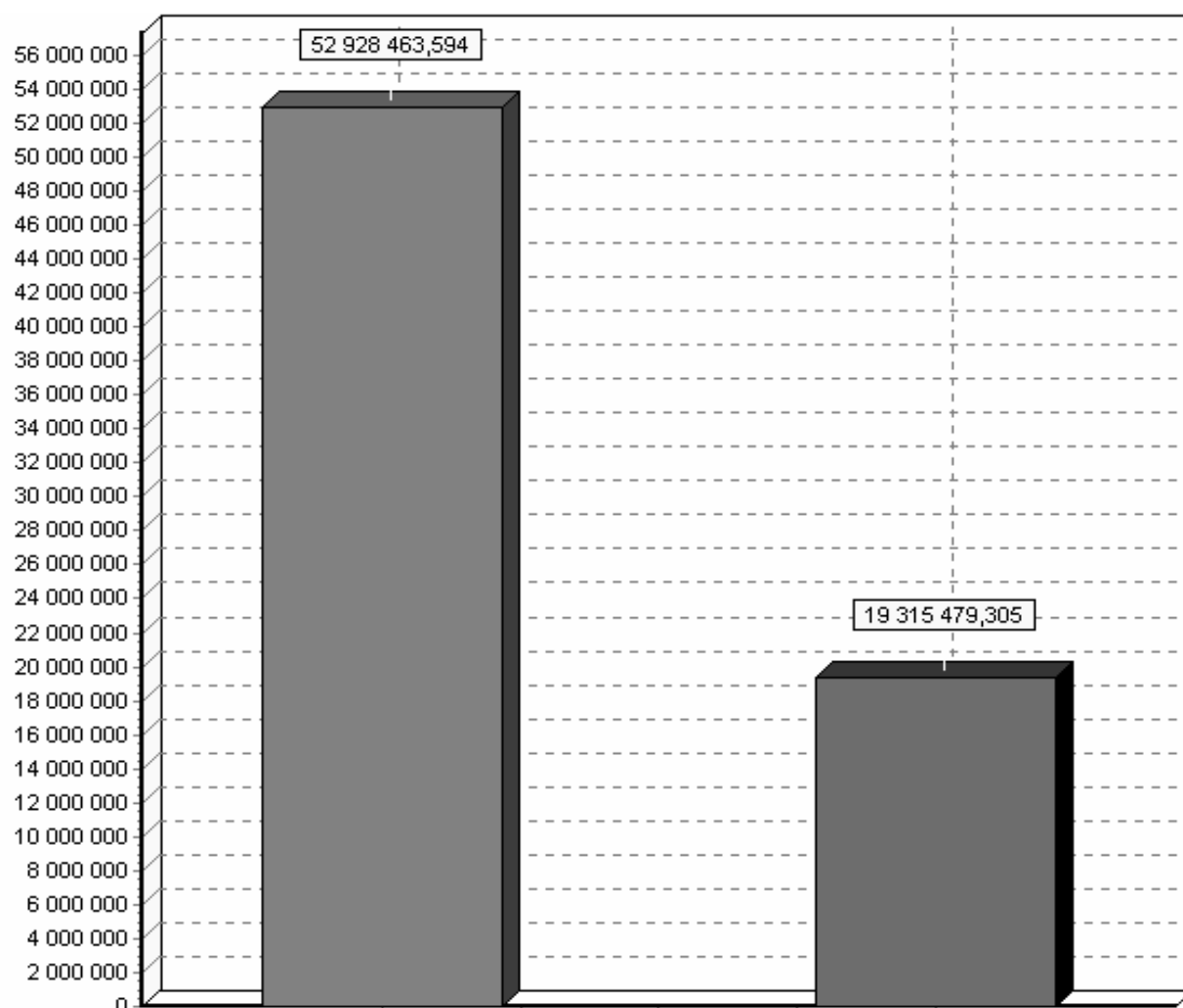
**Table 2.** The values of the parameters of the architecture variant B.

The architecture levels	The amount of components	The component failure probability	The conditional failure probability on different levels	The access, analysis and recovery time	The percentage of component usage
The processes	7	0.1	0.0	5	100
The functions	215	0.35	0.9	12	90
The primitives	63	0.1	0.5	18	110
The data	278	0.1	0.4	10	20
The structures	49	0.05	0.2	11	40
The messages	34	0.05	0.1	14	60

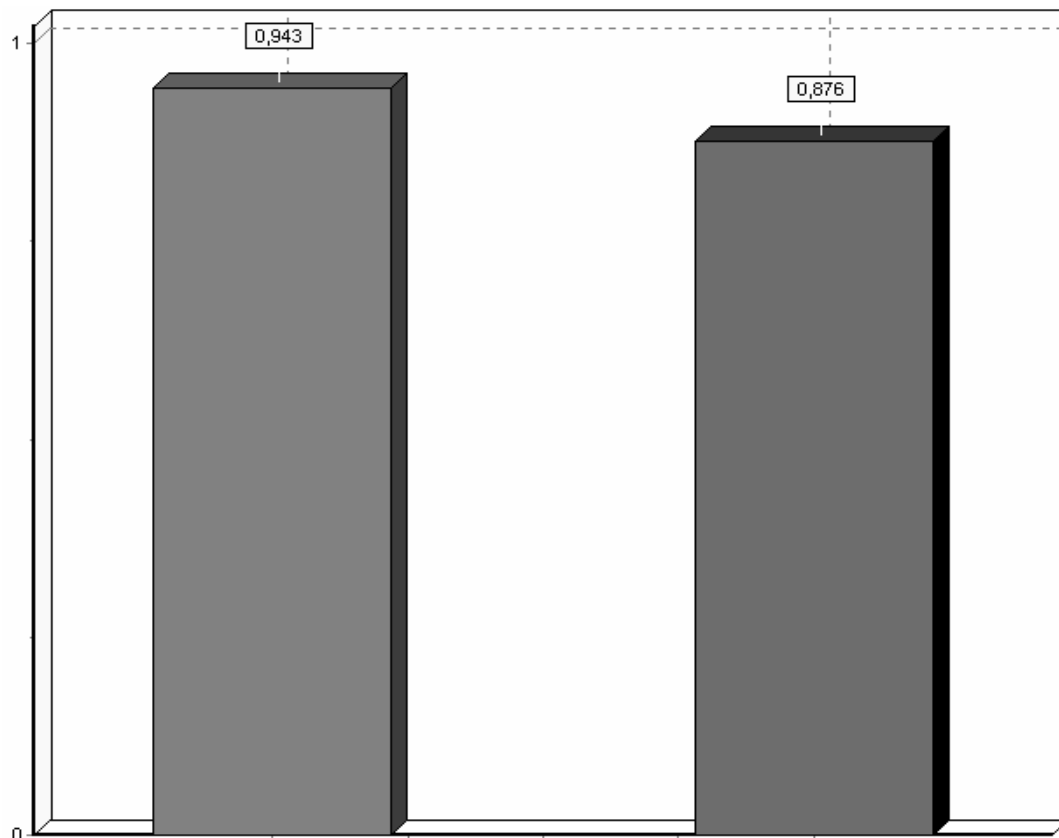
The downtime of the system is shown in "Fig." 1. System B has the minimal downtime having a lesser amount of functions and primitives and a larger amount of data. The decrease in the amount of data causes the most downtime for system A. It means that there is a set of the parameter's values in the large telecommunication software architecture, which decrease system's downtime using the permissible architecture changes. That model's characteristic can be used for the development of the most reliable architecture for the large real-time telecommunication software systems [14-17].



**Figure 1.** The graph of the average systems downtime for both architecture variants.



**Figure 2.** The graph of the average system failure appearing time for both architecture variants.



**Figure 3.** The graph of the system readiness coefficient for both architecture variants.

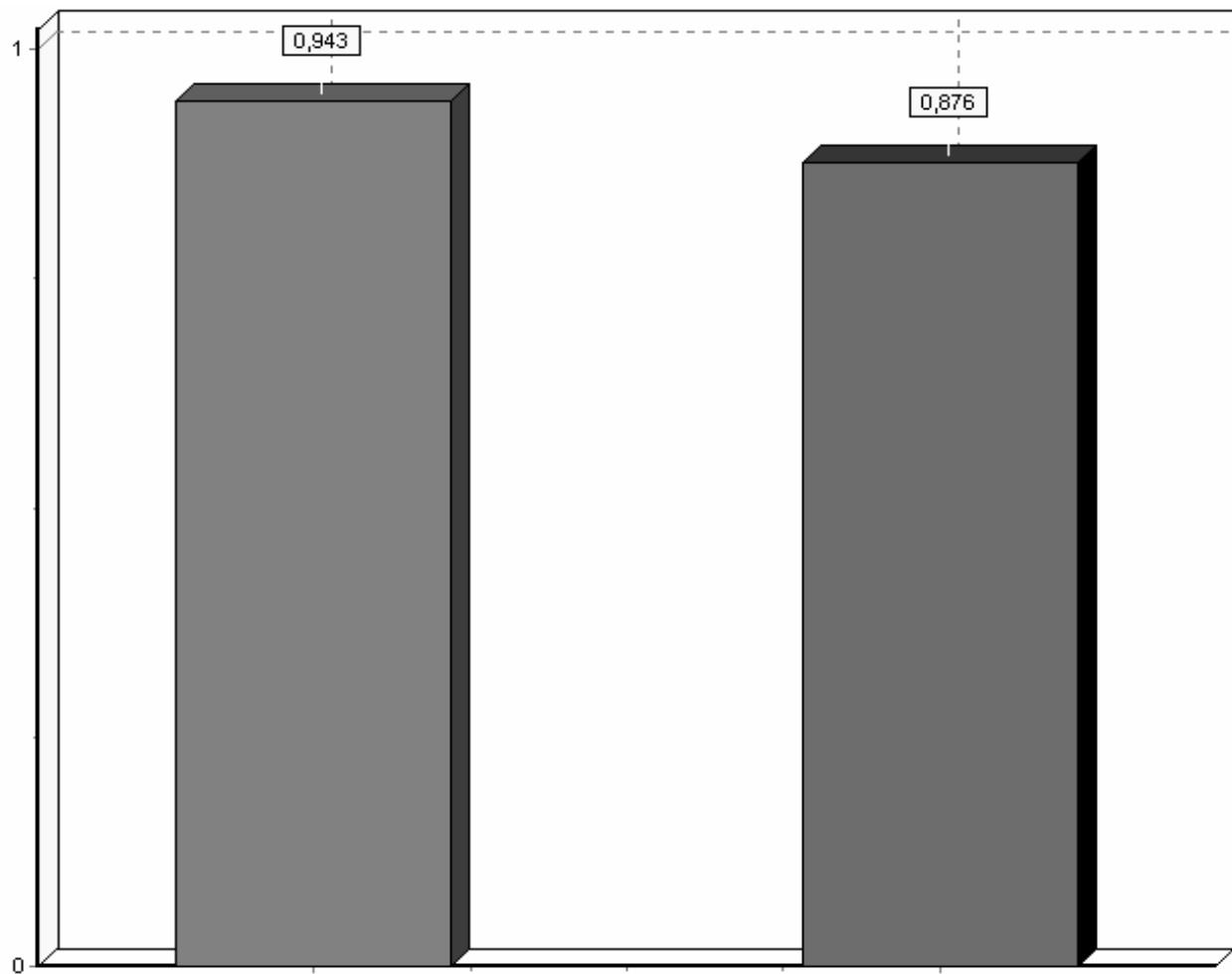
The average system failure appearance time for a software subsystem is shown in "Fig." 2. The average system failure appearance time is increasing then the amount of the components rises. Therefore, architecture variant A consists of 829 components and has the average system failure appearance time of 52 926 463 seconds, while variant B consists of 646 components and has the average system failure appearance time of 315 479 seconds. The average system failure appearing time correlates with the usage time of the components. Therefore a lesser average system failure appearance time does not mean lesser reliability of the large architecture of the real-time data transmission.

The system readiness coefficients for both systems are presented in "Fig." 3. This graph is a generalization of the two graphs described above, because both the average system failure appearance and average systems downtime are used for the system readiness coefficient calculation.

Using the graphs in "Fig." 1 and "Fig." 2 it is hard to decide whether system architecture A or B is more reliable, whereas the graph of the system readiness coefficient for both architecture variants shows that despite higher system downtime, meaning a larger amount of time is required for the system to recover, the architecture variant A is more reliable. The possibility of the fact that in the moment of time  $t$  system A will be functioning well is 0.943 and for system B the possibility is 0.876.

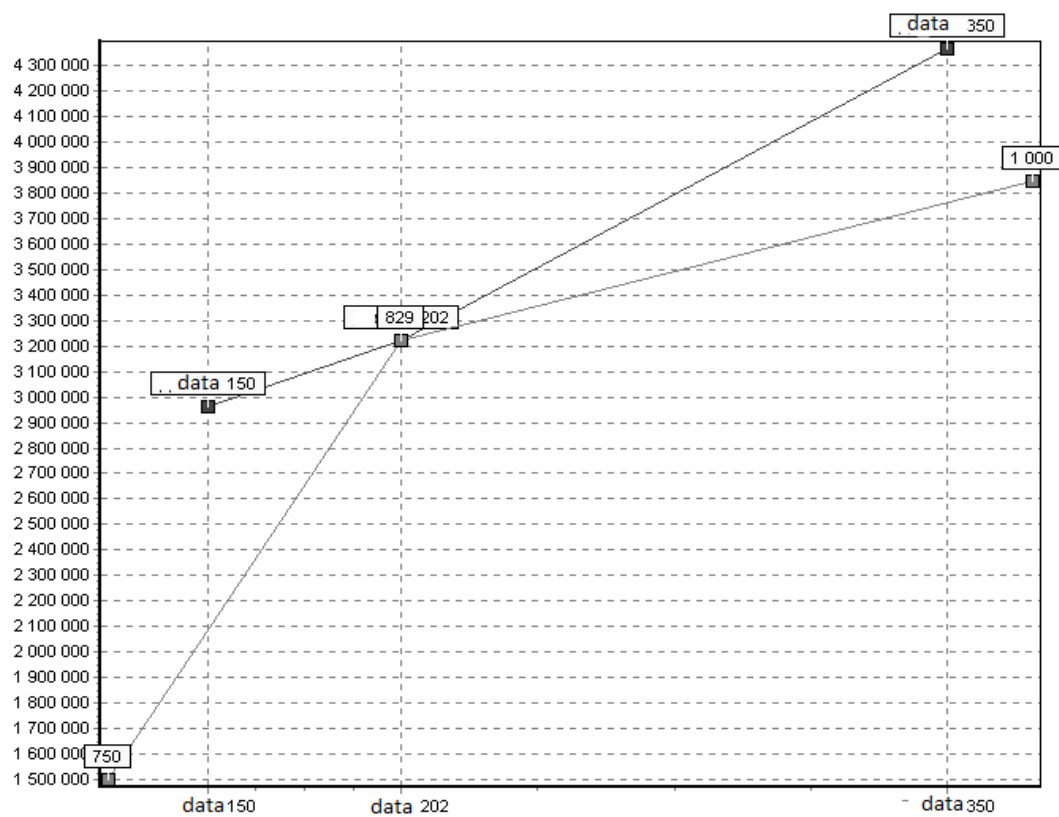
It can be seen in "Fig." 4 that the development cost of variant A is higher than variant B (the development cost of a single component is the same for all components, even for components from different architecture levels).

Variant A is the best choice in terms of reliability, but the most important factor is the development cost of a whole system.

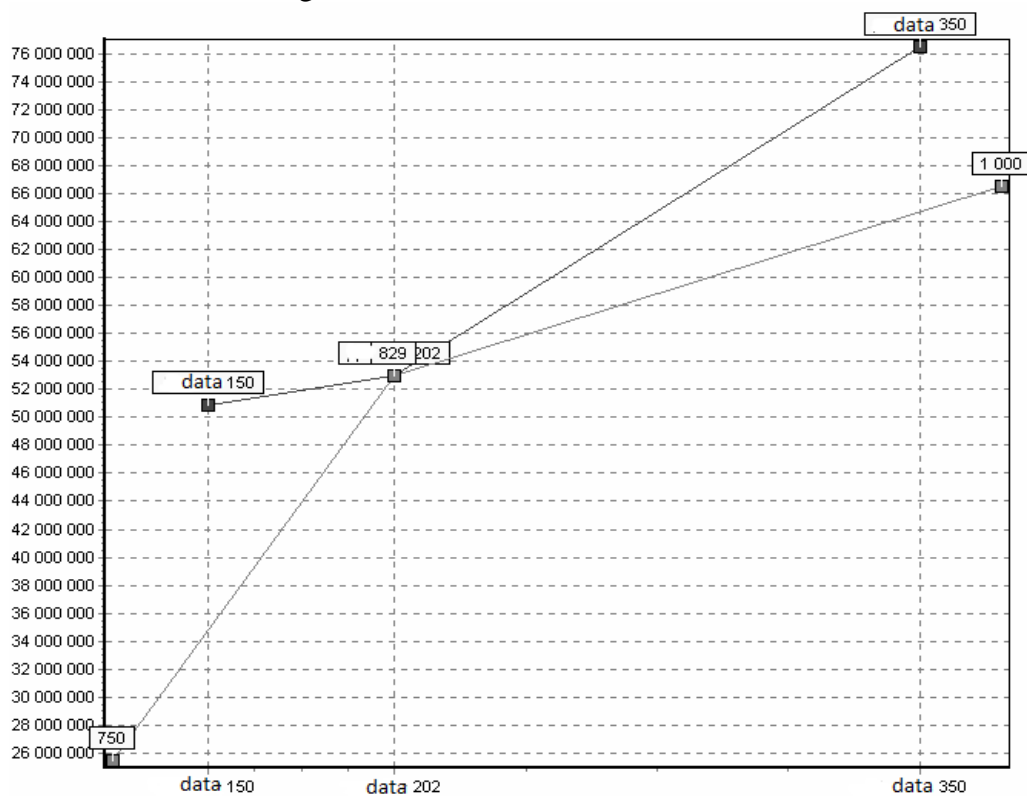


**Figure 4.** The graph of the average system development cost for both architecture variants.

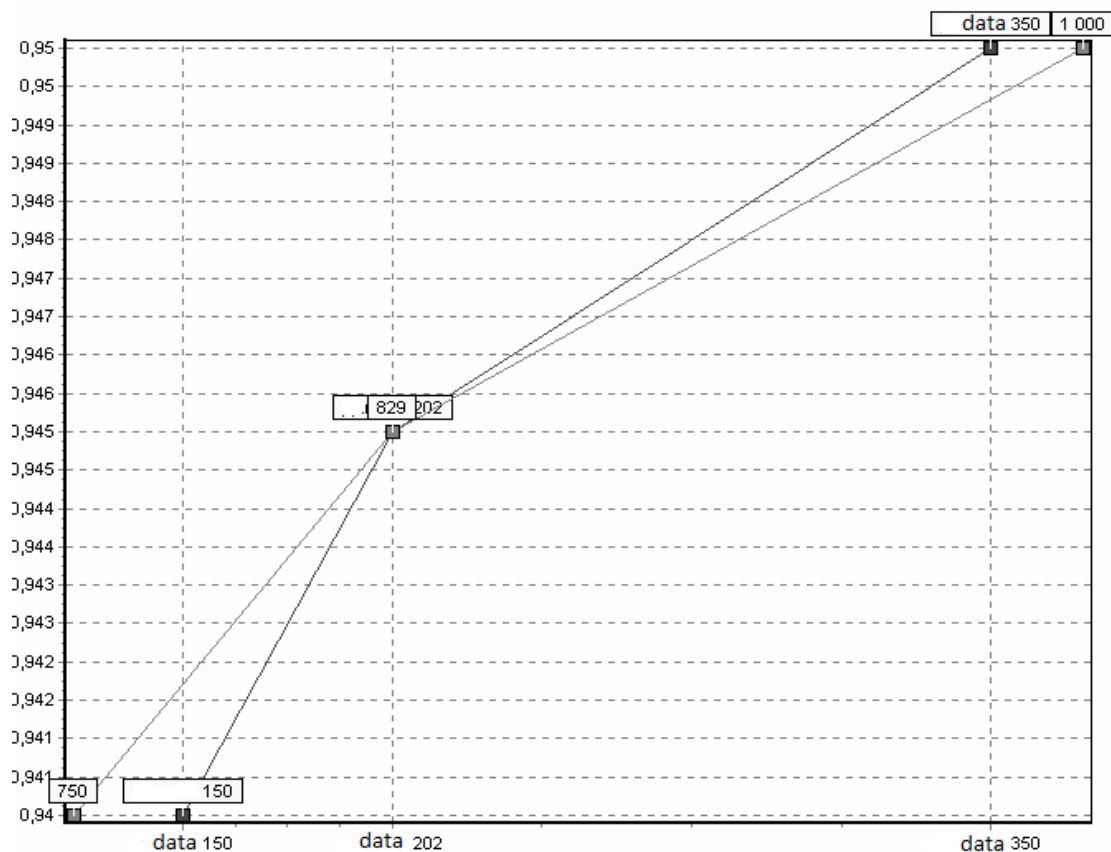
The dialog system of the architecture software reliability express-analysis [10] provides opportunities for more thorough architecture analysis. It can be presented as a graph where the change in the reliability characteristics of an architecture depending on the change of the amount of components on one of many architecture levels is shown. There is a possibility to change the amount of the functions and data for variant A. The dependence of the reliability parameters on the amount of the functions and data graphs are presented in "Fig." 5-7.



**Figure 5.** The graph of the dependence of the change in the length of system downtime on the change in the amount of the functions and data.



**Figure 6.** The graph of the dependence of the failure appearance time on the change in the amount of the functions and data.



**Figure 7.** The graph of the dependence of the system readiness on the change in the amount of the functions and data.

These graphs show that the length of system downtime, the failure appearance time and the system readiness are rising with the increase in the amount of the components. Using the system readiness graph as a generalized indicator it is possible to make the conclusion that the best results can be achieved by increasing the amount of components on the architectural level "Data". Under the condition of a small decrease in the amount of components, a decrease in the system readiness happens.

The conditional and unconditional probabilities of failure and recovery time as well as the usage time of the components are different depending on the amount of components. The express-analysis system can be used to estimate the software reliability for the possible architectural changes and to choose the most reliable architecture.

The possible selection under the condition of the development of complex software systems and the telecommunication, real-time software created using the client-server architecture is limited by many factors of the environment. These factors include the hardware architecture which cannot be changed, the architecture levels which cannot be divided or united, the code of a component which cannot be changed and many other limitations. Therefore, the subset of the possible changes in the software architecture in real environment usually includes a few variants. The model [10] can be applied to many variants to estimate the software reliability while choosing the best software development course.

### References.

- [1] Levendel Y 1990 Reliability analysis of large software systems: Defect data modeling *IEEE Trans. Software Engineering* **16** pp.141-152.



- [2] Kovalev I, Zelenkov P, Ognerubov S 2015 The efficiency analysis of automated lines of companies based on DEA Method. *Lecture notes in economics and mathematical systems* **675** pp. 107-115.
- [3] Kovalev I, Zelenkov P, Ognerubov S, Brezickaya V, Kovalev D 2014 The practical realization of the software architecture reliability analysis *Proceedings of the Work in Progress Session held in Connection with SEAA 2014 the 40th EUROMICRO Conference on Software Engineering and Advanced Applications and DSD 2014 the 17th EUROMICRO Conference on Digital System Design* (Verona (Italy), August 27-29, 2014) SEA-Publications: SEA-SR-40, pp. 17-18.
- [4] Kovalev I V, Zelenkov P V, Karaseva M V, Tsarev M Yu and Tsarev R Yu 2015 Model of the reliability analysis of the distributed computer systems with architecture "client-server" *IOP Conf. Ser.: Mater. Sci. Eng.* **70(1)** art. no. 012009, doi:10.1088/1757-899X/70/1/012009.
- [5] Wattanapongsakorn N 2001 Reliability optimization for software systems with multiple applications *FastAbstract ISSRE and Chillarege Corp. Copyright*.
- [6] Kovalev I, Younoussov R 2002 Fault tolerant software architecture creation Model based on reliability evaluation *Advances in Modeling & Analysis* vol.7 **3** pp 31-44.
- [7] Xie M, Yang B 2000 Regression goodness-of-fit test for software reliability model validation *FastAbstract ISSRE Copyright*.
- [8] Lyu M R 1996 Handbook of software reliability engineering *Edited by Michael R. Lyu Published by IEEE Computer Society Press and McGraw-Hill Book Company* 819 p.
- [9] Kovalev I, Soustin B, Kardashov V 1988 Multicriteria decision model for modular software structure selection *in proc. of the Intern. AMSE Conf., Lyon, France* pp 1.9-1.11.
- [10] Ashrafi N 1992 Optimization models for selection of programs, considering cost & reliability *IEEE Transaction on reliability* vol.41 **2** pp.281-287.
- [11] Zahedi F 1991 Software reliability allocation based on structure, utility, price, and cost *IEEE Trans. on Software Engineering* vol.17 **4** pp.345-356.
- [12] Kovalev I, Grosspietsch K 2000 Deriving the optimal structure of N-version software under resource requirements and cost *Timing Constraints," in proc. of Euromicro '2000, Maastricht, Netherlands* pp 200-207.
- [13] Kovalev I, Vasilenko N, Shabalin N, Davydov I 2000 Using a software reliability model in decision-making support systems for the software structure selection *Advances in Modeling & Analysis* vol.5 **1-2** pp 51-62.
- [14] Kovalev I, Popov A, Shipovalov Ju 2000 Optimization models for reliability of telecommunication software systems *Advances in Modeling & Analysis* vol.43 **3-4** pp 41-46.
- [15] Rosenberg L et al 1988 Software metrics and reliability *Software reliability engineering was presented at the 9-th International Symposium, "Best Paper" Award*.
- [16] Novoy A V, Shtentsel A V 2008 The estimation of reliability of the structure of multi-version software for the processing and control information systems *Vestnik SibGAU* **3(20)** pp. 73-79.
- [17] Kovalev I V, Zelenkov P V and Tsarev M Yu 2015 The control of developing a structure of a catastrophe-resistant system of information processing and control *IOP Conf. Ser.: Mater. Sci. Eng.* **70(1)** art. no. 012008, doi:10.1088/1757-899X/70/1/012008.