USAGE OF SYSTEM ANALYSIS METHODS IN THE SOFTWARE PRODUCTS ENGINEERING

Logunova O. S., Kocherzhinskaya Yu. V., Torchinsky V. E., Sibileva N. S., Arefyeva D. Ya. and Popov S. N.
Power Engineering and Automated Systems Institute, Nosov Magnitogorsk State Technical University, Magnitogorsk, Russia
E-Mail: y.kocherzhinskaya@mail.ru

ABSTRACT

In this article we represent the process of using methods and tools of classical system analysis at various stages of creating a software product, using as an example the development of the first version of UML-editor "UML artist". Project development began with the study of information on the subject area of use of this product, then collected and analyzed the requirements of potential users of the future product. When collecting the processing of data obtained at this stage, we use such methods of system analysis as survey, observation, questioning, ranking of assessments, and determination of consistency of opinions on the sample in relation to the requirements for the product. In the next step, based on the analysis of competing decisions, the paradigm was defined; the conceptual and architectural models of the editor were built. Stages of implementation and testing of the UML editor are considering in terms of satisfying functional requirements, in the paradigm of object-oriented programming and product engineering, with help of the system of building WPF client applications in the Visual Studio IDE. The program code of the editor is in more than 25 source files, its interface has some features that should interest the end user, such as options for aligning components, contextual hints, hotkeys, setting the display of background textures, two languages support (English and Russian), saving and printing created UML-diagrams, etc. We consider the traceability of user requirements from the detection phase to the testing phase of the product. Ehe results of successful experience of using the System Analysis tool for formalizing the process of analyzing the requirements for the product being developed are presented to the attention in the article.

Keywords: software engineering, system analysis, ranking requirements, expert evaluation, Unified Modeling Language, UML-editor, modeling of the development process, software, systems design.

1. INTRODUCTION

The modern level of development of computer systems and information technologies assumes constant perfection and complication both of program components, and the concepts underlying them. It has long been commonplace to use the terms "software system", "software package", "software", which unites the understanding that the scale of the developed product significantly exceeds the size of the accustomed and everyday "Calculator". It is the awareness of the size and multicomponent nature of modern systems that led to a new stage in the development of one of the most useful sciences - system engineering in the field of software development [1].

Moreover, significant preferences from the use of the system approach in this field contributed to the emergence of separate studies in the field of requirements engineering, software components and software testing [2,3]. As before, it does not lose its relevance when developing the best indicators of the classical Trinity Quality-Budget-Terms. In projects where the main indicator is the quality of the product being created, much attention is paid to the implementation and maintenance of the documentation process, throughout the life cycle of the product being created [4].

The software is an integral part of almost all modern complex systems, however, despite the massive nature, the active creation of any product begins with the work of a business analyst whose tasks are to identify focus groups and identify requirements for the software being developed. According to the recommended methods in the most popular sources [5], the work to identify and formulate requirements is poorly formalized. Among the main methods of work in this area are offered interviewing, observation, survey, study of reports on problems [5,6]. Thus, the human factor has a significant impact on the final result of business analysis, not only on the part of the business analyst himself, but also on the part of focus group participants responsible for filling out the error and rejection journal, and so on. To reduce the impact of this kind of negative factors, it is possible, using methods of system analysis and decision theory, successfully applied in the management of various kinds of processes with human participation in other industries [7,8]. Consider this process using the example of designing and developing the UML-editor "UML Artist".

2. OBJECT OF STUDY

Software design is one of the most complex and critical stages in the development life cycle, as the cost of correcting errors made at this step is one of the highest and, according to various sources, can reach half the cost of the entire project.

The practice of creating software most often offers us an object-oriented approach to development, based on three pillars: inheritance, encapsulation, polymorphism. Although the practice of creating software according to the procedural model takes place, all the same, the linking of design and development to the paradigm of object-oriented programming is incomparably more popular [9]. One of the proposed methods to combat the inevitable chaos in the design of automated systems and their modules is the use of the UML (Unified Modeling Language), the feature of which is the targeted focus on solving the primary tasks of the software developer. These include the development of the concept of the software product, its architecture and use scenarios [10].

The UML is a general purpose visual modeling language that is designed for the specification, visualization, design and documentation of software components, business processes and other systems. UML is not a language in the literal sense of the word; it can be defined as a set of rules that are the standard for charting. UML is first and foremost a powerful auxiliary tool for rapid modeling of the architecture of computing systems and the construction of conceptual, logical and graphic models of complex systems for various purposes. This language has incorporated the best qualities of software engineering techniques that have been used successfully in recent years in modeling different systems in terms of scale and complexity [11-14].

UML is accepted for use in many companies-leaders in the software production market (Microsoft, Oracle, Telelogic, Sybase, etc.). In addition, almost all world producers of CASE-tools (software development automation tools) support UML in their products - ARIS (IDS Scheer AG), Rational Rose (IBM), PowerDesigner (Sybase), Oracle Designer, Microsoft Visual Modeler, etc. [15-16].

Despite the fact that the software market already has editors for working with UML diagrams, free versions with the necessary set of qualities: simplicity, brevity, compliance with the latest versions of the UML language - among them a few (perhaps all of the above requirements are only Telelogic Modeler). In addition, the aesthetic and ergonomic component of the human-machine interface of existing applications does not always suit users [16].

Thus, the development of a new software product that implements the design concept using UML is an interesting and relevant task, since it aims to satisfy the target audience, experienced in programming issues.

3. REQUIREMENTS ENGINEERING FOR THE "UML ARTIST"

Within the scope of this project, it was planned to implement the development area of the following types of UML diagrams: class diagram, object diagram, use case diagram, interaction diagram, state diagram, activity diagram, component diagram, package diagram, deployment diagram [17].

Clearly, for a simple correspondence to the level of a UML editor, the product must support all known types of UML diagrams, but it is not so easy to determine which of the user functional requirements the UML editor, developed for the implementation of the diploma project, should support on the terms of the subsequent free use of student programmers [18].

For the primary categorization of requirements for the UML editor, we used the model of the Japanese professor Noriaki Kano [19]. According to it, all requirements can be - mandatory, desirable and pleasing. The mandatory requirements on the first step of software development were formulated. Usually, product's verification based on it. Their execution expecting by default.

In this project, we bounded with time and humans resources limits. That defined the quantity of desirable requirements possible to realization - not more than 5 [5]. To determine the desirable requirements for the software product being developed, it was decided to take the survey and observation methods as a basis [2].

As a focus group in the interview, students of the 2nd course of the master's degree in the direction "Software for Computer Engineering and Automated Systems" were speaking at the rate of 7 people. The choice of participants was based on the fact, they have sufficient competencies in the field of software development, to formulate their requests to the UML editor and evaluate each of the proposed alternatives. We chose for the observation, a well-known thematic resource (https://habrahabr.ru), in which one section of the software developers' users exchange views on certain functions of well-known UML-editors. As a result, of the survey and on the results of monitoring the discussion on the presence of desirable functions in the UML editor, seven most common functional needs were identified. Here they are:

- Using "hot keys" (A1);
- Touchscreen support (A2);
- Contextual hints (A3);
- Support for 2 languages (English and Russian) (A4);
- Functionality for saving and printing of created diagrams (A5);
- Features of component alignment (A6);
- Archiving diagrams by projects (A7).

To determine the importance of each of them, in the opinion of the focus group, a survey was conducted, during which participants evaluated each of the proposed functions, by the criterion of significance, using a 100-point scale. As the most suitable and low-cost method, chose a ranking with subsequent evaluation of the reliability of the results obtained. After ranking all this ratings, the results of which formed the following table (Table-1):

Table-1. Ranking results.

Students in expert role	Ranked alternatives						
	A1	**A2**	**A3**	**A4**	**A5**	**A6**	**A7**
Exp.1	1	7	2	3	4	5	6
Exp.2	1	6	2	4	3	5	7
Exp.3	3	7	2	4	3	5	6
Exp.4	2	7	1	3	4	5	6
Exp.5	1	5	2	3	4	7	6
Exp.6	2	7	1	3	4	5	6
Exp.7	3	7	2	1	4	5	6

Then we calculated the total rank of each alternative as sum of ranks, given by every Expert for this one. After that, determined its rank among others (Table-2).

Table-2. The results of processing the ranks table.

Evaluation	Alternatives						
	A1	**A2**	**A3**	**A4**	**A5**	**A6**	**A7**
Summary Rank (r_j)	13	46	12	21	26	37	43
Re-ranked Alternatives(r_j)	2	7	1	3	4	5	6

To create a chain of preferred requirements, let’s use the results if last row of Table-2:

A3>A1>A4>A5>A6>A7>A2

As a result, to implement the UML-editor "UML artist", the following 5 functions were selected in order of decreasing importance for the target audience:

- Contextual hints (A3);
- Using "hot keys" (A1);
- Support for 2 languages (English and Russian) (A4);
- Functionality for saving and printing of created diagrams (A5);
- Features of component alignment (A6).

The remaining requirements A7 and A2 were classified as “pleasing”. Let us leave them to the future. In conditions of insufficient time, the implementation of such requirements is not reasonable.

After that, using the methods of processing the results of expert assessments given in [20], a quantitative assessment of the consistency of expert opinions was calculated for this sample, the significance of the Kendall Concordance coefficient was determined at the significance level $\alpha = 0.05$ for 7 Alternatives (n) and 7 Experts (m).

Since there are no associated ranks in the rankings, the Kendall's coefficient of concordance is determined by the formula:

$$W = \frac{12S}{m^2(n^3 - n)} \quad (1)$$

where

$$S = \sum_{j=1}^{n}\left(\sum_{i=1}^{m} r_{ij} - \bar{r}\right)^2 \quad (2)$$

Estimating the mathematical expectation in (2) can be determined by the formula:

$$\bar{r} = \frac{1}{n}\sum_{j=1}^{n} r_j - \frac{1}{2}m(n+1) \quad (3)$$

After substituting (3) into (2) and simplifying, and also after applying the formula to the values from Table-2, we get:

S = 225+324+256+49+4+81+225 = 1164;
W = 13968/16464 = 0.848.

To test the hypothesis on the consistency of experts, let us turn to the reference tables. [20] According to them, the threshold for estimating the concordance coefficient for a sample of 7 alternatives and 7 experts at a significance level of $\alpha = 0.05$ $S_T = 394.15$. Our S much above S_T. Thus, there is no reason to reject the hypothesis of consensus opinions of experts.

4. PRACTICAL APPLICATION OF THE RESULTS FOR THE DEVELOPMENT OF THE UML-EDITOR

A. UML Editor development tools

The implementation of the UML diagram editor "UML artist" is mainly characterized by the use of Windows Presentation Foundation (WPF). WPF is a system for building client applications from Microsoft. This system is part of the .NET framework (since version 3.0).

The basic principles of WPF:

- Hardware acceleration. All WPF drawing is done through the DirectX application programming interface, which allows this technology to take advantage of modern graphics cards.

- Independence from permission. WPF technology is so flexible that it can automatically scale up and down, adapting to the preferences of the monitor, depending on the system settings.

- No fixed appearance of the controls. The controls are drawn through the visualization mechanism and are fully customizable. For this reason, WPF controls are often referred to as faceless - they define the functionality of the control, but do not have a rigidly tied appearance.

- Separation of the implementation of appearance and functionality. Appearance of the application is defined using the XAML (Extensible Application Markup Language) language; the functionality is implemented in the C# programming language.

Thus, the use of the WPF system as a project implementation tool was chosen as the most optimal. [18]

B. Results of development of UML-editor

When creating the editor, much attention was paid to the development of a user-friendly interface. After launching the "UML artist" diagram editor, the full-screen mode of the application, Figure-1.

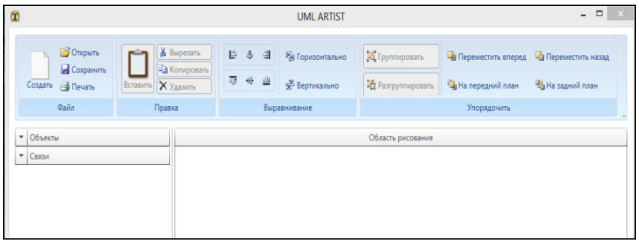

Figure-1. The part of main form “UML artist”.

The interface of the UML diagram editor is designed to be as intuitive as possible and implies the active work of the user through the "mouse". If the user is uncomfortable with full-screen resolution for any reason, the window sizes can be reduced, and the menu items for which there is not enough window width can be expanded using the arrow in the lower right corner of the menu. Thus, a compromise was reached, allowing, without sacrificing the functionality of the main menu, the ability to adjust the size of the window.

To add a new UML entity, the user simply drags the element from the Objects panel to the drawing area by holding the left mouse button. UML entities are created with a default name and standard sizes. The dimensions of the graphic display and its proportions are easily editable by the user. The ability to define a unique name for any object is also taken into account. The creation of UML-links is a drawing of the connecting element from one entity to another. The type of connecting line is determined by means of the "Communication" panel. The route of the trunk is determined by the program.

In the editor there is a strict ordering of objects, which determines which objects are drawn on top, and which are overlapped by graphic displays of other objects. The order is specified for both UML entities and UML-links. To change the order of objects, you can use the menu located in the header of the main window or use the context menu.

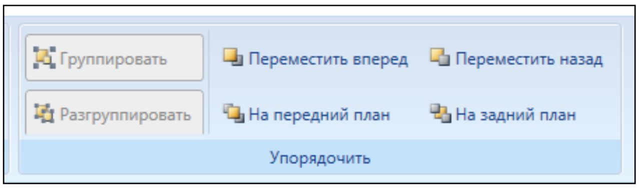

Figure-2. Fragment of the main menu for organizing objects in the editor "UML artist".

A feature of the editor "UML artist" is the possibility of aligning objects with respect to either side or central axes. To implement this functionality, there is an appropriate menu in the editor.

For the convenience of the user UML-editor "UML artist" can function through the use of "hot keys". Keyboard shortcuts are standard for Microsoft applications, which should be convenient for the average user.

In addition to using the main menu located in the header of the program window, the application has the ability to call context menus. In the "UML artist" editor, such menus are defined for the drawing space of diagrams, as well as for accessing UML entities and UML-links. [18]

The listed features can be regarded as interesting to the potential user, due to high aesthetic and functional characteristics

5. CONCLUSIONS

Thus, the results of successful experience of using the System Analysis tool for formalizing the process of analyzing the requirements for the product being developed are presented to the attention. The whole process is clearly illustrated by the example of the development of the software product UML editor "UML artist" as a student qualification work. The results of the implementation of the requirements in the software product are presented.

This area of research in the field of software development technology, project management in software development is promising, as the processes of formalizing the selection of requirements are not sufficiently developed and often based on intuition, emotions and experience. The formalization of this area will have a positive effect on reducing costs and managing the risks of the development process as a whole. Usage of system analysis methods at the initial stages of the life cycle software development should be more intense then now.

REFERENCES

[1] Kosyakov and U. Svit. 2017. System Engineering: Principles and Practices. trans. with English: Moscow. p. 624.

[2] E. Hall, K. Jackson and D. Dik. 2017. Requirements engineering. trans. with English: Moscow. p. 218.

[3] M. Garthner. 2012. ADDT. Software development through the acceptance tests. trans. with English: Moscow. p. 232.

[4] Yu.V. Kocherzhinskaya. 2016. System analysis documentary support software development process. Software of systems in the industrial and social fields, Magnitogorsk. 4-1: 33-41

[5] K. Vigers and D. Bitty. 2014. Development of software requirements. trans. with English: St. Petersburg. p. 736.

[6] A. Cobern. 2014. Modern methods of describing the functional requirements of the system. trans. with English: Moscow, 2014.

[7] H. Hakim, A. Sellami and H. Ben-Abdallah. 2017. Identifying and localizing the inter-consistency errors among UML use cases and activity diagrams: An approach based on functional and structural size measurements. 15th IEEE/ACIS International Conference on Software Engineering Research, Management and Applications. pp. 289-296.

[8] Duzdar and G. Temur. 2017. Comparing of results and implementation of clustering methods of data mining software with a data set. Electric Electronics, Computer Science, Biomedical Engineerings' Meeting.

[9] H. Goma. 2014. UML. Designing real-time systems, parallel and distributed applications. trans. with English: Moscow. p. 704.

[10] D. Khakhar and A. Nayak. 2018. Capturing Performance Requirements of Real-Time Systems Using UML/MARTE Profile. Advances in Intelligent Systems and Computing. pp. 703-714.

[11] D. Rambo. 2007. UML 2.0. Object-oriented modeling and development. trans. with English: St. Petersburg. p. 544.

[12] D. Rambo, A. Yakobson and G. Buch. 2002. UML: special directory. trans. with English: St. Petersburg. p. 656.

[13] G. Buch, D. Rambo and I. Yakobson. 2006. The UML language. User guide. trans. with English: Moscow. p. 496.

[14] A. Yakobson, G. Buch and D. Rambo. 2002. Unified software development process. trans. with English: St. Petersburg. p. 496.

[15] U. Boggs and M. Boggs. 2001. UML and Rational Rose. trans. with English: Moscow. p. 582.

[16] I.V. Chervenchuk. 2006. Information systems and processes, modeling and management. Modeling Information Systems with UML. Omsk. p. 48.

[17] Larman. 2004. Application of UML and design patterns. trans. with English: Moscow. p. 624.

[18] Yu.V. Kocherzhinskaya, V.G. Shamsutdinova. 2014. Software development of UML editor creating. Proceedings of the 2rd International Conference Information Technologies for Intelligent Decision Making Support and the Intended International Workshop Robots and Robotic Systems. - Ufa: Ufa State Aviation Technical University. 3: 13-17.

[19] D. Miloshevich. 2008. A toolsets for project management. Moskow, IT-Company, DMK Press.p. 729.

[20] A.S. Rykov. 2009. System analysis: models and methods of decision making and search optimization. Moscow. p. 608.