Development and Research of Temporal RVTI-Grammar for Workflow Diagrams Processing

A.N. Afanasyev, N.N. Voit, S.Yu.Kirillov, M.E.Uhanova
«Computing technique» department
Ulyanovsk State Technical University
Ulyanovsk, Russia
{a.afanasev, n.voit }@ulstu.ru, kirillovsyu@gmail.com, mari-u@inbox.ru

Abstract— **This paper describes the development and study of temporal RVTI-grammar, which allows to take into account time parameters in the process of document flow analysis. Methods of neutralization of the revealed errors, translation of conceptual schematic models of the automated systems presented in widely used visual languages into the diagram models based on formal languages are considered.**

I. Introduction

In the design of automated systems (as) are actively used diagram models presented in the artifacts of visual graphic languages BPMN, UML, IDEF and others. Visual (in the form of diagrams) form of presentation of business processes is designed to help designers in the development and analysis of design solutions, project preparation of production, technological preparation of production with the help of the logical justification of specific complex business processes. One of the components that complicate the workflow is the time factor. This can be a predefined value for the operation or a time-out after which the worker thread continues to force execution. There are also situations when you need to meet with the implementation of all operations in a certain period or allocate a certain proportion of time. Failure to do so may result in delays or downtime, which will have a significant impact on the expected outcome. The mathematical apparatus allowing to analyze the work flows with time characteristics is proposed. RV grammar will be used as a basis [1].

Checking workflows for structural and semantic errors is a computational task, so different formal approaches and languages can be used. However, the approach used for validation must support the workflow language. Due to the computational complexity of the problem (polynomial, exponential), only a few approaches successfully cope with checking workflows with constraints, including time, for all types of workflow graphs. The main language in which development will be carried out is the well-known BPMN. The standard model business process and notation (BPMN) will provide businesses the capability of understanding their internal business procedures in a graphical notation and will give organizations the ability to communicate these procedures in a standard manner. [2] BPMN Specification also provides a mapping between the graphics of the notation and the underlying constructs of execution languages, particularly the language of the business processes (BPEL) .

II. Related work

Design and processing of the work flow associated with the technology Rational Unified Process (RUP) [5], the PBWD methodology, languages Unifeid Model modeling Language (UML) [6], extended Event Driven Process Chain (eEPC), BPMN, IDEF0, IDEF3, Am-ber, Promela, YAWL, the Booch Methodology [7], Hierarchical Object Oriented Requirement Analysis (HOORA) [8], Jacobson Method [9], Object Modeling Technique (OMT) [10], Planguage [11], Shlaer-Mellor Object-Oriented Analysis Method [12], Software Cost Reduction requirements method (SCR) [13], software Requirements Engineering Methodology (SREM) [14], Storyboard Proto-typing [15], Structured Analysis and Design Technique (SADT) [16], and Structured Analysis and System Specification (SASS), Volere method, WinWin approach, and Component-based methods (COTS-Aware Requirements Engineering (CARE), Off-the-Shelf Option (OTSO)) [17].

Karpov Yu. G.in his work [18] uses the Model Checking approach for the analysis, control, modeling and reengineering of business processes, in which the main drawback is the study of the model, and not the system itself, so the question arises about the adequacy of the model to the system, while the complexity of solving the above problems is exponential.

In the work of Neda Saeedloei and Gopal Gupta [19] applied a temporal machine that implements the temporal context-free grammar for the analysis of cyber-physical systems with the subsequent translation of this grammar into a program for the interpreter of Prolog.

Yuan Wang and Yushun Fan [20] propose to use temporal logic of actions to describe workflows in a graph form, which requires a description of all routes of a graph in the formulas of temporal logic of action. Apply linear temporal logic to formalize the route of the tasks of branching and, OR, and similarities JOIN, however, question the adequacy of building the description of the workflow in graphical form remains not settled.

The database of tools for analysis and control of cyber-physical systems, as well as work flows is available at [21, 22]. In addition, there are tools CPN Tools [23], "Roméo - A tool for Time Petri Nets analysis" [24], TimesTool [25], the Tina Toolbox [26], Visual Object Net++ [27].

The traditional workflow management system includes ProBis [28]. To dynamic systems of flow control of design works according to the works [29, 30, 31], include YAWL (Yet Another Workflow Language), iPB. All such systems use diagrammatic representations of work flows. This solves the problem of analyzing the structure (syntax) and meaning (semantics) of diagrams. Thus, in [32] color Petri nets are used for dynamic semantic analysis of work flows, and in [33] PI-Calculus approach formalizing work flows into algebraic statements of first order logic is used. Currently, pi-calculus (pi-calculus) is a promising, but still very young and developing theory, it has a lot of open questions and unsolved problems. Petri nets have the following limitations:

- there is no universal framework for modeling and analysis of project work flows based on Petri nets. In order to analyze different properties (liveliness, reachability, security), workflows are modeled in different types of Petri nets, which is ad hoc.
- there is no mechanism that would help the designer in modeling and ensure the successful completion of the task with the necessary requirements (properties).

The model checking method has found a wide application of work flow analysis in the development of error-free systems at the design stage. However, it is intended for experienced scientists and engineers, as it is difficult to understand and operate [20].

III. TEMORAL RVTI-GRAMMAR AND METHODS OF NEUTRALIZATION, TRANSLATION

Project flows are a powerful tool for analyzing the business processes of the enterprise and contain design tasks addressed to specific departments and performers of the enterprise. Such design work can be carried out simultaneously in different departments and different performers, so the problems of synchronization, blocking resources, deadlocks, bottlenecks, etc., found in the theory of Informatics, arise in the field of management of business processes of the enterprise. Such flows of design work can be represented on a temporary basis "before", "during", "after", setting the schedule of works of the enterprise in accordance with the parameter. Temporary RVTI-the grammar of the language L (G) is an ordered non-empty sets eight

$$G = (V, \Sigma, \tilde{\Sigma}, C, E, R, T, r_0) \quad (1)$$

where $V = \{v_e, e = \overline{1, L}\}$ is auxiliary alphabet; $\Sigma = \{a_t, t = \overline{1, T}\}$ is terminal alphabet graphic language; $\tilde{\Sigma} = \{\widetilde{at}, t = \overline{1, T}\}$ is quasi terminal alphabet; C is set of clock identifiers; E is the set of temporal relations "Before", "During", "After" (initialization hours $\{c := 0\}$, relations form $\{c \sim x\}$, where x is a variable (clock identifier), x is a constant, $\sim \in \{=, <, \leq, >, \geq\}$); $R = \{r_i, i = \overline{0, I}\}$ is the grammar of the G schema (the set of product complex names, each r_i complex consists of a subset P_{ij} of products $r_i = \{P_{ij}, j = \overline{1, J}\}$); $T \in \{t_1, t_2, \ldots, t_n\}$ is a set of time stamps; $r_0 \in R$ is RV-axiom grammar [3].

In a mechanism of the grammar introduces an additional tape, which will contain information on the amount for the current item. When you return a label link, you must retrieve the value from the ribbon associated with the current item. We assume that time is spent only when the operation is performed. Add operation $W_1(ts^{t(5)})$, in which "ts" is the pre-calculated sum of two numbers: the sum of the current time for the element and its time characteristic. Also, when returning, $W_2(b^{1m}, b^{t(6)})$ is performed, that is, reading from the storage of the element itself and reading information from the corresponding tape about its amount of time determining the "ts". An example of a diagram that can be analyzed using RVTI-grammar is presented in Fig. 1. The timestamp for each object from the terminal alphabet is the main feature.

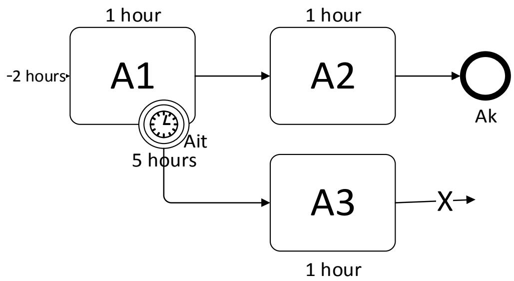

Fig. 1. Temporal BPMN diagram example

RVTI-grammar was developed for the basic elements of BPMN notation. The table form of this grammar is presented in Table 1.

Table 1. An example grammar for a simple BPMN diagrams

N	State	Quasi term	Next state	Operation with memory
1	r_0	A0	r1	o
2	r_1	rel	r3	o
3	r_2	labelEG	r3	$W_2(b^{1m}, b^{t(6)})$
4		labelPG	r3	$W_2(b^{2m}, b^{t(6)})$
5	r_3	Ai	r1	o
6		Aim	r1	o
7		Ait	r1	$W_1(t_s{}^{t(6)})$
8		Akl	r2	$W_3(e^{1m}, e^{2m})$
9		Ak	r4	o
10		A	r1	$W_1(t_s{}^{t(6)})$
11		EGc	r1	$W_1(t^{1m^{(n-1)}})/W_3(k = 1)$
12		EG	r2	$W_1(1^{t(1)}, k^{t(2)})/W_3(e^{t(2)}, k != 1)$
13		_EG	r2	$W_1(inc(m^{t(1)})/W_3(m^{t(1)} < k^{t(2)})$
14		_EGe	r1	$W_1(t^{1m^{(n-1)}})/W_3(m^{t(1)}=k^{t(2)}, p != 1)$
15		_EGme	r1	$o/W_3(m^{t(1)}=k^{t(2)}, p = 1)$
16		PGf	r1	$W_1(t^{2m^{(n-1)}})/W_3(k = 1)$
17		PG	r2	$W_1(1^{t(3)}, k^{t(4)})/W_3(e^{t(3)}, k != 1)$
18		_PG	r2	$W_1(inc(m^{t(3)})/W_3(m^{t(3)} < k^{t(4)})$
19		_PGe	r1	$W_1(t^{2m^{(n-1)}})/W_3(m^{t(3)}=k^{t(4)}, p != 1)$
20		_PGje	r1	$W_1(t^{2m^{(n-1)}})/W_3(m^{t(3)}=k^{t(4)}, p = 1)$
21	r_4	no_label	r5	*
22	r_5			

The next step is to develop a method to neutralize the errors of the modified grammar. Efficient error-handling technologies are essential to enable graphical language analyzers to detect as many errors as possible in a single pass. The method of error neutralization is based on the method of workarounds (finding a detour). This method involves ignoring part of the graph to try to find a state in which you can resume parsing the graph [4]. When designing the grammar, the types of graphical objects are distinguished, the instances of which will be used as successors of the analysis. It is clear that it is important to find the "Golden mean" when determining the number of types of analysis continuers, because when analyzing an incorrect chart, an excessively large number of objects - the successors of the analysis will affect the speed of analysis (spend a lot of time), but most of the chart will be analyzed. A small number of subscribers, on the contrary, will reduce the analysis time, but a smaller part of the chart will be analyzed. The basic algorithm of offset error shown in Fig. 2.

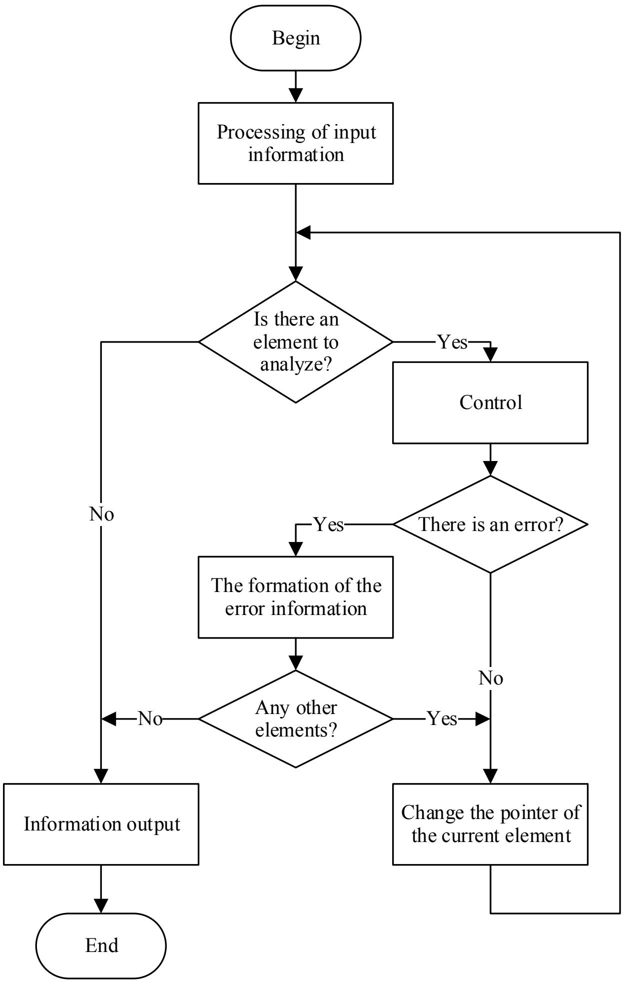

Fig. 2. The algorithm of errors neutralization

It should be noted that the proposed grammatical modifications support neutralization. The following errors that can occur when adding work with time characteristics are defined:

1. The first type of error-the block has no time characteristic. Following the above algorithm, sufficient neutralization of the algorithm is reduced to taking an empty value for zero.

2. The second type of error can occur when the connection is disconnected or in another similar situation, when it is necessary to resort to the search for continuers of the analysis. It can be assumed that the transition to a successor will lose the already calculated time indicators. However, following the algorithm, there is always a current value from which to continue working with the time parameters.

Neutralization of other types of errors is already checked on RV-grammar. The next problem is that most syntactically formal graphical languages are not semantically formal. Such languages are flexible and allow you to build diagrams that can be applied to different subject areas. The flexibility of languages is due to the incompleteness or informality of their description, as a result of which the resulting diagrams can be interpreted ambiguously. Machine processing of graphic schemes of such languages is difficult. Most researchers suggest two methods of solving this problem:

1. To specialize in the language. To give it some language features, to simplify it, to give new features, etc. This will have a positive impact on formalization.

2. The definition of dynamic semantics. Dynamic semantics involves converting the diagrams of the underlying graphical language into the target language.

The first method is effective if there is a need to introduce highly specialized and low-budget funds to automate the design process. The second method has great prospects, since there is no need to change the language, and it is more universal. For this purpose, a library with primitives representing some interpretation of graphic images in terms of the target graphic language is developed. Moreover, the target language is chosen as a more formal language relative to the base language. For the translation mechanism to work, the following functions must be introduced:

1) generate_input(): forming a set of input connection points, except for the one on which this graphic object was reached. It is performed at the initial analysis of graphical objects containing more than one input.

2) generate_output(): formation of a set of outgoing connection points. It is performed during the initial analysis of graphical objects containing more than one output, or when only one output is supposed to be used as a label reference, i.e. the direction of the analysis must be changed.

3) select_output(): select the outgoing connection point of the element as the successor of the destination language chain. From these points select the event function, which is performed for the graphic schemes with dynamically varying number of points outgoing connection after the formation of the set of connection points for outgoing connections. In General, the selection algorithm is not regulated, i.e. the choice is arbitrary.

4) stick_connection_points(): to associate a connection point and the object. An event function that is performed when analyzing secondary graphics that contain multiple inputs. Binds an incoming reference to an object connection point that is stored in internal memory.

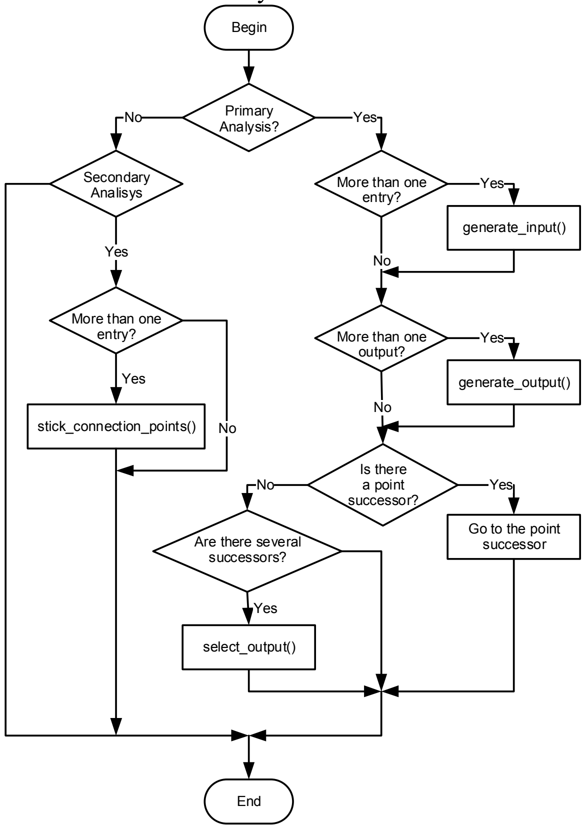

Fig. 3. The basic translation algorithm

The last thing to mention is the speed of the algorithm. The RV analyzer has a linear time characteristic of the control, i.e. the time spent on the analysis of the input list of chart elements is determined by the linear function of its length and calculated by the formula:

$$t = c \cdot L_s, \tag{1}$$

or

$$t = c \cdot k \cdot L_k, k = L_s \cdot L_k, \tag{2}$$

where L_s is the number of transitions on the state machine, L_k is the number of elements of the diagram, and C is constant implementation of the algorithm, which shows how many commands (operators) spent on the analysis of one object in the implementation of the control algorithm on a particular PC.

An experiment was conducted in which 500 diagrams were generated, for each of which the number of generated elements was determined and the number of states of the analyzing machine was calculated. The result of the experiment is presented in the form of a scatter plot in Fig. 4 confirms the linear dependence of the number of applications of grammatical rules on the number of graphical objects and links. In addition, the expectation of the coefficient of equal and average deviation was determined.

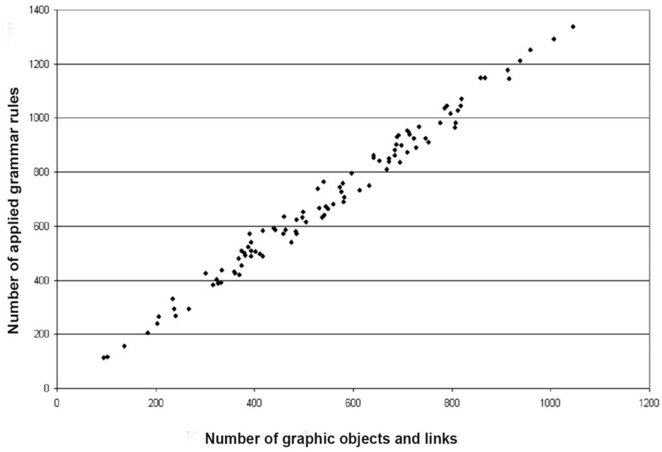

Fig. 4. Experimental values of linear coefficients for 500 diagrams

CONCLUSION

The temporal grammar of RVTI proposed by the authors has a linear characteristic of the time of the analysis of working processes, takes into account the language of the process description and can be applied to any diagram. Allows time parameters to be taken into account. This will help to identify logical planning errors in the development and design of automated systems. The proposed method of neutralization applied to this grammar allows to identify several errors during the passage. Translating the chart into a temporary Petri net adds new possibilities for analysis. Further work is the extension possibilities of semantic analysis chart patterns with the point of view of matching text attributes diagrams to the project documentation.

ACKNOWLEDGMENT

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

REFERENCES

[1] A.G. Sharov, A.N. Afanas'ev, “Avtomatnaya graficheskaya grammatika,” Vestnik Ul'yanovskogo gosudarstvennogo tekhnicheskogo universiteta, no. 1 (29). pp. 54-56, 2005.

[2] BPMN Specification - Business Process Model and Notation. http://bpmn.org

[3] A. N. Afanasyev, N. N. Voit and S. Y. Kirillov, “Development of RYT-grammar for analysis and control dynamic workflows,” International Conference on Computing Networking and Informatics (ICCNI), 2017. doi: 10.1109/ICCNI.2017.8123797, URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8123797&isnumber=8123766.

[4] O.G. Sharov, A. N. Afanas'ev, "Neutralization of syntax errors in the graphic languages," Programming and Computer Software. 2008, vol. 1, pp. 61-66, 2008.
[5] P. Kruchten, The Rational Unified Process. Addison-Wesley, 1998.
[6] G. Booch, I. Jacobson, J. Rumbaugh, The Unified Modeling Language User Guide. Addison-Wesley, 1998.
[7] G. Booch, Object-oriented Analysis and Design with Applications, 2nd edition. Addison-Wesley, 1994.
[8] HOORA's homepage. URL: http://www.hoora.org/
[9] I. Jacobson, M. Christerson, P. Jonsson, O. Gunnar, Object-Oriented Software Engineering: A Use Case Driven Approach. Addison-Wesley, 1992.
[10] J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, W. Lorensen, Object-Oriented Modeling and Design. Prentice Hall, 1991.
[11] T. Gilb, Competitive Engineering. Addison-Wesley, 2003.
[12] S. Shlaer, S. Mellor, Object-Oriented System Analysis: Modeling the World in Data. Yourdon Press Computing Series. Prentice-Hall, 1988.
[13] D.L. Parnas, Software Requirements for the A-7 Aircraft. Technical Report 3876. Naval Research Laboratory. Washington DC, 1978.
[14] M. Alford, A Requirements Engineering Methodology for Real Time Processing Requirements. IEEE Transaction on Software Engineering, vol. 3, no.1, pp. 60-69, 1977.
[15] S. Andriole, Storyboard Prototyping for Systems Design: A New Approach to User Requirements Analysis. Q E D Pub Co, 1989.
[16] D.A. Marca, C.L. McGowan, SADT: Structured Analysis and Design Techniques. McGraw-Hill, 1988.
[17] P. Parviainen, H. Hulkko, J. Kaariainen, J. Takalo, M. Tihinen, Requirements engineering inventory of technologies. VTT PUBLICATIONS, 2003.
[18] Program Verification and temporal logics. URL: http://logic.pdmi.ras.ru/~yura/modern/034.pdf (in Russian)
[19] N. Saeedloei, G. Gupta, "Timed definite clause omega-grammars," LIPIcs-Leibniz International Proceedings in Informatics, vol. 7, Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2010.
[20] Y. Wang, Y. Fan, "Using temporal logics for modeling and analysis of workflows," E-Commerce Technology for Dynamic E-Business, 2004. IEEE International Conference on. IEEE, 2004.
[21] YAHODA. URL: http://web.archive.org/web/20120220001353/http://anna.fi.muni.cz/yahoda/
[22] Workflow Handbook 2005:/ Layna Fischer (editor), Published by Future Strategies Inc., Book Division.
[23] CPN Tools. URL: http://cpntools.org/
[24] Roméo. URL: http://romeo.rts-software.org
[25] TimesTool. URL: http://www.timestool.com/documentation.shtml
[26] Tina Toolbox. URL: http://projects.laas.fr/tina/
[27] Visual Object Net++. URL: https://www.techfak.uni-bielefeld.de/~mchen/BioPNML/Intro/VON.html
[28] T. Andersson, A. Andersson-Ceder, I. Bider, "State flow as a way of analyzing business processes-case studies," Logistics Information Management, 15(1), pp.34-45, 2002.
[29] YAWL Foundation, 2004. YAWL. URL: http://www.yawlfoundation.org/
[30] I. Bider, "Analysis of Agile Software Development from the Knowledge Transformation Perspective," 13th International Conference on Perspectives in Business Informatics Research (BIR 2014), Lund, Sweden. Springer, LNBIP, 2014.
[31] IbisSoft, iPB Reference Manual. URL: http://docs.ibissoft.se/node/3
[32] A. Jalali, P. Wohed, C. Ouyang, "Aspect Oriented Business Process Modelling with Precedence,". In Business Process Model and Notation, LNBIP, Vol. 125, Springer, pp.23-37, 2012.
[33] Hasso Plattner Institut. http://bpt.hpi.uni-potsdam.de