Model of the special software development and modification based on account factors of complexity, workability, qualification of programmers and resource limitations

Vitalii Fedoriienko *[1] [A]

[*]**Corresponding author:** [1] Senior Researcher, e-mail: delesh918@gmail.com, ORCID: 0000-0002-0921-3390

[A] National Defence University of Ukraine named after Ivan Cherniachovskyi, 28, Povitroflotsky, ave, Kyiv, 03049, Ukraine

Received: June 11, 2021 | Revised: June 22, 2021 | Accepted: June 30, 2021

DOI: 10.5281/zenodo.5544382

Abstract

The article includes a comprehensive model based on the functional dependence of the programming subject structural and parametric synthesis. The study includes special software as an object that is the result of modified application or standard software to propose the requirements of the particular industry, such as defense. After all, it is known that large software systems are created for various purposes, which must be high-quality, interoperable ets. This shows that information technology is becoming more widespread, and software is a key component. Such projects are costly, in which most of the funds are directed to the creation of the specific software product. Accordingly, the developing special software process is crucial.

The research is based on statistics obtained from the ERP system. These data were processed and based on regression models for three levels of complexity of the special software and for and three qualification classes of the programmers for each of them. For a better approximation, the approach of a separate study of two parts was used, which show the degradation and growth of software performance. The regression model obtained by the method of least squares is the sum of exponential and logistic mathematic functions. This describes the process of developing and modifying special software and relevant the nature of the phenomenon.

The parameters influencing the classification of programmers on the basis of the theory of program reliability for the construction of a similar stochastic dependence, which confirmed the adequacy of the regression function, are determined. The division into classes and refined coefficients of the deterministic function, taking into account the qualifications of performers, resource constraints made it possible to calculate the useful effect of special software based on the application of the hyperbolic tangent function. The obtained group dependences of the number of programmers allowed creating three-dimensional models in the limitations on the level of efficiency of special software, time, cost and structural composition of the team of three classes of programmers. Practical graph-analytical calculations on the use of the model of creation and modification of special software in the conditions of resource constraints are given, which allowed obtaining the range of optimal values of the rational selection of programmers in the team.

Key words: special software, software quality, resource limitations, programmer classes, program performance level.

Introduction

Today, informatization is carried out in various areas in the defense industry around the world, especially where program and organizational components play a key role. Trends that are the impetus for this development is a network-centric concept of combat operations. It includes the core based on a software platform to which various subsystems are connected on the principle of interoperability. The structure of construction for different NATO member countries is different, but the most common example is the C4ISR (Command, Control, Communications, Computers, Intelligence, Surveillance and Reconnaissance) system. It combines at the software level various systems, such as GIS (Geographic Information System), ERP (Enterprice Resource System). This requires the mandatory of programmers to further developing or adjusting (to modify or adapt) to national and industry needs. The amount of the refinement is due to the specifics of the industry itself. When organizations purchase the rights to use standard large-scale software (such as ERP) for a particular industry, usually one installation is not enough.

At the same time, the DevOps (development and operations) methodology has become widespread in the world, which includes a combination of a number of practices designed to revitalize the interaction of developers with IT professionals and bring their workflows closer to each other, and is based on the idea of the interdependence between the development and using of software to create and update programs and services swiftly. When creating complex software systems of narrow specification (defense or other industry) according to the methodology of software development described by industry standards, the stages of design, coding, testing, operation, including the processes of software development, implementation and maintenance. During these stages, the development, modification and evaluation of the quality of software code for the needs of a particular industry, it means that special software is created.

According to general statistics, about 70% of the total amount spent during the life cycle of the software product is its maintenance. And also 50-70% of the amount for its creation falls on the salaries of programmers. This raises questions about the skillful management of the software development and maintenance process. At the same time, there is ambiguity in the approaches to the selection of IT staff to support and maintain software systems with saving of the properties of quality, for instance, the reliability of the software component. These circumstances emphasize the importance of the researching on the organization process for development and modification of special software.

The relevance of this study is determined by the need to comprehensively create methods and models for managing the development and modification of the program code of special software. This study is especially relevant along with the trends of transition to Agile methodology of software lifecycle management, where flexible management of subjects and objects of the programming in terms of the resource constraints are carried out.

This study analyzes the experience of developing, implementing and maintaining complex software systems. It has been found that in the development, modification and evaluation of the quality of special software, the property of "qualification of programmers" is the key term. It is based on the analysis of existing research, the author found a contradiction between the need to ensure the required level of performance of special software, on the one hand, and the lack of existing methods of resource management, on the other. On the other hand, we have the contradiction that consider the process of improving of the methods for developing special software based on models that take into account the qualifications of programmers and resource constraints is **relevant**.

A significant contribution to the development of information technology to substantiate the theoretical and applied aspects of development and modification of special software was developed by A. Klimenko, S. Russell, G. Myers, T. Gilb, V. Golyan, K. Lavrishcheva, and others. But their research concerned the quality of software in terms of improving the reliability of software, the development of software engineering theory, and applied recommendations for the use of metrics in the creation of software. Sphere of management of the information systems with the use of optimization was investigated by V. Shevchenko, O. Barabash, Y. Kravchenko (Barabash, O., Kravchenko, Y., Mukhin, V., Kornaga, Y., Leshchenko, O., 2017; Shevchenko V. L., 2004).

Weinberg (Weinberg G. M.) conducted experiments in which several groups of programmers were given the same task, but different criterias for success were established. The author proposed to provide a basis for the study of human factors in software engineering, while summarizing the literature on the human factor in software engineering, but the search for better values in conditions of limited resources was not performed.

Ruhe's work (Ruhe C., 2019) on the basis of empirical research is devoted to general optimization issues. To obtain input data, Ruhe suggests using a checklist of managed processes, including to analyze (evaluate) return on investment (ROI) and forecast the cost of the decision, but the factors of consideration of the qualification level were not given.

The method presented by Klimenko (Klimenko, A., Klimenko, V., 2009) is devoted to solving the problem of optimizing the software development process by changing the number of developers. The disadvantage is the division between the two categories of a novice and an expert without group composite consideration of the qualification division by levels. As well as it is the lack of consideration of the impact of the profession of programming programmers on the speed and reliability of the formation of expert assessment of the quality of the prepared software.

In his work (Russell, S., Bennett, T. D., Ghosh, D., 2019), Russell focuses on software engineering in the medical field, in particular, on software testing and software optimization on the example of reproducible software tests of the R package (Comprehensive R Archive Network packages). The disadvantage is the disregard for the profession of the programmer who prepared the program in its evaluation.

The joint work of Kernighan B. W. and Plauger P. J., devoted to programming style (Kernighan B. W. Plauger P. J.) and is an example of practical programming assistance to achieve the highest quality performance of software. But the issues of resource constraints and the qualification division of specialists are superficial.

A similar idea was used in compiling a series of research reports from CIRAD (A Study of Fundamental Factors) on work in the field of software operational reliability. Among the programming methods are considered, for example, conceptual grouping, modular construction; as the characteristics of the software that determine the ease of its operation, called consciousness, uniformity, compactness, intelligibility and others. The disadvantage is that the issues of resource constraints and expert evaluation are superficial.

Myers' research is devoted to the problems of creating software metrics and analyzing its quality based on the criterion of reliability in the design of software systems (Myers G. J.). When assessing the reliability of the influence of the qualification factor of programmers is not fully disclosed.

Haltead's work on the basics of software theory (Haltead M. H.) summarizes successful research aimed at establishing and experimentally verifying the overall complexity of software as a function of the number of different operators and operands and the frequency of their use. But the influence of the factor of qualification of programmers when working with software of different complexity is not disclosed.

The purpose and objectives of the study: to increase the efficiency of the processes of development, modification and evaluation of the quality of special software, taking into account the qualifications of programmers and resource constraints.

Material and methods

Methods of observation, system analysis and expert evaluation were used to analyze the approaches to the qualification division of the programmers and approaches to the assessment of the quality of the special software. There are deterministic and stochastic research methods for using to study of the software development and modification processes. To build a mathematical model of the functional dependence of the level of recovery of programs used the methods of regression analysis and comparison. Methods of the theory of research of operations, mathematical modeling are used for construction of probabilistic dependences of working special software for *N* programmers. It was used for the development and modification of special software by different classes of programmers with resource constraints, optimization methods and system analysis during the development of models for finding the optimal solution.

Results and discussion

Analysis of software life cycle models. General processes of software development and modification during software life cycle (LC) are described in the standards ISO / IEC 12207, ISO / IEC 15404, ISO / IEC 9126, ISO / IEC 25010, etc., structured by levels and directions. But they are not related to the existing methods and tools of software engineering for software development. This makes it possible to compare them with the usual paradigms and methods of the development (object, component, service, etc.).

The software life cycle of software is a set of separate working stages that is carrying out in a given order during a period of time that begins with the decision on software development and finishes with the termination of software use (Lavryshcheva K. M.).

The main models of the life cycle are waterfall (cascade) or sequential (waterfall), model V, model Rapid Application Development (RAD), Agile, iterative, spiral (Boehm) model and the model of the evolutionary prototyping. Their analysis and purpose are given in (Golian, V., Kravchenko, O., 2019; Bhuvaneswar, T., Prabaharan, S., 2013). It is known that each of the models of the lifecycle software includes the main and additional stages of software development and modification. But in a number of analyzed algorithms of the lifecycle software creation – there is no factor of qualification division of the programmers during development, modification and quality check.

A programmer – is a professional who writes and adjusts programs for computers (Kvalifikatsiyni kharakterystyk) (any computing devices), i.e. programming. The breadth of the subject area and the quality of skills for each programmer are individual. Therefore, it is common practice to classify them by professional level.

The study will consider a programmer of different qualifications, who develops and modifies special software, which is a type of application software (Stasev, Y., Sidchenko, S., Zalkin, S., 2006). Special software is a set of programs that are developed to create information technology for a specific functional purpose.

In the course of the research, statistical data on the number of developed and modified running programs depending on the qualification level of programming subjects from ERP system logs were collected and processed. Next, a regression analysis of the function of the relative level of performance of special software on a given time interval by the method of the least squares, examples of programmers of II and III classes are shown in Fig.1.

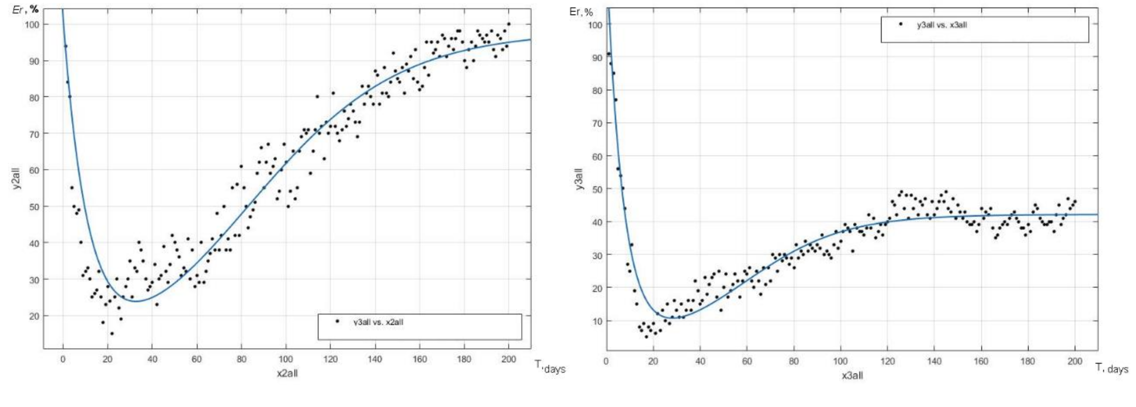


Figure 1 – Regression analysis of corrected errors of the special software by programmers of II (left) and III (right) classes for the whole period

The sum of exponential (for the first part of the curve – left to extremum) and logistic SL-type (for the second part of the curve – right of extremum) functions is recognized as a regression function that would correspond to the nature of the research (Shevchenko, V., Fedoriienko, V., Kirpichnikov, Y., Golovchenko, O., 2019). The first (exponential) function degrades, and the second (logistic) grows (fig. 1):

$$f(x) = a \cdot e^{(b \cdot x)} + \frac{s}{1+e^{-\frac{2}{t}\cdot(x-k)}}. \quad (1)$$

As a factor of gradation of complexity of the special software it is offered to use its division into three levels (s = 1 – a simple level of complexity, s = 2 – complex, s = 3 – the most complex). It is formed on the basis of quality metrics of hybrid software and load logs on the technical component of the system. The values of the coefficients of the regression function (1) for programmers of I, II and III classes are calculated for all three complexities of the SDR, an example for the most complex software is given in the table below.

Table – Parameters of function (1) for the dependence of the software performance on the qualification classes of programmers (for the complexity rate of the software s = 3)

Qualification class of the programmers (software complexity)	Coefficients of function (1), their values				
	a	b	s	t	k
I class (s=3)	103,7	-0,1453	98,74	41,61	58,35
II class (s=3)	93,08	-0,08997	97,99	68,50	81,66
III class (s=3)	118,00	-0,1415	42,16	43,26	57,94

A general algorithm and a model for assessing the reliability of special software software components of the information system were developed to verify the adequacy of the found regression functions. This model is based on the probabilistic dependence of running programs based on the method of queuing system with queue. The Monte Carlo method was used to generalize iterations with the section of programmers by class. The parameters of the intensity of errors in the special software programmer were proposed as: for the Class I: λinput = 0.1; Class II: λinput = 0.3; Class III: λinput = 0.7 (Shevchenko, V., Fedoriienko, V., Kirpichnikov Y. and others, 2013); Shevchenko, V., Fedoriienko, V., Kirpichnikov, Y., Golovchenko, O., 2014).

In general, no discrepancies were found between the model of special software performance and the simulation model of special software reliability according to the Monte Carlo method.

For the class III programmer, the initial complexity of the software is too high and it is impossible to bring it to the level of uncorrected errors (100% mark). Moreover, a very popular decision of the responsible persons to quickly launch special software into operation is the readiness of the software after recovery or time limit (Fig. 2).

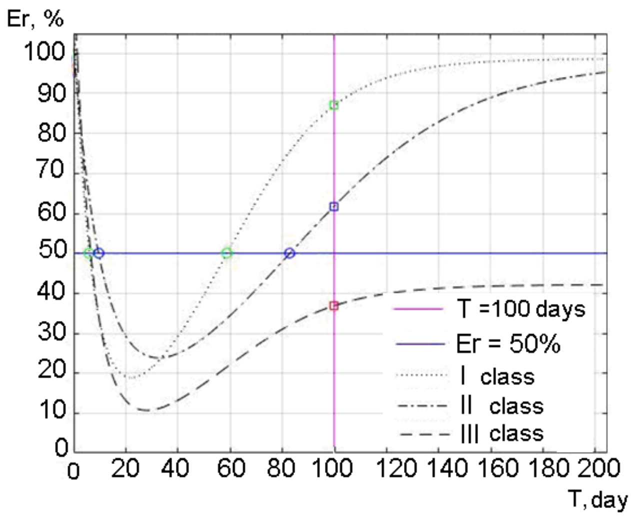


Figure 2 – Models of efficiency of special software, Er, depending on time T, (restriction T = 100 days), taking into account qualification of executors and complexity of the software s = 3 (restriction Er = 50%)

We present a time period in the form of an estimate of direct costs of programmers, which is more than 50% of the cost of the entire informatization project. This value corresponds to Agile sprints during software development.
The transition from resource constraint time to cost is equal to:

$$W = \text{ЗП} \cdot t + \text{УВ} \cdot t + f(t), \quad (2)$$

where W – is the cost of the project in money, *ЗП* – is the programmer's salary, t is the project execution time, *УВ* – is the lost benefit, $f(t) = const$ – are the constant additional payments. The found dependence (1) is listed in cost indicators and is shown in fig. 3.

Assume that there is a given number of programmers of the project team, it is necessary to determine their ratio by class. Let their total number be known and equal to 10 people. (the number is taken from the experiment based on the waterfall model of the software life cycle).

To determine the maximum productivity of the programmer, it is proposed to find the value of the partial useful effect, which is the highest level of development of the logistics function (1). Namely, the search for the ordinates of the points of contact. The use of the partial useful effect of recovery of special software is due to the subsequent transition of the logistic SL-shaped curve of the programmer of the *i*-th class in the saturated state in the upper part of the S-shaped asymptotic bend. Here the performance of the programmer is expected to fall. We take the value of the relative level of efficiency of programs as a useful effect. Then the total effect is calculated as:

$$Ef_0 = Ef_1 N_1 + Ef_2 N_2 + Ef_3 N_3, \quad (3)$$

where *Ef_1, Ef_2, Ef_3 – partial useful effects* of the programmers of different classes, or the value of Y of the maximum values of the effects of the curves SL_i of each of the three classes of programmers. This consistent approach allows you to see the growth of the effect of additive curtailment of achievements (recovery of programs) of each of the programmers.

Given the maximum number of programmers allowed in each of the classes, it is possible to plot the dependences of integer values using the hyperbolic tangent function. For a different number of programmers, we substitute the corrected values of the parameters of the logistics SL-like function, and, accordingly, we obtain the value of the hyperbolic tangent function for the logistics function, which will take the form:

$$f(\text{x})_0 = th(Ef_0). \quad (4)$$

Adding the values of the SL and n_N functions will allow you to search for *tanh(Ef)* functions with the different numbers of programmers.
For example, for the first-class programmers, the dependence of group benefits will take the form of Fig. 3.

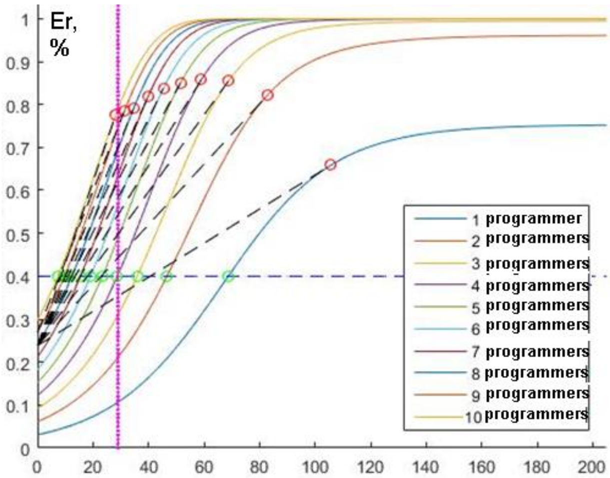


Figure 3 – Dependence of working programs of special software on the number of class I programmers with time constraints on decision-making to start work on the recovery of special software (vertical line)

Directly parallel to the ordinate axis (Fig. 4), shows the extremum of the point of dynamic dependence (1) of the performance of special software restored by a programmer of a certain class at the intersection with the curves.

Approbation of the model of dynamic dependence of recovery of special software by the programmer of the *i*-th class in value of cost indicators is expressed in application of the found partial useful effects of executors and includes calculations of the real data taken from typical ERP system which served as an environment for experiment.

To find the points of intersection, we equate the function of the hyperbolic tangent to the limiting direct level of efficiency $y = l$, where l = 0.5 = 50% (partial case of the considered condition) for each value $N_n = \{1,\ 10\}$.

$$y = th(SL_i \cdot N_n). \qquad (5)$$

To visualize the time and cost indicators, Fig. 4 shows a combined graph of the partial beneficial effects of dynamic and cost dependencies of three classes of programmers.

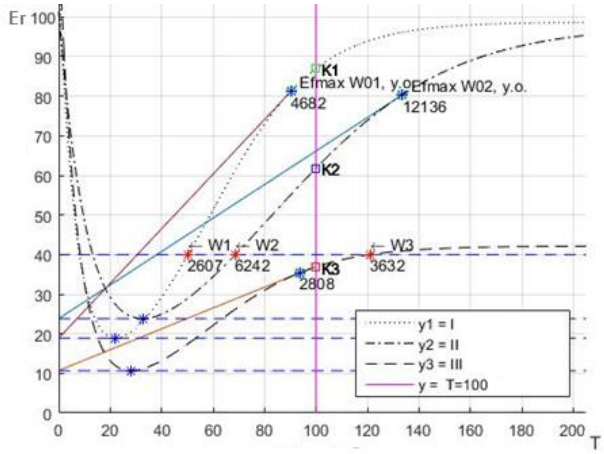


Figure 4 – Model for determining the maximum cost efficiency to ensure a given level of software performance, Er = 40%, T = 100 (days), for s = 3

The model of dependence of the spent calendar time on correction of errors in SDR on combinations of programmers of various specialty is shown in fig. 5. At the heart of three-dimensional graphs, the X and Y axes reflect the values of the number of programmers of I and II classes, Z – the values of the partial useful effects.

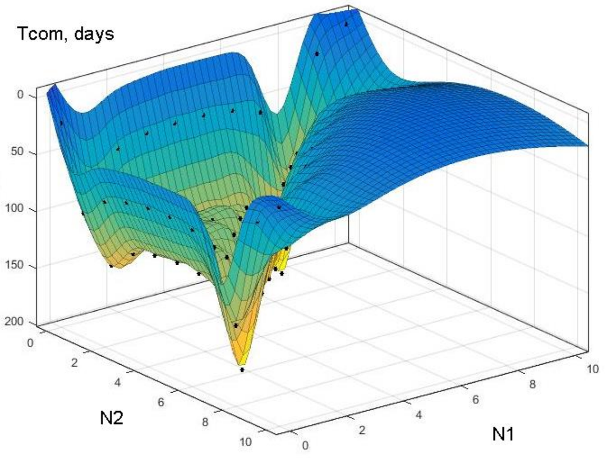


Figure 5 – Special software performance model (calendar time) depending on the structure of the programmer team (reversion)

For convenience of calculation of time and expenses, it is necessary to normalize indicators of calendar time:

$$Nr_t = T_{com} / T_{norm}, \qquad (6)$$

where Nr_t – is the time norm, T_{com} – calendar time, T_{norm} – normalized time.

Graphic models of development and modification of special software using the hyperbolic tangent function from the sum of known achievable effects of programmers in the indicators of the cost of regulatory time are shown in Fig. 6.

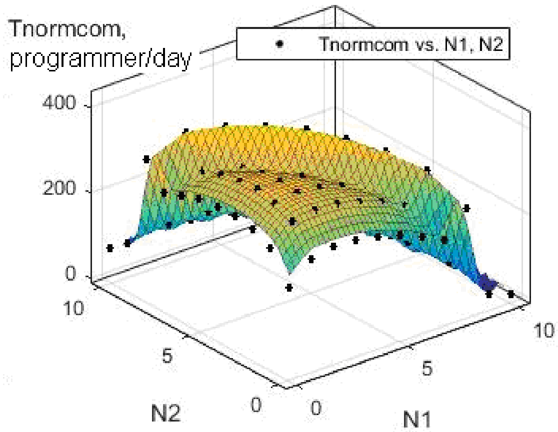
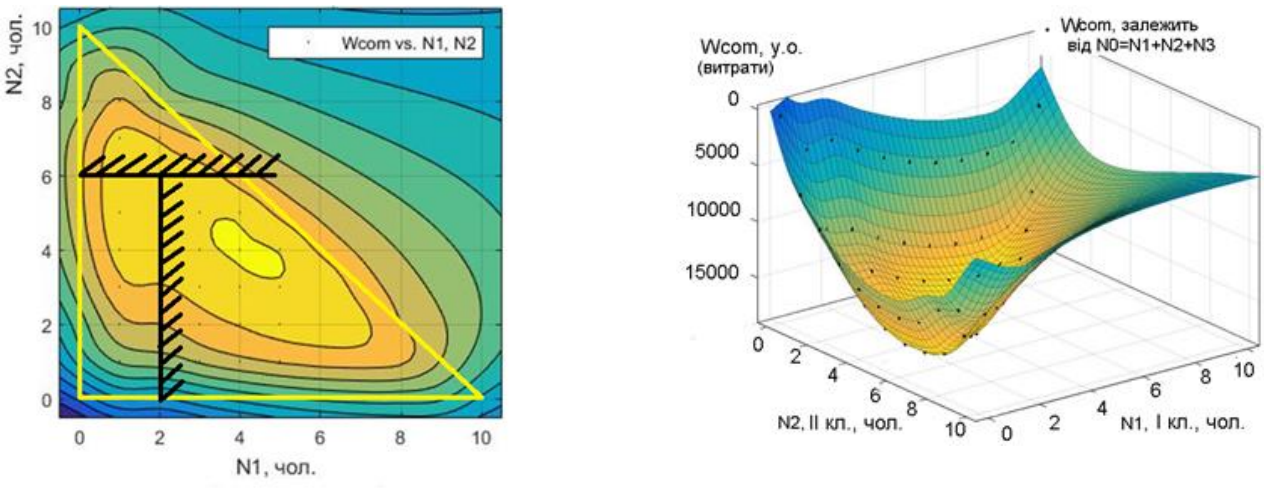


Figure 6 – Special software performance model (normalized time) depending on the structure of the team of the programmers

The completeness of the relief of the three-dimensional figure of the value dependence on the plane is revealed by the contour graph (Fig. 7 a). The Z axis (Fig. 7 b) shows the cost, *W*, of wages for different combinations of the programmers of the different professional level (classes) provided the total number of their working places in the team N_0 = 10 (programmers).

The search for the optimal team structure with restrictions on the size of individual classes of programmers (class I programmers should be N1≤2 (people), class II programmers should be N2≤6) in units of cost costs are shown in Fig. 7, a. The general view of the three-dimensional model is shown in Fig. 7, b.

Fig. 7 shows the search for the optimal command structure while limiting the size of the individual classes of the programmers.



a)

b)

Figure 7 – Contour type of the cost dependence on combinations of the programmers of three classes in one team (a) and reversible type of the dependence of the remuneration of the team of the programmers on error correction (b)

Conclusions

Thus, the model of development and modification of SDRs taking into account the qualifications of programmers and resource constraints is built taking into account five criteria (1 – the number of programmers in three qualification classes; 2 – software complexity; 3 – level of software performance; 4 – time; 5 – cost). The use of this model has reduced costs by choosing of the best option for the use of specialists and resources of the organization – time and money.

The constructed models first of all contain a practical component. Namely, the significance of the obtained results lies in the development of the specific engineering solutions using the proposed method and models to optimize the process of creating, modifying and accepting the results of the experts in the industrial special software operation. The presented models allow to choose the composite composition of the different programmers depending on the expected time and cost of special software development and modification, which allows to increase the level of special software implementation at the coding, testing and operation stages of the software lifecycle by reducing costs by 3-6% of the rational use of the programmers and resources of the organization's budget.

References

Shevchenko V. L. (2011). Optymizatsiyne modelyuvannya v stratehichnomu planuvanni. *Collection of the scientific paper of the Center for Military and Strategic Studies.* 2011. 283 s.

Shevchenko V. L. (2004). Yspol’zovanye SL-zavysymostey dlya postroenyya modeley razvytyya tekhnolohyy y uproshchenyya protsedur poyska optymal’nykh reshenyy na prymere uravnenyya Bellmana // Systemni tekhnolohiyi. Rehional’nyy mizhvuzivs’kyy zbirnyk naukovykh prats’. Vol.6 (35). Dnipropetrovs’k: DNVP "Systemni tekhnolohiyi". P. 148-153.

Barabash, O., Kravchenko, Y., Mukhin, V., Kornaga, Y., Leshchenko, O. (2017). Optimization of Parameters at SDN Technologie Networks. *International Journal of Intelligent Systems and Applications*. Vol. 9. № 9. Hong Kong: MECS

Publisher, 2017. P. 1 – 9.

Weinberg G. M. The Psychology of Improved Programmer Performance, Datamation, P. 82–85 (November 1986).

Ruhe, C. (2019). Optimizationin Software Engeneering: A Pragmatic Approach / Cuenther Ruhe. *Contemporary Empirical Methods in Software Engineering.* Cuenther Ruhe. – Calgary: The University of Calgary, 2019. pp. 235–261.

Klimenko, A., Klimenko, V. (2009). Metodika optimizatsii protsessa razrabotki softwarenogo obespecheniya. *Bulletin of the Taganrog Institute of Management and Economics*. 2009. №1. P. 108–117. – Resource access mode: https://ideas.repec.org/a/scn/030999/14050614.html.

Russell, S., Bennett, T. D., Ghosh, D. (2019) Software engineering principles to improve quality and performance of R software. *Peer Computer Science*. №5. С. 1–26. Resource access mode: https://peerj.com/articles/cs-175/.

Kernighan B. W. Plauger P. J. The Elements of Programming Style McGraw – Hill, 1984.

A Study of Fundamental Factors Underlying Software Maintenance Problems, CIRAD, Inc., December 1981.

Myers G. J. Software Reliability: Principles and Practices, Wiley. – New York, 1986.

Haltead M. H. Elements of Software Science, Elsevier North-Holland, Inc., 1987.

ISO/IEC/IEEE 12207:2017 Systems and software engineering — Software life cycle processes. / ISO/IEC JTC 1/SC 7 Software and systems engineering, 2014. 37 c.

Lavryshcheva K. M. Prohramna inzheneriya / Kateryna Mykhaliyivna Lavryshcheva. – Kyiv: National Academy of Sciences of Ukraine, 2008. – 319 p. – Resource access mode: http://cyb.univ.kiev.ua/library/books/lavrishcheva-6.pdf.

Golian,V., Kravchenko, O. (2019). Comparison of models of life cycle of software with the purpose of identifying the most effective. *Information Processing Systems*. №2157. P. 63–70. DOI: 10.30748/soi.2019.157.08/.

Bhuvaneswar, T., Prabaharan, S. (2013). A Survey on Software Development Life Cycle Models. *International Journal of Computer Science and Mobile Computing*. Vol. 2, Issue. 5. P. 262-267.

Kvalifikatsiyni kharakterystyk profesiy pratsivnykiv [Electronic resource]: 16.02.1998 N 24. – Official issue — Kyiv.: Ministry of Social Policy, 1998. – (Normative document of the Ministry of Labor and Social Policy of Ukraine. Handbook). 11 p.

Stasev, Y., Sidchenko, S., Zalkin, S. (2006). Special analysis and transformation of text information software. *Systems of Arms and Military Equipment*. №2. P. 79–83. – Resource access mode: http://www.hups.mil.gov.ua/periodic-app/article/944/eng.

Shevchenko, V., Fedoriienko, V., Kirpichnikov, Y., Golovchenko, O. (2019). Determination of the beneficial effect of the dynamic dependence of the performance of special software on the specialty of programmers. *Collection of the scientific paper of the Center for Military and Strategic Studies.* №1(65) P. 75-85. – Resource access mode: http://znp-cvsd.nuou.org.ua/article/view/178979

Shevchenko, V., Fedoriienko, V., Kirpichnikov Y. and others (2013). Approaches to assessing the reliability of the main software component of a unified information environment. *Collection of the scientific paper of the Center for Military and Strategic Studies.* №3. (49) P. 15–21. – Resource access mode: http://nbuv.gov.ua/UJRN/Znpcvsd_2013_3_4.

Shevchenko, V., Fedoriienko, V., Kirpichnikov, Y., Golovchenko, O. (2014). The model for evaluating the reliability of software components a single information environment. *Collection of the scientific paper of the Center for Military and Strategic Studies.* №1. – P. 144–151. – Resource access mode: http://nbuv.gov.ua/UJRN/Znpcvsd_2014_1_25.