Grammar-algebraic approach to analyze workflows

Alexander Afanasyev[1] and Nikolay Voit[1][0000-0002-4363-4420]

[1] Ulyanovsk State Technical University, Ulyanovsk, Russia
{a.afanasev, n.voit}@ulstu.ru

Abstract. Improving the lifecycle of automated systems and reducing their development time are an important production problem in a large enterprise. We have created a new approach to the analysis and transformation of their processes on the basis of author's principles, grammar, method of the design process for narrowing the semantic gap between business process analysis and business process execution. This approach allows designers to improve the quality and reduce the time spent on the lifecycle of automated systems.

Keywords: Workflow, Process, Grammar.

1 Introduction

Project enterprises often need to dynamically reconfigure their internal processes to improve the efficiency of the business flow. However, modifications of the workflow usually lead to several problems (deadlocks) in terms of the degree of freedom, completeness and security of decisions. Therefore, the design, analysis, monitoring and modeling of dynamic workflows is an urgent task concerned with the need to quickly respond to changing business situations. The dynamic development of complex automated systems is associated with the adaptation of work processes to changes in system requirements. It is generally accepted that the agility (liveliness) of an enterprise/business as a property of an enterprise functions in a dynamically developing world. Enterprises should develop in two directions: adjust to changes in the surrounding environment; discover new opportunities constantly appearing in the dynamic world for launching completely new products (services).

Becoming agile requires a new approach that allows project managers to discover changes and opportunities for the development of complex automated systems and to react on them appropriately. The need to develop such an approach arose when the degree of change in development requirements increased. Large corporations like Whitestein Technologies, Magenta Technologies, SkodaAuto, Volkswagen, Saarstahl AG note that industry and technology (progress) move too fast, and requirements change very quickly, and traditional (monolithic) methods can no longer manage a product's lifecycle and project's workflows. The efficiency of business process can be improved by dynamical reconfiguration of enterprise's business processes as workflows. Design, analysis, checking, modeling, and transformation of dynamic workflows result in workflows' modifications. When modifying workflows, it is required to solve problems such as deadlock, security, and etc. [1].

The design and development of automated systems should include the adaptation to agile requirements of the environment. In work [2], the agility is the main property of production. There are two behaviors of an enterprise. One of them is to change the enterprise's business processes of production. Another is to create new marketable products. Enterprises should often change their business processes in order to increase product quality and get new market outlets. The work [3] notes that the speed of development in industry and technology should change monolithic approaches. A lot of large enterprises like IBM, ARIS note that the monolithic product lifecycle management systems with static workflow automation tools have reached their limits; almost all possible process configurations and reconfigurations are not only slow and costly, but often impossible [4]. The consequences are ill-fitting processes and soaring process development and improvement costs. ProBis [6] has monolithic workflows. Dynamic workflows are presented in works [7-9].

We used a definition given in [5] for a dynamic workflow as a process of adaptation to the current environment. We define a new RVT-grammar as a temporal finite state grammar using a memory as stacks and tapes to analyze workflows. There are two main principles in this work. They are: the principle of the ensemble of hybrid dynamic workflows. It includes the use of heterogeneous types and distribution in the space of hybrid dynamic workflows; the principle of adaptive design. It has a continuous in time structural and parametric analysis and synthesis of hybrid dynamic workflows.

Article has the following structure. In Introduction, the list of standard problems with workflows is submitted briefly. Related work has an overview of works on this topic. In Method to analyze workflows based on RVT-grammar, and Method of business-processes transformation, authors describe the approach with an example. Discussion has an overview about analyzing and managing manufacturing and workflows of cyber-physical systems. Outputs and the further directions of researches are presented in the conclusion.

2 Related works

We have studied many research works considered with the workflows' specification, verification and translation. Some of them focus on formal semantics and verification methods for workflows using Petri nets, process algebra, abstract state machine [10, 11]. In [12], Decker and Weske offer a formalism based on Petri Nets to define properties as reliability and promptness, and a method for testing these two properties. However, they only describe the synchronous relationship and do not have any research comparisons for high-level interaction modeling languages as BPMN. In [13], the behavior of BPMN from a semantic point of view is studied and several BPMN templates are proposed. This work is theoretically unjustified and is not complete, which considers only a few models. Lohmann and Wolff [14] offer the analysis using existing templates and monitoring them using compatible templates. In [15], the authors draw attention to the translation of BPMN into the process algebra for analyzing choreographies using the help model and checking equivalence.

The main limit of the methods considered is that they do not work in the presence of different types of diagrams at the same time, which means that in some cases the input diagrams can not be analyzed.

3 Method to analyze workflows based on Temporal RVT-grammar

Temporal RVT-grammar is defined as the tuple

$$G = (V, \Sigma, \tilde{\Sigma}, C, E, R, \tau, r_0). \tag{1}$$

where $V = \{v_e,\ e = \overline{1.L}\}$ is an additional alphabet for the operation onto a memory; $\Sigma = \{(a_l, t_l), l = \overline{1.T}\}$ is an alphabet (words) of events; $\tilde{\Sigma} = \{(\tilde{a}_n, \tilde{t}_n),\ n = \overline{1.\tilde{T}}\}$ is a quasi-term alphabet, extending Σ; $C = \{c_i, c_i = c_i + t_{l-1}, i \in N\}$ is a set of a time identifier, and a beginning $c_i = 0$; E is a set of the temporal relations as $\{c_i \sim t_l\}$, where c is a variable (a time identifier), $\sim \in \{=, <, \leq, >, \geq\}$; $R = \{r_i,\ i = \overline{0.I}\}$ is a rule of this grammar G (a set of production rule's complexes), where this complex r_i has a subset P_{ij} of the production rule $r_i = \{P_{ij},\ j = \overline{1.J}\}$; $\tau = \{t_l \in [0; +\infty], l = \overline{1.T}\}$ is a set of timestamps, where $c_i \in \tau \times \sim \times \tau$; $r_0 \in R$ is an axiom of this grammar (a name of the first production rule), $r_k \in R$ is the last production rule. The production rule $P_{ij} \in r_i$ has a view as

$$(a_l, t_l) \xrightarrow{\{W_\gamma(v_1,\ \dots,\ v_n)|E\}} r_m. \tag{2}$$

where $W_\gamma(v_1,\ \dots,\ v_n)$ is *n*-relation, that defines a type of an operation over memory, depending on $\gamma = \{0,1,2,3\}$ (0 – operation is not performed, 1 – write, 2 – read, 3 – compare); (a_l, t_l) is a word as a pair of an event and a timestamp; $r_m \in R$ is a name of a target production rule. The language $L(G)$ of this grammar has words as (a_l, t_l) and presents a trace $\sigma = \{a_0, 0\} \rightarrow \{a_l, t_l\} \rightarrow \{a_k, t_T\}$. The grammar for UML AD is shown in Table 1.

We can check 23 errors. The following semantical errors are: The cyclic connection; Mutually exclusive links; Multiple communication; Remote context error; Control transfer failure; Error in the multiplicity of inputs; Error multiplicity of outputs; Invalid link; Communication error; Access level error; Error transmitting the message; An error in the delegation of control; A quantitative error in the elements of the diagram; Excluding links of the wrong type; A call directed to the life line; Collapsed connection; Violation of the multiplicity of dependencies; Mutually exclusive links; Synchronous call before receiving a response; Great synonymy; The antonymy of objects; Conversion of relations; Inconsistency of objects.

Table 1. Temporal RVT-grammar for UML AD

Prev. state	**Quazi-term**	**Next state**	**Operation**
r_0	A_{0i}	r_1	insert()/$W_3(k^{t(1)}$==1)
	A_0	r_1	Ø
r_1	rel	r_2	Ø
	find	r_2	$W_2(t^{1m})$
r_2	A_i	r_1	insert()/$W_3(k^{t(1)}$==1)
	A_r	r_1	replace()/$W_3(k^{t(1)}$==2)
	A_d	r_3	delete(), $W_1(l^{1m})$)/ $W_3(k^{t(1)}$==3
	A	r_1	Ø
	A_k	r_5	Ø
r_3	drel	r_4	Ø
r_4	A_i	r_1	(change_rel(),insert())/$W_3(k^{t(1)}$==1
	A_r	r_1	(change_rel(), replace())/$W_3(k^{t(1)}$==2
	A_d	r_3	delete_with_link()/$W_3(k^{t(1)}$==3
	A	r_1	change_rel()
	A_k	r_5	change_rel()
r_5	no_label	r_k	*

In order to correct errors during the grammar design, types of graphic objects are entered into the alphabet $\tilde{\Sigma}$. The graphic objects have more than one input or output, the samples of which will be used as an analysis continuer. Such graphic objects are key, because the main graphic representations are built on them. In addition, the large number of outcoming links allow one to cover the most part of the diagram for analysis. The algorithm for business process analysis using RVT-grammar is shown in Fig.1.

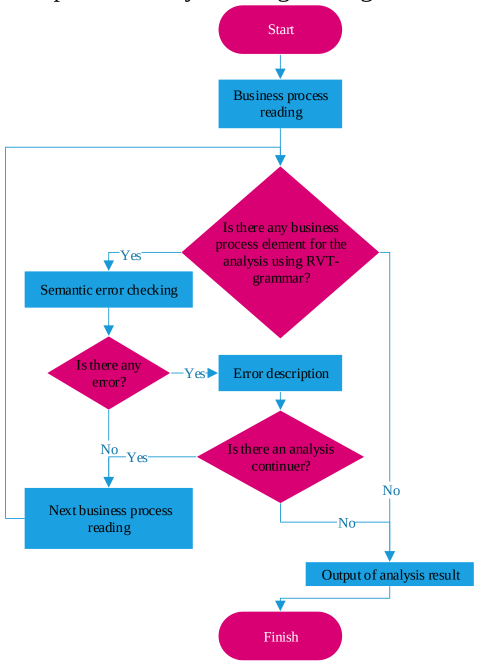
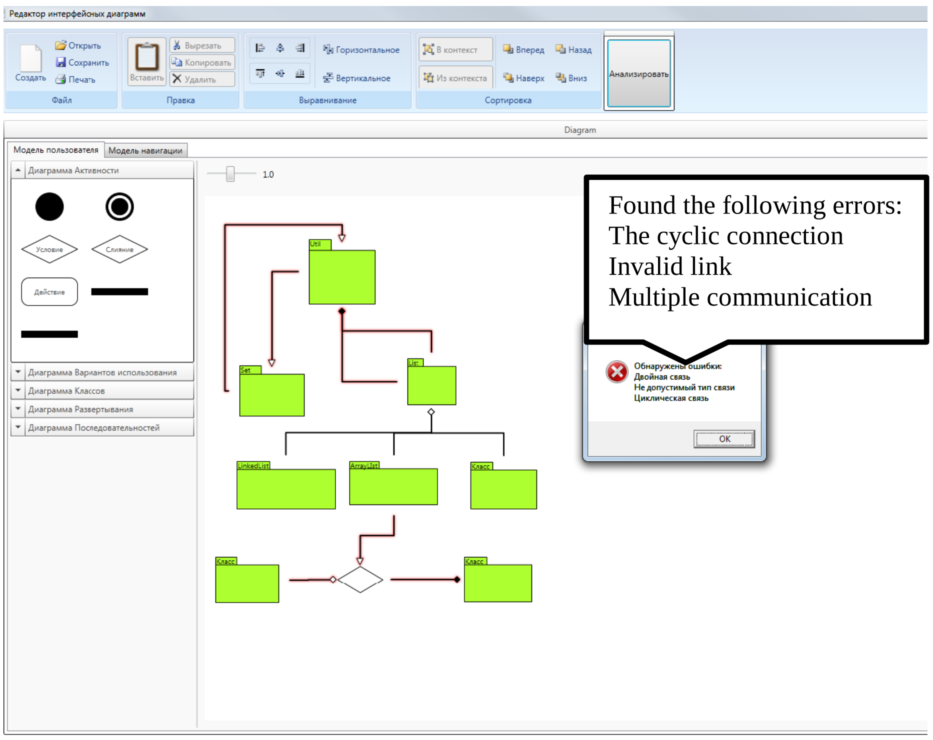

Start
Business process reading
Is there any business process element for the analysis using RVT-grammar?
Yes
Semantic error checking
Is there any error?
Yes
Error description
Is there an analysis continuer?
No
Yes
Next business process reading
No
No
Output of analysis result
Finish

Fig. 1. Algorithm for business process analysis using RVT-grammar.

To analysis and control the errors the Plugin has developed [20]: syntax oriented analyzer of UML diagrams for MS Visio, and network system of diagrams analysis and control, offering a full set of a functional to analyze and control syntax and semantic errors (Fig.2). The Plugin allows to perform follows:

- Analyses any diagrams (BPML, UML, IDEF0, IDEF3, Petri net) using Temporal RVT-grammar;
- Be integrated as plug-ins into MS Visio;
- Checks 23 types of errors;
- Gives recommendations to a designer for improve a diagram.



Found the following errors:
The cyclic connection
Invalid link
Multiple communication

Обнаружены ошибки:
Двойная связь
Не допустимый тип связи
Циклическая связь

Fig. 2. Plugin to analyze workflows for MS Visio.

4 Method of business-processes transformation

In order to dynamically reconfigure business processes, a mechanism for diagrams' transformation should be developed. It will allow programmers to gain agility, to improve functionality, and to increase business processes efficiency of a company. We propose to transform the diagram's structure via the procedures "Delete", "Insert" and "Replace" with saving a link for (before, after, and etc.) a special period of time. All graphic primitives must have a timestamp which will allow programmers to determine the diagram’s transformation time. As a rule, graphic primitives of BPMN, eEPC, IDEF0, UML AD, and etc. contain a description (see UML AD) which can be defined as a time variable.

Orchestration is the internal workflows in an enterprise or a company that presents its internal business processes [16]. IBM, Microsoft, Oracle, BEA Systems develop tools as BPEL4WS, XLANG, WSFL to describe the business logic [17-19]. In order to control these workflows in the enterprise’s business processes, only one manager for orchestration is required (Fig. 3).

Choreography is the external workflows in a lot of enterprises and companies that have relationship with each other. Each member of choreography can describe a role and his place in workflows. All choreography relationships are monitored in log.

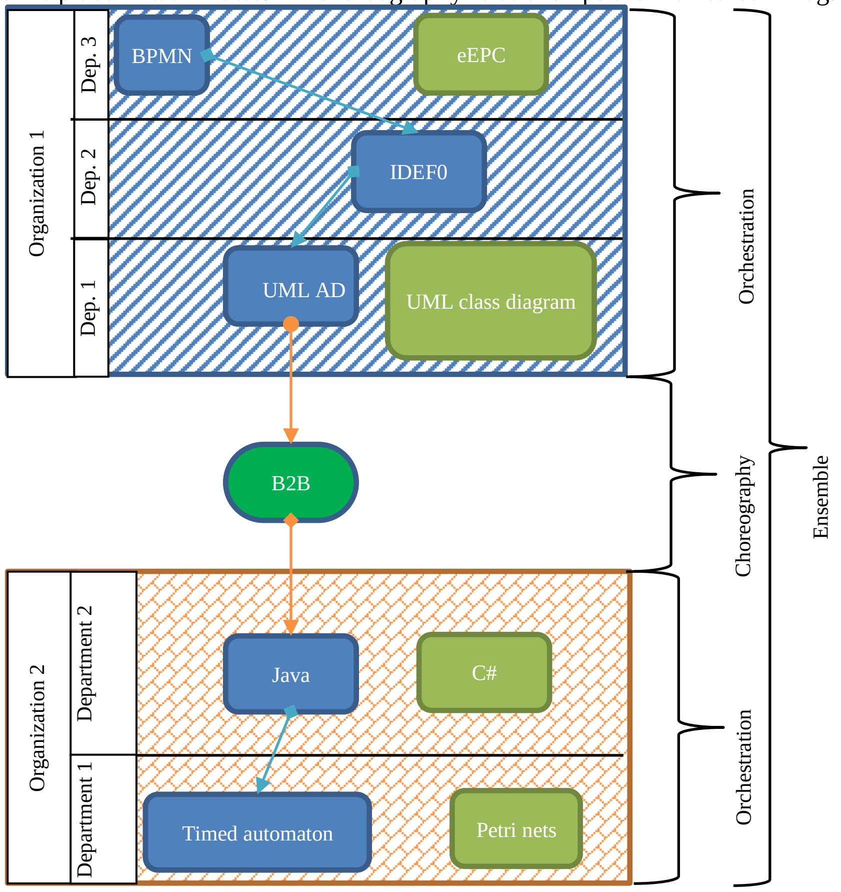


Fig. 3. The ensemble of diagrams and the hybrid orchestration.

The dynamics of workflows is presented in two aspects: orchestration and a sample of choreography that combine into an ensemble. An emerged sample of choreography should be associated with the developed diagrams. Organization 1 has BPMN, IDEF0, UML AD, eEPC, UML diagram in the orchestration. BPNM, IDEF0, UML AD are only in the ensemble [20, 10, 21-24]. Organization 2 has Java, C#, Timed automata, Petri nets in the orchestration, but Java and Timed automata are only in the ensemble. Hybrid orchestration means that we use different types of diagrams. We can create Temporal RVT-grammar for BPMN, IDEF0, eEPC etc., and transform them. Let us examine an example of a UML AD diagram (Fig. 4).

Graphic primitives (elements) A1 and A2 are denoted by the timestamp t1. This means that at a certain point in time t1 these elements will be transformed in the following ways: (1) Insert, (2) Replace, (3) Delete.

It is logical to assume that only one element can be transformed in one period of time. Therefore, each timestamp is given a tape where three variants can be indicated for one element: 1 – Insert, 2 – Replace, 3 – Delete.

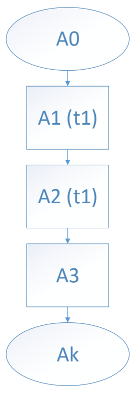


Fig. 4. UML AD diagram with a timestamp t1.

Using Insert and Replace operations, additional information will be saved at an extended tape allowing it to save both numbers and quasi-terms.

Additional function Insert() is used for the operation 1. It provides necessary information extracted from the extended tape and forms an inserted fragment based on subgrammar.

Operation 2 is a complex operation that is represented as a set of operations “Delete” and “Insert”. Additional function Replace() is introduced.

Let us consider function Delete at the initial stage. At time t1, the diagram becomes as follows (Fig. 5).

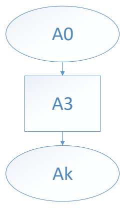


Fig. 5. Elements’ deletion at a diagram.

It can be deleted an infinite number of elements. In order to delete an element, let us use the following method. If the automaton encounters an element with a timestamp, a link to this element is saved in a stack. Then the automaton continues processing elements until it meets the element without a timestamp. In this case, the special function change_rel() is performed. It finds a link to the first deleted element in the stack and ties it to a current element. This process is shown in Fig. 6.

The function Delete() removes elements in a diagram not to leave elements, which should be deleted, as hanging elements when processing quasi-terms which should be deleted. It should be separately introduced the function delete_with_link() in order to delete elements with an inlink.

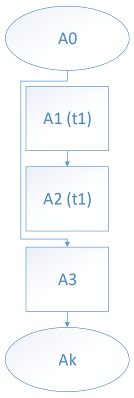


Fig. 6. Links' assignment in deleting an element.

5 Discussion

Applying timed grammar for designing, specification, controlling and analyzing real-time systems is well-known practice [25]. Timed and hybrid automatons is used for analyzing and managing manufacturing (MM) and workflows of cyber-physical systems (WCPS) [26].

When solving the tasks of MM&WCPS's design, specification, control and analysis, there are problems with access to resources, blocking, liveliness limitation (liveness, reversibility, boundedness, reachability, dead transitions, deadlocks, home states). The examples of MM&WCPS's tasks are the control of the nuclear reactor's temperature and the control of the railway level-crossing gate [27], and electronic workflows. These tasks successfully apply timed context-free grammars. A large number of MM&WCPS sets the task of monitoring and analyzing. It can be accomplished by various mathematical methods based on workflows [5]. At present, π-calculus is a promising but very young and evolving theory. It has many open questions and unresolved problems. Petri nets which are widely used do not have a universal framework for MM&CPS's modeling and analyzing. In order to analyze various properties (liveliness, attainability, safety), MM&CPSs are modeled in different types of Petri nets. In order to analyze MM&CPS in the error-free systems' development in the conceptual design phase, the model checking method is widely used. However, it is mainly developed for experienced scientists and engineers, since it is complex to understand and use [25]. MM&CPSs are also specified by managers who do not have training in formal models and informatics. Formal analysis requires a detailed representation of the process model in a formal language.

Although, relevant and having big practice famous is a problem researching mechanisms to analyze and control MM&CPS.

Modern tools can only check 16 types of errors [20]. Thus, we develop a new Temporal RVT-grammar to check and fix all type of errors, which has a linear require of time to analysis oppositely with other grammars with exponential and polynomial requires of time. Fig. 7 shows the efficiency. Graphic objects are graphic figures as a circle, a rectangle, a rhombus, a square, a line and etc. We propose a formula for calculation of the efficiency that can be written as:

$$\text{Required time} = c \cdot L_s, L_s = \sum_{i=1}^{m} \left(\sum_{j=1}^{V_i} v_{in_{ij}} + \sum_{j=1}^{V_i} v_{out_{ij}} \right) + \sum_{i=m+1}^{t} \left(V_i + \sum_{j=1}^{V_i} v\_out_{ij} \right) + no\_label. \quad (3)$$

where c – the constant of realization of algorithm, which determines a quantity of time (operators) that are spent to analysis one graphic object; L_s – the number of graphic objects; V_i – the number of graphic objects of i-type; v_in_{ij} – the number of inputs to j-graphic objects of i-type; v_out_{ij} – the number of outputs from j-graphic objects of i-type; t – a total of object types; m – a quantity of object types that have more than one output.

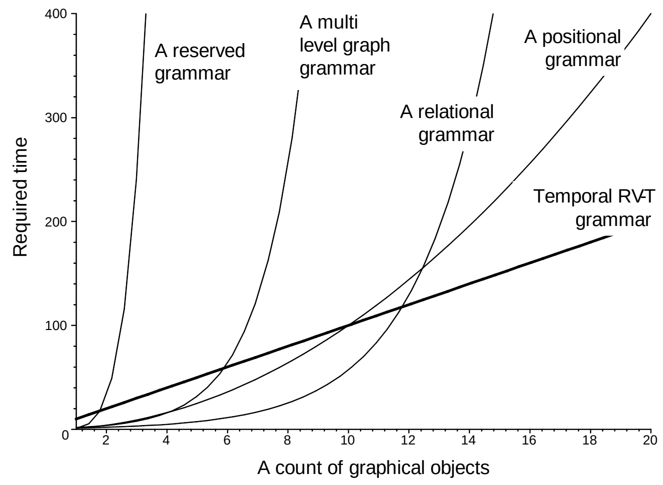


Fig. 7. Efficiency of checking and fixing errors with help Temporal RVT-grammar into diagrams.

Conclusion and future works

There is a problem with workflows for checking and exchanging various formats between large industrial enterprises, and also between their departments. Time begins to play a major importance for production, which often uses the Internet. This is why we develop a temporal grammar for analyzing, managing and reconfiguring dynamic design workflows, where time is a known role. We create this grammar as a temporal state grammar that uses the memory as a stack. Thus, this grammar allows us to remove several semantic errors (structural-behavioral) at the stage of conceptual design in complex computer systems, as well as solve problems of reengineering for reactive systems using real time. The transformation can check errors in a diagram and correct this diagram. We describe a simple example of using UML AD. The scientific significance of the approach is represented by a grammar and algebraic model that takes into account the temporal nature of workflows, provides structural and behavioral and semantic analysis and reconfiguration of workflows, and expands the class of errors. In future works, we plan to conduct experiments on BPMN and UML AD and other graphical temporal languages for the complex system development, taking into account the design data and technological preparation for the production of a real enterprise. We will identify new typical structural-behavioral errors and describe them in our future works.

Acknowledgments. The reported study was funded by RFBR according to the research project № 17-07-01417.

References

1. Aguilar, J. C. P., Hasebe, K., Mazzara, M., & Kato, K. Model Checking of BPMN Models for Reconfigurable Workflows. arXiv preprint arXiv:1607.00478 (2016).
2. Sherehiy, B., Karwowski, W., Layer, J. K.: A review of enterprise agility: Concepts, frameworks, and attributes. International Journal of Industrial Ergonomics 37, 446-460 (2007).
3. Highsmith, J., Orr, K., Cockburn, A.: Extreme programming. E-Business Application Delivery, 4-17 (2000).
4. A global Swiss company offering advanced intelligent application software for multiple business sectors, http://whitestein.com/, last accessed 2018/03/14.
5. Bider, I., Jalali, A.: Agile Business Process Development: Why, How and When Applying Nonaka's theory of knowledge transformation to business process development. Information Systems and e-Business Management. 14(4), 693-731 (2014). doi:10.1007/s10257-014-0256-1.
6. Andersson, T., Andersson-Ceder, A., Bider, I. State flow as a way of analyzing business processes-case studies. Logistics Information Management, 15(1), 34-45, (2012), http://www.ibissoft.com/publications/Cases.pdf, last accessed 2018/03/14.
7. YAWL Foundation, YAWL. http://www.yawlfoundation.org/, last accessed 2018/03/14).
8. Bider, I., Johansson, B., Andersson, B., Holmberg, N., Eds.: Analysis of Agile Software Development from the Knowledge Transformation Perspective. Perspectives in Business Informatics Research. BIR 2014. Lecture Notes in Business Information Processing, 194 (2014).
9. IbisSoft. iPB Reference Manual. http://docs.ibissoft.se/node/3, last accessed 2018/03/14.
10. Afanasyev, A. N., Voit, N.N., Kirillov, S. Y.: Development of RYT-grammar for analysis and control dynamic workflows. 2017 International Conference on Computing Networking and Informatics (ICCNI) 2017, 1-4 (2017), doi: 10.1109/ICCNI.2017.8123797, http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8123797&isnumber=8123766, last accessed 2017/03/15.

11. Voit, N. N.: Development of timed RT-grammars for analysis of business process at manufacturing and in cyber-physical systems. 2017 International Conference on Computing Networking and Informatics (ICCNI) 2017, 1-5 (2017), doi: 10.1109/ICCNI.2017.8123798, http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8123798&isnumber=812376, last accessed 2017/03/15.
12. Orchestration and Workflow, https://www.cloudenablers.com/blog/orchestration-and-workflow/, last accessed 2017/03/15.
13. ISO. LOTOS – A Formal Description Technique Based on the Temporal Ordering of Observational Behaviour. Technical Report 8807, ISO 1989.
14. Lohmann, N.,Wolf, K.: Realizability Is Controllability. WS-FM'09 2010 6194, LNCS, pp. 110–127 (2010).
15. Poizat, P., Salaün, G.: Checking the Realizability of BPMN 2.0 Choreographies. SAC'12, 1927-1934 (2012).
16. Orchestration and Workflow, https://www.cloudenablers.com/blog/orchestration-and-workflow/, last accessed 2017/12/29.
17. Van der Aalst, W. M. P.: Don't go with the flow: Web services composition standards exposed. IEEE intelligent systems 2003, 18(1), 72-76 (2003), http://www.martinfowler.workflowpatterns.com/documentation/documents/ieeewebflow.pdf, last accessed 2017/12/29.
18. Marca, D. A., McGowan, C. L.: SADT: structured analysis and design technique. McGraw-Hill, Inc. (1987), http://dl.acm.org/citation.cfm?id=31837, last accessed 2017/12/29.
19. TP026B, Rev. Rational Unified Process, https://www.ibm.com/developerworks/rational/library/content/03July/1000/1251/1251_bestpractices_TP026B.pdf, last accessed 2017/12/29.
20. Afanasyev, A.N., Voit, N.N., Gainullin, R.F.: Diagrammatic models processing in designing the complex automated systems. 10th IEEE International Conference on Application of Information and Communication Technologies (AICT) 2016, 441-445 (2016), doi: 10.1109/ICAICT.2016.7991737
21. Afanasyev, A., Voit, N.: Intelligent Agent System to Analysis Manufacturing Process Models. First International Scientific Conference «Intelligent Information Technologies for Industry» (IITI'16), 451, 395-403 (2016), doi: 10.1007/978-3-319-33816-3_39
22. Afanasyev, A.N., Voit, N.N., Voevodin, E.Yu., Gainullin, R.F.: Control of UML diagrams in designing automated systems software. 9th IEEE International conference on Application of Information and Communication Technologies: AICT-2015, 285-288 (2015), doi: 10.1109/ICAICT.2015.7338564
23. Afanasyev, A., Voit, N.: Multi-agent system to analyse manufacturing process models. International conference on Fuzzy Logic and Intelligent Technologies in Nuclear Science - FLINS2016. France, 444-449 (2016), doi: 10.1142/9789813146976_0072
24. Afanasyev, A., Voit, N., Gainullin, R.: The analysis of diagrammatic models of workflows in design of the complex automated systems. International conference on Fuzzy Logic and Intelligent Technologies in Nuclear Science - FLINS2016. – France, 509-514 (2016), doi: 10.1007/978-3-319-33609-1_20
25. Heitmeyer, C. L. and Lynch, N. A.: The generalized railroad crossing: A case study in formal verification of real-time systems. IEEE RTSS, 120-131 (1994).
26. Karpov, Yu. G.: MODEL CHECKING. Verifikaciya parallel'nyh i raspredelennyh programmnyh system. 560 s (2010, in Russian).
27. Lee, E. A.: Cyber-physical systems: Design challenges. ISORC (2008).