Analysis and Control of Hybrid Diagrammatical Workflows

Alexander Afanasyev(✉), Nikolay Voit, Oksana Timofeeva, and Vyacheslav Epifanov

Ulyanovsk State Technical University, Ulyanovsk, Russia
{a.afanasev,n.voit,o.timofeeva}@ulstu.ru

Abstract. The paper presents a method for analyzing hybrid project workflows based on the author's RV-grammar by using BPMN as an example, as well as translations of such project workflows into languages describing business processes (e.g., BPEL with the author's RVTt-grammar). The effectiveness of author's grammars for hybrid project workflows' analysis and translation at large design and manufacturing enterprises is assessed.

Keywords: Diagrammatic workflows · Grammars · Analysis · Translating

1 Introduction

Workflows are an activity in solutions of project tasks and reengineering tasks of business process's representation. A person, device or program can generate this activity. Workflow control systems are represented by frameworks that provide the workflows accumulation, as well as workflows' scheduling and presentation as graphic notations, e.g., BPMN, UML, IDEF0, WPDL, etc. A visual presentation (diagrams) of business processes is designed to assist designers with project solutions' development and analysis, designing preproduction, preproduction engineering with logical reasoning on specific complex business processes. Many project workflow control systems were developed for different paradigms and have a scientifically closed research and application [1]. The workflow analysis method depends on the workflow modeling method. As a rule, methods are divided into modeling and non-modeling [2]. Non-modeling methods include activity diagrams of UML, WPDL, BPMN, in which there are no techniques and properties for analysis. Modeling methods include Petri nets (as the main representative) and π-calculus. At present, π-calculus is a promising but still very young and evolving theory. It has many open questions and unresolved problems. Petri nets have the following limitations:

The reported study was funded by RFBR and Government of Ulyanovsk Region according to the research project № 16-47-732152. 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 according to the research project № 17-07-01417.

A. Abraham et al. (eds.), *Proceedings of the Second International Scientific Conference "Intelligent Information Technologies for Industry" (IITI'17)*,
Advances in Intelligent Systems and Computing 679, DOI 10.1007/978-3-319-68321-8_13

- there is no a universal framework for project workflows' modeling and analysis on the basis of Petri nets. In order to analyze various properties (liveliness, attainability, safety), workflows are modeled in different types of Petri nets, which are ad hoc.
- there is no a technique that can assist a designer in modeling and ensure the successful completion of the task with the necessary requirements (properties).

The model checking method was extensive use for workflow analysis in error-free systems development at conceptual design stage. However, it is intended for experienced scientists and engineers, since it is complicated to understand and operate [2].

When designing and developing complex computer-based systems (for example, Web services in accordance with SOA), developers identify orchestration and choreography in the system architecture [3]. Orchestration is a description of an enterprise's internal business process as an interaction flow between enterprises' internal and external Web services. The point of view on this process is purely internal, e.g., the management staff's point of view or the business process owner's point of view. BPEL (Business Process Executive Language), XPDL (XML Process Definition Language), UML are modeling languages for describing the orchestration. In order to execute an orchestration, it is required the presence of a central processor that calls Web services. In this case Web services "do not know" that they participate in a more global business process. Choreography is a definition of a condition sequence in compliance with several independent participants' exchange messages in order to perform some general business tasks (e.g., B2B). Accordingly, WS-CDL (Web-services - Choreography Description Language) and ebXML (Electronic Business using eXtensible Markup Language) are modeling languages for describing the choreography. There is no a central coordinator in business process choreography, since each Web service "knows" when to perform its operations and with what a Web service it interacts.

Thus, it can be said that there are many different (hybrid) languages describing business processes in orchestration and choreography, as well as in modeling, analysis, control and optimization of project workflows. In order to analyze and translate hybrid project workflows with greater efficiency, the authors developed a framework solving these problems based on the author's RV-, RVTt-grammars. The author's framework presents both the models of hybrid design workflows and their properties.

The paper has the following structure. Section 2 has related work that short describes major relevant research studies. Section 3 provides a brief overview of the orchestration, choreography and the translation's purpose. Section 4 contains RV-, RVTt-grammars for hybrid project workflows analysis and translation. Section 5 presents experimental results of the RV-, RVTt-grammars' application on chosen function. Conclusions and further research directions are presented in Conclusion.

2 Related Work

Authors research 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., [4–14]. In [11], Decker and Weske offer a Petri Net-based formalism for specifying choreographies, properties as realizability 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. Bultan and Fu [15] define sufficient condition to analyse realizability of choreographies specified with UML collaboration diagrams (CDs). In [16], Salaün and Bultan modify and extend this work with method to analyse realizability by adding synchronization message among peers. This method controls realizability of CDs for bounded asynchronous communication. The realizability problem for Message Sequence Charts (MSCs) has also been studied (e.g., [17, 18]). [18] proposes bounded MSCs graphs that are limited BPMN 2.0, because branching and cyclic behaviors are not supported by CDs and MSCs (absent a choice in CDs, absent some cyclic behaviors in MSCs, and only self-loops in CDs). In work, [19] studies BPMN behaviors from a semantic point of view, and proposes several BPMN patterns. This work is not theoretically grounded and is not complete, that reviews only a few patterns. Lohmann and Wolf [20] propose an analysis with help existed patterns and to control these with help compatible patterns. In [21], authors have focused on the translation of BPMN into process algebra to analyse 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.

3 Orchestration and Choreography

The messaging tools, by which several independent agents tend to achieve a desired state, are called “choreography”, and the services’ interaction is called “orchestration”. Special programming tools (BPEL4WS, XLANG, WSFL) were developed for orchestration (i.e., for the business logic’s description). These tools were developed with the participation of the largest vendors such as IBM, Microsoft, Oracle and BEA Systems. When using choreography, the messages’ sequences between several participants and sources are tracked (Fig. 1).

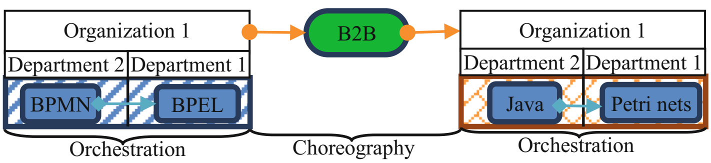


Fig. 1. An example of the place and role of orchestration and choreography in manufacturing plants

Translation programs are used to convert hybrid workflows. For example, the BPEL description of a business process is translated from BPMN (Fig. 2), and then the

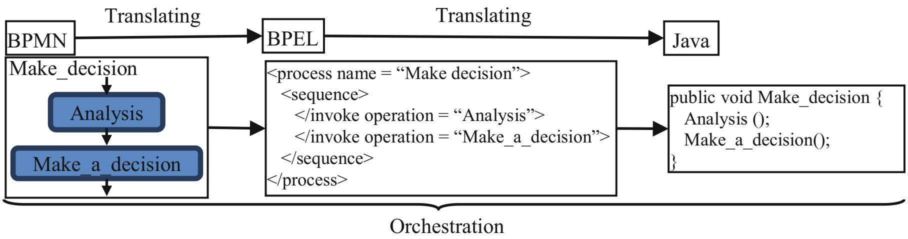


Fig. 2. Translating BPMN into BPEL and translating BPEL into Java

source code in a specific language (e.g., Java) is translated from the BPEL description of a business process. The authors developed RVTt-grammar for this purpose.

4 RV-, RVTt-Grammar

4.1 Review of Grammars

The authors review a reserved graph grammar, a positional grammar, a relational grammar, and a Web grammar to diagram business process and workflow by the tools [22–26]. A positional grammar: it uses a plex-structure, which does not have attaching points, and does not consider links of inputs\outputs, therefore, a designer cannot use it for graphical languages. A relational grammar: it generates a non-exhaustive list of errors that cannot be defined to analyze. A reserved graph grammar and a Web grammar: they save sentential form of a diagram that takes a lot of time to analyze it. For a reviewed grammar, the authors summarize problems as: an increasing number of rules to describe the grammars take much exponential or polynomial time for analysis; reviewed grammars have a sentential form to diagram; there are no tools for a semantic analysis of diagram’s attributes.

4.2 RV-Grammar

The authors have developed automaton grammar, called RV-grammar, to analysis and check (control) diagrams for the tools, described in the papers [27–30].

RV-grammar has a basis of the L (R) language grammar, which can be written as:

$$G = \left(V, \Sigma, \tilde{\Sigma}, R, r_0\right), \qquad (1)$$

where $V = \left\{v_l, l = \overline{1, L}\right\}$ is an auxiliary alphabet (the alphabetical operations with the internal memory); $\Sigma = \left\{a_t, t = \overline{1, T}\right\}$ is a terminal alphabet, which is the union of its graphic objects and links (the set of primitives); $\tilde{\Sigma} = \left\{\tilde{a}_t, t = \overline{1, \widetilde{T}}\right\}$ is a quasi-terminal alphabet, extending the terminal alphabet. The alphabet includes: quasi-terms of graphic objects; quasi-terms of graphic objects with more than one input; quasi-terms of links, marked with the specific semantic; quasi-term for the end of an analysis;

$R = \{r_i, i = \overline{1, I}\}$ is the scheme of the grammar (a set of rules, where each complex r_i consists a subset P_{ij} of rules, where $r_i = \{P_{ij}, j = \overline{1, J}\}$); $r_0 \in R$ is an axiom of RV-grammar (the initial complex of rules), $r_k \in R$ is a final complex of rules.

The complex of rules $P_{ij} \in r_i$ is given as:

$$\tilde{a}_t \xrightarrow{W\gamma(\gamma_1,\ldots,\gamma_n)} r_m, \qquad (2)$$

where $W_\gamma(\gamma_1, \ldots, \gamma_n)$ – n-ary relation, defining an operation with the internal memory depending on $\gamma \in \{0, 1, 2, 3\}$; $r_m \in R$ is the receiver of rules.

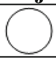
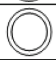
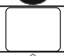
4.3 RVTt-Grammar

RVTt-grammar is a translating grammar that allows the system to make a syntax-oriented translation of graphical language diagrams into textual formal descriptions. This is a grammar, for which the target is a textual formal description (programs in the programming language, text in a specific markup language, etc.). RVTt-grammar is the development of the RV-grammar, in which production rules of a grammar scheme are expanded to store correspondences in terms of the target formal description, and the internal memory stores the information that is necessary for the translation process.

RVTt is a grammar of the L (G) language. It is the ordered seven non-empty sets:

$$G = (V, U, \Sigma, \tilde{\Sigma}, M, R, r_0), \qquad (3)$$

Table 1. Mapping BPMN-object into a tag of BPEL

A graphical object of BPMN	Quasi-term	A tag of BPEL
	A0	<receive>
	Ai	<wait>
	Aim	<receive>
	Ak	-
	A	<extensionActivity>
	EG	<if>
	_EG	<flow>
	_EGe	-
	labelEG	
	labelPG	
	no_label	

where the set of V, Σ, $\tilde{\Sigma}$, R, r_0 are inherited from the RV-grammar and are used to determine the syntactical correctness of the analyzed diagram; $U = \left\{\nu_e,\ e = \overline{1,K}\right\}$ is an alphabet of operations over the internal memory used by the target language; $M = TT \cup TN$ is a union of the alphabets of terminal (TT) and nonterminal (TN) symbols of the target language. The sets of U, M are necessary for providing the grammar with translating functions. They create a new formalism by expanding the RV-grammar. Production rule $P_{ij} \in r_i$ is given as

$$P_{ij} : \tilde{a}_t \xrightarrow{W_v(\gamma_1,\ldots,\gamma_n)} \gamma_m\{\chi\}, \tag{4}$$

Table 2. Set m and χ for RVTt-grammar

m	**χ of BPEL-script**
A	<sequence standard-attributes>
B	</sequence>
V	standard-elements
G	<if standard-attributes>
D	<condition> bool-expr</condition>
E	<elseif>*
Zh	</elseif>
Z	<else>?
I	</else>
K	</if>
L	<receive partnerLink="getPartnerLink()" operation="getOperationName()" createInstance="yes"? standard-attributes>
M	<receive partnerLink="getPartnerLink()" operation="getOperationName()" standard-attributes>
N	</receive>
O	<flow standard-attributes>
P	<links>?
R	<link name="NCName">+
S	</links>
T	</flow>
U	<extensionActivity>
F	<action standard-attributes>
H	</action>
Ch	</extensionActivity>
Sh	<wait standard-attributes>
Shi	<until expressionLanguage="anyURI"?>deadline-expr</until>
E	</wait>

where $W_\nu(\gamma_1, \ldots, \gamma_n)$ is n-th ratio that determines the type of an operation over the internal memory depending on $\nu \in \{0, 1, 2, 3\}$; $r_m \in R$ is a name of the complex of production rules-successor; χ is a display of quasi-term in terms of the target language (a symbols' set $m \in M$).

5 Experimental Results

The RVTt-grammar for translating from BPMN into BPEL is given in Tables 1, 2 and 3.

Figure 2 shows the efficiency. Graphic objects are graphic figures as a circle, a rectangle, a rhombus, a square, a line and etc. The authors propose a formula [27] for calculation of the efficiency that can be given as:

$$\mathrm{Required\_time} = c \cdot L_s, L_s = \sum\nolimits_{i=1}^{m} \left(\sum\nolimits_{j=1}^{V_i} v_{in_{ij}} + \sum\nolimits_{j=1}^{V_i} v_{out_{ij}} \right) + \sum\nolimits_{i=m+1}^{t} \left(V_i + \sum\nolimits_{j=1}^{V_i} v\_out_{ij} \right) + no\_label \quad (5)$$

where c is a constant of the algorithm's realization, which determines a quantity of time (clock) taken to analyze one graphic object; L_s is the number of graphic objects; V_i is the number of graphic objects of i-type; v_in_{ij} is the number of inputs to j-graphic objects of

Table 3. RVTt-grammar for translating BPMN into BPEL

№	Start state	Quasiterm	End state	RVTt-ratio	m
1	r0	A0	r1	o	A, L, N
2	r1	rel	r3	o	–
3	r2	labelEG	r3	$W_2(b^{1m})$	–
4		labelPG	r3	$W_2(b^{2m})$	–
5	r3	Ai	r1	o	Sh, Shi, E
6		Aim	r1	o	M, N
7		Ak	r4	o	B
8		A	r1	o	U, F, H, Ch
		EGc	r1	$W_1(t^{1m^{(n-1)}})/W_3(k = 1)$	G, D
9		EG	r2	$W_1(1^{t(1)}, k^{t(2)})/W_3(e^{t(2)}, k\ != 1)$	E, D
10		_EG	r2	$W_1(inc(m^{t(1)})/W_3(m^{t(1)} < k^{t(2)})$	Zh, E, D
11		_EGe	r1	$W_1(t^{1m^{(n-1)}})/W_3(m^{t(1)} = k^{t(2)}, p\ != 1)$	K, G, D
		_EGme	r1	$o/W_3(m^{t(1)} = k^{t(2)}, p = 1)$	K
12		PGf	r1	$W_1(t^{2m^{(n-1)}})/W_3(k = 1)$	O, A
		PG	r2	$W_1(1^{t(3)}, k^{t(4)})/W_3(e^{t(3)}, k\ != 1)$	B, A
13		_PG	r2	$W_1(inc(m^{t(3)})/W_3(m^{t(3)} < k^{t(4)})$	B, A
14		_PGe	r1	$W_1(t^{2m^{(n-1)}})/W_3(m^{t(3)} = k^{t(4)}, p\ != 1)$	B, T, O, A
		_PGje	r1	$W_1(t^{2m^{(n-1)}})/W_3(m^{t(3)} = k^{t(4)}, p = 1)$	B, T
15	r4	no_label	r5	*	–
16	r5				–

i-type; v_out_{ij} is the number of outputs from j-graphic objects of i-type; t is a total of object types; m is the number of object types with more than one output (Fig. 3).

Thus, RVTt-grammars possess a linear characteristic of time expenditure unlike known, for example, position, saving, relational, having exponential time of the analysis [27–30]. Regarding error control of RVTt-grammar allow to record the errors called a rupture of a context, connected with use of logical communications, such as 'AND', 'OR', 'XOR', and also context-dependent errors which will not be found by means of the mentioned tools can be in the text (notations) of any diagrams. They become "expensive" errors in design.

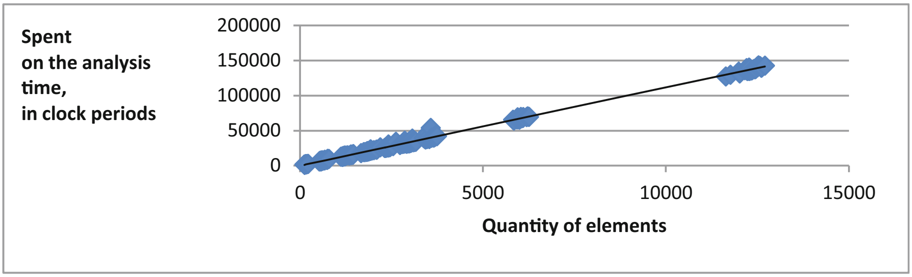


Fig. 3. Efficiency of analysis errors and translating with the help RVTt-grammar

6 Conclusion and Future Work

In this paper, the authors present a framework for the hybrid design workflows' analysis and translation. The analysis of existing approaches and tools was carried out and showed the advantage of RV-, RVTt-grammars. In future research works, the authors will examine the timed grammars of the hybrid design workflows' analysis, control and translation in which time factor occupies a significant place. The use of such grammars will eliminate a number of semantic errors at the stage of conceptual design of complex computer-based systems.

References

1. Georgakopoulos, D., Hornick, M., Sheth, A.: An overview of workflow management: From process modeling to infrastructure for automation. J. Distrib. Parallel Database Syst. **3**(2), 119–153 (1995). doi:10.1007/BF01277643
2. Wang, Y., Fan, Y.: Using temporal logics for modeling and analysis of workflows. In: Proceedings of IEEE International Conference on E-Commerce Technology for Dynamic E-Business (2004). doi:10.1109/CEC-EAST.2004.72
3. Bock, C.: Introduction to business process and definition metamodel. U.S. National Institute of Standard and Technology. Manufacturing Engineering (2008). https://www.nist.gov

4. Poizat, P., Salaün, G., Krishna, A.: Checking business process evolution. In: 13th International Conference on Formal Aspects of Component Software (FACS), <hal-01366641> (2016). https://hal.inria.fr/hal-01366641
5. Martens, A.: Analyzing web service based business processes. In: FASE 2005, pp. 19–33 (2005). doi:10.1007/978-3-540-31984-9_3
6. Raedts, I., Petkovic, M., Usenko, Y.S., van der Werf, J.M., Groote, J.F., Somers, L.: Transformation of BPMN models for behaviour analysis. In: MSVVEIS 2007, pp. 126–137 (2007). http://jmw.vdwerf.eu/_media/public/transformationforbehaviouranalysis.pdf
7. Dijkman, R.M., Dumas, M., Ouyang, C.: Semantics and analysis of business process models in BPMN. Inf. Softw. Technol. **50**(12), 1281–1294 (2008). doi:10.1016/j.infsof.2008.02.006
8. Wong, P.Y.H., Gibbons, J.: A process semantics for BPMN. In: ICFEM 2008, pp. 355–374 (2008). doi:10.1007/978-3-540-88194-0_22
9. Wong, P.Y.H., Gibbons, J.: Verifying business process compatibility. In: QSIC 2008, pp. 126–131 (2008). doi:10.1109/QSIC.2008.6
10. Decker, G., Weske, M.: Interaction-centric modeling of process choreographies. Inf. Syst. **36**(2), 292–312 (2011). doi:10.1016/j.is.2010.06.005
11. Decker, G., Weske, M.: Local enforceability in interaction petri nets. In: BPM 2007, vol. 4714. LNCS, pp. 305–319 (2007). doi:10.1007/978-3-540-75183-0_22
12. Güdemann, M., Poizat, P., Salaün, G., Dumont, A.: VerChor: a framework for verifying choreographies. In: FASE 2013, pp. 226–230 (2013). doi:10.1007/978-3-642-37057-1_16
13. Mateescu, R., Salaün, G., Ye, L.: Quantifying the parallelism in BPMN processes using model checking. In: CBSE 2014, pp. 159–168 (2014). doi:10.1145/2602458.2602473
14. Kossak, F., Illibauer, C., Geist, V., Kubovy, J., Natschläger, C., Ziebermayr, T., Kopetzky, T., Freudenthaler, B., Schewe, K.-D.: A rigorous semantics for BPMN 2.0 process diagrams (2014). doi:10.1007/978-3-319-09931-6_4
15. Bultan, T., Fu, X.: Specification of realizable service conversations using collaboration diagrams. Serv. Oriented Comput. Appl. **2**(1), 27–39 (2008). doi:10.1109/SOCA.2007.41
16. Salaün, G., Bultan, T.: Realizability of choreographies using process algebra encodings. In: IFM 2009, vol. 5423. LNCS, pp. 167–182 (2009). doi:10.1007/978-3-642-00255-7_12
17. VBPMN Framework. https://pascalpoizat.github.io/vbpmn/
18. Alur, R., Etessami, K., Yannakakis, M.: Realizability and verification of MSC graphs. Theor. Comput. Sci. **331**(1), 97–114 (2005). doi:10.1016/j.tcs.2004.09.034
19. ISO. LOTOS – A Formal Description Technique Based on the Temporal Ordering of Observational Behaviour. Technical report 8807, ISO (1989)
20. Lohmann, N., Wolf, K.: Realizability is controllability. In: WS-FM 2009, vol. 6194. LNCS, pp. 110–127 (2010). doi:10.1007/978-3-642-14458-5_7
21. Poizat, P., Salaün, G.: Checking the realizability of BPMN 2.0 choreographies. In: SAC 2012, pp. 1927–1934 (2012). doi:10.1145/2245276.2232095
22. Fu, K.: Structural Methods of Pattern Recognition, p. 319. Mir Moscow(1977)
23. Costagliola, G., Lucia, A.D., Orece, S., Tortora, G.: A parsing methodology for the implementation of visual systems. http://www.dmi.unisa.it/people/costagliola/www/home/papers/method.ps.gz
24. Wittenburg, K., Weitzman, L.: Relational grammars: theory and practice in a visual language interface for process modeling (1996). http://citeseer.ist.psu.edu/wittenburg96relational.html
25. Zhang, D.Q., Zhang, K.: Reserved graph grammar: a specification tool for diagrammatic VPLs. In: IEEE Symposium on Visual Languages, Proceedings. IEEE, pp. 284–291 (1997)
26. Zhang, K.B., Zhang, K., Orgun, M.A.: Using graph grammar to implement GlobalLayout for a visual programming language generation system (2002)

27. Afanasyev, A., Voit, N.: Intelligent agent system to analysis manufacturing process models. In: Proceedings of the First International Scientific Conference "Intelligent Information Technologies for Industry" (IITI 2016), vol. 451. Advances in Intelligent Systems and Computing, AISC, pp. 395–403 (2016). doi:10.1007/978-3-319-33816-3_39
28. Afanasyev, A., Voit, N., Gaynullin, R.: The analysis of diagrammatic models of workflows in design of the complex automated systems. In: Proceedings of the First International Scientific Conference "Intelligent Information Technologies for Industry" (IITI 2016), vol. 450. Advances in Intelligent Systems and Computing, AISC, pp. 227–236 (2016). doi:10.1007/978-3-319-33609-1_20
29. Afanasyev, A.N., Voit, N.N., Gainullin, R.F.: Diagrammatic models processing in designing the complex automated systems. In: 10th IEEE International Conference on Application of Information and Communication Technologies (AICT), pp. 441–445 (2016)
30. Afanasyev, A., Voit, N.: Multi-agent system to analyse manufacturing process models. In: Proceedings of International Conference on Fuzzy Logic and Intelligent Technologies in Nuclear Science, pp. 444–449 (2016). doi:10.1142/9789813146976_0072