Parallel Business Process Modeling Languages

Georgiy Kalyanov [0000-0003-2429-0703]

V.A. Trapeznikov Institute of Control Sciences of Russian Academy of Sciences
Profsoyuznaya st., 65, 117997 Moscow, Russia
kalyanov@mail.ru

Abstract. The modern state of the theory of business processes is analyzed, its main directions are classified. The problem of parallelism in business processes is analyzed, and classes of parallel business processes are identified and analyzed. Requirements for business process modeling languages are formulated. As the basis of the conceptual model of the parallel business process modeling language, we propose a model of DFD technology that integrates structural models of various purposes based on a DFD diagram, and allows you to reflect the functional, informational, and behavioral aspects of the simulated object.
The article also analyzes the syntax, semantic and pragmatic aspects of the introduced modeling language. For a formal description of the syntax, it is proposed to use the apparatus of mixed grammars, which are a combination of graph and normal grammars. The article describes the grammar that generates the simplest dialect of DFD technology, informally describes the semantic aspects of the language, in particular the semantics of relations between objects of the language.

Keywords: Business process (BP), parallel business process, pipeline, synchronous and asynchronous business processes, conceptual model of the BP modeling language, graph and mixed grammars, syntax and semantics of the BP modeling language.

1 Introduction

The term " business process" (BP) was introduced in the early 90's of the last century (M.Hammer, J.Champy) and became widely used after the publication of a monograph on a new approach to the BP reorganization - business process reengineering (BPR). However, studies of similar objects called organizational processes, activities, operations, etc., experts have begun to do for a long time. In particular, Frank and Lillian Gilbreth in 1921 in their report to the ASME (American Society of Mechanical Engineers) proposed a notation of process maps (flow process chart), which with small modifications is still used today. A significant stage in the development of BP science was the period of the late 60s – early 70s the last century was when the widely used modeling languages as SADT-IDEF0, DFD, CFD, clusters (prototypes of classes in the object-oriented approach), and the names of their authors (Ross, Yodan, DeMarco, Gein, Sarson, Liskov, etc.) are known to any specialist in

Proceedings of the of the XXIII International Conference "Enterprise Engineering and Knowledge Management" (EEKM 2020), Moscow, Russia, December 8-9, 2020.

the field under consideration. However, the priority belongs unconditionally to the domestic edition published in 1858 (N. A. Dobrolyubov. Guide to a visual study of the administrative order of securities in Russia. Moscow, 1858), which describes not only the predecessor of the modeling languages listed above, but also provides a number of designs for modern EML-class BP modeling languages (for example, "swimming paths").

2 Parallel business processes

Modern business process theory [1] is one of the sections of process theory and it is based on such areas of mathematical programming theory as formal grammars and languages; parallel processes and methods of parallelization of computer programs; software testing methods; methods of optimization, verification, analysis and evaluation of software quality; theory of databases and knowledge bases; structural and object-oriented methods of analysis and design, etc.

The main directions and sections of the theory of business processes are considered in the works [2, 3], we will give only a brief list of them here:

- modeling languages and BP models,
- modeling technologies,
- structuring/decomposition methods,
- engineering/reengineering methods,
- analysis and verification methods,
- methods for the transition from BP requirements models to BP automation models.

However, the above models and methods practically do not affect the issues of parallelism in the BP. It should be noted that even Hammer and Champi in [4] noted the effectiveness of parallelization of BP, for example, the rejection of the linearity of BP execution and the transition to a "natural" (in fact – parallel) order of execution in order to dramatically improve their characteristics.

On the other hand, parallelism is inherent in the very nature of BP. Moreover, such synchronization mechanisms as Dijkstra semaphores, conditional (critical) intervals, monitors, and control expressions [5] are informally present, for example, in the rules of work of performers with documents. Another thing is that such constructions are practically not used in BP modeling languages, which excludes the possibility of designing and studying parallel BP.

Another area of interest is the reorganization of BP parallelization methods, including parallelization of their linear sections and repetitive structures. The effectiveness of such parallelization may exceed the effect of “horizontal and vertical compaction” [4] – the two most effective mechanisms for reengineering business processes by Hammer and Champi.

To solve problems related to parallelism, the following classes of the computer architectures were identified in [1], based on the well-known Flynn computer classification:

- SISD (Single Instruction stream and Single Data stream) - sequential PSUs; MISD (Multiple Instruction stream and Single Data stream) - pipeline PSUs;

- SIMD (Single Instruction stream and Multiple Data stream) - synchronous (vector and matrix) PSUs;
- MIMD (Multiple Instruction stream and Multiple Data stream) – asynchronous PSUs.

It is obvious that the first and last classes do not need comments, the corresponding fragments are available in almost any business process. Conveyor and synchronous PSUs are also quite common. A classic example of a conveyor business process is the process of preparing an board for departure: a large number of operations (maintenance, baggage loading, passenger boarding, etc.) are performed on each instance of a single object. An example of a synchronous process is the process of writing a report in a large scientific organization that executes government orders, called projects. Each of the projects is divided into topics, each of the topics, in turn, is divided into works that are performed within the organization's divisions. By order of the management, reports on planned work are generated synchronously in each of these departments by a certain date, then topic managers form (also synchronously) reports on topics based on them, and then project managers – project reports. It should be noted that there may be links between works and topics. However, this does not contradict the concept of SIMD-parallelism, moreover, such features are directly embedded in the architectural solutions of SIMD-computers.

Thus, work on parallel business processes can develop in two directions: the creation of languages that allow describing parallel constructions, and the development of methods for parallelizing descriptions in traditional modeling languages. In turn, it is obvious that two approaches are possible to solve the first problem – the extension of existing languages in order to orient them to parallel BP and the creation of new parallel BP modeling languages.

Examples of parallel extensions include the corresponding constructions in the DFD technology [6] and the UML language [7]. In both cases, parallel behavior modeling is reduced to describing parallel control flows and ways to interact between them using the finite state machine apparatus. In DFD technology, these flows are objects of CFD (Control Flow Diagrams) diagrams at the top level and state transition diagrams at the lower levels, respectively. UML provides a range of special tools designed for more subtle, detailed, and explicit modeling of behavior during parallel execution of a process (in this case, computational). However, state transition diagrams are also used here due to their expansion using parallel composite states and the introduction of a number of additional structures, such as the active state, the configuration of active states, the area of the parallel composite state, the composite transition, and the synchronizing state [7].

Modern BP modeling languages (EML - Enterprise Modeling Language) organically include parallel process modeling tools that have the same level of abstraction as all other language tools. For example, one of the most commonly used languages of this class, BPML, based on the BPMN notation [8], contains such constructs as parallel execution without synchronization, parallel execution with synchronization, execution coordination, and a number of completion patterns that collectively implement the corresponding synchronization mechanisms.

In cases of creating new languages for describing MISD parallelism, in [9] developed a language for modeling pipeline BP, describing the process in the form of an acyclic oriented graph, the vertices of which can be functional operations or trigger functions. The set of functional primitives that allow describing conveyor BP includes such constructions as operation, linear conveyor, multiplication-reduction, conjunction-disjunction, distribution- reception functions, etc.

It should be noted that the constructions of all the above-mentioned parallel BP modeling languages without exception have a significantly lower level of abstraction compared to widely used diagram techniques such as IDEF0, DFD, and others, and are no longer able to play the role of languages "for transmitting understanding", in fact being classical design languages and being with the first type languages in the "assembler – high-level language" ratio.

Note that for SIMD parallelism (synchronous PD), there are currently no language tools for organizing it. However, as a basis for such parallelism, we can use the results of the programming theory in terms of the organization of vector and matrix data structures and methods of their placement and processing, as well as programming languages for vector computing [10]. To solve this problem, an appropriate extension of the BP information models (in particular, ERD - entity-relationship diagram) is necessary.

3 Conceptual model of the language

The conceptual model of one of the BP modeling languages proposed in [6] contains four basic components:

- language dictionary;
- language syntax;
- set of abstract semantic rules/procedures;
- aspects of language pragmatics.

The language dictionary includes three types of basic building blocks: objects, relationships and diagrams. Objects are basic indivisible elements (the alphabet of the language), relationships link objects into semantic blocks (words), and diagrams group words into "meaningful" phrases and sentences.

In its most general form, the language has 3 types of objects: a functional object (process, subsystem, minispecification, module, discriminator, etc.), an information object (external entity, storage, information channel, entity, event, data area etc.), and a behavioral object (control process, state etc.).

Relationships between objects determine their interaction through information flows and control signals, or provide a structural organization of object conglomerates (hierarchy, generalization etc.).

A diagram, in turn, is a collection of words represented as a directed graph with vertices corresponding to objects and edges corresponding to relations.

The language syntax defines the rules for forming words, phrases and sentences in the language.

The semantics of a language defines the meaning of language constructs and is defined by entering the specifications of each of the building blocks. At the same time, the range of methods for setting specifications for various objects varies from strictly formal ones (for example, specifying a data stream using the Backus-Naur form) to formalized ones at the level of the list of necessary attributes of the corresponding object (for example, setting restrictions on the information channel bandwidth). The corresponding semantic rules allow you to correctly and unambiguously define:

- identify objects names;
- name scopes and visibility;
- the equivalence of names;
- integrity and consistency of objects.

Pragmatics defines the subject area (scope) of language constructs – type of models (functional, informational, behavioral, etc.), modelling object (business model, organizational model, information systems model, etc.), stage of the life-cycle (requirements analysis, conceptual and detailed design) etc.

Traditionally, the formalism of graph grammars is used to define the syntax of visual languages [11, 12], which are a generalization of Chomsky grammars to graphs. A graph grammar is the quadruple (T, N, P, s), where T – the set of terminal symbols, N – set of nonterminal symbols, P is the set of rules of the form L ::= R (L is a nonempty sequence of terminal and nonterminal symbols, R is an arbitrary sequence of terminal and nonterminal symbols), s € N is the initial symbol. In such grammars, the role of traditional symbols is played by graphs/subgraphs of various types (namely, oriented graphs, multigraphs, pseudographs, Hi-graphs, metagraphs and hypergraphs).

At the same time, [13] proposed a model of DFD technology in the form of a mixed graph with different types of vertices and edges for an adequate description of organizational and management systems, and developed a special parallel attribute generating grammar for a business process that allows generating variants (scenarios) of its execution under various constraints.

To formalize the syntax of the business process modeling language, an intermediate variant is proposed, namely a mixed grammar, whose symbols can be not only graphs/subgraphs, but also fragments of visual models in various notations (within the framework of DFD technology) up to the atomic symbols of the language. This corresponds to the introduction of two types of terminal objects into the grammar: detailed (pseudo-terminal-terminal within a specific diagram) and non-detailed (terminal symbols).

As mentioned above, relationships between objects are divided into two types:

- linking objects at the same level of the model (relation-information flow, relation-control flow, relation-transition, relation-link);
- establishing inter-level relationships (decomposition relations of various types, categorization relations).

The semantics of relations of the first type consists in transmitting data or control (control signals) between objects of a specific level. In this case, the composition and structure of the transmitted data is determined by the corresponding grammatical rule. And the semantics of the control thread is determined by its type: a thread of type A starts the process, a thread of type B can both start and stop the process, and a thread of type C also starts and stops the process, but through different channels. For comparison, an analog of A-stream is a light switch that lights up until something happens inside the running process, for example, a light bulb burns out, an analog of B-stream is a traditional switch with light on / off functions, an analog of C-stream is a switch with two buttons, one for turning on and the other for turning off the light.

The semantics of decomposition relations consists in inter-level balancing, i.e., in fact, in linking first-type relations between model levels, the main rule of which is that all first-type relations associated with the object being detailed must be displayed (and linked to the corresponding objects) at the detailing level. The categorization relation is essentially a classical generalization relation.

The semantics of the expanded flat model are set at the lower level by finite automata, algorithmic languages, and relational algebra relations to describe behavioral aspects, functionality, and structure of information objects, respectively.

References

1. Kalyanov G.N. On the theory of business processes. *Software Engineering*. 2018; 9(3):99-109.
2. Kalyanov G.N. The theory of business processes: formal models and methods. *Economics, statistics and informatics*. 2016; (4):19-21.
3. Kalyanov G.N. Models and methods of the business processes theory (review). *Open education*. 2015; (6):4-9.
4. Hammer M., Champy J. Reengineering the Corporation: A Manifesto for Business Revolution. N.Y.: Harper-Collins; 1993.
5. Trahtengerts E.A. Software of parallel processes. Moscow: Nauka; 1987.
6. Kalyanov G. N. A conceptual model DFD-technology. *Open Education*. 2017; (4):21-26.
7. Gromov Y.Y., Ivanova O.G., Belyaev M.P., Danilkin S.V. Methods and tools for designing information systems. Object-oriented approach. Tambov; 2013.
8. Fedorov I.G. Modeling of business processes in BPMN 2.0 notation. Moscow: MESI; 2013.
9. Kupriyanov B.V. Method of effective analysis of the recursive conveyor process model. *Automatics and telemechanics*. 2017; (3):63-79.
10. Razbegin V.P., Kalyanov G.N., Kupriyanov B.V. Programming system for vector computing. *Programming*. 1985; (4):25-32.
11. Rekers J., Schürr A. A graph grammar approach to graphical parsing. In: *Proceedings of Symposium on Visual Languages*. pp. 195-202, 1995. doi: 10.1109/VL.1995.520809
12. Zhang D.-Q., Zhang K., Cao J. A context-sensitive graph grammar formalism for the specification of visual languages. *The Computer Journal*. 2001; 44(3):186-200.
13. Kalyanov G.N. Theory and practice of business process reorganization. Moscow: SINTEG, 2000.