Semantic features of processing hybrid dynamic workflows of design

A N Afanasyev[1], N N Voit[2] and S Yu Kirillov[3]

[1] Doctor of Engineering, First vice-rector, vice-rector of distance and further education, Ulyanovsk State Technical University, 32, Severny Venets, Ulyanovsk, 432027, Russia
[2] Ph. D., associate professor of «Computer engineering», Ulyanovsk State Technical University, 32, Severny Venets, Ulyanovsk, 432027, Russia
[3] PhD.student of «Computer engineering», Ulyanovsk State Technical University, 32, Severny Venets, Ulyanovsk, 432027, Russia

Abstract. The article examines the semantic peculiarities of grammatical processing models of visual languages like RC ASCON-Volga, BPMN and eEPC as denotative and significative semantics. It is offered author's structure of the denotation and significata. Methods of control, analysis of qualitative and quantitative characteristics of workflows, transformation and interpretation of workflows are proposed by authors. A temporary automatic grammar of RC ASCON-Volga, BPMN and UML AD visual languages is proposed by authors for semantic processing of hybrid dynamic workflows, as well as for analysis and control of denotative and significative semantic errors.

1. Introduction

The paradigm associated with the hybrid dynamic nature of their nature is increasingly dominating in the creation of project workflows. Hybridity is defined not only as the development of models using different diagram bases (for example, UML AD [1], BPMN [2], IDEF0 [3]), but also as the composition of orchestration and choreography [4, 5] in the form of an ensemble. Dynamism is determined by the need for immediate response to emerging production requests and contains the concept of «time», so the elimination of errors in the flow of work is a significant scientific and technical problem. Under treatment refers to the analysis, control, transformation, and interpretation workflows, and analysis and control diagrammatically models of the error associated with denotative and significative the semantics. Denotative semantics of diagram models is represented by a sequence of temporal words of the formal automaton language in the form of traces and defines antonymy, synonymy of these words in order to identify errors in the events of diagram models. The significative semantics of diagram models reveals the relations of isomorphism, homomorphism of these traces for the purpose of localization of structural errors in diagram models, and also for their subsequent transformation. Changing the design of the diagram models is possible with the identified synonyms, antonyms of graphic words, isomorphic traces and the possibility of combining such traces into a single track. A necessary condition for the transformation of the flow of design work is the presence of a semantic error. Detection of such errors is possible with the help of step-by-step automated interpretation (tracing) of the project work flow in debug mode. Automata-based grammars allow you to present a diagram model of the flow of design work in the form of a graph with vertices, arcs and in a visual form to represent the process of interpreting the flow of design work as a system of transitions. The methods of analysis can be used to study the qualitative and quantitative characteristics of project work flows. Under qualitative characteristics refers to the logical-algebraic correctness of workflows, formalized using graph theory, networks, workflows, matrix matching, graphical modeling languages, including Unified Model Language, Business Process Management Notation, IDEF0 and eEPC, etc., and also the evolutionary approach, logic statements, etc. Quantitative characteristics represent the effectiveness of the execution of the workflow in the parameters, such as average service time, the utilization rate of production capacity (downtime), etc. The efficiency of workflows is evaluated using simulation modeling (Petri nets), Markov chains and Queuing theory (Queuing systems).

In this paper, the authors developed a temporary automatic grammar of visual languages of RC ASKON-Volga, BPMN and eEPC, as well as denotative and significative semantics of diagram models of visual languages as a general structure for semantic processing of diagram models of hybrid dynamic design workflows. Processing will reduce the time, improve the success and quality of charting models. The mathematical apparatus of processing of diagram models of hybrid dynamic flows of design work allows to simulate the process of workflows in the form of a finite state machine. The work contains an introduction, the author's description of denotative and significative semantics of workflows in the visual languages of RC ASCON-Volga, BPMN and eEPC. The review of qualitative and quantitative methods for estimating the characteristics of workflows presents mathematical tools for the analysis, control and modeling of workflows. In the section of the Temporal automaton RVTI-grammar author presents automata-based temporal grammar for visual language RC ASKON-Volga, BPMN and UML AD. In the section Transformation the author's method of transformation of diagrammatic models of workflows by means of the developed author's temporal automatic grammar is offered. In conclusion, briefly concludes the work and identifies future directions of work.

2. Related work

The authors investigate some works that consider the specification of document flow, verification and translation. Several papers focused on the definition of formal semantics and validation methods for workflows using Petri nets, process algebra, abstract state machine, see for example [6-16]. In [12, 13], Decker and Weske propose a Petri net-based formalism for determining choreographies, properties as realizability and local applicability, and a method for verifying these two properties. However, they consider only synchronous communication and does not explore the association with languages modeling of interaction of a high-level BPMN. Bultan and Fu [17] determine a sufficient condition for analyz-ing the feasibility of choreographies defined using UML collaboration diagrams (CD). In [18], Salaün and Bultan modify and extend this work with the feasibility analysis method by adding a synchronization message among peers. This method controls the realizability of CDs for bounded asynchronous communication. The feasibility problem for Message sequence diagrams (MSCs) has also been studied (e.g. [19, 20]). In [20], the authors offer bounded MSCS graphs which are bound-ed by BPMN 2.0 because branching and looping behavior are not supported by CDs and MSCs (there is no selection in CDs, there are no some looping behaviors in MSCs, and only Self-loops in CDs). In [21] BPMN behavior is studied from the semantic point of view and several BPMN patterns are proposed. This work is not theoretically justified and is not complete, it discusses only some of the laws. Lohmann and Wolf [22] propose to analyze existing patterns and control them with compatible patterns. In [23], the authors focused on the translation of BPMN into the algebra of processes for the analysis of choreography using model checking and equivalence. The main limitation of these methods is that they do not work when there are different types of diagrams at the same time, which means that in some cases the input diagrams cannot be analyzed. There are the following paradigms of analysis and control of quality characteristics of work flows: model checking; equivalence check; deductive verification (Prolog language). The model checking approach is intended for the analysis, control of workflows by means of formal check of whether the given logical formula is executed on the given structure (whether the given logical formula *f* will be true for the given system of transitions *M*, i.e. whether *M* will be model *f*). The main disadvantage of the campaign is the study of the model, not the system itself, so the question arises about the adequacy of the model to the system, and the complexity of the solution of the above problems is exponential. Deductive verification involves checking the correctness of the workflow, which is reduced to proving theorems in a suitable logical system with the help of axioms and inference rules (for example, with the help of the Prolog language, automatic grammars, etc.). This very complex procedure can not be fully automated, it requires the participation of a person acting on the basis of assumptions and guesses, using intuition in the construction of invariants and non-trivial choice of alternatives. The equivalence test determines the equivalence of formal models of specification, implementation and execution (behavior) of workflows based on the algebra of processes (Calculus of interacting systems). Simulation modeling is a flexible approach to analysis and is applied almost always to the analysis of workflows, which is reduced to determining the path in the reachability graph, taking into account the probability of distribution. Multiple execution of workflows using a computer provides «ease» of understanding of the functional people who do not have mathematical training. Visual representation and analysis of workflows is available in many tools for modeling workflows. Usually use Petri nets in the modeling and analyzed the following properties: reachability (reachability) – which sets out that the final state of the system is reached when any sequence of transitions from positions i. This property also implies that upon reaching the final position of the network there are no chips in the intermediate positions; security (safety), establishes that the processes do not exist hangs (deadlocks), looping dead ends; vitality (liveness) – specifies that the system does not contain unnecessary items that will never be fulfilled. The lack of liveliness means either redundancy of the business process in the designed system, or indicates the possibility of loops, deadlocks, locks. Flow rate, transmission rate, waiting time, service time, and capacity utilization can be calculated using queue theory. If we are interested in the formation of a separate queue to multiple resources of the same type, it is necessary to confine the system with a single queue. When considering the entire flow, Queuing systems are best used. The main models used in Queuing theory are single-and multi-channel Queuing systems (QSOS). The most simple model of the workflow to determine complexity of the corresponding phase can be obtained if we accept the assumption about the absence of consequences in the process, meaning that the next job in the flow depends only on the current state and does not depend on previous States. In this case, the flow of work becomes a Markov process determined by a variety of inherent conditions and the matrix of transition probabilities, and a probability distribution of states in the initial moment of time.

3. Denotative and significative semantics of diagram models of visual languages of RC ASCON-Volga, BPMN and eEPC

Denotative semantics [24-26] of any visual language is represented by denotates in the form of graphical objects (words). The denotate of a word in the theory of visual languages is understood as an instance of a class with specific values of properties characterizing the belonging of a word to a subject. The authors have developed a general structure of the denotation and significata graphic words. The general structure of the graphical class instance of a word is displayed in listing 1.

Listing 1. Generalized structure of the graphic denotation of the word.

```
The class name class=start
  beginning
    property 1=value 1
    property 2=value 2
    property n=value n
  end
```

The structural units of the parameters (properties) of the graphical word represent the signature of the word, and the properties themselves are combined into a class (listing 2).

Listing 2. Generalized structure of significata graphic words.

```
Class name: text
  beginning
    property 1: type 1
    property 2: type 2
    property n: type n
  end
```

The description of denotates and significations in the set-theoretic language has the following form.

$$NameOfDenotate = (ValueOfProperty\_1, ValueOfProperty\_2, ValueOfProperty\_3, \dots, ValueOfProperty\_N), \quad (1)$$

where $NameOfDenotate$ is the name of the denotation (class instance) the graphic word; $ValueOfProperty\_1$ is the value of the first property of the graphic word; $ValueOfProperty\_2$ is the value of the second property of the graphic word; $ValueOfProperty\_3$ is the value of the third property of the graphic word; $ValueOfProperty\_N$ is the value of n-th property of a graphical word.

$$Significatum = (Property1, Property2, Property3, \ldots, PropertyN), \qquad (2)$$

where $Significatum$ is the name of significata (class) graphic words; $Property1$ is structure of the first property of the graphic word; $Property2$ is structure of the second property of the graphic word; $Property3$ is structure of the third property of the graphic word; $PropertyN$ is structure of n-th property of a graphical word.

The difference between the significate and the denotate is that the denotate is an instance of the significate, i.e. the properties (structure) of the denotate have specific values. Further in tables 1, 2, 3 denotative and significative semantics of grammatical models of hybrid dynamic streams of design works of visual languages of RC ASKON Volga, BPMN, eEPC are presented [27-35].

Table 1. Denotative and significative semantics of the visual language of RC ASCON-Volga.

Graphic word	Concept	Class	Class properties	The values of class properties

⬜	A collapsed subprocess that can be called multiple times. Has only one inbound link of type «go to procedure»	Procedure	Input
			Algorithm
			Output
				Input=work thread
				Algorithm=ishodnyh
				Output=Potocari

🟨	Collapsed subprocess, the action is required to be performed by the user	Task	Input
			Text of task
			Output
				Input=work thread
				Textpane=scriptside
				Output=work thread

🟩	Collapsed subprocess, the implementation of which is required repeatedly	Iteration	Algorithm	Algorithm=source code
Call	Used in conjunction with «go to procedure» and «procedure» block	Procedure call	Adres call procedure	Adres call procedure=number

Thread	Used in conjunction with «go to procedure» and «procedure» block	Thread creation	Name
			Adresie
				Name=stream_name
				Adresie=number

E	The operation performed by the user	No transit	opcode operand address 1 operand address 2	opcode=number operand address 1=number operand address 2=number
A	Automated (automatic) execution of operations	The script (auto operation)	Algorithm	Algorithm=source code
IF	It has only two outgoing connections. True and False, respectively	Branching	Condition	Condition=source code

🔵	Allows you to connect parts of the chart at different levels of nesting. In fact a connection	Phantom	From level
			Against the level
				Isorena=numerowana
				Karouny=numerowana

Ev	Used in conjunction with «waiting». Three incoming, one of them «Synchro action», notifying about the event.	Event	Time	Time=number

Se	Used in conjunction with «waiting». Three incoming, one of them «Synchro action», notifying about the beginning and completion of events.	Semaphore	Beginning
			Completion
				Start=number
				Completion=number

F	Has two outgoing branches, one of them «Synchro action», notifying «Event» on successful completion of the event	Activate	Output	Output=Boolean
W	It has two outgoing branches, one of them «Synchro action», which monitors the status of the event. Skips the stream only if the event succeeds	Expectation	Duration	Duration=number

Inc	It has two outgoing branches, one of them «Synchro action», notifying «Semaphore» about the beginning of the event execution	Increment	With
			On
			Step
				C=the number
				By=number
				Step=number

Dec	It has two outgoing branches, one of them «Synchro action», notifying «Semaphore» about the completion of the event	Decrement	From
			To
			Step
				From=number
				Up=number
				Step=number

Table 2. Significative and denotative semantics of the visual language of BPMN.

Graphic word	Concept	Class	Class properties	The values of class properties
○	Has no incoming threads	Starting event	Output	Output=work thread

◎	Only one incoming and outgoing stream	Intermediate event	Input
			Operation
			Output
				Input=work thread
				Operation=algorithm
				Output=work thread

✉	Can be a trigger	Intermediate event «Message»	Input
			Event
			Output
				Input=work thread
				Event=algorithm
				Output=work thread

●	Has no outgoing streams	Final event	Input	Input=work thread

▭	Incoming stream	Action	Input
			Procedure
			Output
				Input=work thread
				Procedure=algorithm
				Output=work thread

◇X	Only one outgoing branch is allowed to flow	Exclusive gateway	Input
			Condition
			Output
				Input=work thread
				Condition=algorithm
				Output=work thread

◇⬠	It can be only one trigger event	Gateway the Event-based	Input
			Event
			Output
				Input=work thread
				Event=algorithm
				Output=work thread

◇+	Activated only if there is a thread on each incoming branch	Parallel gateway	Input 1 Input 2 Input 3 Input s Output	Input 1=work stream Input 2=work stream Input 3=work stream Input=work thread Output=work thread
🗋	Can connect to any flow element by association	Data object	Document	Document=text

⟶	The relationship between the graphical objects	A regular relationship	From
			To
				From=graphical object
				To=graphical object

⇢	Allows you to monitor an invalid sequence of elements	Event-based gateway communication	From
			To
				From=graphical object
				To=graphical object

┈	Associative relationship between graphical objects by key	Association	From
			To
				From=graphical object
				To=graphical object

Table 3. Significative and denotative semantics of the visual language eEPC.

Graphic word	Concept	Class	Class properties	The values of class properties
🟥	It is the fact of accomplishment of something, and not having duration in time, or this time to aspire to zero (or does not matter)	Event	Time	Number
🟩	Execution of a function always ends with the event	Function	Function	The function=algorithm

▭	Workflow direction	Through a process	From
			To
				From=work thread
				To=work thread

🟡	The object reflects the various organizational units of the enterprise	Unit	Name	Name=text

XOR	A logical operator that defines the relationship between events and functions within a process. Allows you to describe the branching process	exclusive or	From
			To
				From=work thread
				To=work thread

∨	A logical operator that defines the relationship between events and functions within a process. Allows you to describe the branching process	OR	From
			To
				From=work thread
				To=work thread

∧	A logical operator that defines the relationship between events and functions within a process. Allows you to describe the branching process	AND	From
			To
				From=work thread
				To=work thread

🟦	The object reflects the media	Information (Material)	Document	Document=text

🟨	Core business process	Main process	From
			To
				From=graphical object
				To=graphical object

🟨	Structural and functional element	Component	Name	Name=text
🟡	Area of research	Subject area	Name	Name=text
⟩	A set of simple processes, collected on the same basis and solve the same problem	Process group	Name	Name=text

└	The relationship between the graphical objects	Dynamic connection line	From
			To
				From=graphical object
				To= graphical object

Semantic errors of grammatical models of work flows include the following errors [36]. *Synonym mismatch (denotative error).* Temporal words of visual language $(\tilde{a}_l, t_i)$ and $(\tilde{a}_k, t_i)$ are synonyms if and only if $\tilde{a}_l \neq \tilde{a}_k, \tilde{a}_l \equiv \tilde{a}_k, t_i < t_j$ and indicated synonymy of the words as$(\tilde{a}_l, t_j) \equiv (\tilde{a}_k, t_j)$. The identical equality of the word determines the similarity (similarity) of the structure and values of denotate features. A mistake is the situation when the name of the denotates of words in two temporal traces of the graphical language are similar, but the values of other features are very different. In practice, this situation is presented as follows: the analysis of the grammatical model of the visual language reveals the structural similarity of words and names of denotates, but the values of other features of denotates of words are different. To present variants of the composition of products under different conditions: in versions, substitutability and interchangeability, in this situation, the implementation of interchangeability of such words in the diagram model of the visual language is a mistake of non-conformity of synonyms. *Discrepancy of antonyms* (denotative error). Temporal words of visual language $(\tilde{a}_l, t_j)$ and $(\tilde{a}_k, t_j)$ are antonyms if and only if $\tilde{a}_l = \neg\tilde{a}_k, t_i < t_j$ and is denoted by antonyms to words as $(\tilde{a}_l, t_l) \equiv (\neg\tilde{a}_k, t_k)$. Identical to the opposite of the two words defines the similarity (similarity) of the structure and the opposite (inversely) characteristic value of the denotation. Generally, the words «Beginning» and «End» in charting are antonyms of the graphical language. A mistake is the project situation when the name of denotates of words in two temporal traces of graphic language are opposite (inverse), but the values of other signs are very similar. In practice, this situation is presented as follows: when analyzing the diagram model of the visual language, structural similarity of words and inversion of denotate names are revealed, but the values of the other signs of denotate words are similar. In this situation, the interchangeability of such words in the diagram model of the visual language is a mistake of discrepancy of antonyms. Conversionist relations is to bind antonyms diagrammatically models of visual languages that describe the same design situation, but with different roles. *Error conversionist relations* is significative, i.e. structural (structural), and is defined as the lack of these relations between antonyms diagrammatically models describing the same design situation, but with different roles. *The inconsistency of the objects* is significatively mistake. Is the absence of a relationship between dependent temporal words.

4. Temporal automata-based RVTI grammar

The temporal automaton RVTI grammar of the language L (G) is called the ordered eight of non-empty sets $G = (V, \Sigma, \tilde{\Sigma}, C, E, R, \tau, r_0)$, where $V = \left\{v_e,\ e = \overline{1.L}\right\}$ is auxiliary alphabet (the alphabet of operations on internal memory, represented by the store or elastic band); $\Sigma = \left\{a_n,\ n = \overline{1.T}\right\}$ is alphabet of graphic symbols (objects); $\tilde{\Sigma} = \left\{\widetilde{a_n},\ n = \overline{1.\tilde{T}}\right\}$ is quasi-terminal alphabet, which is an extension of the terminal alphabet $\sum$; $C = \{c, c = c + t_l | \exists t_0 = 0 \to c = 0\}$ is clock ID (counter); $\tau = \left\{t_l \in [0\,; +\infty], l = \overline{1.K}\right\}$ is a lot of timestamps, and $c \in [t_l\,; t_{l+1}]$;E is temporal aspect ratio $\{c \sim t_l\}$, where variable c (ID hours), the ratio $\sim \in \left\{=, \langle, \leq, \rangle, \geq\right\}$), describing the condition of the occurrence of an event t_l; $R = \left\{r_i,\ i = \overline{0, I}\right\}$ is scheme of grammar G (the set of names of products of complexes, each complex r_i consists of a subset P_{ij} products' $r_i = \left\{P_{ij},\ j = \overline{1.J}\right\}$); $r_0 \in R$ is the axiom of RVTI-grammar (name of the initial complex of products), $r_k \in R$ is the final set of products. Product $P_{ij} \in r_i$ have the form $\tilde{a}_l \xrightarrow{\{W_\gamma(v_1,\ \ldots, v_n) | E\}} r_m$, where $W_\gamma(v_1,\ \ldots,\ v_n)$ is n-ary ratio, which determines the type of operation on the internal memory, depending on $\gamma = \{1,2,3\}$ (1 – write, 2 – read, 3 – a comparison), $E = \emptyset$ and $c \neg\sim t_l$; $(\tilde{a}_l, t_l)$ is words as a pair of quasi-symbol and timestamp; $r_m \in R$ is name of the successor product complex. The language of this grammar contains words of the form $(\tilde{a}_l, t_l)$ and $E \neq \emptyset$ and $\tilde{a}_l$ and $E = \emptyset$, represents the alignment $\sigma = \left\{\tilde{a}_0, 0\right\} \to \left\{\tilde{a}_l, t_l\right\} \vee \tilde{a}_l \to \left\{\tilde{a}_k, t_K\right\}$. In tables 4 and 5 present the temporal grammars for specific language RC ASKON Volga and language BPMN.

Table 4. Temporal RVTI grammar language RC ASKON Volga.

Complex-source	Quasi term	The complex receiver	The memory operation
r_0,t_0	A_0	$r_3,t3$	$\varnothing/E$
r_1,t_1	$return$	r_2,t_4	$w_2(b^{4m})$
r_2,t_2	vA	r_1,t_1	$w_1(s^{1m},t^{4m})$, $CALL\ vA/E$
	vIT	r_1,t_1	$w_1(s^{1m},t^{4m})$, $CALL\ vIT/E$
	Ak	r_4,t_4	$\varnothing$
	Akm	r_5,t_5	$w_1(1^{t(1)},i^{t(2)})/w_2(e^{t(1)})/E$
	$\_Akm$	r_5,t_5	$w_1(inc(m^{t(1)}))/w_3(m^{t(1)}<k^{t(2)}-1),E$
	$Akme$	r_4,t_4	$w_1(inc(m^{t(1)}))/w_3(m^{t(1)}=k^{t(2)}-1),E$
	CL	r_6,t_6	$w_1(t^{4m})$
	TH	r_6,t_6	$w_1(1^{t(7)}, i^{t(8)}, t^{4m})$
	SC	r_3,t_3	$\varnothing$
	SCm	r_5,t_5	$w_1(1^{t(3)},i^{t(4)})/w_2(e^{t(3)}),E$
	$\_SCm$	r_5,t_5	$w_1(inc(m^{t(3)}))/w_3(m^{t(3)}<k^{t(4)}-1),E$
	$SCme$	r_3,t_3	$w_1(inc(m^{t(3)}))/w_3(m^{t(3)}=k^{t(4)}-1),E$
	C	r_7,t_7	$w_1(t^{2m})$
	EV	r_3,t_3	$w_1(0^{t(5)}, 0^{t(9)},0^{t(11)})/w_2(e^{t(5)}),E$
	S	r_3,t_3	$w_1(0^{t(6)}, 0^{t(10)},0^{t(12)})/w_2(e^{t(6)}),E$
	F	r_{11},t_{11}	$w_1(t^{3m})$
	W	r_9,t_9	$w_1(t^{3m})$
	IN	r_{11},t_{11}	$w_1(t^{3m})$
	D	r_{12},t_{12}	$w_1(t^{3m})$
r_3,t_3	rel	r_2,t_2	$\varnothing$
r_4,t_4	no_label	r_{17},t_{17}	*
r_5,t_5	$labelC$	r_2,t_2	$w_2(b^{2m})$
r_6,t_6	$prel$	r_{13},t_{13}	$\varnothing$
r_7,t_7	$nrel$	r_2,t_2	$\varnothing$
r_8,t_8	$PHsp$	r_6,t_6	$\varnothing$
r_9,t_9	$arel$	r_{14},t_{14}	$\varnothing$
r_{10},t_{10}	$PHsa$	r_9,t_9	$\varnothing$
r_{11},t_{11}	$airel$	r_{15},t_{15}	$\varnothing$
r_{12},t_{12}	$adrel$	r_{16},t_{16}	$\varnothing$
r_{13},t_{13}	vPR	r_1,t_1	$w_1(s^{1m})$, $CALL(vPR)/E$
	$PHep$	$r8,t8$	$\varnothing$
r_{14},t_{14}	THa	r_2,t_2	$w_1(inc(m^{t(7)}))/w_3(m^{t(7)}<k^{t(8)}),E$
	$PHea$	r_{10},t_{10}	$\varnothing$
	EVa	r_2,t_2	$w_1(1^{t(9)}), w_2(b^{3m})$
	Sa	r_2,t_2	$w_1(1^{t(10)}), w_2(b^{3m})$
r_{15},t_{15}	EVa	r_2,t_2	$w_1(inc(m^{t(5)}), 1^{t(11)}), w_2(b^{3m})$
	Sa	r_2,t_2	$w_1(inc(m^{t(6)}), 1^{t(12)}), w_2(b^{3m})$
r_{16},t_{16}	Sa	r_2,t_2	$w_1(dec(m^{t(6)}), 1^{t(12)}), w_2(b^{3m})$

Table 5. RVTI grammar for BPMN.

Complex-source	Quasi term	The complex receiver	The memory operation
r_0	A_0	r_1	Ø
r_1	rel	r_3	Ø
r_2	$label_{EG}$	r_3	$W_2(b^{1m}, b^{t(6)})$
	$label_{PG}$	r_3	$W_2(b^{2m}, b^{t(6)})$
r_3	A_i	r_1	Ø
	A_{im}	r_1	Ø
	A_{it}	r_1	$W_1(t_s^{t(6)})$
	A_{kl}	r_2	Ø $/W_3(!e^{1m}, !e^{2m})$

Complex-source	Quasi term	The complex receiver	The memory operation
	A_k	r_4	Ø
	A	r_1	$W_1(t_s^{t(6)})$
	A_{it}	r_3	$W_1(t_s^{t(6)})$
	EG_c	r_1	$W_1(t^{1m^{(n-1)}})/W_3(k = 1)$
	EG	r_2	$W_1(1^{t(1)}, k^{t(2)})/W_3(e^{t(2)}, k != 1)$
	$\_EG$	r_2	$W_1(inc(m^{t(1)})/W_3(m^{t(1)} < k^{t(2)})$
	$\_EGe$	r_1	$W_1(t^{1m^{(n-1)}})/W_3(m^{t(1)}=k^{t(2)}, p != 1)$
	$\_EG_{me}$	r_1	$o/W_3(m^{t(1)}=k^{t(2)}, p = 1)$
	PG_f	r_1	$W_1(t^{2m^{(n-1)}})/W_3(k = 1)$
	PG	r_2	$W_1(1^{t(3)}, k^{t(4)})/W_3(e^{t(3)}, k != 1)$
	$\_PG$	r_2	$W_1(inc(m^{t(3)})/W_3(m^{t(3)} < k^{t(4)})$
	$\_PG_e$	r_1	$W_1(t^{2m^{(n-1)}})/W_3(m^{t(3)}=k^{t(4)}, p != 1)$
	$\_PG_{je}$	r_1	$W_1(t^{2m^{(n-1)}})/W_3(m^{t(3)}=k^{t(4)}, p = 1)$
r_4	no_label	r_5	*
r_5			

An instance of the diagram model (table 4, table 5) can be used to construct an ontology [27, 30], the classes of which are words (concepts) and have the following form: $Class_0 = (\tilde{a}_k, t_0), \ \ldots, Class_k = (\tilde{a}_k, t_k)$, where a couple $(\tilde{a}_i, t_i)$ is a temporal word. Classes have properties, which, for example, are represented as follows: $Property = \{Name, OrientationTime, ProcessingTime\}$, where $Name$ is the name of the field (inherited from the name of the notation diagrammatically model); $OrientationTime$ is the start time of the stream; $ProcessingTime$ is the duration of the thread.

5. Transformation

Dynamic reconfiguration of business process need to have a mechanism for transformation of diagrams reaching flexibility, improving a functional and an efficiency of enterprise's business process. In work [19-21] the problem of reconfiguration has been researched both theoretical and practical. Authors offer applying the structure transformation of a diagram with help procedures: delete, insert and replace with saving a connection during an interval of time. It is necessary all graphic element have a timed label where we can define time of the transformation. As rule BPMN, eEPC, IDEF0, UML AD etc. graphic elements contain a description (notes in UML AD) which can be define as a timed variable. Let's see an example of UML AD diagram (figure 1).

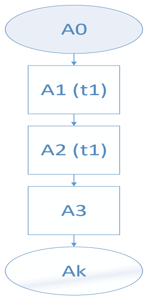

Figure 1. UML AD diagram with t1 timed label.

Graphic elements A1 and A2 have t1 timed labels. This means that a current element will be transform at t1 time with help operations: (1) Insert, (2) Replace, (3) Delete. Reasoning to suppose that only one operation cab be performed at one element. Therefore, timed label is assigned to a tape where an element has their variants: number 1 – Insert, number 2 – Replace, number 3 – delete. Additional information when Insert and Replace saved at extended tape allowing to save both numbers and quasi-terms. Additional Insert() function is used for the operation 1 allowing to get needed information from extended tape and form inserted fragment. Operation 2 is a complex operation that represents an aggregate of removing and inserting operations. Replace() additional function is brought for ease. Deleting is considered in a start. The diagram has a form in t1 time (figure 2).

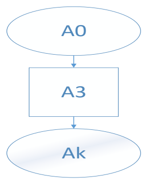

Figure 2. Deleting elements at diagram.

The chain including deleting element can be infinite size. Authors suggest the approach to perform deleting. If we meet element with timed label, then timed label is put in a stack. Next step an automaton follows about elements while not getting element with absent timed label. In this case it perform change_rel() special function that pop up from the stack timed label at deleting element and assign its with a current element. This algorithm is shown in figure 3. In order not to leave deleting elements suspended in a diagram when to pass deleting quasi-term delete() function perform that delete elements from the diagram. delete_with_link() function performs deleting elements with an enter link.

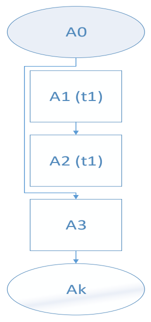

Figure 3. Assignment of links where deleting an element.

The grammar for that diagram is shown in table 5.

Table 5. Timed RVTI-grammar for UML AD.

Prev. state	**Quazi-term**	**Next state**	**Operation**
r_0	*A0i*	r_1	*insert()/*$W_3(k^{t(1)}==1)$
	A0	r_1	*o*
r_1	*rel*	r_2	*o*
r_2	*Ai*	r_1	*insert()/*$W_3(k^{t(1)}==1)$
	Ar	r_1	*replace()/*$W_3(k^{t(1)}==2)$
	Ad	r_3	*(delete(),* $W_1(l^{1m})$*)/* $W_3(k^{t(1)}==3)$
	A	r_1	*o*
	Ak	r_5	*o*
r_3	*drel*	r_4	*o*
r_4	*Ai*	r_1	*(change_rel(),insert())/*$W_3(k^{t(1)}==1)$
	Ar	r_1	*(change_rel(), replace())/*$W_3(k^{t(1)}==2)$
	Ad	r_3	*delete_with_link()/*$W_3(k^{t(1)}==3)$
	A	r_1	*change_rel()*
	Ak	r_5	*change_rel()*
r_5	*no_label*	r_k	*

6. Conclusion

The semantic features of hybrid dynamic design processes are analyzed in terms of their denotative and significative representations using the visual languages RC ASCON-Volga, BPMN and eEPC. The mathematical description of denotations and significative diagrammatically models of visual languages is given by authors, as well as the table description of denotations and significative diagrammatically models of visual languages RC ASKON-Volga, BPMN and eEPC. The authors have expanded the list of semantic errors that occur in the workflow, four types of errors such as Synonym mismatch (denotative error), Discrepancy of antonyms (denotative error), Error conversionist relations (significative error), The inconsistency of the objects (significative error). Paradigms of the analysis and control of qualitative and quantitative characteristics of workflows are investigated. The author has developed automata-based temporal grammar for visual language RC ASKON-Volga, BPMN and UML AD, analysing and controlling these structural and semantic errors. The method of workflow transformation on the example of visual language UML AD is presented. In future works it is supposed to carry out researches of dynamic model of representation of processes of the automated systems on the basis of the temporal automatic grammar providing the mathematical description of hybrid dynamic design workflows for the analysis, control, transformation and interpretation that will allow to define the place of an error in diagram model.

7. References

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

[2] Model B P 2011 Notation (BPMN), v. 2.0 *OMG* www.omg.org/spec/BPMN/2.0

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

[4] Samuilov K E, Serebrennikova N V, Chukarin A V, Yarkina N V 2008 Osnovy formal'nyh metodov opisaniya biznes-processov *Ucheb. posobie. M.: RUDN* 130 s (in Russian)

[5] Bock C 2008 Introduction to business process and definition metamodel *U.S. National Institute of Standard and Technology. Manufacturing Engineering* https://www.nist.gov

[6] Poizat P, Salaün G, Krishna A 2016 Checking business process evolution *In International Workshop on Formal Aspects of Component Software Springer Cham* pp 36-53 https://hal.inria.fr/hal-01366641

[7] Martens A 2005 Analyzing web service based business processes *In International Conference on Fundamental Approaches to Software Engineering Springer Berlin Heidelberg* pp 19-33 doi: 10.1007/978-3-540-31984-9_3

[8] Raedts I, Petkovic M, Usenko Y S, van der Werf J M E, Groote J F, Somers L J 2007 Transformation of BPMN Models for Behaviour Analysis *MSVVEIS* pp 126-137.

[9] Dijkman R M, Dumas M, & Ouyang C 2008 Semantics and analysis of business process models in BPMN *Information and Software technology* 50(12) pp 1281-1294 doi: 10.1016/j.infsof.2008.02.006

[10] Wong P Y, & Gibbons J 2008 A process semantics for BPMN *In International Conference on Formal Engineering Methods Springer Berlin Heidelberg* pp 355-374 doi: 10.1007/978-3-540-88194-0_22

[11] Wong P Y, & Gibbons J 2008 Verifying business process compatibility (short paper) *In Quality Software QSIC'08 The Eighth International Conference on* IEEE pp 126-131 doi: 10.1109/QSIC.2008.6

[12] Decker G, & Weske M 2011 Interaction-centric modeling of process choreographies *Information Systems* 36(2) 292-312 doi: 10.1016/j.is.2010.06.005

[13] Decker G, & Weske M 2007 Local enforceability in interaction petri nets *In International Conference on Business Process Management Springer Berlin Heidelberg* pp. 305-319 doi: 10.1007/978-3-540-75183-0_22

[14] Güdemann M, Poizat P, Salaün G, & Dumont A Verchor 2013 A framework for verifying choreographies *In International Conference on Fundamental Approaches to Software Engineering Springer Berlin Heidelberg* pp 226-230 doi: 10.1007/978-3-642-37057-1_16

[15] Mateescu R, Salaün G., & Ye L 2014 Quantifying the parallelism in BPMN processes using model checking *In Proceedings of the 17th international ACM Sigsoft symposium on Component-based software engineering* pp 159-168 doi: 10.1145/2602458.2602473

[16] Kossak F, Illibauer C, Geist V, Kubovy J, Natschläger C, Ziebermayr T, Schewe K D A 2014 Rigorous Semantics for BPMN 2.0 Process Diagrams *In A Rigorous Semantics for BPMN 2.0 Process Diagrams* pp 29-152 *Springer Cham* doi: 10.1007/978-3-319-09931-6_4

[17] Bultan T, & Fu X 2008 Specification of realizable service conversations using collaboration diagrams *Service Oriented Computing and Applications* 2(1) pp 27-39 doi: 10.1109/SOCA.2007.41

[18] Salaün G, & Bultan T 2009 Realizability of choreographies using process algebra encodings *In International Conference on Integrated Formal Methods Springer Berlin Heidelberg* pp 167-182

[19] VBPMN Framework https://pascalpoizat.github.io/vbpmn/

[20] Alur R, Etessami K, & Yannakakis M 2005 Realizability and verification of MSC graphs *Theoretical Computer Science: Automata, Languages and Programming* 331(1) 97 doi: 10.1016/j.tcs.2004.09.034

[21] Lotos I S O 1989 A formal description technique based on the temporal ordering of observational behaviour *ISO8807, 1XS989*

[22] Lohmann N, & Wolf K 2009 Realizability is controllability *In International Workshop on Web Services and Formal Methods Springer Berlin Heidelberg* pp 110-127 doi: 10.1007/978-3-642-14458-5_7

[23] Poizat P, & Salaün G 2012 Checking the realizability of BPMN 2.0 choreographies *In Proceedings of the 27th Annual ACM Symposium on Applied Computing* pp 1927-1934 doi: 10.1145/2245276.2232095

[24] Kotcova E E 2002 Leksicheskaya semantika v sistemno-tematicheskom aspekte *Arhangel'sk Pomor. gos. un-t* p 203 (in Russian)

[25] Krongauz M A 2005 Semantika: uchebnik dlya stud. lingv. fak. vyssh. ucheb. zavedenii *2-e izd. ispr. i dop M Izdatel'skii centr «Akademiya»* p 171 (in Russian)

[26] Kobozeva I M 2004 Lingvisticheskaya semantika *URSS* (in Russian)

[27] Klein M 2001 Combining and relating ontologies: an analysis of problems and solutions *IJCAI-2001 Workshop on ontologies and information sharing* pp 53-62

[28] Volkova G A 2013 Sozdanie «ontologii vsego». Problemy klassifikacii i resheniya *Novye informacionnye tehnologii v avtomatizirovannyh sistemah* 16 (in Russian)

[29] Mitrofanova O A, Konstantinova N S 2015 Ontologii kak sistemy hraneniya znanii (in Russian)

[30] Euzenat J et al. 2007 Ontology matching *Heidelberg Springer* 18

[31] Mizoguchi R 2000 Shag v napravlenii inzhenerii ontologii *Novosti iskusstvennogo intellekta* 1-2 S11- 36 (in Russian)

[32] Malinovkskii V P 2005 Ispol'zovanie ontologicheskogo podhoda pri modelirovanii zhiznennogo cikla znanii v sisteme korporativnoi pamyati organizacii *Novosti iskusstvennogo intellekta* 3 S pp 31- 41 (in Russian)

[33] Fischer L Workflow Handbook 2005 *Workflow Management Coalition*

[34] Karpov Yu G 2010 MODEL SHECKING. Verifikaciya parallel'nyh i raspredelennyh programmnyh system *SPb BHV Peterburg* p 560 (in Russian)

[35] Kalyanov G N 2006 Modelirovanie, analiz, reorganizaciya i optimizaciya biznes-processov *Uchebnoe posobie M Finansy i statistika* p 240 http://www.twirpx.com/file/2204790/ (in Russian)

[36] Afanasyev A N, Voit N N, Kirillov S Y 2017 Development of RYT-grammar for analysis and control dynamic workflows *International Conference on Computing Networking and Informatics (ICCNI)* pp 1-4 Lagos doi: 10.1109/ICCNI.2017.8123797

Acknowledgments
The reported study was funded by RFBR according to the research project № 17-07-01417 and Russian Foundation for Basic Research and the government of the region of the Russian Federation, grant № 18-47-730032.