Development of timed RT-grammars for analysis of business process at manufacturing and in cyber-physical systems

N.N. Voit
«Computing technique» department
Ulyanovsk State Technical University
Ulyanovsk, Russia
n.voit@ulstu.ru

***Abstract*—Author offers timed automaton RT-grammars with a memory to analyze complex business process at manufacturing of the large industrial enterprise. The practical application of RT-grammars to the well-known generalized railroad crossing problem is presented too**

***Keywords*—timed grammars; cyber-physical systems; analysis; workflows; business process**

I. Introduction

Applying timed automaton for designing, specification, controlling and analyzing real-time systems is well-known practice [1]. Timed and hybrid automatons is used for analyzing and managing manufacturing (MM) and workflows of cyber-physical systems (WCPS) [2].

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 [3], 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 [6]. 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 [1]. 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. Author offers that mechanism based on automaton RT-grammars with a memory.

II. Related works

Author researches some works which consider workflow specification, verification and translation. Several works have focused on specifying formal semantics and verification methods for workflows using Petri nets, process algebra, abstract state machine, see, e.g., [5,6,7,8,9,10,11,12,13,14,15,19,25]. In [12], Decker and Weske offer a Petri Net-based formalism for specifying choreographies, properties as reliability and local enforceability, and method to check these two properties. However, they consider only synchronous communication, and have not research mappings to high-level interaction modeling languages as BPMN. In work, [16] studies BPMN behaviors from a semantic point of view, and proposes several BPMN pat-terns. This work is not theoretically grounded and is not complete, that reviews only a few patterns. Lohmann and Wolf [17] propose an analysis with help existed patterns and to control these with help compatible patterns. In [18], author has focused on the translation of BPMN into process algebra to analyze choreographies with help model and equivalence checking.

The main limit of considered methods are that they do not work in the presence of different diagram types at once, meaning that in some cases, the input diagrams cannot be analyzed.

III. Definition timed RT-grammars

The automaton timed RT-grammar is a R-grammar extension [20-24]. It is represented by six following components:

$$G = (V, \sum, C, E, R, r_0), \tag{1}$$

where $V = \{\nu_e,\ e = \overline{1, H}\}$ is an auxiliary alphabet (the alphabet of operations over internal memory); $\sum = \{a_l,\ l = \overline{1, L}\}$ is a terminal alphabet of a language; C —is a finite set of clock identifiers; $E-$ is a set of time expressions C (clock limitation and clock reset), is limited by the following expressions: from the beginning $\{c := 0\}$ and onwards $\{c \sim x\}$,

978-1-5090-4642-3/17/$31.00 ©2017 IEEE

and c is a variable, and x is a constant, $\sim \in \{=,<,\leq,>,\geq\}$; $R = \{r_i,\ i = \overline{0,I}\}$ is grammar scheme G (a set of names of production rules' complexes, each complex r_i consists of subset P_{ij} of production rules $r_i = \{P_{ij},\ j = \overline{1,J}\}$); $r_0 \in R$ is a RT-grammar axiom (the name of the initial complex of production rules), $r_k \in R$ is a conclusive complex of production rules. Production rule $P_{ij} \in r_i$ is given as $P_{ij}: a_l \frac{W_v(\gamma_1,\dots,\gamma_n)}{E} r_m$, where $W_v(\gamma_1, \dots, \gamma_n)$ is a n-th relation, which determines the type of operation over the internal memory depending on $v \in \{0,1,2\}$ (respectively, 0 – operation is not performed, 1 – record, 2 – read); $\gamma_1, \dots, \gamma_n \in V$; $r_i \in R$ is the name of the complex of a production rule's source; $r_m \in R$ is the name of the complex of a production rule's successor.

Example 1. The author deals with a language in which each sequence a is accompanied by the same number of sequence b, and there are at least two a and two b. For each pair of sequences a and b, the first symbol b should appear within 5 units of time from the first symbol a, and the final symbol b should appear within 20 units of time from the first character a. The context-free timed grammar of this language [10] is defined as:

$$
\begin{aligned}
S &\to R\,S,\\
R &\to a\,\{c := 0\}T\,b\,\{c < 20\},\\
T &\to a\,T\,b,\\
T &\to a\,b\,\{c < 5\}.
\end{aligned}
$$

The author gives a definition of this language:

$$L = \{a^n(b, c < 5)(b^m, c < 20) | m > 0, n = m + 1\} \quad (2)$$

Graphical RT-grammar representation (as a stack-like internal memory automata) analyzing the language (3) is shown in Fig. 1.

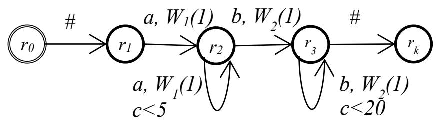


Fig. 1. Timed stack memory RT-automata

Example 2. A classic example of a generalized railroad crossing problem [3] can be described by a timed automaton (Fig. 2).

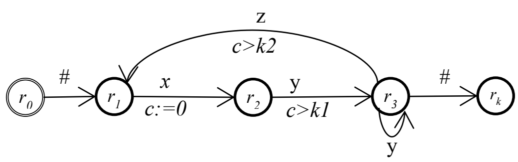


Fig. 2. A timed automaton that formalizes a generalized railroad crossing problem

The train's states (locations) r_0 and r_k are axioms. In location r_1 the train is approaching to the railroad track crossing; the clock is set to zero. The location r_2 is before the railroad track crossing, so the railroad track is open only for the train located in this location; moreover, other railroad tracks are closed after time k_1. Then the train is in location r_3 up to time k_2, after that it leaves the location r3, and this track is open for the next train.

The context-free grammar [10] is given as:

$$
\begin{aligned}
S &\to R\,S,\\
R &\to x\,\{c := 0\}\,T\,z\,\{c > k_2\},\\
T &\to y(c > k_1)T,\\
T &\to yT,\\
T &\to y.
\end{aligned}
$$

A timed context-free language describing the cyber-physical system of railroad track crossing (see Fig. 2) is defined as:

$$L = \{(x, c := 0)(y, c > k_1)y^n(z, c > k_2) | n, k_1, k_2 > 0\} (3)$$

IV. The Temporal Business Process Model to Coordinate Electronic Workflows

The stage to coordinate electronic workflows contains two levels: upper (Fig. 2) and lower (Fig. 3), that is written in special visual language.

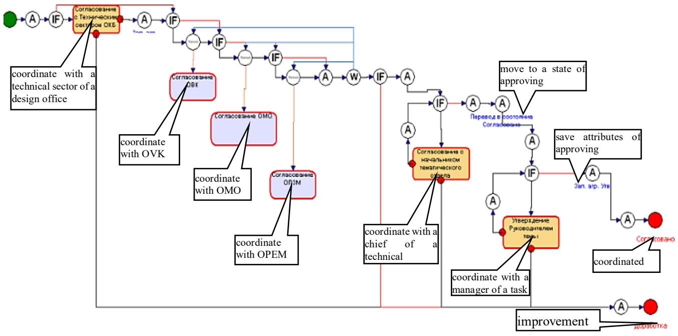


Fig. 3. Top level of the coordination

The top level represents the laboratory coordination in respect of a correctness of the scheme (verification of electric circuits, nomenclatures, etc.). The bottom level represents the coordination of construct (technology of radio mounting, etc.). The specified workflows are presented in the specialized language allowing to organize conditional and side-by-side execution of works. The topological correctness (especially in respect of deleted "And", "OR" branching and their merges) is offered to be carried out by means of the author's device of RT-grammars [20-24].

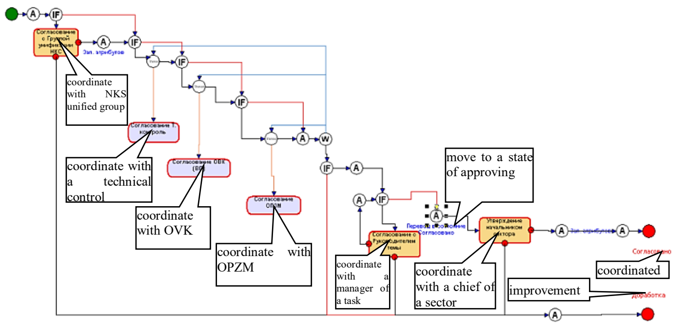
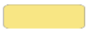


Fig. 4. Bottom level of the coordination

The specialized visual language allows to organize an enclosure of processes at the expense of several elements. The curtailed sub process is developed for visualization of work-flows in the form of diagrams in software products of ASCON [26]. Such specialized visual language is used by one of the large industrial enterprise as JSC “Ulyanovsk mechanical plant”. A creation of parallel workflows is supported in this language. Control of such workflows is possible by means of two elements “Events” and “Semaphore” which work together with the expectation element and additional type of communication of “Synchro action”. There is “Phantom” element which allows to connect parts of the diagram at the different levels of an enclosure. A number of graphic elements of language is shown in Table 1.

TABLE I. ELEMENTS OF THE SPECIALIZED LANGUAGE

Name	Graphical element	Description
Procedure		The curtailed subprocess which is possible for causing repeatedly. Has only one entering communication like “To pass into the procedure”
Task		The curtailed subprocess in which action is obligatory to performance by the user
Iteration		The curtailed subprocess which performance is required repeatedly
Call procedure		It is used together with “to pass into the procedure” and the procedure block
Create workflow		Create a new workflow
Intransitive		The operation which is carried out by the user
Script (Auto operation)		
Branching		Has only two proceeding communications. True and false respectively
Phantom		Allows to connect parts of the chart at the different levels of an enclosure. In fact, is communication
Event		It is used together with “Expectation”. Rub entering, one of them “Synchro action” notifying on performance of an event.
Semaphore		It is used together with "Expectation". Rub entering, one of them “Synchro action” notifying on the beginning and completion of events.
Activate		Two proceeding branches, one of them “Synchro action” notifying “Event” on a successful completion of an event have
Waiting		Two proceeding branches, one of them “Synchro actrion” monitoring the status of performance of an event have. Passes a workflow only in case of a successful completion of an event
Increment		Two proceeding branches, one of them “Synchro action” notifying “Semaphore” on the beginning of execution of an event have
Decrement		Two proceeding branches, one of them “Synchro action” notifying “Semaphore” on completion of execution of an event have

The developed automaton RT-grammar of language (Table 2) allows to carry out the analysis of topology of diagrams of the specified language and to reveal mistakes.

TABLE II. RT-GRAMMAR OF THIS VISUAL LANGUAGE

Start State	Quasiterm	End State	Operations with memory
r0	A0	r3	o
r1	return	r2	$w_2(b^{4m})$
r2	vA	r1	$w_1(s^{1m},t^{4m})$, CALL vA
	vIT	r1	$w_1(s^{1m},t^{4m})$, CALL vIT
	Ak	r4	o
	Akm	r5	$w_1(1^{t(1)},i^{t(2)})/w_2(e^{t(1)})$
	_Akm	r5	$w_1(inc(m^{t(1)}))/w_3(m^{t(1)}<k^{t(2)}-1)$
	Akme	r4	$w_1(inc(m^{t(1)}))/w_3(m^{t(1)}=k^{t(2)}-1)$
	CL	r6	$w_1(t^{4m})$
	TH	r6	$w_1(1^{t(7)}, i^{t(8)}, t^{4m})$
	SC	r3	o
	SCm	r5	$w_1(1^{t(3)},i^{t(4)})/w_2(e^{t(3)})$
	_SCm	r5	$w_1(inc(m^{t(3)}))/w_3(m^{t(3)}<k^{t(4)}-1)$
	SCme	r3	$w_1(inc(m^{t(3)}))/w_3(m^{t(3)}=k^{t(4)}-1)$
	C	r7	$w_1(t^{2m})$
	EV	r3	$w_1(0^{t(5)}, 0^{t(9)},0^{t(11)})/w_2(e^{t(5)})$
	S	r3	$w_1(0^{t(6)}, 0^{t(10)},0^{t(12)})/w_2(e^{t(6)})$
	F	r11	$w_1(t^{3m})$
	W	r9	$w_1(t^{3m})$
	IN	r11	$w_1(t^{3m})$
	D	r12	$w_1(t^{3m})$
r3	rel	r2	o
r4	no_label	r17	*
r5	labelC	r2	$w_2(b^{2m})$
r6	prel	r13	o
r7	nrel	r2	o
r8	PHsp	r6	o
r9	arel	r14	o
r10	PHsa	r9	o
r11	airel	r15	o
r12	adrel	r16	o
r13	vPR	r1	$w_1(s^{1m})$, CALL(vPR)
	PHep	r8	o
r14	THa	r2	$w_1(inc(m^{t(7)}))/w_3(m^{t(7)}<k^{t(8)})$
	PHea	r10	o
	EVa	r2	$w_1(1^{t(9)}), w_2(b^{3m})$
	Sa	r2	$w_1(1^{t(10)}), w_2(b^{3m})$
r15	EVa	r2	$w_1(inc(m^{t(5)}), 1^{t(11)}), w_2(b^{3m})$
	Sa	r2	$w_1(inc(m^{t(6)}), 1^{t(12)}), w_2(b^{3m})$
r16	Sa	r2	$w_1(dec(m^{t(6)}), 1^{t(12)}), w_2(b^{3m})$

Since the notation of the language has sub processes, the grammar is hierarchical, the CALL and RETURN functions are used to organize the analysis. A global nesting depth independent store is introduced, in which the automaton state is stored before entering the sub process. Some elements can generate the main type of connection “Go to” many times. All other types of connections come from the elements one time. Therefore, when going to elements with different types of communication, initially there is a transition to “non-typical” element with a reference to the continuation of the main stream

The grammar uses 4 stacks and 12 tapes. The purpose of the stacks is as follows.

A global stack is to store the state before entering the sub-process; for the “Branch” element; to return to the items after the transition using the type of the “Synchronization” connection, to store the quasiterm, from which the analysis will continue after the return from the sub process.

The purpose of the tapes is as follows.

Three tapes are to count the analyzed incoming branches of the elements “Exit Point”, “Auto Operation” and “Create Flow”.

Three tapes are to store the total number of incoming branches of the elements “Exit Point”, “Auto Operation” and “Create Stream”.

Two tapes are to count the status of elements “Event” and “Semaphore”.

Two tapes are to control the communication of event elements and “Wait for event”.

Two tapes are to control communication and controlling events.

The condition for successful execution (*) is empty stacks; completely calculated branches of the elements “Exit point”, “Auto-operation” and “Create flow”.

The value 1 for each passed “Event” element in the corresponding tape.

The value 0 for each passed “Semaphore” element in the corresponding tape.

A value of 1 for each passed element in the tapes control-ling relationships of the event elements.

The developed RT-grammar allows defining the following types of topological errors: absence of communication, hang-ing connection, control transfer error, inadmissible connection of elements, “Deadlock”, hang, uncertainty, absence of flow path, violations in complex structures of parallel-sequential processes and others.

Analysis of the text component of the diagram blocks is performed using the method [12], which consists in the dynamic construction of the graph of concepts and the coordination with it of the next text component.

V. CONCLUSION AND FUTURE WORKS

Author researches for domain of MM&WCPSs from point of view problems as deadlock, reachability and others. So, he developed automaton RT-grammar with a memory to analyze and check (fix) complex electronic workflows of manufacturing at the large industrial enterprise and for cyber-physical systems. There are some examples that are how to use this RT-grammar for famous problems as generalized railroad crossing and coordination of electronic workflows at the large industrial enterprise. There is a specialized visual language at Ulyanovsk Mechanical Plant using for the production-support work, which author tries to formalize and gives RT-grammar for it. As result of using this RT-grammar at that plant, the production-support work becomes more efficiency (less time is spent on to coordinate the electronic workflows).

In the future works, the author is going to deal with timed grammars for the analysis, control and translation of dynamic project workflows’ problems, in which the time factor occupies a significant place. Use of such grammars will eliminate a number of semantic errors at the stage of conceptual design of complex computer systems.

ACKNOWLEDGMENT

This research is supported by the grant of the Ministry of Education and Science of the Russian Federation, the project № 2.1615.2017/4.6. The reported study was funded by RFBR and Government of Ulyanovsk Region according to the research project № 16-47-732152. The reported study was funded by RFBR according to the research project № 17-07-01417.

REFERENCES

[1] C. L. Heitmeyer and N. A. Lynch, “The generalized railroad crossing: A case study in formal verification of real-time systems,” IEEE RTSS, 1994, pp. 120-131.

[2] Yu. G. Karpov, “MODEL CHECKING. Verifikaciya parallel'nyh i raspredelennyh programmnyh system,” SPb.: BHV-Peterburg, 560 s, 2010 (in Russian).

[3] E. A. Lee, “Cyber-physical systems: Design challenges,” ISORC, 2008.

[4] F. Abouzaid, A. Bhattacharyya, N. Dragoni, J. S. Fitzgerald, M. Mazzara, and M. Zhou, “A Case Study of Workflow Reconfiguration: Design, Modelling, Analysis and Implementation,” Newcastle University, Technical report CS-TR-1265, July 2011. URL: http://www.cs.ncl.ac.uk/publications/trs/papers/1265.pdf Cited October 18, 2017.

[5] A global Swiss company offering advanced intelligent application software for multiple business sectors. URL: http://whitestein.com/ Cited October 18, 2017.

[6] I. Bider, A. Jalali, “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,” Springer-Verlag Berlin Heidelberg, 2014. URL: https://www.researchgate.net/publication/266078141 Cited October 18, 2017.

[7] T. Andersson, A. Andersson-Ceder, and I. Bider, “State flow as a way of analyzing business processes-case studies,” Logistics Information Management, vol. 15(1), pp. 34-45, 2002. URL: http://www.ibissoft.com/publications/Cases.pdf Cited October 10, 2017.

[8] YAWL Foundation, YAWL. URL: http://www.yawlfoundation.org/ Cited October 10, 2017.

[9] I. Bider, “Analysis of Agile Software Development from the Knowledge Transformation Perspective,” in Perspectives in Business Informatics Research. BIR 2014. Lecture Notes in Business Information Processing, vol 194, Johansson B., Andersson B., Holmberg N., Eds. Springer, Cham, 2014.

[10] IbisSoft, “iPB Reference Manual”. URL: http://docs.ibissoft.se/node/3 Cited October 18, 2017.

[11] T. Bultan “Analyzing Interactions of Asynchronously Communicating Software Components,” in Formal Techniques for Distributed Systems. Lecture Notes in Computer Science, vol 7892, Beyer D., Boreale M., Eds. Springer, Berlin, Heidelberg, 2013. URL: https://link.springer.com/chapter/10.1007/978-3-642-38592-6_1 Cited October 18, 2017.

[12] Orchestration and Workflow. URL: https://www.cloudenablers.com/blog/orchestration-and-workflow/ Cited October 18, 2017.

[13] W. M. P. Van der Aalst, “Don’t go with the flow: Web services composition standards exposed,” IEEE intelligent systems, vol. 18, no. 1, pp. 72-76, 2003. URL: http://www.martinfowler.workflowpatterns.com/documentation/documents/ieeewebflow.pdf Cited October 18, 2017.

[14] D. A. Marca and C. L. McGowan. SADT: structured analysis and design technique. McGraw-Hill, Inc., 1987. URL: http://dl.acm.org/citation.cfm?id=31837 Cited October 18, 2017.

[15] TP026B, Rev. "Rational Unified Process". URL: https://www.ibm.com/developerworks/rational/library/content/03July/1000/1251/1251_bestpractices_TP026B.pdf Cited October 18, 2017.

[16] ISO. LOTOS – A Formal Description Technique Based on the Temporal Ordering of Observational Behaviour. Technical Report 8807, ISO, 1989.

[17] N. Lohmann and K. Wolf, “Realizability Is Controllability,” WS-FM’09, vol. 6194 of LNCS, 2010, pp. 110–127.

[18] P. Poizat and G. Salaün, “Checking the Realizability of BPMN 2.0 Choreographies,” SAC'12, 2012, pp. 1927-1934.

[19] Y. Wang, Y. Fan, “Using Temporal Logics for Modeling and Analysis of Workflows,” Proceedings of E-Commerce Technology for Dynamic E-Business, IEEE International Conference, 2004. URL: http://ieeexplore.ieee.org/abstract/document/1388319/ Cited October 18, 2017.

[20] A. Afanasyev, N. Voit, “Intelligent Agent System to Analysis Manufacturing Process Models,” Proceedings of the First International Scientific Conference «Intelligent Information Technologies for Industry» (IITI’16) vol.451 of the series Advances in Intelligent Systems and Computing. Russia, 2016, pp. 395-403.

[21] A. Afanasyev, N. Voit, R. Gaynullin, “The Analysis of Diagrammatic Models of Workflows in Design of the Complex Automated Systems,” Proceedings of the First International Scientific Conference «Intelligent Information Technologies for Industry» (IITI’16) vol. 450 of the series Advances in Intelligent Systems and Computing. Russia, 2016, pp. 227-236.

[22] A.N. Afanasyev, N.N. Voit, R.F. Gainullin, “Diagrammatic models processing in designing the complex automated systems,” 10th IEEE International Conference on Application of Information and Communication Technologies (AICT). Baku, Azerbaijan, 2016, pp. 441-445.

[23] A.N. Afanasyev, N.N. Voit, E.Yu. Voevodin, R.F. Gainullin, “Control of UML diagrams in designing automated systems software,” Proceedings of the 9th IEEE International conference on Application of Information and Communication Technologies: AICT – 2015, 2015, pp. 285-288.

[24] A.N. Afanasev, N.N. Voit, E.Yu. Voevodin, R.F. Gainullin, “Analysis of Diagrammatic Models in the Design of Automated Software Systems,” Object Systems – 2015: Proceedings of X International Theoretical and Practical Conference (Rostov-on-Don, 10-12 May, 2015) / Edited by Pavel P. Oleynik. – Russia, Rostov-on-Don: SI (b) SRSPU (NPI), 2015, pp. 124-129.

[25] N. Saeedloei and G. Gupta, “Timed definite clause ω-grammars,” Technical Communications of the International Conference on Logic Programming, 2010, pp. 212–221. URL: http://www.floc-conference.org/ICLP-home.html Cited October 18, 2017.

[26] ASCON. URL: http://ascon.net/main/ Cited October 18, 2017.