UDC 004.4'22 + 658.5

Stanislav S. Velykodniy[1], Candidate of Technical Sciences, Associate Professor, Associate Professor of the Department of Information Technologies, E-mail: velykodniy@gmail.com,
ORCID ID: https://orcid.org/0000-0001-8590-7610
Zhanna V. Burlachenko[1], post-graduate student of the Department of Information Technologies,
E-mail: 7035373@ukr.net, ORCID ID: https://orcid.org/0000-0001-8451-5527
Svitlana S. Zaitseva-Velykodna[1], post-graduate student of the Department of Informatics,
E-mail: svetlana.zaytseva@gmail.com, ORCID: https://orcid.org/0000-0001-7453-8821
[1]Odessa State Environmental University, Odessa, Lvivska st. 15, Odessa, Ukraine, 65016

SOFTWARE FOR AUTOMATED DESIGN OF NETWORK GRAPHICS OF SOFTWARE SYSTEMS REENGINEERING

Annotation. *The subject of the work is the construction of a graphical network model of reengineering of the software system. Purpose of the paper is development of software for increasing the level of automation of designing network charts for the organization of production by reengineering software systems in the framework of project management. Network planning is one of the forms of graphical representation of the work content and the duration of implementation of strategic plans and long-term complexes of project, planning, organizational and other activities of the enterprise. Along with linear charts and table calculations, network planning methods are extensively used in the development of long-term plans and models for the creation of complex production systems and other objects of long-term use. The "Nodes-Activity" type of graphs is used in modern specialized packages of planning and operational management computer programs. The task before creating a software tool is the ability to work with all types of network charts with the possibilities of their comprehensive transformation. Methods. The article is based on methods of network planning for the PERT (Program Evaluation and Review Technique) methodology, the use of elements of graph theory and the Gantt chart method as an accounting method for project management. Simulation of the system software architecture is carried out within the UML (Unified Modeling Language) 2.5 methodology using the CASE toolkit Enterprise Architect 14. Project decisions proposed by the authors are the results of the article. The content of the design part is determined, firstly, by the specifics of the planning of software projects reengineering, and secondly, by the features of specific technical proposals for a project that is manageable. The architecture (project "Frame") software for managing network planning of software project reengineering is designed in the article. Conclusions. The architecture is developed in the form of several structural and behavioral diagrams, namely: use case diagram, which provides an analyst with a detailed idea of the software field of application; sequence diagram that is designed to create a programmer's imagination on how to perform actions when working with a future program tool; statechart diagram that is required for a visual representation of those states in which the software can be at different times; class diagrams that are used to design the main form filling of the future software; component diagram that is designed to examine the composition of the components of the future software and indicate the sequence of compilation and assembly of individual modules. The numerical and temporal estimation of the planning parameters is based on the data obtained from the Gantt design charts.*

Keywords: *project management; graph; network schedule; software; reengineering; CASE-tool; UML-diagram*

Introduction

The process of a project creation, prototype, and reimage of the future object, condition and methods of its production is called design. In engineering, design is understood as the development of project, construction and other technical documentation designed to provide for the creation of new types and patterns. In designing a systematic approach is used, which consists in the establishment of the structure of the system, such as links, defined attributes, and the analysis of the effects of the environment. In the design process, technical and economic calculations, charts, graphs, explanatory notes, estimates, calculations and descriptions are made.

One of the main components of design is planning, defined as a pre-scheduled procedure or an optimal allocation of resources necessary to achieve the goal.

Network planning is one of the forms of graphical representation of the content of work and the duration of implementation of strategic plans and long-term complexes of project, planning, organizational and other activities of the enterprise.

Along with linear charts and table calculations, network planning methods are widely used in the development of perspective plans and models for the creation of complex production systems and other objects of long-term use. Network plans of enterprises to create new competitive products include not only

 DOI://10.15276/hait.02.2019.2
ISSN 2663-0176 (Print)
ISSN 2663-7731 (Online)

the total duration of the whole complex of design and production and financial and economic activities, but also the duration and sequence of the implementation of individual processes or stages, as well as the need for the necessary economic resources.

Analysis of research and publications

For the first time, scheduled plan were developed in 1958 by the consulting company Booz, Allen, and Hamilton, together with Lockheed Corporation, at the request of the subunit of the United States Navy Special Projects Office (US Navy) for the project to create the Polaris missile system, which was the answer to the crisis that occurred after the launch of the first space satellite by the Soviet Union [1].

According to [2], the concept of "network graph" is a dynamic model of the production process, which reflects the technological dependence and sequence of the implementation of a complex of works that coordinates their completion in time, taking into account the cost of resources and the cost of work with the allocation of at the same time narrow (critical) places. At the same time, the network graph is a graph the vertices of which reflect the conditions of a certain object (for example, construction), and arcs are the works carried out on this object [3]. Each arc compares the time during which work is carried out and/or the number of workers who carry out work. Often, the network graph is constructed so that the location of the vertices horizontally corresponds to the time of reaching the condition corresponding to the given vertices (popular component of the PERT methodology – Program (Project) Evaluation and Review Technique).

The network schedule is based on the use of a mathematical model – a graph. Graphs (outdated synonyms: network, maze, map, etc.) are called by the mathematicians “a set of vertices and a set of ordered or disordered pairs of vertices” [4]. Speaking more familiar to the designer (but less accurate) language, the graph is a set of circles (rectangles, triangles, etc.) connected by directional or non-directional segments. In this case, the circles themselves (or other shapes used) in the terminology of the theory of graphs will be called “vertices”, and connecting their non-directional segments – “edges” directed (arrows) – “arcs”. If all the segments are directed – the graph is called oriented, if not directed – non-oriented [5]. Any sequence of work in a network graph, in which the eventual event of each work of this sequence coincides with the initial event of the subsequent work, is called the path.

Graph is a collection of objects with links between them [6]. Objects are considered as vertices, or nodes of the graph, and the links – as arcs, or edges [7]. For different areas of use, the types of graphs may differ in orientation, limitations on the number of links and additional data about the vertices or edges. A large number of structures of practical value in mathematics and computer science can be represented by graphs [8]. For example, the structure of Wikipedia can be modeled using an oriented graph, in which the vertices are articles, and arcs (oriented edges) – links to other articles.

Several standardized languages are used to describe the graphs for purposes suitable for machine processing and, at the same time, convenient for human perception – several standardized languages are used [9], among which are: DOT (language), GraphML, Trivial Graph Format, GML, GXL, XGMML, DGML, and Java [10]. There should be noted specialized programs for constructing graphs [11]. The most successful include the commercial ILOG, GoView, Lassalle AddFlow, LEDA. Free of charge, you can note the Boost Graph Library. There can be used Graphviz to visualize graphs (according to experts, it works well for orgraphs) or LION Graph Visualized [12]. Special attention shall be given to the program Graph Analyzer – a Russian-language program, with a very simple user interface [13].

Gantt chart (tape diagram, bar chart) is a popular type of diagram used to illustrate a plan, schedule work for any project. These diagrams are one of the methods of project planning and management [14-16]. The first chart format was developed by Henry L. Gantt (1861-1919) in 1910. Gantt Chart is a section (graphic spacers) placed on a horizontal time scale. Each section corresponds to a specific task or subtask. Tasks and subtasks, components of the plan, are placed vertically. The beginning, end, and length of the segment on the timeline correspond to the beginning, end, and duration of the task. Some diagrams of Gantt also show the relationship between tasks. The diagram can be used to represent the current status of the work: the part of the rectangle corresponding to the task is dazzled, noting the percentage of the task; shows the vertical line corresponding to the "today" moment [17]. Often, the Gantt chart is used in conjunction with a worksheet whose rows correspond to a particular task depicted in the diagram, and the columns contain additional information about the task [18].

PERT – technique of evaluation and analysis of programs (projects) used in project management [19;

ISSN 2663-0176 (Print)
ISSN 2663-7731 (Online)

20]. PERT is a way of analyzing the tasks required to complete a project, in particular, analyzing the time needed to complete each individual task, as well as determining the time minimum required for the entire project [21]. PERT has been developed primarily to simplify paper scheduling and create schedules for large and complex projects [22]. PERT is intended for large-scale, unique, complicated, unwritten projects. The method assumed the existence of uncertainty, giving the opportunity to develop a work schedule of the project without the exact knowledge of the details and the required time for all its components. The most popular part of PERT is a critical path method based on building of a network graph (PERT network diagram) [23].

The critical path method is an effective tool for scheduling and managing project terms. At the core of the method is the definition of the longest sequence of tasks from the beginning of the project to its completion, taking into account their interconnection. The tasks that lie on the critical path (critical tasks) have a zero run-time reserve and in case of changing their duration, the terms of the whole project change. In this regard, in carrying out the project critical tasks require more thorough monitoring, in particular, timely detection of problems and risks affecting the timing of their implementation and, therefore, on the terms of the project as a whole [24]. In the process of project implementation, the critical path of a project may change because, when changing the duration of tasks, some of them can be in a critical path. The most famous part of the PERT is the diagrams of interconnections between works and events. PERT suggests using chart diagrams with works on nodes, works on arrows (network graphs), as well as Gantt charts.

Determination of previously unsettled parts of the general problem

The general content of the graphs is as follows: linear or ribbon graphs on a horizontal axis at a selected time scale, postponed the duration of work in all stages and phase of production, the content of the work cycles depicted on the vertical axis with the required degree of their division into separate parts or elements.

At domestic enterprises, cycles or linear charts, as well, are used in the process of short-term or operative planning of production activities. The main disadvantage of such plans-schedules is the lack of the possibility of close interconnection of individual works in a single production system or the overall process of achieving the planned end-goals of the company (firm). Network graphs serve not only to plan a variety of long-term jobs, but also for their coordination between executives and project implementers [25], and network charts are needed to determine the necessary productive resources and their rational use.

Automated Enterprise Resource Planning (ERP) systems typically include computer programs [26] that automate some of the steps in drawing up and updating network graphs in one form or another, but such programs have a fairly high licensed price and are not suitable for a domestic producer.

Thus, the **purpose of the work** is to development of software for increasing the level of automation of designing network charts for the organization of production by reengineering software systems in the framework of project management.

The object of work is the network planning of the production process.

The subject of the work is the construction of a graphical network model of reengineering of the software system.

The task before creating a software tool is the ability to work with all types of network charts with the possibilities of their comprehensive transformation.

Materials and methods

The article deals with network planning for the PERT methodology, use of elements of graph theory and Gantt chart method as an accounting method for project management. The simulation of the system software architecture is carried out within the UML (Unified Modeling Language) 2.5 methodology using the Enterprise Architect 14 CASE.

Research results

The development of the system architecture of a software tool for network planning management of a software project reengineering (hereinafter referred to as software) shall be started with design of a Use-Case Diagram (UCD). UCD's are used to provide an analyst with a detailed picture of the application development industry. With the help of UCD, it becomes clear what the product is intended for, subsystems and modules it operates, links which are the elements and the entity in it. The central element of the projected UCD is a network graph (NG), depicted as an actor with a stereotype "business actor" (Fig. 1). The actor "NG" has a variety of connections, most of which associations with the stereotype "uses", but also there are dependencies. It can be seen from UCD that NG's are used in all applications associated with the list mentioned.

Let's turn to the consideration of the essence, which is connected with the NG type of connection dependence "dependency". On the diagram of use

case it is a package “Software Development”, the implementation of which is really based (depends) on NG. The composition of the package includes many entities, united by links, the collection of which is the enclosed designed subdigraph of the diagram of use case (Fig. 2), which has a graph similar to the topology.

Software engineering and computer science in general are one of the areas where NG is most often used; therefore the package is submitted and presented as a separate dependent structure (Fig. 2). The complexity and the large number of modules and protocols in modern software products greatly complicate the understanding of their work, management and optimization. Therefore, NG programs are often compiled, and most often it is done automatically by assemblers, compilers, or parsers.

Sequence Diagrams (SD) are designed to create a programmer's imagination on how to perform actions when working with future software, which are formed by the system architect. There are, as a rule, two types of SDs: the SD for displaying programmer actions when developing the software and the SD for displaying user actions while continuing to work with future software. SDs of the second type is used to create user instructions that are an integral part of software and methodological complexes.

This type of SD is designed in the given article (Fig. 3), since for the implementation of the reengineering of a software project, it is necessary to understand the principles of the software work.

State Machine Diagram (SMD) is required for a visual representation of those states of software in which it can be at different times. In other words, we can say that states describe the behavior of software, which, in turn, can depend both on user instructions and on the computational procedures of the software itself.

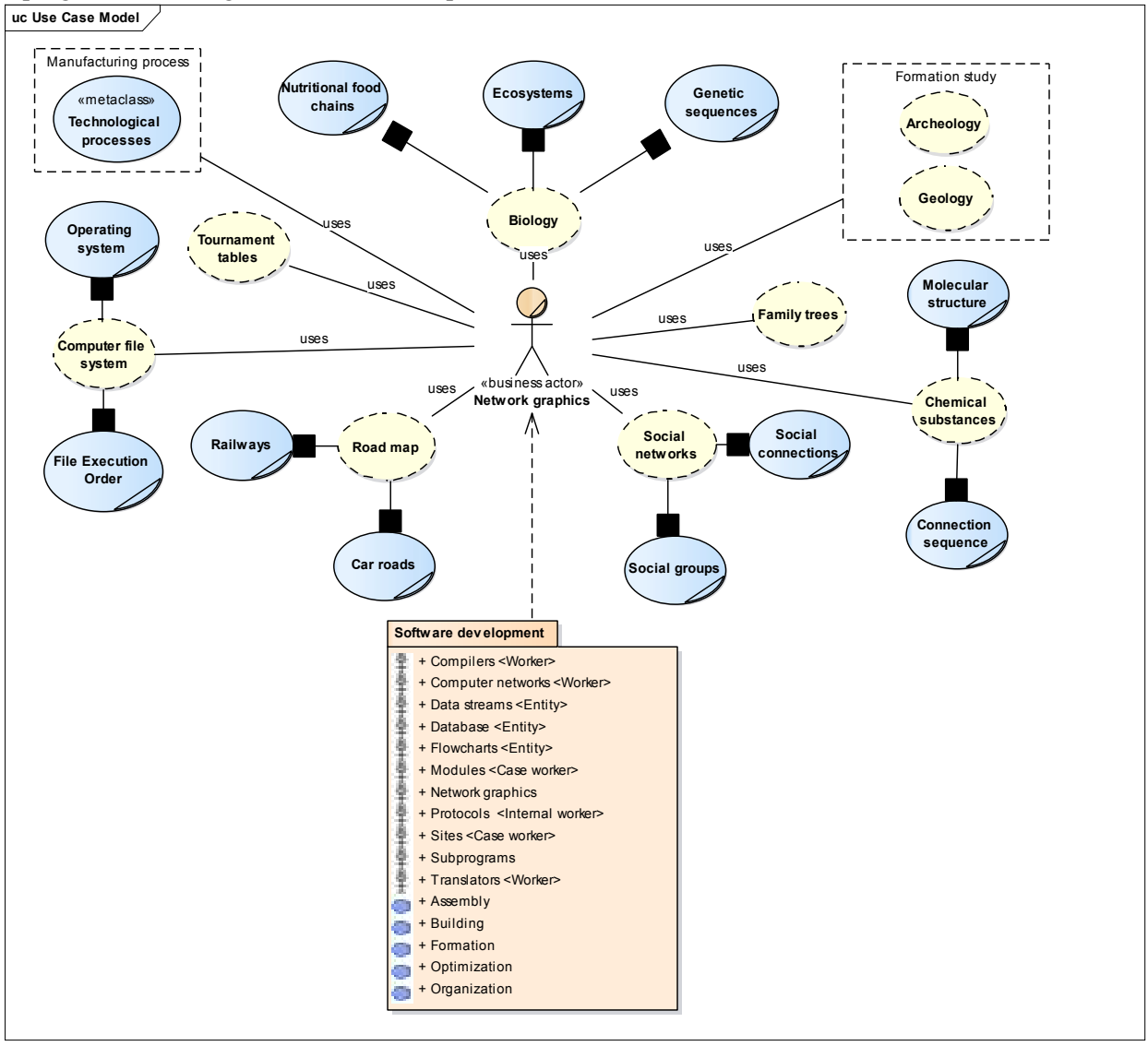

Fig. 1. Use-Case Diagram

ISSN 2663-0176 (Print)
ISSN 2663-7731 (Online)

As can be seen from the name of the chart – the key points in it – are state (state), representing, of its kind, the point of stopping the work of the software or the collection of statistical and technical information (logging). Each state has entry points and exit to this state, and there may be several such points (both for inputs and outputs) – precisely for fixing these positions within the state – there is such a concept as the "state history", which optionally turns on in the specification of each state and has the designation: the letter "H" (history), which is presented in a circle.

In addition, important for the SMD, as well, there are transitions, which are the relationships between states. Each transition has a syntax that reveals the mathematical or physical process that caused this transition. In turn, the syntax, in addition to the name of the transition, may contain a limiting condition (it is given in rectangular brackets [ ]) – in case of failure of which – the transition is impossible.

The projected DS for software is shown in Fig. 4. The DS, as shown in the figure, is a cascade model of the design of the software, where, as a gradual stage of the implementation of the states, the main diagonal of the diagram with many branches (depending on the points of exit from the concrete states) is in different directions.

Activity diagram (AD) is intended to reflect the activity of a particular object, during the analysis of the action of interest. With the help of AD, you can research and protocol the flow of information flows that are required for the creation of further software.

Activity diagram emulate some aspects of the SD and SCD, but only in AD is the ability to design complete algorithmic cycles, based on which the blocks "Decision" will stand.

The main essence of AD is the so-called activity (Activity), which is the essence similar to the state of the SCD. Activity shows the specific actions of the software process. Also, on AD there are transitions that have the same nature as the transitions to the SCD, with the same limiting conditions. In addition, there are activity paths on AD, which indicate the allocation of boundaries and activity zones of each of the objects.

Designed AD for the main information flow ("I-Flow") of software design of NG is shown in Fig. 5, as activity paths "User", "Interface" and "Compute and Memory Area". Their names are in the upper part of AD, and the activity limits are separated by solid lines.

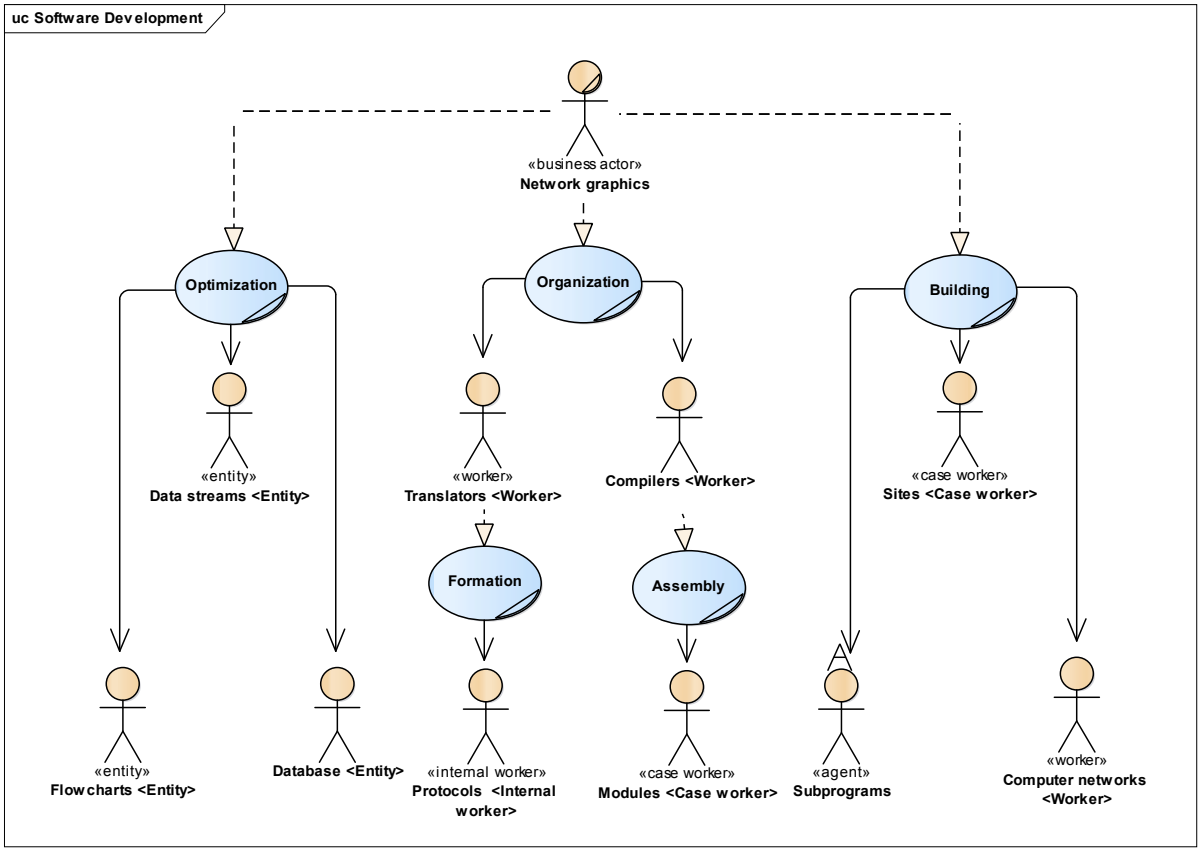

Fig. 2. Diagram-Package "Software Development"

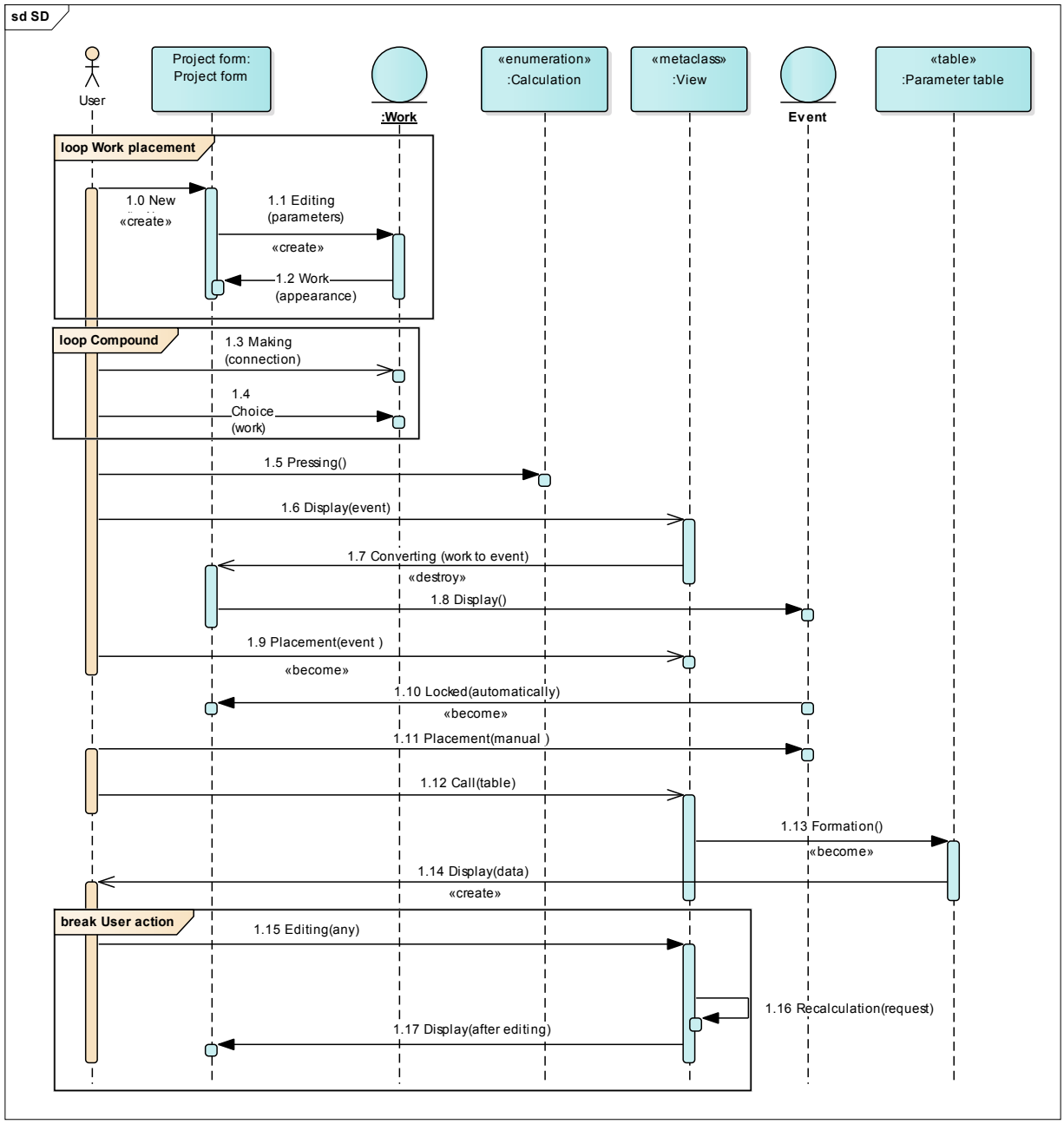

Fig. 3. Software Sequence Diagram

Class diagrams (CD) are used to design the main form of contents of future software. As for UML notation, classes contain attributes (self-encapsulated data of different origin) and operations (actions on these or other data).

Each specific attributes and class operations that are designed in the provided software are shown in Fig. 6. In addition to the classes on the CD, connections or relationships are also important for analysis (this is a more precise definition for CD). In general, entities or relationships, in relation to the methodology of CD – there is a rich diversity, but let us dwell only on those that are present in the designed CD of software (Fig. 6) and need additional clarification.

The central class on the CD is a class interface ("Interface UMP"), which is graphical user interface software. This class first appears to the user in the form of a bootable primitive class "Project form", which has a compositional relationship with the class "Interface" and defines the primary geometric dimensions of the interface window. Also in front of the user will appear a menu bar, whose elements – are attributes of the class "Interface UMP".

ISSN 2663-0176 (Print)
ISSN 2663-7731 (Online)

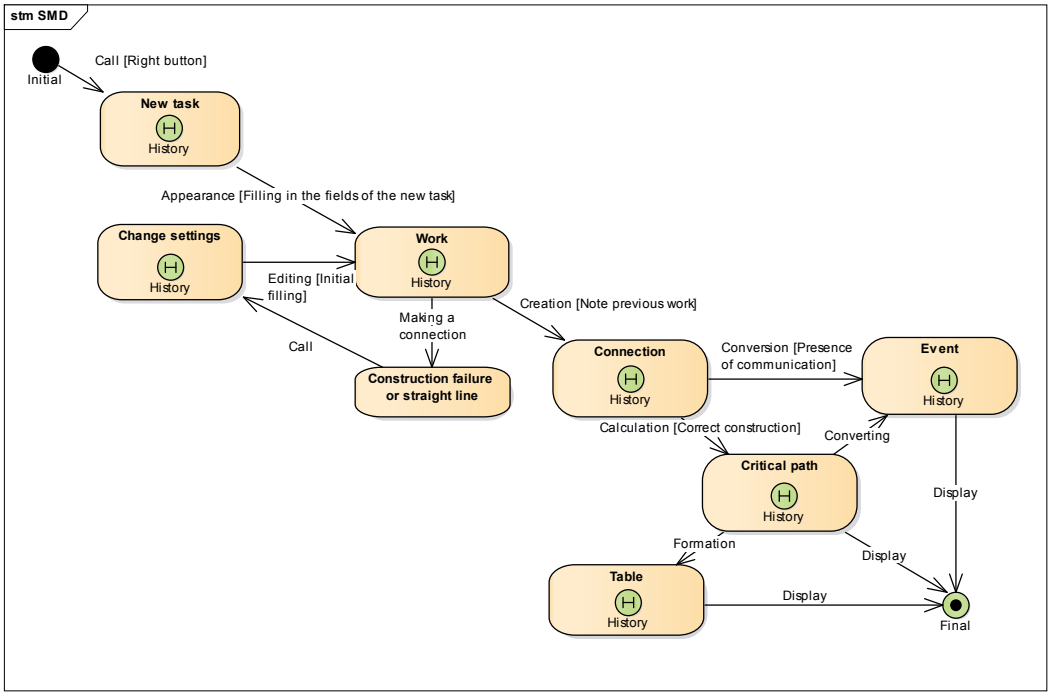

Fig. 4. State Machine Diagram for software

Calculation class (enumeration) “Calculation”, which is related to the compositional relationship with the “Interface UMP” class, implements the numerical (tabular representation parameters) and graphical (critical path) of the NG parameters.

The “New Task” class attached (relation “Bind”) to “Project Forms” is intended to create new and subsequent edits of an existing data type “Work”, which is a graphical representation of the “Work” contents.

The nesting class “Type of Work” is attached to the “New Task” class. It forms the types of work that differ in the presence or absence of incoming and outgoing links.

Component diagram (CPD) is intended to study the composition of future software components and to indicate the sequence of compilation and assembly of individual modules. The main requirement for CPD is standardized – there are no cycles, that is, the consistency of the components should be clear and transparent. The programmer works with the components that may be available to him – in reverse order.

Designed by Component diagram software is shown in Fig. 7

Detailed study shall be applied to the structure of Component diagram. The first component that the programmer deals with is the artifact or the shortcut for the “Link for UMP (*. lnk)” attached to the executable “UMP.exe” that launches the new project through the exposed interface components “Interface Visibility”.

This component serves as the interface to the “Interface UMP”, which includes port help and port driver. Port help has connected to it document aircraft “Help index . Gid”, which may be optionally called by the user in case of a complicated situation with the project. Through the Port Driver – an artifact containing special data is connected – Device Driver Profile “Device Driver . ddp”. From this driver, the mode and support of the solving ability of the entire software interface depends.

Also, the “Device Driver . ddp” component depends on the form of the Delphi Module “Module Form . dfm” and the package component of the “Delphi Compiled Module . dcu”, which also depends on “Module Form . dfm” . Thus, the package component “Compiled module Delphi (*.dcu)” has a double arrangement (Fig. 7).

The package component “Compiled module Delphi (*.dcu)” is main for the range of the components, namely for:

a) executable file “UMP . exe”, described above;
b) document artifact “Status Information”, which contains technical information on the current and final state of the project;
c) object “Default project . dpr,” which is downloaded as a template when creating a new project and which (if necessary) can be modified and supplemented.

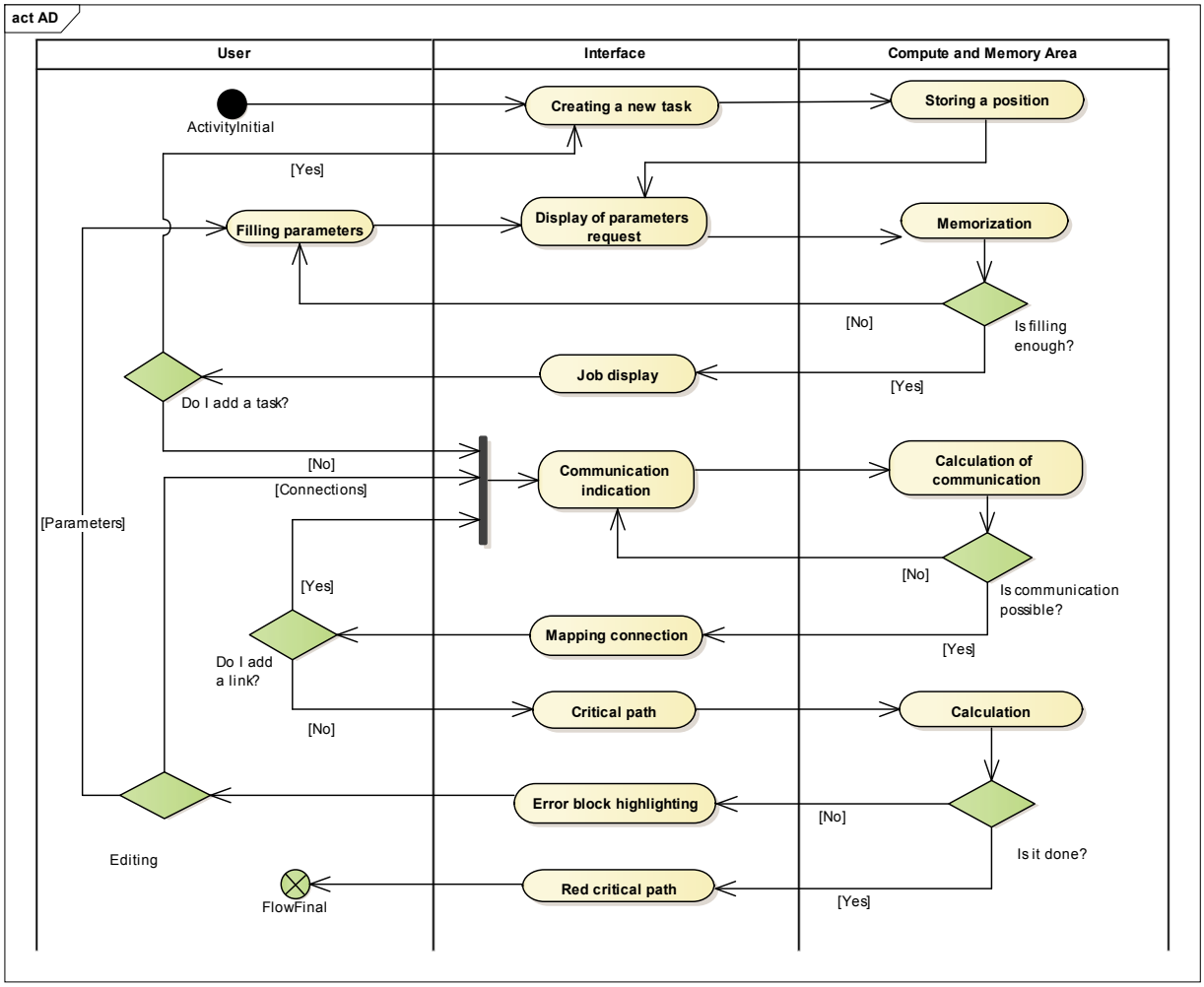

Fig. 5 Activity Diagram of the main information flow

Cumulative database component “Dbase resource . res” has double arrangement and is compiled each time on addition or change of information in document artifact “Status Information” and object “Default project . dpr”.

Eexecutable file “UMP. exe” is main for:

a) artifact or contact shortcut “Link for UMP (* . lnk)” described above;
b) document artifact software options and parameters “Options . ini”, which contains technical information on the latest geometric size of the project form set by the user;
c) Web-document artifact “Graf . htm”, which reflects the final formed topology of the placement of tasks on NG; thus Web-document artifact “Graf . htm” forms (realize) the object itself “Graf . bmp” as an independent (standalone) bmp-file that can be saved (printed, sent, etc.) and edited.

Thus, document artifact “Options. ini” and Web-document artifact “Graf. Htm” are arranged as abstraction dependency, which means creating them as essences only for the needs of the user and with the necessity of his participation (making changes in size, printing teams or Web-conversion.

Discussion

When analyzing and discussing the results of the project, system architects expressed a lot of thoughts about the means of implementing the software. These thoughts related to the technology and implementation language, the number of encoders and testers involved, etc., but all of this is part of a new phase in the implementation of the software. At this stage, regardless of the choice of means, languages and software coding technologies, its architecture has already been developed and ready. When you select programming languages, you will only need to process the component diagram.

ISSN 2663-0176 (Print)
ISSN 2663-7731 (Online)

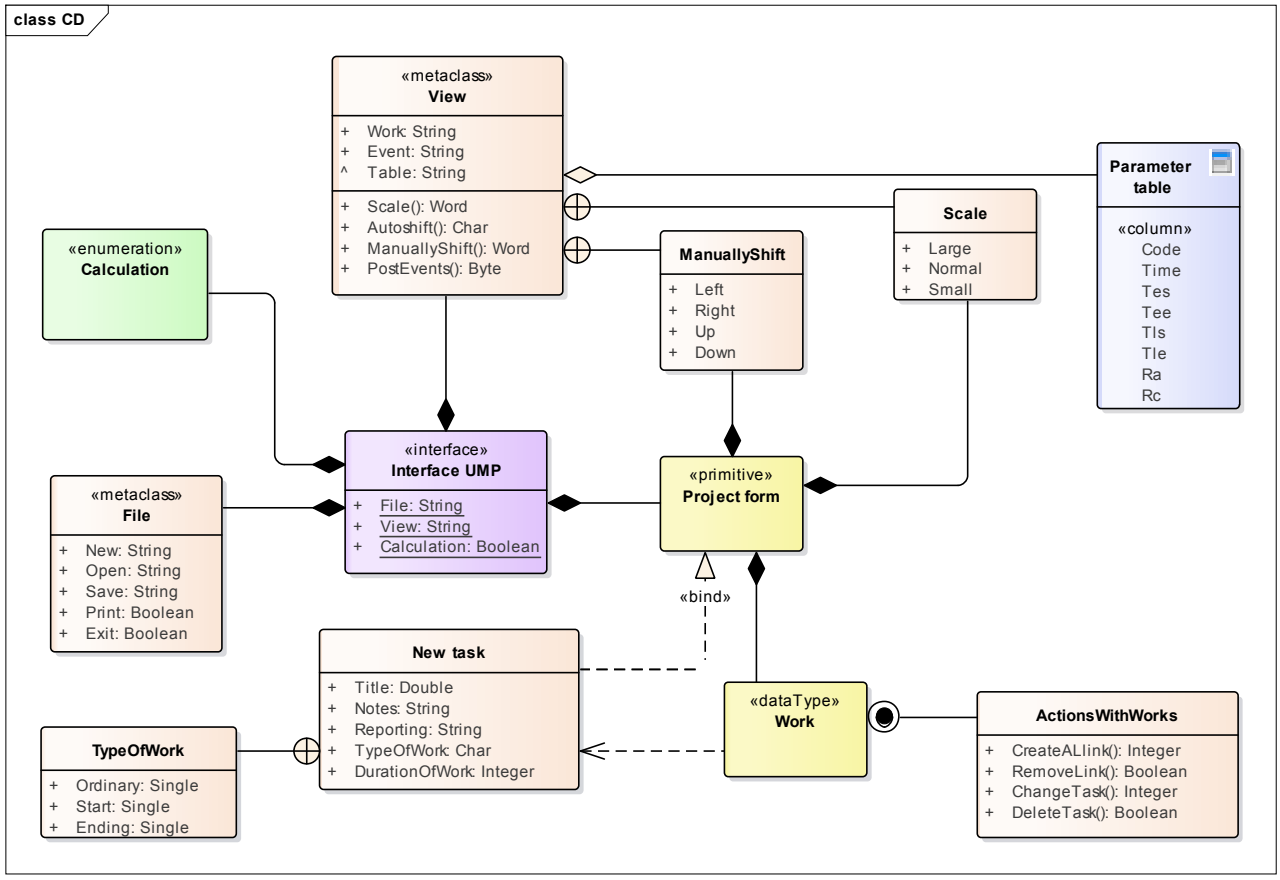

Fig. 6. Class Diagram

Conclusions and perspectives of the further development

It should be noted that although in today's paid specialist packages of computer programs of planning and operational management, the type of "Activity-on-arrow network" graphs is used mainly – the software designed is suitable for all types of network graphs with the possibilities of their comprehensive transformation.

The result of this article is a project decision suggested by the authors. The content of the design part is determined, firstly, by the specifics of the planning of reengineering of software projects, and secondly, the features of specific technical proposals for a project that is manageable.

In the given article the designer of the software for management of network planning of reengineering of the software project has been designed. The architecture is developed in the form of several diagrams of various natures, executed with the observance of UML 2.5 notation using the CASE toolkit Enterprise Architect 14. The development of the basis includes the methods of network planning for the PERT methodology and the use of the elements of the theory of graphs. The numerical and temporal estimation of the planning parameters is based on the data obtained by the Gantt chart method, as an account for the management of software projects.

The prospects for creating this software consists in the product encoding for a final industry user (software project manager), which is important to know only the sequence of reprogramming work on the software system and the duration of each stage, and it does not matter how the graph is formed, that is, which it type, since the type of network graphics itself can be mutually converted.

For work with the software will be created a program and methodical complex, which will be developed by user instructions on the application of the necessary software, supplemented by comments on the work of the software designed.

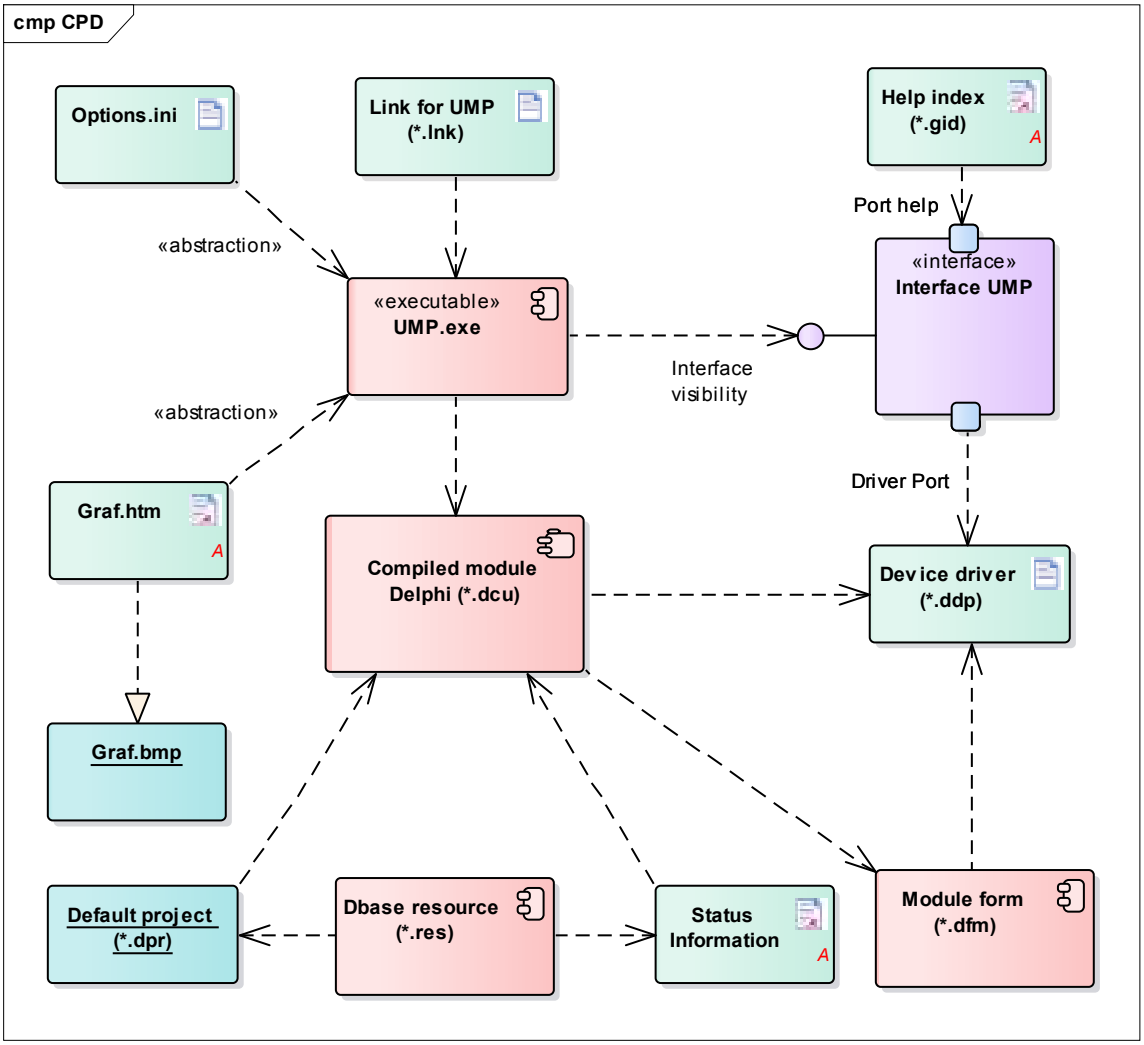

Fig. 7. Component Diagram

References

1. (2019). Program Evaluation and Review Technique (PERT), [Electronic Resource]. – Access mode: https://www.inc.com/encyclopedia/program-evaluation-and-review-technique-pert.html. – Active link – 01.04.2019.

2. Glushkov, V. M., Amosov, N. M., & Artemenko, I. A. (1974). Entsiklopediya kibernetiki, [Cybernetics Encyclopedia], Vol. 2, Kiev, Ukraine, *Publ. Glavnaya redaktsiya USE,* 624 p. (in Russian).

3. Berezina, L. Yu. (1979). Grafy i ikh primenenie: posobie dlya uchiteley, [Graphs and their application: a manual for teachers], Moscow, Russian Federation, *Publ. Prosveshcheniye,* 143 p. (in Russian).

4. Ore, O. (1968). Teoriya grafov, [Graph theory], Moscow, Russian Federation, *Publ. Nauka,* 336 p. (in Russian).

5. Uilson, R. (1977). Vvedenie v teoriyu grafov, [Introduction to graph theory], Moscow, Russian Federation, *Publ. Mir,* 208 p. (in Russian).

6. Bondy, J. A. & Murty, U. S. R. (1986). “Graph Theory with Applications”, New York, *Publ. North-Holland*, 270 p.

7. Bondy, J. A. & Murty, U. S. R. (2008). “Graph Theory”. San Francisco, *Publ. Springer,* 655 p., doi: https://doi.org/10.1007/978-1-84628-970-5.

8. Emelichev, V. A., Mel'nikov, O. I., & Sarvanov, V. I., (2009). Lektsii po teorii grafov, Izd. 2, [Lectures on graph theory], 2nd ed., Moscow, Russian Federation, *Publ. Nauka*, 392 p. (in Russian).

ISSN 2663-0176 (Print)
ISSN 2663-7731 (Online)

9. Jungnickel, D. (2013). “Graphs, Networks and Algorithms”, 4th ed., Berlin, *Publ. Springer,* 677 p., doi: https://doi.org/10.1007/978-3-642-32278-5.

10. Sedgewick, R. (2003). “Algorithms in Java”, 3rd ed., Part 5, Graph Algorithms, Boston, *Publ. Addison Wesley*, 528 p.

11. (2019). Kommercheskie spetsializirovannye programmy dlya postroeniya grafov, [Commercial specialized programs for graph construction], [Electronic Resource]. – Access mode: http://www.boost.org/. – Active link – 02.04.2019 (in Russian).

12. (2019). LION Graph Visualizer, [Electronic Resource]. – Access mode: http://lion.disi.unitn.it/intelligent-optimization/visualizer.html. – Active link – 03.04.2019.

13. (2019). Grafoanalizator – sreda vizualizatsii grafov, [Graph analyzer – graph visualization environment], [Electronic Resource]. – Access mode: http://grafoanalizator.unick-soft.ru/. – Active link – 03.04.2019 (in Russian).

14. Jalote, P. (2005). “Software project management in practice”, Indianapolis, *Publ. Addison-Wisley,* 242 p.

15. Cohn, M. (2005). “Agile Estimating and Planning”, *Publ. Prentice Hall*, NY, 368 p.

16. Karner, G. (1993). “Resource Estimation for Objector Projects: project report*,* Objective Systems”, San Francisco, *AB,* 9 p.

17. Grover, V. & Malhotra M. (2019). “Business process reengineering: A tutorial on the concept, evolution, method, technology and application”, [Electronic Resource]. – Access mode: https://www.sciencedirect.com/science/article/abs/pii/S0272696396001040. – Active link – 23.04.2019, doi: https://doi.org/10.1016/S0272-6963(96)00104-0

18. Velykodniy, S. (2019) Metod predstavlennia otsinky reinzhynirynhu prohramnykh system za dopomohoiu proektnykh koefitsiientiv, [Method of presenting the assessment for reengineering of software systems with the project coefficients help], *Innovative Technologies and Scientific Solutions for Industries*, No. 1 (7), pp. 34-42, doi: https://doi.org/10.30837/2522-9818.2019.7.034 (in Ukrainian).

19. Punmia, B. C. & Khandelwal, K. K. (2016). “Project Planning and Control with PERT and CPM”. New Delhi, *Lame Publications*, 258 p.

20. Manganelli, R. & Klein, M. (2019). “The Reengineering Handbook: A Step-by-Step Guide to Business Transformation”, [Electronic Resource]. – Access mode: https://www.researchgate.net/publication/304544531_The_Reengineering_Handbook_A_Step-by-Step_Guide_to_Business_Transformation. – Active link – 23.02.2019, doi: https://doi.org/10.1097/01445442-199503000-00011.

21. Kerzner, H. (2003). “Project Management: A Systems Approach to Planning, Scheduling, and Controlling”, 8ht ed., New Jersey, *Publ. John Wiley & Sons*, 914 p.

22. Clemmons, R. (2016). “Project Estimation with Use Case Points”, *Publ. Cross Talk*, Vol. 2, Issue February, pp. 18-22.

23. Selby, R. W. (2017). “Software Engineering: Barry W. Boehm's Lifetime Contributions to Software Development, Management and Research”, *Publ. John Wiley & Sons*, New Jersey, 818 p.

24. Velykodniy, S. S., Tymofieieva, O. S., & Zaitseva-Velykodna, S. S. (2018). Metod rozrakhunku pokaznykiv otsinky proektu pry vykonanni reinzhynirynhu prohramnykh system, [The calculation method for indicators project estimation in the implementation of software systems reengineering], *Radio Electronics, Computer Science,* Control, No. 4, pp. 135-142, doi: https://doi.org/10.15588/1607-3274-2018-4-13 (in Ukrainian).

25. Velykodniy, S. (2015). Reinzhiniring sistem monitoringu ta distantsiynogo upravlinnya sudnovimi energetichnimi ustanovkami, [Reengineering of SCADA-systems by shipping energy plants], *22th International Conference “Automatic 2015”*, 10-11 sep. proceedings, Odessa, Ukraine, pp. 133-134 (in Ukrainian).

26. Nevlyudov, I. Sh., Velykodniy, S. S., & Omarov, M. A. (2010). Ispol'zovanie CAD/CAM/CAE/CAPP pri formirovanii upravlyayushchikh programm dlya stankov s ChPU, [Using CAD / CAM / CAE / CAPP when forming control programs for CNC machines], *Eastern-European Journal of Enterprise Technologies,* Vol. 2, Issue 2 (44), pp. 37-44 (in Russian).

Received 11.03.2019

УДК 004.4'22 + 658.5

[1]**Великодний, Станіслав Сергійович,** кандидат технічних наук, доцент, доцент кафедри інформаційних технологій, E-mail: velykodniy@gmail.com,
ORCID ID: https://orcid.org/0000-0001-8590-7610
[1]**Бурлаченко, Жанна Вікторівна,** аспірант кафедри інформаційних технологій,
E-mail: 7035373@ukr.net, ORCID ID: https://orcid.org/0000-0001-8451-5527
[1]**Зайцева-Великодна, Світлана Сергіївна,** аспірант кафедри інформатики,
E-mail: svetlana.zaytseva@gmail.com, ORCID: https://orcid.org/0000-0001-7453-8821
[1]Одеський державний екологічний університет, вул. Львівська, 15, Одеса, Україна, 65016

ПРОГРАМНИЙ ЗАСІБ ДЛЯ АВТОМАТИЗОВАНОГО ПРОЕКТУВАННЯ МЕРЕЖЕВИХ ГРАФІКІВ РЕІНЖИНІРИНГУ ПРОГРАМНИХ СИСТЕМ

Анотація. *Предмет роботи – побудова графічної мережевої моделі реінжинірингу програмної системи. Мета роботи – розробка програмного засобу для підвищення рівня автоматизації проектування мережевих графіків з організації виробництва по реінжинірингу програмних систем у рамках управління проектами. Мережеве планування – це одна з форм графічного відображення змісту робіт й тривалості виконання стратегічних планів і довгострокових комплексів проектних, планових, організаційних та інших видів діяльності підприємства. Поряд з лінійними графіками й табличними розрахунками мережеві методи планування знаходять широке застосування при розробці перспективних планів і моделей створення складних виробничих систем та інших об'єктів довгострокового використання. У сучасних спеціалізованих пакетах комп'ютерних програм планування та оперативного управління використовується тип графіків «вершини-роботи». Завданням перед створенням програмного засобу є здатність працювати з усіма типами мережевих графіків із можливостями їх всебічної трансформації. Методи. В основу статті закладено методи мережевого планування за методологію PERT (Program (Project) Evaluation and Review Technique), використання елементів теорії графів та методу діаграм Ганта, як облікового для здійснення управління проектами. Моделювання системної архітектури програмного забезпечення виконується у рамках методології UML (Unified Modeling Language) 2.5 із використанням CASE-інструментарію Enterprise Architect 14. Результатами статті є проекті рішення, що запропоновані авторами. Зміст проектної частини визначається, по-перше, специфікою планування реінжинірингу програмних проектів, по-друге, особливостями конкретних технічних пропозицій до проекту, що піддається управлінню. У статті спроектована архітектура (проектний «каркас») програмного засобу для управління мережевим плануванням реінжинірингу програмного проекту. Висновки. Архітектура розроблена у вигляді декількох структурних та поведінкових діаграм, а саме: діаграма варіантів використання, що надає аналітику детальної уяви про галузь застосування програмного засобу; діаграма послідовності, яка призначена для формування уяви програміста про порядок виконання дій при роботі з майбутнім програмним засобом; діаграма станів, що необхідна для наочного подання тих станів, у яких програмний засіб може знаходитися у різні моменти часу; діаграма класів, яка використовуються для проектування основного формового наповнення майбутнього програмного засобу; діаграма компонентів, що призначена для вивчення складу компонентів майбутнього програмного засобу та вказівки послідовності компіляції та збірки окремих модулів. Чисельна та часова оцінка параметрів планування будується за даними, що отримані із проектних діаграм Ганта.*

Ключові слова: *управління проектом; граф; мережевий графік; програмний засіб; реінжиніринг; CASE-засіб; UML-діаграма*

УДК 004.4'22 + 658.5

[1]**Великодный, Станислав Сергеевич,** кандидат технических наук, доцент, доцент кафедры информационных технологий, E-mail: velykodniy@gmail.com,
ORCID ID: https://orcid.org/0000-0001-8590-7610
[1]**Бурлаченко, Жанна Викторовна,** аспирант кафедры информационных технологий,
E-mail: 7035373@ukr.net, ORCID ID: https://orcid.org/0000-0001-8451-5527
[1]**Зайцева-Великодная, Светлана Сергеевна,** аспирант кафедры информатики,
E-mail: svetlana.zaytseva@gmail.com, ORCID: https://orcid.org/0000-0001-7453-8821
[1]Одесский государственный экологический университет, ул. Львовская, 15, Одесса, Украина, 65016

ISSN 2663-0176 (Print)
ISSN 2663-7731 (Online)

ПРОГРАММНОЕ СРЕДСТВО ДЛЯ АВТОМАТИЗИРОВАННОГО ПРОЕКТИРОВАНИЯ СЕТЕВЫХ ГРАФИКОВ РЕИНЖИНИРИНГА ПРОГРАММНЫХ СИСТЕМ

Аннотация. *Предмет работы – построение графической сетевой модели реинжиниринга программной системы. Цель работы – разработка программного средства для повышения уровня автоматизации проектирования сетевых графиков организации производства по реинжинирингу программных систем в рамках управления проектами. Сетевое планирование – это одна из форм графического отображения содержания работ и продолжительности выполнения стратегических планов и долгосрочных комплексов проектных, плановых, организационных и других видов деятельности предприятия. Наряду с линейными графиками и табличными расчетами сетевые методы планирования находят широкое применение при разработке перспективных планов и моделей создания сложных производственных систем и других объектов долгосрочного использования. В современных специализированных пакетах компьютерных программ планирования и оперативного управления используется тип графиков «вершины-работы». Заданием перед созданием программного средства является способность работать со всеми типами сетевых графиков с возможностями их всесторонней трансформации. Методы. В основу статьи заложена методы сетевого планирования по методологии PERT (Program (Project) Evaluation and Review Technique), использование элементов теории графов и метода диаграмм Ганта, как учетного для осуществления управления проектами. Моделирование системной архитектуры программного обеспечения выполняется в рамках методологии UML (Unified Modeling Language) 2.5 с использованием CASE-инструментария Enterprise Architect 14. Результатами статьи являются проектные решения, предложенные авторами. Содержание проектной части определяется, во-первых, спецификой планирования реинжиниринга программных проектов, во-вторых, особенностями конкретных технических предложений к проекту, что подлежит управлению. В статье спроектирована архитектура (проектный «каркас») программного средства для управления сетевым планированием реинжиниринга программного проекта. Выводы. Архитектура разработана в виде нескольких структурных и поведенческих диаграмм, а именно: диаграмма вариантов использования, иллюстрирующая аналитику детальное представления об области применения программного средства; диаграмма последовательности, предназначенная для формирования представления программиста о порядке выполнения действий при работе с будущим программным средством; диаграмма состояний, необходимая для наглядного представления тех состояний, в которых программное средство может находиться в разные моменты времени; диаграмма классов, использующаяся для проектирования основного формового наполнения будущего программного средства; диаграмма компонентов, предназначенная для изучения состава компонентов будущего программного средства и указания последовательности компиляции и сборки отдельных модулей. Численная и временная оценка параметров планирования строится по данным, полученные из проектных диаграмм Ганта.*

Ключевые слова: *управление проектом; граф; сетевой график; программное средство; реинжиниринг; CASE-средство; UML-диаграмма*