Converting Diagram to a Timeline Ontology

Nikolay Voit
Ulyanovsk State Technical University
Russia, Ulyanovsk
+7(8422)77-88-33
n.voit@ulstu.ru

Sergey Kirillov
Ulyanovsk State Technical University
Russia, Ulyanovsk
+7(8422)77-88-33
kirillovsyu@gmail.com

Semen Bochkov
Ulyanovsk State Technical University
Russia, Ulyanovsk
+7(8422)77-88-33
bochkovsi@ido.ulstu.ru

ABSTRACT

CAD systems design and development is not a simple process. It consists of large amount of works, most of them are interconnected and should be performed either simultaneously or sequentially, some of them depend on success of previous works etc. Design workflows allow to describe works in visual form and calculate their quantitative and qualitative parameters. They significantly increase the design process efficiency and the product quality due to the usage of participants interaction language unification. However, modern workflow management tools lack of some important functions especially in part of temporal analysis and ontology-based timeline diagrams.

In this paper, we describe the novel method to convert any diagram to a timeline structure like an ontology. The method includes converting algorithm and allows engineers to get the issue of workflows in which they are involved thus giving them a help to design complex CAD systems. It is shown that any diagram describing complex system behavior may be converted into simple view as a timeline ontology. An illustrated example is given in the article.

CCS Concepts

• **Software and its engineering→Software notations and tools→Context specific languages→Visual languages**

• **Theory of computation→Design and analysis of algorithms→Graph algorithms analysis→Dynamic graph algorithms**

Keywords

analysis; workflows; business process; computer-aided design.

1. INTRODUCTION

Automated systems are defined in GOST R 34.601-90 as an organizational and technical system that provides solutions based on automation of information processes in various fields of activity (management, design, production, etc.) or their combinations, as well as in the international standard IEEE 1471 as complex systems that use software intensively. In the business process management theory, theoretical computer science, and design automation theory, the stages of creating such systems are represented by design workflows (by designers who use software (computer) development tools intensively). The processing of such workflows (business processes) in the end-to-end digital design paradigm contains key design procedures: analysis and synthesis, which are among the latest research areas and significantly affect the result and success of the design. At the same time, the problem of the project solutions success in these theories has been dealt with for more than 30 years, such attention to the problem is caused by a high degree of output of development (project solutions) beyond the planned time, financial and functional parameters. The existing theory identifies the reasons and makes recommendations for improving the success of designing complex automated systems. however, according to the Standish Group international company, which is engaged in research on the success of automated systems development, currently only 40% of developments are completed successfully.

2. RELATED WORK

In modern graphic visual languages theory representing workflows, a logical model (behavioral model) is used [1, 2], which contains graphical objects and links between them. The following graphic languages are widely used in large enterprises: UML [3], BPMN [4], AMBER [5], IDEF [6, 7], eEPC [7, 8], and PERT [9]. In [5], the AMBER language is described, which has a simple data structure (there are no arrays, records, and classes), so it is not possible to implement complex business process structures on IT. The structural approach is embedded in the IDEF methodology [6] and has been developed in the UML, BPMN, eEPC languages and the specialized language Pilot Workflow ASCON (Russian developer of work flow management systems, ASCON firm [8, 10]) in terms of inheriting the object-oriented paradigm and introducing the concept of "time" into the diagram of workflow models [11]. However, in the most common tools for creating and processing diagram models, such as Microsoft Visio [12], Visual paradigm UML Tool [13], Aris Toolset [14], IBM Rational Software Architect (RSA) [15], pilot Workflow ASCON, the analysis of diagram models is performed using direct methods, requiring several "passes" depending on the type of error being controlled, there is no analysis of the structural features of complex diagram models and operational semantic analysis of attached software modules of diagram models of dynamic workflows. In particular, the workflow model designing tools at the Ulyanovsk mechanical plant JSC use specialized software, Workflow Designer, which is a component of the project management system developed by the RC ASKON-Volga, which has the following problems:

1. In terms of editing workflows.

* removing unnecessary business process blocks (tasks and procedures). The function not implemented in Workflow Designer. It is implemented by creating a new version of the business process procedure (function), which starts new workflows, and the old ones continue to run in the old version of the business process procedure (function). To solve this problem, it is necessary to analyze the running workflows for the possibility of transferring them to a new version of the procedure (function) of the business process. Completed tasks are usually not changed.

* adding new context variables for tasks and procedures. In Workflow Designer, adding new variables is only possible if a new version of the business process procedure (function) is created, with all the consequences described above.

* deleting context variables for tasks and procedures. Variables are defined at the start level of the workflow, so deleting variables is difficult. If the information entered earlier is not currently necessary, then deleting it should not cause any difficulties.

* it is almost impossible to add or remove predefined (constant) variables that operate at the level of the entire business process. In this case, it is suggested to create a new version of the business process itself, with restarting all static (constantly running) tasks.

2. Lack of functionality in Workflow Designer for analyzing the integrity of the business process for the presence of freezes, loops, and finiteness.

At the moment, there is a large amount of grammars designed or adapted for analyzing and controlling the flow patterns of visual languages [16, 17]. The most well-known are Web grammar [18], Positional grammar [11], Relational grammar, multi-level graph grammar [19], and preserving graph grammar [20]. Positional grammars are the simplest. Developing on the basis of Plex structures [18], they inherited their shortcomings. These grammars do not involve the use of join regions. They cannot be used for graphical languages whose graphical objects have a dynamic variable number of inputs/outputs. they cannot be used to control the syntax of graphical languages that contain parallelism. The advantage of relational grammars is that they can handle errors, but they do not have a mechanism for neutralizing such errors. Multi-level and persistent graph grammars can provide analysis of graphical languages with a "deep" context dependency, which is necessary in languages that allow to specify the synchronization of performed actions. Examples of such languages are the ones of process graph diagrams and message Sequence Charts. Common disadvantages of the above grammars are the following.

1.Increasing the number of productions when constructing a grammar for unstructured graphic languages, i.e. if the number of primitives of a graphic language is constant, there is a significant increase in the number of productions, since it is necessary to determine all possible variants of unstructured language.

2.The complexity of constructing a grammar (increasing the complexity of products and their number), and for some formalisms, the impossibility of constructing a grammar, for graph schemes with unstructured parallelism.

3.Large time complexity. Analyzers built on the basis of the considered grammars offer polynomial or exponential time for analyzing graphical languages diagrams.

In [21, 22], a syntactically-oriented approach based on a family of RV-grammars is proposed for processing visual languages. However, there are no mechanisms for analyzing and synthesizing structural and semantic features of diagrams in terms of their integrity and consistency between themselves and the conceptual model, including text attributes. The problem of error neutralization and its solution is well reflected in classical works on compilers, for example, [23]. A method of error neutralization for RV-grammars is also proposed [24]. However, the issues of neutralization for diagrammatic models of dynamic distributed workflows are not resolved in them. Translation of visual language models into another target language based on RV-grammars is solved in [25]. However, the task of translating several interrelated diagrammatic models of workflows presented in different languages is not considered in the target language. Temporal automatic RVTI-grammar [26] considers the timestamps of diagrammatic models of workflows, and not the events of business processes, so it does not take into account the facts of work performed in the analysis.

With the development of software, maintenance becomes the most expensive component of software system development. Studies by Myers, Linz, Swanson, and Shah have shown that from 1/2 to 2/3 of the costs over the life of a software system are spent on maintenance [27]. The maintenance stage also accounts for the highest costs associated with correcting detected errors [28]. Weinberg's analysis of the most expensive errors in the history of programming showed that the worst three were caused by changing exactly one line of code that was not tested after the change was made. Regression testing is the most important stage of software verification, aimed at rechecking the correctness of the modified program, creating assurance that the modified system meets the requirements [29, 30]. The analysis of works devoted to regression testing has shown that the main direction of research is the formation of an optimal tests set to confirm the program's performance, while solving the problems of minimizing test sets, determining priorities and selecting tests [31, 32]. There are several approaches to solving these problems.

* McCarthy's approach. Changes are analyzed at the level of the entire module. The link between program elements and tests is set manually by the developer.

* The Rothermel and Harrold approach [33], and the Ball approach [34]. Changes are analyzed at the node level of the program's control flow graph. The relationship of program elements to tests is based on dynamic information about the execution of each test.

• AST-based approach. The analysis of changes takes place at the level of the vertices of the abstract syntax tree (AST) of the program. The Association of program elements with tests is set at the AST level based on dynamic information about the execution of each test.

* Approach based on the firewall concept [35]. Changes are analyzed at the level of entire modules. The Association of program elements with tests is not taken into account.

However, existing regression testing approaches do not take into account the specifics of business processes in the system flow control works large industrial enterprises (for example, network coordination documentation, coordination of changes of design and technological documentation, changes in the structure and the list of performers and others), so do not allow you to automatically check the implementation of business processes, their compliance with the existing spec flow control works.

3. METHOD TO CONVERT A DIAGRAM INTO A TIMELINE ONTOLOGY

The diagram is the input data for the method. It can be developed in any graphical language and in any design tool. As a rule, such diagrams are presented in an XML description and are materialized in files [36-40]. Below the algorithm for converting a diagram into a timeline ontology is shown.

1. Delete vertices without external links.

2. Hide verbs that do not contain verbs (see Fig. 1).

a. To the top of the unparsed.

b. Check whether the vertex is an action, if so, before moving to the next vertex.

c. Get an unanalyzed incoming edge.

d. If there are none, go to step *j*.

e. Get the initial vertex for this edge.

f. Get an unanalyzed outgoing edge.

g. If there are no such edges, delete the outgoing edge and go to step *c*.

h. Adding a link between the initial vertex of the incoming edge and the final vertex of the outgoing edge.

i. Go to step *f*.

j. Delete all outgoing edges.

k. Delete the vertex.

3.Calculate the minimum number of steps to each vertex of the diagram.

4.Display vertices on the timeline according to the minimum number of steps to reach them.

The example diagram of technological processes in project management is shown in Fig. 2.

After deleting vertices without links the diagram will be set as shown in Fig. 3.

Next, vertices without actions (e.g. input and output data), edges without vertices and undirected edges are removed. The result can be seen in Fig. 4.

For each process step a minimum reaching time in units from the initial vertices is calculated. Corresponding values are appended to the diagram (Fig. 5).

Fig. 6 shows the final timeline ontology.

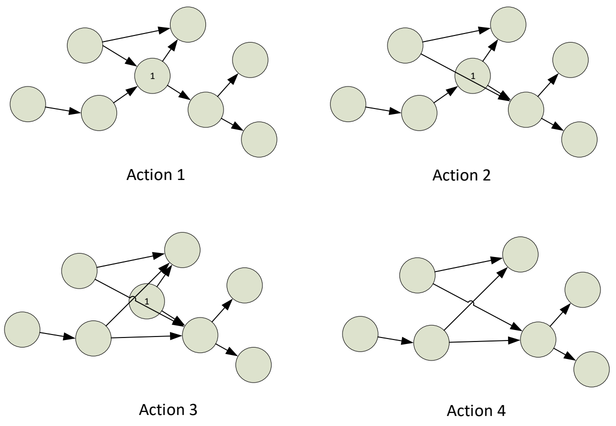

Figure 1. Example of hiding vertex 1.

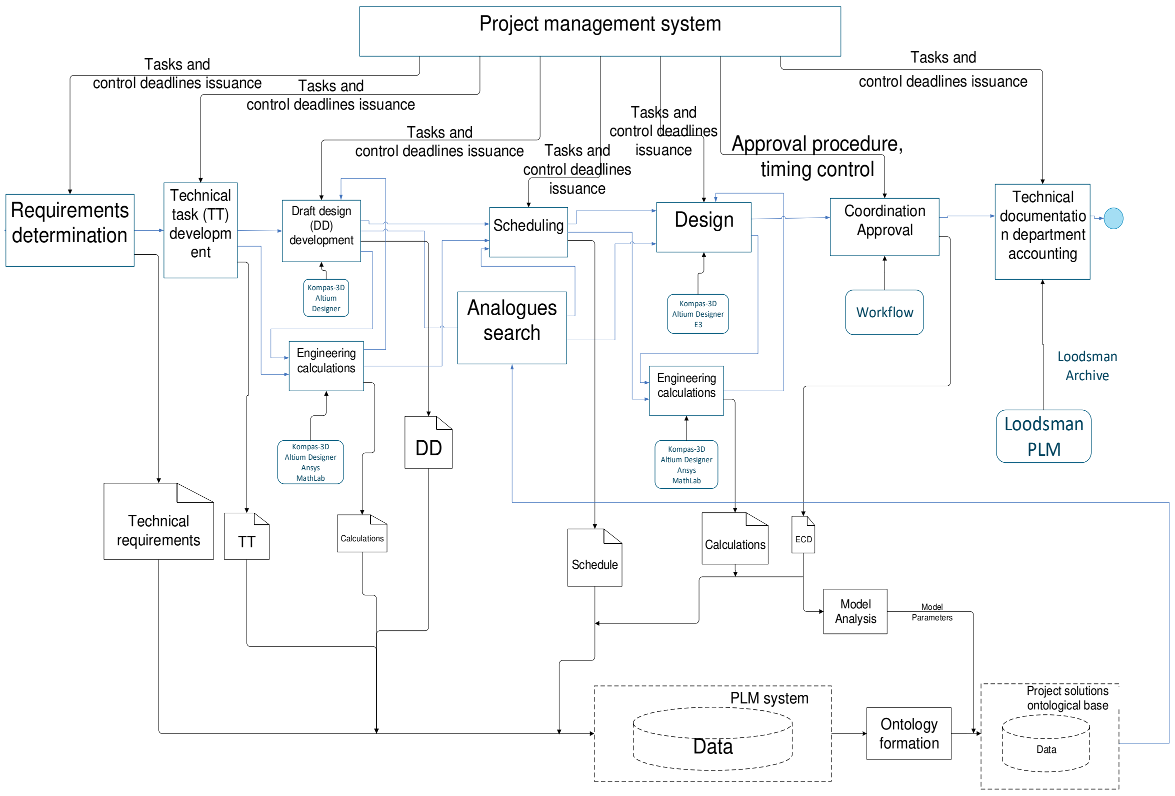
Project management system
Tasks and control deadlines issuance
Tasks and control deadlines issuance
Tasks and control deadlines issuance
Tasks and control deadlines issuance
Tasks and control deadlines issuance
Tasks and control deadlines issuance
Approval procedure, timing control
Requirements determination
Technical task (TT) developm ent
Draft design (DD) development
Scheduling
Design
Coordination Approval
Technical documentatio n department accounting
Kompas-3D Altium Designer
Analogues search
Kompas-3D Altium Designer E3
Workflow
Engineering calculations
Engineering calculations
Loodsman Archive
Kompas-3D Altium Designer Ansys MathLab
Kompas-3D Altium Designer Ansys MathLab
Loodsman PLM
DD
Technical requirements
TT
Calculations
Schedule
Calculations
ECD
Model Analysis
Model Parameters
PLM system
Data
Ontology formation
Project solutions ontological base
Data

Figure 2. Project management system.

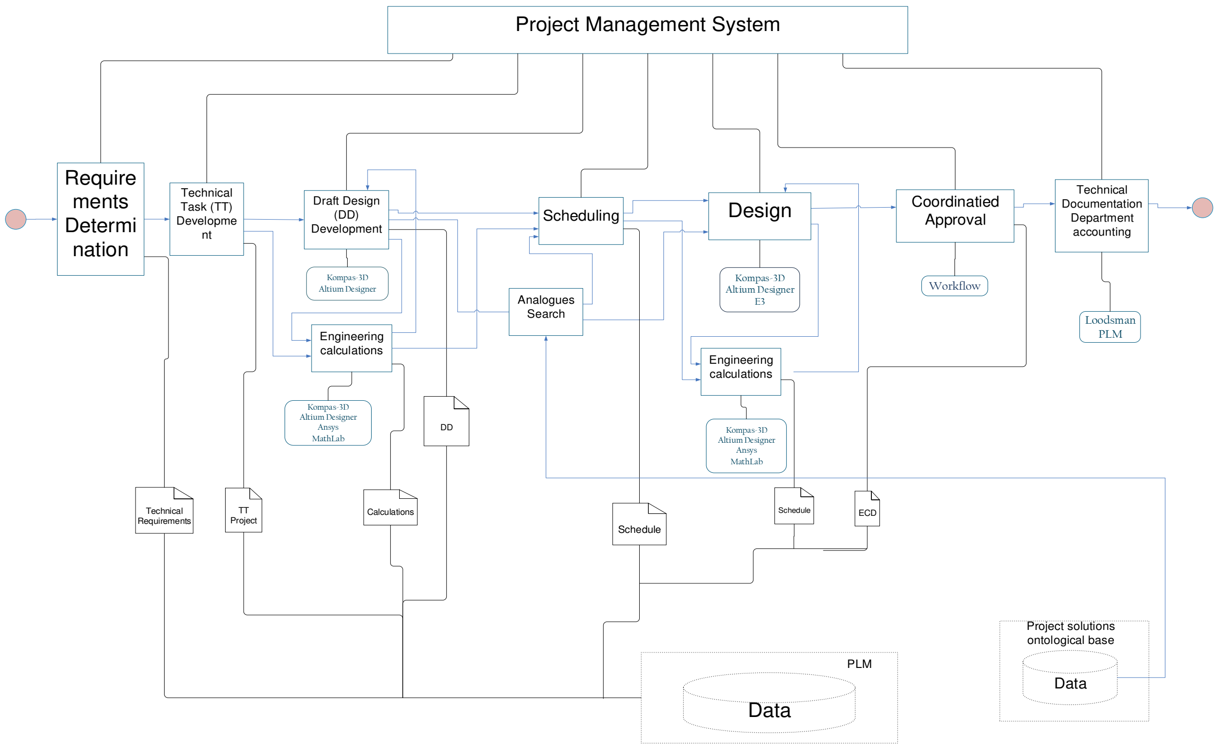
Project Management System
Require ments Determi nation
Technical Task (TT) Developme nt
Draft Design (DD) Development
Scheduling
Design
Coordinatied Approval
Technical Documentation Department accounting
Kompas-3D Altium Designer
Analogues Search
Kompas-3D Altium Designer E3
Workflow
Loodsman PLM
Engineering calculations
Engineering calculations
Kompas-3D Altium Designer Ansys MathLab
DD
Kompas-3D Altium Designer Ansys MathLab
Technical Requirements
TT Project
Calculations
Schedule
Schedule
ECD
Project solutions ontological base
Data
PLM
Data

Figure 3. Deleting vertices without links.

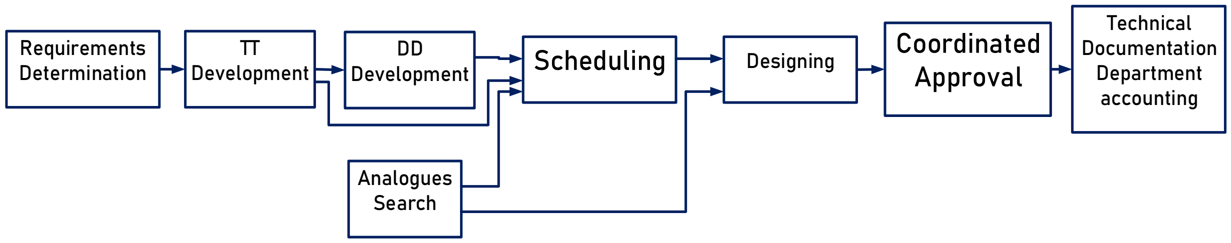

Figure 4. Hiding vertices without actions, edges without vertices, and undirected edges.

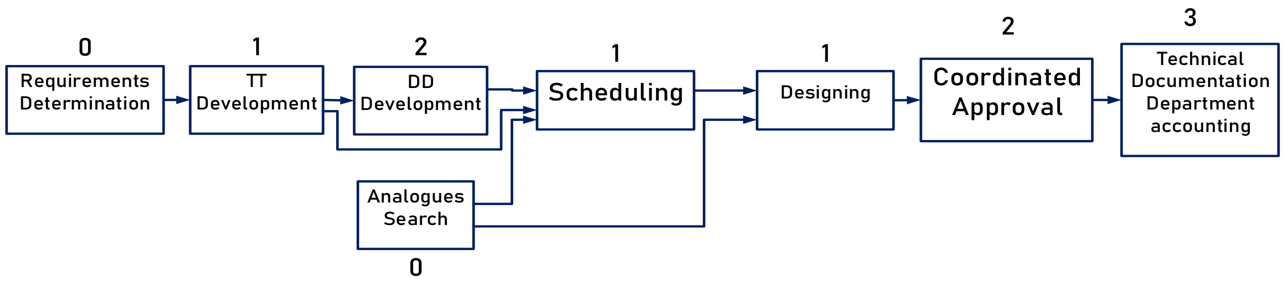

Figure 5. Calculation of the minimum number of steps from the initial vertices.

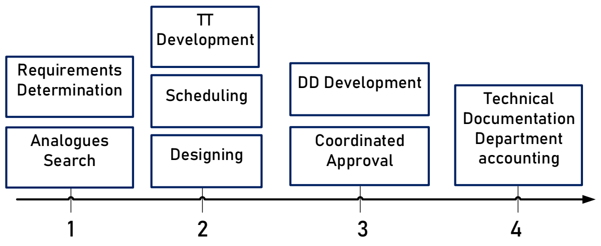

Figure 6. Building a timeline ontology.

4. CONCLUSION

A method for automated synthesis of the timeline ontology from diagram is presented. Diagrams are workflows in the basis of graphic languages in computer-aided systems (CAD). This method automates the production of an ontological model and presents the denotative semantics of diagrams of workflows in CAD. The method creates a timeline ontology as taxonomy (the first conceptual level of ontology description). The method helps the designer, analyst, and expert to identify errors in understanding the flow of design work, reduce the semantic gap between the stages of conceptual design, layout, and prototype manufacturing, and increase the digital engineering culture of joint design of complex technical automated systems. The method differs from the existing ones in that it takes into account the hybridity and dynamic nature of grammatical models of project workflows, takes into account the concept of "time" and the parameter "hours".

In future works authors describe the method to restructure any diagram for checking errors.

5. ACKNOWLEDGMENTS

The reported research was funded by Russian Foundation for Basic Research and the government of the region of the Russian Federation, grant № 18-47-730032.

6. REFERENCES

[1] Fischer, L. 2005. *Workflow Handbook*. Workflow Management Coalition.

[2] Van Der Aalst, W., and van Hee, K. M. 2004. *Workflow management: models, methods, and systems*. MIT Press, Cambridge, Massachusetts, USA.

[3] Booch, G., Jacobson, I., and Rumbaugh, J. 1998. *The Unified Modeling Language User Guide*. Addison-Wesley.

[4] Business Process Model and Notation (BPMN) Version 2.0, https://www.omg.org/spec/BPMN/2.0/PDF, last accessed 2020/03/02.

[5] Janssen, W., Mateescu, R., Mauw, S., and Springintveld, J. 1998. Verifying business processes using SPIN. In *Proceedings of the 4th International SPIN Workshop* (Paris, France, November 2 1998) SPIN'98, 21-36.

[6] Mayer, R. J., Painter, M. K., and de Witte, P. S. 1994. *IDEF family of methods for concurrent engineering and business re-engineering applications*. Knowledge Based Systems, College Station, Texas, USA.

[7] Samuilov, K. E., Serebrennikova, N. V., Chukarin, A.V., and Yarkina, N. V. 2008. *Fundamentals of formal methods for describing business processes*. RUDN, Moscow.

[8] LOODSMAN:PLM, https://ascon.ru/products/889/review/, last accessed 2020/03/02.

[9] Pozewaunig, H., Eder J., and Liebhart W. 1997. ePERT: Extending PERT for workflow management systems. In *Proceedings of the First East-European Symposium on Advances in Databases and Information Systems* (Siant-Petersburg, Russia, September 02 – 05, 1997). ADBIS'97. Nevsky Dialect, Saint-Petersburg, Russia, 217-224. DOI=https://doi.org/10.14236/ewic/ADBIS1997.34

[10] ASCON. https://ascon.ru, last accessed 2020/03/02.

[11] Costagliola, G., Lucia, A. D., Orefice, S., and Tortora, G. 1998. Positional grammars: a formalism for LR-like parsing of visual languages. In *Visual Language Theory,* Marriott K., Meyer B., Eds. Springer, New York, NY, 171-191. DOI= https://doi.org/10.1007/978-1-4612-1676-6_5

[12] Roth, C. 2011. *Using Microsoft Visio 2010*. Pearson Education.

[13] Easy-to-Use UML Tool, https://www.visual-paradigm.com/features/uml-tool/, last accessed 2020/03/02.

[14] Santos, P. S., Almeida, J. P. A., and Pianissolla, T. L. 2011. Uncovering the organizational modeling and business process modeling languages in the ARIS method. *International Journal of Business Process Integration and Management*. 5, 2 (May. 2011), 130-143. DOI= https://dx.doi.org/10.1504/IJBPIM.2011.040205

[15] Hoffmann, H.-P. 2012. Deploying model-based systems engineering with IBM® rational® solutions for systems and software engineering. In *Proccedings of 2012 IEEE/AIAA 31st Digital Avionics Systems Conference* (Williamsburg, VA, USA, October 14-18, 2012). DASC'12. IEEE, 1-8. DOI= https://doi.org/10.1109/DASC.2012.6383084

[16] Zhang, D. Q., and Zhang, K. 1997. Reserved graph grammar: A specification tool for diagrammatic VPLs In *Proceedings of 1997 IEEE Symposium on Visual Languages* (Isle of Capri, Italy, September 23-26, 1997). VL'97. IEEE, 284-291. DOI= https://doi.org/10.1109/VL.1997.626596

[17] Kalyanov, G. N. 2006. *Modeling, analysis, reorganization and optimization of business processes*. Finance and statistics, Moscow.

[18] Fu, K. 1977. *Structural Methods of Pattern Recognition*. Mir, Moscow.

[19] Rekers, J., and Schurr, A. 1997. Defining and parsing visual languages with layered graph grammars. *Journal of Visual Languages and Computing*. 8, 1 (February. 1997), 27-55. DOI=https://doi.org/10.1006/jvlc.1996.0027

[20] Zhang, D.-Q., Zhang, K., and Cao J. 2001. A context-sensitive graph grammar formalism for the specification of visual languages. *The Computer Journal*. 44, 3 (January. 2001), 186-200. DOI= https://doi.org/10.1093/comjnl/44.3.186

[21] Sharov, O. G., and Afanasiev, A. N. 2005. Syntactically-oriented implementation of graphic languages based on automatic graphic grammars. *Programming and Computer Software*. 6, 31 (2005), 56-66.

[22] Sharov, O. G., and Afanasiev, A. N. 2011. Methods and means of translating graphic diagrams. *Programming and Computer Software*. 3, 37 (2011), 65-76.

[23] Aho, A., Sethi, R., Ullman, J., and Lam, M. 2006. *Compilers: Principles, Techniques, and Tools. 2nd edition*. Addison-Wesley.

[24] Sharov, O.G., and Afanasyev A.N. 2008. Neutralization of syntax errors in graphic languages. *Programming and Computer Software.* 1, 37 (2008), 61-66.

[25] Sharov, O. G., and Afanasiev A. N. 2011. Methods and means of translating graphic diagrams. *Programming and Computer Software*. 3, 37 (2011), 65-75.

[26] Afanasyev, A., Voit, N., and Ukhanova, M. 2018. Control and analysis of denotative and significant semantic errors diagrammatic models of design flows in designing automated systems. *Radioengineering*. 6, (2018), 84-92.

[27] Myers, G. J., Sandler, C., and Badgett, T. 2011. *The art of software testing*. John Wiley & Sons, Hoboken, New Jersey, USA.

[28] Karpov, Yu. G. 2010. *MODEL CHECKING. Verification of concurrent and distributed software systems*. BHV-Petersburg, Saint-Petersburg.

[29] Brooks, F. 1995. *The mythical man-month: essays on software engineering. Anniversary edition*. Addison Wesley Longman.

[30] Felderer, M., Matthias, B., Johns, M., Brucker, A. D., Breu, R., and Pretschner, A. 2016. Security testing: A survey. In *Advances in Computers, Vol. 101*, A. Memon, Ed. Elsevier, 1-51. DOI= https://doi.org/10.1016/bs.adcom.2015.11.003

[31] Yoo, S., and Harman, M. 2012. Regression testing minimization, selection and prioritization: a survey. *Software Testing, Verification and Reliability*. 22, 2 (2012), 67-120. DOI= https://doi.org/10.1002/stvr.430

[32] Kotlyarov, V. P., and Kolikova T. V. 2009. *Fundamentals of modern software testing. Textbook manual for universities*. Binom. Knowledge laboratory, Moscow, Russia.

[33] Rothermel G., and Harrold, M. J. 1997. A Safe, Efficient Regression Test Selection Technique. *ACM Transactions on Software Engineering and Methodology*. 6, 2 (April. 1997), 173-210. DOI= https://doi.org/10.1145/248233.248262

[34] Ball, T. 1998. On the limit of control flow analysis for regression test selection. In *Proceedings of the 1998 ACM SIGSOFT international symposium on Software testing and analysis* (Clearwater Beach, Florida, USA, March 02 - 04, 1998). ISSTA '98. Association for Computing Machinery, New York, NY, USA, 134-142. DOI= https://doi.org/10.1145/271775.271802

[35] White, L.J., and Leung, H.K.N. 1992. A Firewall Concept for Both Control-flow and Data-flow Inregression Integration Testing. In *Proceedings of the Conference on Software Maintenance* (Orlando, Florida, USA, November 09 – 12, 1992). ISCM '98. IEEE, 262-271. DOI= https://doi.org/10.1109/ICSM.1992.242535

[36] Afanasyev, A., Voit, N., Ukhanova, M., and Ionova, I. 2017. Development and analysis of design-engineering workflows (mentioned as an instance a radio engineering enterprise). In *2017 IEEE 11th International Conference on Application of Information and Communication Technologies* (Moscow, Russia, September 20 – 22, 2017). AICT'17. IEEE. DOI=https://doi.org/10.1109/icaict.2017.8687262

[37] Voit, N., Kirillov, S., Kanev, D. 2019. Automation of Workflow Design in an Industrial Enterprise. In *Computational Science and Its Applications – ICCSA 2019*. ICCSA '19, vol. 11623. Springer, Cham, 551-561. DOI= https://doi.org/10.1007/978-3-030-24308-1_44

[38] Afanasyev, A., Voit, N., and Kirillov, S. 2019. Temporal Automata RVTI-Grammar for Processing Diagrams in Visual Languages as BPMN, eEPC and Askon-Volga. In *Proceedings of the 2019 5th International Conference on Computer and Technology Applications* (ICCTA 2019). ICCTA'19. Association for Computing Machinery, New York, NY, USA, 71-75. DOI= https://doi.org/10.1145/3323933.3324067

[39] Voit, N., Ukhanova, M., Kirillov, S., Bochkov, S. 2019. Method to Create the Library of Workflows. In *Proceedings of the 14th International Conference on Interactive Systems: Problems of Human-Computer Interaction*,. IS'2019. UlSTU, Ulyanovsk, 97-107.

[40] Afanasyev, A., Gladkikh, A., Voit, N., Kirillov, S. 2019. Processing of Conceptual Diagrammatic Models Based on Automation Graphical Grammars. In *Proceedings of the Third International Scientific Conference "Intelligent Information Technologies for Industry" (IITI'18)*. IITI'18. Advances in Intelligent Systems and Computing, vol 875. Springer, Cham, 369-378. DOI= https://doi.org/10.1007/978-3-030-01821-4_39