Business process modeling based on entity life cycles

Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 138 (2018) 462–469

Procedia Computer Science

www.elsevier.com/locate/procedia

CENTERIS - International Conference on ENTERprise Information Systems / ProjMAN - International Conference on Project MANagement / HCist - International Conference on Health and Social Care Information Systems and Technologies, CENTERIS/ProjMAN/HCist 2018

Business process modeling based on entity life cycles

Giorgio Bruno*

Politecnico di Torino, Corso Duca degli Abruzzi 24, 10129 Torino, Italy

Abstract

This paper presents an approach to business process modeling that aims to benefit from research on the artifact-oriented perspective and on case management. It draws the notion of entity life cycle from the former and the notion of hierarchical stage from the latter. The main purpose of the approach, which is called ELBA (Entity-Lifecyle Based Approach), is to leverage the notion of dataflow to coordinate the life cycles of the entities involved in a business process. For this reason, the approach takes advantage of a number of patterns, structural and functional, which are illustrated with the help of two examples, one related to the handling of papers submitted to conferences and the other concerning the many-to-many mapping between requisition orders and procurement ones in build-to-order processes.

Selection and peer-review under responsibility of the scientific committee of the CENTERIS - International Conference on ENTERprise Information Systems / ProjMAN - International Conference on Project MANagement / HCist - International Conference on Health and Social Care Information Systems and Technologies.

Keywords: entity life cycle; information model; dataflow; hierarchical stage; flow pattern; selection pattern.

* Corresponding author. Tel.: +39 0110907003.
E-mail address: giorgio.bruno@polito.it

Selection and peer-review under responsibility of the scientific committee of the CENTERIS - International Conference on ENTERprise Information Systems / ProjMAN - International Conference on Project MANagement / HCist - International Conference on Health and Social Care Information Systems and Technologies.
10.1016/j.procs.2018.10.064

1. Introduction

Three major perspectives have brought relevant contributions to the modeling of business processes. The well-known activity-oriented viewpoint, whose champion is BPMN [1], emphasizes both the tasks that carry out the purposes of a business process and the control flow that determines their ordering; business entities are not given a conceptual representation and are simply denoted by variables in process instances.

The artifact-oriented perspective [2] puts the artifacts in the foreground: an artifact is a business entity that evolves over time through a life cycle made up of states (also called stages), and the transitions between states take place through tasks.

The notion of case as a situation requiring customized treatment has inspired the recent CMMN standard [3]: the emphasis is on the participants in the process, who can decide the order of execution of tasks, and can also assign tasks to each other.

The research reported in this paper aims to integrate the contributions of the above-mentioned perspectives by leveraging the notion of dataflow. The dataflow provides the inputs to the tasks of the process and receives their outputs; it is made up of business entities whose types are defined in a companion information model showing their attributes and relationships.

The entities used in a process may be divided into two major categories: the active ones are generated and modified during the execution of the process, while the passive entities are mainly intended to receive connections from active ones. For example, in a process handling papers submitted to a conference, the conference entity and the paper entities are active while the tracks of the conference are passive entities in that they are determined before the process begins and are not affected by its execution. Active entities follow specific life cycles to which relevant tasks are associated: the dataflow is represented by the states of the active entities.

A process model results from the coordination of the life cycles of its active entities and the analysis of recurring situations may lead to the discovery of useful patterns.

The approach, called ELBA (Entity-Lifecyle Based Approach), presented in this paper takes advantage of a number of patterns: one is structural and the others are functional. The structural pattern, called hierarchical life cycles, is used when there are subordination constraints between the entities of a process. In the above-mentioned process handling paper submissions, paper entities are subordinate to the conference entity because the progress of papers depends on that of the conference. With the structural pattern, the states of subordinate entities are included in the states of the primary entity. For example, a conference entity goes through a number of states, such as paper submissions, paper assignments, and so on. In state paper submissions, papers may be submitted, modified and also withdrawn by their authors; in state paper assignments, papers are assigned to reviewers.

Functional patterns include flow patterns and selection ones: the former are about merging, decomposing, and recomposing flows and have strong similarities with those classified as Enterprise Integration Patterns [4]. Selection patterns come into play when the inputs of a task have to be selected from among those available by the task performer, or when the same inputs may be handled with two or more alternative tasks and it is up to the performer to select the most suitable one. Human choices are grounded on these patterns.

This paper is organized as follows. Section 2 is about the related work. Section 3 introduces hierarchical life cycles with the help of the paper-submissions process, and section 4 illustrates flow patterns and selection ones on the basis of an order-handling process. Section 5 contains the conclusion.

2. Related work

The key terms of the research reported in this paper are business entity, life cycle, dataflow, and flexibility. Business entities are called artifacts in the artifact-oriented perspective [2]. Artifacts are associated with business goals and the analysis of how they progress toward their goals helps determine their life cycles [5]. Life cycles improve communication among stakeholders and let them focus on the primary purposes of the business [6]. Several notations have been proposed, such as GSM (Guard-Stage-Milestone) [7], COREPRO [8] and PHILharmonicFlows [9]. They differ in the coordination between life cycles: GSM leverages events and rules, COREPRO is based on

hierarchical (container-components) relationships, and PHILharmonicFlows introduces macro processes in order to coordinate the state transitions of the entities involved in the process.

In ELBA, coordination is achieved by means of flow patterns meant to merge, decompose and recompose flows of entities. The latter two draws on Enterprise Integration Patterns.

The dataflow is a kind of "afterthought" [10] in business process notations such as BPMN. On the contrary, in ELBA, the dataflow supersedes the control flow: the activation of a task does not depend on the completion of a previous task, but on the availability of suitable input entities.

Leveraging the dataflow may help address situations that are not adequately supported by notations based on a rigid control flow: an example is the many-to-many mapping between requisition orders and procurement ones in build-to-order processes. The difficulty of using BPMN has been pointed out [11]; what is needed is a solution based on flow patterns such as the one presented in section 4 of this paper.

Dataflow models have also been presented in other papers [12, 13] but there the dataflow is subordinate to the control flow. Dataflow patterns have also been addressed [14]: the purpose, however, is to analyze the exchange of information between process variables and tasks.

Flexibility is of paramount significance when addressing knowledge-intensive processes [15]. Flexibility means that the execution of tasks can be decided by the participants in the process, and is not subjected to a rigid control flow as it occurs in routines.

Case handling [16] and case management [17] are grounded on flexibility. The recent case management standard CMMN [3] stresses the abilities of case workers to decide the order of execution of tasks and to assign tasks to each other. CMMN processes are made up of stages that are groups of tasks: the opening and closing of stages are based on events, such as the completion of a task, a time event or a human decision. However, the stages are not related to the life cycles of business entities and moreover the dataflow is missing.

In ELBA, flexibility means handling situations in which participants need to be able to select the inputs when a task may need more than one input, or the task when two or more alternatives for the same inputs are available.

3. Hierarchical life cycles

The term "hierarchical life cycles" denotes a situation in which there are subordination constraints between the entities in the process under consideration. Such constraints imply that the progress of the entities of a certain type, say, B, depends on the progress of the entities of another type, say, A. A is a master type with respect to B and B is a subordinate of A. From a notational viewpoint, the states of B are included in the states of A.

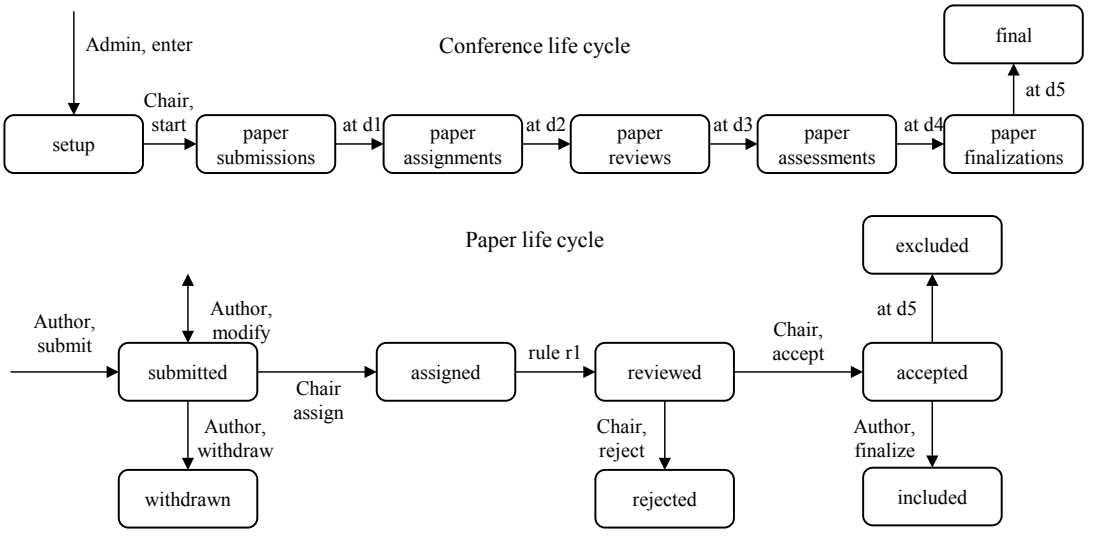
An example is given by a process handling the submissions of papers to a conference; this example is a simplification of the actual process. The active entity types are Conference, Paper and Review. Papers are assumed to be assigned to three reviewers each; a reviewer receives an empty review for each paper that has been assigned to him or her, and is meant to return the reviews completed. In addition, there are three role entities, i.e., Chair, Author and Reviewer, which represent the participants in the process. These six entity types along with their relationships and attributes make up the information model of the process.

The life cycles of the Conference and Paper entities may be shown separately, as it takes place in Fig.1. This picture gives a view that is easy to understand on the dynamics of the process. A life cycle is a state-transition diagram. States are depicted as rectangles with rounded corners: the state name is written inside. Transitions have labels indicating the reason of the state change: the reason may be the execution of a task, a deadline, or the firing of a rule. The initial state has an input arc with no source, which is called initial transition.

The Conference life cycle develops through a sequence of states. It is assumed that a conference entity is generated by the administrator of the system who also associates the chair, represented by a Chair entity, with the conference. That is the purpose of the enter task that labels the initial transition. This is a human task and, as such, it is preceded by the role of the people who can perform it.

The purpose of the Conference states is as follows. In the setup state, the chair is meant to enter information about the conference as well as to enter the reviewers. In state paper submissions, authors may register with the conference and then they may submit papers and also withdraw or modify them. In state paper assignments, the chair assigns each paper to three reviewers by providing them with empty reviews, and, in state paper reviews, they complete their

reviews. In state paper assessments, the chair decides the acceptance or rejection of the papers. In state paper finalizations, the authors of accepted papers may provide the final versions of their papers (along with payment proofs). Only the papers that have been finalized are included in the conference.



rule r1: when 3 reviews for the paper are available
at d5: when the current day matches deadline d5 of the conference the paper has been submitted to

Figure 1. The life cycles of Conference and Paper entities

The transition from state setup to state paper submissions is driven by task start which is performed by the chair; the subsequent transitions are driven by timing constrains which are taken from the attributes of the conference entity; these attributes (d1..d5) are dates defined in the setup state. The label "at d", where d is a date attribute of the entities contained in the source state of the transition, means that the transition will affect the entities in the source state at the end of the day matching the deadline written in their attributes d.

The Paper life cycle is also shown in Fig.1. The states and transitions are derived from the description above. The double arrow in transition "Author, modify" means that the source and destination of the transition coincide.

There are two transitions that are not caused by the execution of a task. The one from state assigned to state reviewed is determined by the firing of a rule, while that from state accepted to state excluded is due to a timing constraint. The rules and the timing constraints are informally described in the annotations of the diagram.

The state transitions of the papers take place in the time framework defined for the conference. For this reason, the life cycle of Paper has to be distributed into the Conference one. In addition, the life cycle of Review has to be included as well: the purpose is to place the work of reviewers in the proper time frame.

The resulting process model is shown in Fig.2. It basically keeps the high-level life cycle of Conference and shows the expansion of its hierarchical states.

In a process model, the name of a state is preceded by the name of the type of the entities that the state can contain. If a state is expanded, type and name are shown above the state icon. Moreover, in a process model, tasks are depicted as rectangles when they have two or more input states or two or more output states.

In state paper submissions, the authors must first register with the conference and then they can submit papers. Task registerAsAuthor enables a user of the system to register with the conference as an author. The input state and the output one of the task coincide with the enclosing state; therefore, the rectangle icon is needed and the input and output states are omitted. The execution of the task enables the user to select a conference from among those in state paper submissions; the user is then registered at the conference with the Author role.

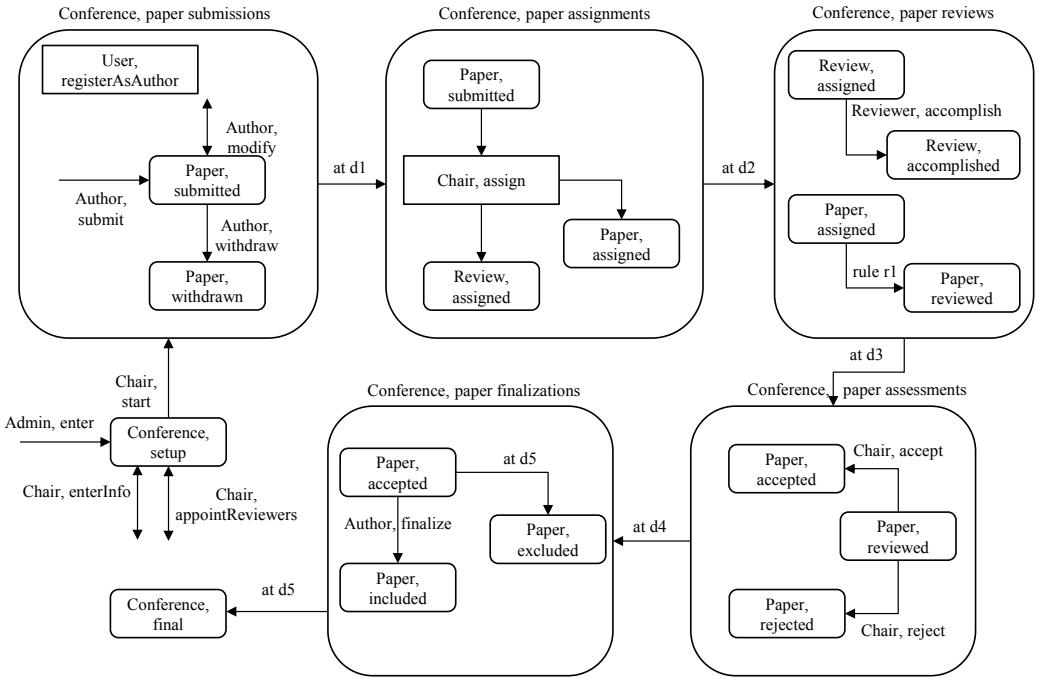


Figure 2. The Conference process with hierarchical states

Authors may then submit papers and may also modify or withdraw their papers.

In state paper assignments, the chair can assign papers to reviewers; he or she does so by generating three empty reviews for each paper; a review is directed to a reviewer selected from among those that the chair has associated with the conference by executing task appointReviewers in state setup. State assigned is the initial state of the Review life cycle and contains the empty reviews.

In state paper reviews, the reviewers accomplish their reviews and the papers are moved from state assigned to state reviewed when rule r1 fires. This rule has been informally described in Fig.1.

In state paper finalizations, the accepted papers may be finalized by their authors: these papers enter state included. If an accepted paper is not finalized before deadline d5 of the conference it is moved into state excluded.

4. Functional patterns

Functional patterns include flow patterns and selection patterns. The former are about merging, decomposing, and recomposing flows. Selection patterns are about selecting the inputs when a task may need more than one, or the task when two or more alternatives for the same inputs are available.

These patterns are illustrated with the help of an order-handling process that enables a distributor to operate in the middle of a supply chain between customers and suppliers. The requisition orders coming from customers are handled by the account managers working with the distributor. Orders are made up of a number of lines, each one pointing to a product type.

Each customer is served by a specific account manager who can bundle the lines of incoming orders into procurement orders directed to suppliers. An account manager has the choice of adding lines to a newly generated procurement order (which is then put in the pending state) or to a pending one. Eventually he or she places the order with the supplier associated with it. When the supplier has fulfilled an order, its lines are considered to be served. When all the lines related to a requisition order have been served, this order is considered to be served as well.

Three life cycles are needed; they refer to entity types RequisitionOrder (ROrder), ProcurementOrder (POrder) and Line. The process model is flat as those life cycles take place at the same level.

Three flow patterns are used: a splitter to obtain the lines from an order, an aggregator to recompose the served lines into a served requisition order, and a matching pattern to add lines to pending orders. The splitter and aggregator tasks are automatic ones and draw on Enterprise Integration Patterns.

The information model of the process is shown in Fig.3; it is basically a UML class model [18]. In the process model shown in Fig.4, role AccountMgr is abbreviated to AM, and state handled of Line is shown twice to avoid crossing connections.

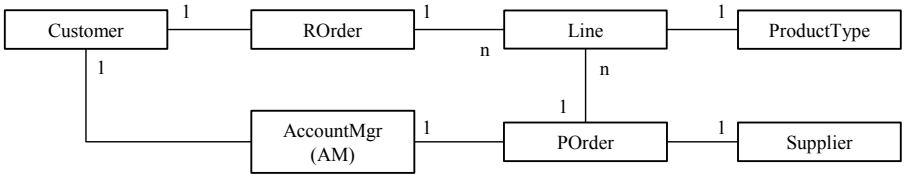


Figure 3. Information model of the order-handling process

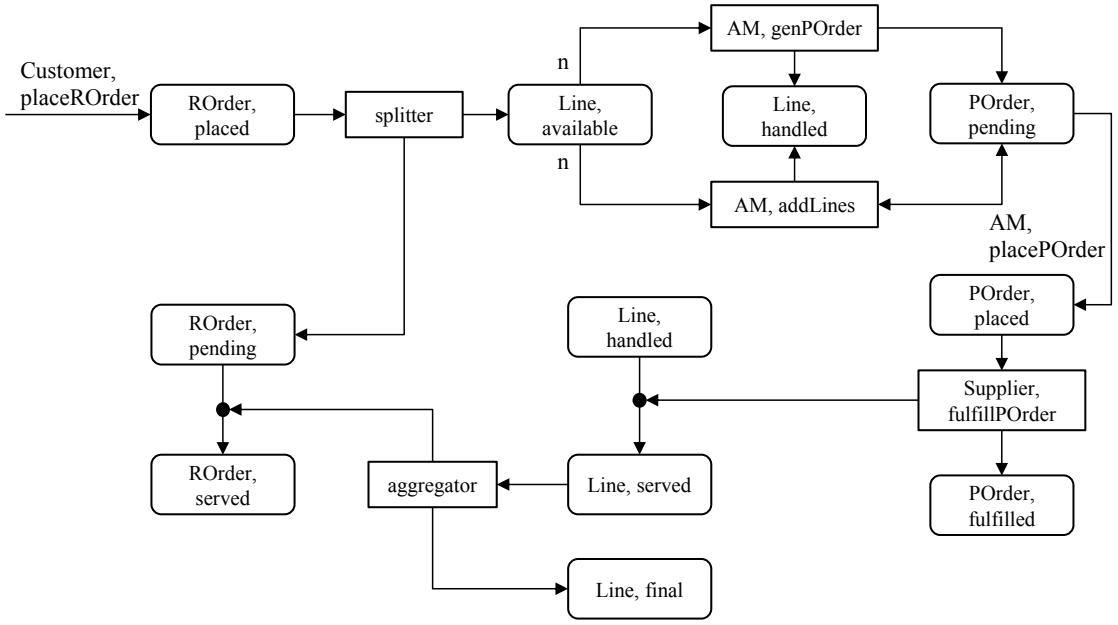


Figure 4. The model of the order-handling process

There are several points of interest in the process model, as follows.

The splitter task has one input state and two output ones: one output state is needed to show the change of state of the input entity, and the other receives all the entities related to the input one on the basis of the relationship between the input type and the output one. In the example, state available of Line gets all the lines related to the ROrder entity taken by the splitter from state placed. State available is the initial state of the Line life cycle.

Task genPOrder enables an account manager to bundle a number of available lines so as to generate a new POrder (in state pending) with which the lines are associated. Task addLines is a matching task in that it enables an account manager to associate a number of available lines with a pending procurement order. The choice of the lines

and of the order is up to the account manager. Weights n on the output links of state available of Line indicate that the output tasks of the state can take more than one line at each execution.

The two tasks above involve the human choice of the input entities; in fact, it is up to the account manager to select the lines to be associated with a procurement order. Constraints may be added as pre-conditions to those tasks so as to make sure that the product types indicated by the lines are provided by the supplier the procurement order is directed to.

From the viewpoint of an account manager, tasks genPOrder and addLines are two alternatives that share some of the input entities, i.e., the lines. The choice of the task is up to the account manager. This is a case of human choice related to the selection of the task with which to handle the same input entities when two or more alternatives are available.

The fulfillment of a procurement order involves two changes of state: one affects the input order and the other affects the lines associated with it. The order becomes fulfilled and the lines become served. Task fulfillPOrder turns out to coordinate the POrder life cycle and the Line one. It is not a matching task because the lines do not have to be selected from among those in state handled. The lines are those related to the input procurement order. The transition of the lines from state handled to state served is called correlated transition: it is an automatic transition following that of the procurement order the lines are related to. For this reason, a correlated transition has no label but has a small node in the middle: the node receives a link from the task that triggers the transition.

State served of Line is the input of an aggregator task. An aggregator has one input state and one output state: they have the same type. Moreover, it is connected to a correlated transition in the life cycle of a different entity type referred to as aggregator type. The aggregator works as follows. When the input state contains all the entities that are related to an entity of the aggregator type (this entity is referred to as aggregating entity), it changes the state of such input entities, and it also changes the state of the aggregating entity on the basis of the correlated transition. In the example, the aggregating entity of the lines is the ROrder entity they refer to. When all the lines referring to the same ROrder entity are contained in state served, the aggregator moves the lines into state final and the order into state served.

The process model coordinates three life cycles, whose states are as follows; ROrder: placed, pending, served; Line: available, handled, served, final; POrder: pending, placed, fulfilled. All the tasks in the process model but placeROrder and placePOrder take part in the coordination of the entity life cycles.

5. Conclusion

This paper has presented the ELBA approach, which leverages the notion of artifact life cycle so as to emphasize the flow of entities in business process models. Two models have been illustrated. The first one shows a master life cycle that controls the life cycles of the other entities (the subordinate ones): the states of the master life cycle include the states of the subordinate life cycles. The second model consists of peer life cycles that are coordinated by means of flow patterns and correlated transitions. In addition, flexibility issues have been discussed as far as the human selection of input entities or alternative tasks are concerned.

A taxonomy of task patterns and the extension of CMMN with entity life cycles have been presented in previous papers of the author [19, 20].

Current work is devoted to the design and implementation of a simulation environment whose purpose is to validate the logic of the process under consideration on the basis of java lambdas that are associated with the tasks so as to simulate their behavior.

References

[1] BPMN, Business Process Model and Notation, V.2.0.2. Retrieved February 4, 2018, from http://www.omg.org/spec/BPMN/2.0.2/
[2] Hull, R. (2008) “Artifact-centric business process models: brief survey of research results and challenges.” LNCS 5332: 1152–1163. Heidelberg: Springer
[3] CMMN. Case Management Model and Notation, V.1.0. Retrieved February 4, 2018, from http://www.omg.org/spec/CMMN
[4] Hohpe, G., Woolf., B. (2004) “Enterprise Integration Patterns.” Addison-Wesley

[5] Nigam A., Caswell, N. S. (2003) "Business artifacts: an approach to operational specification." *IBM Systems Journal* **42**: 428–445
[6] Chao, T., et al. (2009) "Artifact-based transformation of IBM Global Financing." LNCS 5701: 261–277. Springer, Heidelberg
[7] Hull, R., et al. (2011) "Introducing the Guard-Stage-Milestone approach for specifying business entity life cycles." LNCS 6551: 1–24. Springer, Heidelberg
[8] Müller D., Reichert M., Herbst J. (2007) "Data-driven modeling and coordination of large process structures." LNCS 4803: 131–149. Springer, Heidelberg
[9] Künzle ,V., Reichert, M. (2011) "PHILharmonicFlows: towards a framework for object-aware process management." *Journal of Software Maintenance and Evolution: Research and Practice* **23**: 205–244
[10] Sanz, J.L.C. (2011) "Entity-centric operations modeling for business process management - A multidisciplinary review of the state-of-the-art." In: 6th IEEE Int. Symposium on Service Oriented System Engineering, pp. 152–163. IEEE Press, New York
[11] Meyer, A., Pufahl, L., Fahland, D., Weske, M. (2013) "Modeling and enacting complex data dependencies in business processes." LNCS 8094: 171–186. Springer, Heidelberg
[12] Kucukoguz, E., Su, J. (2011) "On life cycle constraints of artifact-centric workflows." LNCS 6551: 71–85. Springer, Heidelberg
[13] Kumaran, S., Liu, R., Wu, F. Y. (2008) "On the duality of information-centric and activity-centric models of business processes." LNCS 5074: 32–47. Springer, Heidelberg
[14] Russell, N., ter Hofstede, A. H., Edmond, D, van der Aalst, W.M.P. (2005) "Workflow data patterns: identification, representation and tool support." LNCS 3716: 353–368. Springer, Heidelberg
[15] Di Ciccio, C., Marrella, A., Russo, A. (2015) "Knowledge-intensive processes: characteristics, requirements and analysis of contemporary approaches." *Journal on Data Semantics* **4**: 29–57
[16] Van der Aalst, W.M.P., Weske, M., Grünbauer, D. (2005) "Case handling: a new paradigm for business process support." *Data & Knowledge Engineering* **53** (**2**): 129–162
[17] Marin, M., Hauder, M., Matthes, F. (2015) "Case management: an evaluation of existing approaches for knowledge-intensive processes." In 4th International Workshop on Adaptive Case Management and other non-workflow approaches to BPM. Springer, Heidelberg
[18] UML. Unified Modeling Language, V.2.5.1. Retrieved February 4, 2018, from http://www.omg.org/spec/UML/
[19] Bruno, G. (2015) "Data flow and human tasks in business process models." *Procedia Computer Science* **64**: 379–386
[20] Bruno, G. (2017) "Extending CMMN with entity life cycles." *Procedia Computer Science* **121**: 98–105