Events and Activities: Is there an Ontology behind BPMN?

Emilio M. SANFILIPPO [a,b,1] Stefano BORGO [b] and Claudio MASOLO [b]
[a] *Institute of Industrial Technologies and Automation, ITIA CNR, Italy*
[b] *Laboratory for Applied Ontology, Institute of Cognitive Sciences and Technologies ISTC CNR, Italy*

Abstract. In the context of business process modelling, the Business Process Model and Notation (BPMN) is a de-facto standard with more than 70 commercial tools that currently support its use. Amongst its main modelling constructs, BPMN includes *activities* and *events*. However, the focus of the standard is on providing an intuitive graphical language, rather than formal semantics specifications. This results in semantic ambiguities regarding the interpretation of its modelling constructs. We investigate whether the main building blocks of BPMN commit to an ontological theory of the domain entities at hand, eventually clarifying this commitment by the approach of ontological analysis.

Keywords. Business Process, BPMN, Event, Activity

Introduction

Business process modelling (BPM) concerns the analysis and representation of the activities by which companies coordinate their internal organisation and work, produce goods of business value, interact with each others and customers. Scholars, communities, process modelling languages and terminological systems [21,22,23] have different understanding of BPM, and agree on just a minimal view about business processes (BP) in terms of: "a set of activities whose coordinated execution contributes to the realization of a business function in a technical and organizational environment" [22, p.73]. Nevertheless, stakeholders, business analysts and business people need to share a common conceptual modelling language that can be easily understood to facilitate business communication intra- and inter-organisations.

The Business Process Model and Notation (BPMN) [18] has been proposed over the years to support BPM and is a OMG standard[2] with more than 70 commercial tools that currently supports its use[3]. There are five categories of elements in BPMN: *Flow objects*, *Data*, *Connecting Objects*, *Swimlanes* and *Artifacts*. In this paper, we study only two of these modelling constructs, namely *event* and *activity*, which are defined within the

[1] Corresponding Author: Laboratory for Applied Ontology, ISTC-CNR, via alla cascata 56C, Povo, Trento, 38123, Italy ; E-mail: emilio.sanfilippo@itia.cnr.it.
[2] http://www.omg.org
[3] http://www.bpmn.org

category of *Flow objects* and play a key role for process modeling. In particular, a BPMN activity represents "a work that is performed within a business process" [18, p.29] and can be either atomic (*tasks*), or non-atomic (*sub-processes*). A BPMN event "is something that "happens" during the course of a process [...] and usually have a cause [trigger] or an impact [result]" (ibid., p.233). Events affect the flow of the model: *throw* events cause something to happen, *catch* events are caused to happen. Moreover, depending on their position in the process flow, they are: *start events*, *end events*, or *intermediate events*.

Despite its large application, BPMN main focus is on graphical constructs rather than on formal semantics. This choice has some well-known drawbacks: BPMN presents conceptual ambiguities regarding the interpretation of its metamodel; the supporting software tools are not guarantee to interoperate; and there is no guideline for using the modeling language in ontology-based systems. Different communities, in particular within the Semantic Web and knowledge management domains, have been working on BPMN-like ontologies for disparate application purposes. However, due to BPMN's conceptual ambiguities and the differences across the communities, no credible candidate has yet emerged as the ontological counterpart for BPMN. Here we follow a different route and apply ontological analysis to dwell into the backbone elements of BPMN. The goal is to investigate whether the standard is (possibly implicitly) committed to some coherent ontological perspective.

The remainder of the paper is organised as follows. Section 1 gives the state of the art relatively to the ontological analysis of BPMN. Section 2 looks at a BPMN process diagram to highlight some problematic issues regarding the BPMN conceptualisation. Section 3 gives the ontological analysis of our target notions. Section 4 indicates directions for future work.

1. State of the art

Different communities have proposed implementations of BPMN in some ontology modeling language and have discussed the concepts of BPMN at least to some extent.

Following [14], one can identify three goals in the study of BPMN. These lead to three types of analysis, each focusing on different knowledge about the modeled processes: (i) the *structural* or *syntactic* analysis, (ii) the *behavioral* or *execution* analysis, and (iii) the *ontological* or *conceptual* analysis. The structural analysis aims at formalizing the meta-model of BPMN, i.e., at defining the structural constraints that BPMN-models must satisfy—e.g., the fact that a Start Event has no incoming Sequence Flow (solid arrow) or an AND-split has at least two outgoing Sequence Flows. At this level, a BPMN-model is seen as a labeled graph which is well-formed if validated by a certain set of rules for the combination of the basic graphical objects (circles, arrows, boxes etc.) [9,4,1]. In [9], the authors present the BPMNO meta-ontology implemented in the Web Ontology Language (OWL) [15]. The authors aim to allow reasoning with the semantically annotated processes [5] and thus their formalization enriches BPMN with, e.g., temporal information. An analysis of the BPMN concepts is not in the scope of this work. Similarly, the Semantic Business Process Modelling Notation (sBPMN), developed within the SUPER project [1], provides a the meta-ontology encoded in the Web Service Modeling Language (WSML) [3]. In this case, neither an ontological nor logical attempt is made to clarify the BPMN notions. More generally, although the syntactic constraints can be

motivated by semantical considerations, these remain largely implicit in the structural analyses of BPMN.

The behavioral analysis looks at what can happen during the execution of a well-formed BPMN model like, e.g., the information or physical items that are produced or consumed during the execution of the process [14] or the existence of deadlocks or live-locks [6]. This (static) analysis of a process model considers the semantics underlying the schema only for procedural information and is not relevant for our work in this paper.

Finally, the ontological analysis focuses on the characterization of the primitives of the language. This goal can be achieved by studying BPMN as a self-standing system or by looking at correspondences with the categories of a foundational ontology. In the latter case, the analysis is a sort of *ontological annotation* or *semantic mapping*. [14,10] and [19] fall into the latter case: [14] discusses the OPAL reference framework and characterizes the specific kinds of activities or events present in a given model, it lacks a characterization of the general difference between BPMN-activities and BPMN-events since both BPMN activities and events are mapped to OPAL processes; [10] uses ABDESO, a foundational ontology for agent-based discrete event simulation, to distinguish actions from other events (only the first are intentionally performed by agents). The purpose is to assess whether BPMN is suitable for business system simulation and the the authors find quite a few ambiguous and redundant elements, as well as missing concepts in BPMN. Finally, [19] looks at the mutual relationship between BPMN and the Bunge-Wand-Weber (BWW) foundational ontology. The paper highlights some ontological shortcoming in the first release of the standard with respect to ontological completeness, construct overload, construct excess and construct redundancy. It concludes that BWW-Event, BWW-ExternalEvent e BWW-PoorlyDefinedEvent should be seen as subclasses of BPMN events and the elements of BWW-Transformation as BPMN activities and tasks. In all these ontological studies, the characterization of the differences between the BPMN-primitives is based on a comparison with other frameworks.

[17] presented a development of an OWL ontology based on the version 2.0 of BPMN. In this work, it emerges with a certain emphasis the problems of developing a BPMN-like ontology given the conceptual unclarity of the standard. However, this work is not based on ontological analysis and the ontological status of BPMN notions is not discussed.

In this paper we are interested in the ontological analysis of BPMN with the aim of clarifying how one can understand the notions of activity and event. We carry out our study in two ways: first by ontologically analyzing the information provided by the BPMN standard, and then by characterizing our findings on these concepts with the DOLCE ontology [13]. Note that we are not proposing an evaluation of the BPMN models with respect to an ontology; we use the ontology to find (possibly implicit) commitments of the standard, identify business related elements that the standard does not address, and to highlight the variety of interpretations that are consistent with these constraints.

2. Activities and events in BPMN

The BPMN diagram in Figure 1 represents a process with four participants (pools): *Purchaser*, *Service provider A*, *Service provider B* and *Service provider C*. The process starts when the event type *None* in the *Purchaser* pool happens. This is followed by the exe-

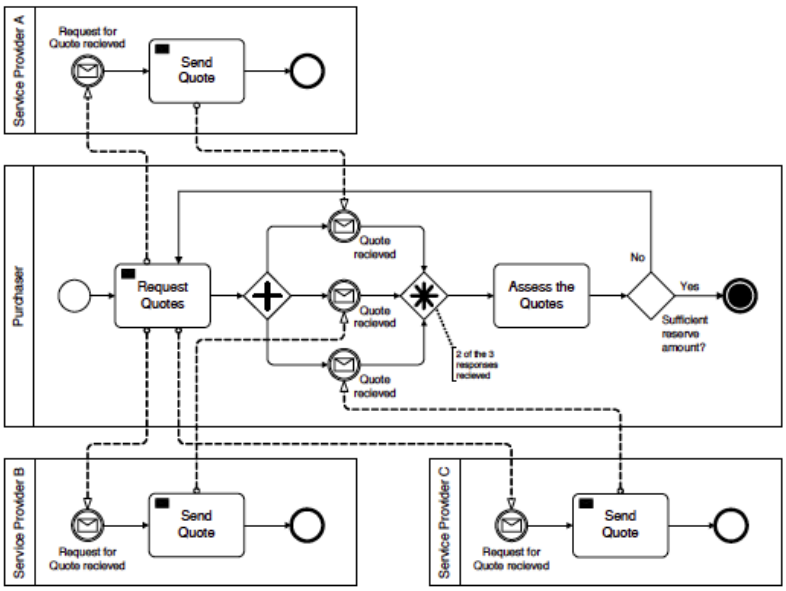

Figure 1. BPMN process diagram, taken from [18, p.362]. Circles are events; rectangles are tasks and diamonds are gateways. Solid arrows indicate the process flow within a pool (identified by a rectangle labeled on the left); dashed arrows indicate interactions across pools. Each process is contained within a rectangle, that is, a pool.

cution of task *Request Quotes* which ends with the sending of a message to each service provider participant in the process. Once a service provider receives the message, it starts its own process consisting in sending a quote to the *Purchaser* (the task *Send Quote*). After this, the service provider process ends. When the *Purchaser* receives at least two quotes (the gateway marked with an asterisk), it executes the *Assess the Quotes* task after which the process ends for the *Purchaser* provided the condition *Sufficient reserve amount?* of the last gateway is satisfied. Otherwise, the process is back to the *Request Quotes* and flows again as described above.

Already at this level, some observations arises. The process model in Figure 1 is composed of four processes each associated to a pool. However, the model does not tell how to interpret the pools; it is unclear whether, once the process is started, the *Service provider A, B, C* are to be understood as frozen with specific agents (organizations, services) or if these can change during the execution. The problem is evident when there is a cycle: should the *Purchaser* send the message to the same individuals playing *Service provider A, B, C* or to new individuals assuming those roles?

Furthermore, the reason why some parts of the process are activities and other events is not clear. In Figure 1, messages are exchanged between the process pools by using tasks of type *Send message* (these tasks are marked with a filled envelope icon). However, it is well known that among its graphical elements, BPMN includes *Message* as a throw event, which is used to model the sending of a message as well. Since both *Message task* (an activity) and *Message throw event* (an event) model the same sending of a message, people have discussed whether there are differences in meaning and what are the guidelines to use one or the other construct. According to Ferreira: “It is possible to use either one or the other. In general, an event conveys the idea that *when something happens...*

whereas an activity places more focus on the idea that *something needs to be done*" [7, p.325]. The idea seems to suggest that events represent changes in the domain being modelled, while activities refer to a participant's commitment towards the fulfilment of a specific goal. Unfortunately, this distinction is not grounded on discriminating attributes between events and activities. The BPMN Quick Guide[4] states that "[...] an Event maps to a time point on a time line [while] a Task [or an activity] maps to a time interval". This relationship between BPMN constructs and the temporal line suggests that events are instantaneous while activities last in time, so that temporal atomicity is a discriminating property among these two types of entities. Unfortunately (once more), BPMN does not commit to a theory of time points or intervals, every reference to time beyond atomicity remains vague within its modelling framework. Another possibility is to understand activities and (at least some) events in terms of endogenous vs exogenous entities: the first are happenings controlled within the poll the latter are out of the control of the pool.

But what are the possible readings of a BPMN process? How can we identify them and evaluate their coherence? In the next paragraph we start answering these questions by digging deeper into the notions of event and activity (task) as used in BPMN. Our conceptual tools are based on formal ontology considerations and are guided, when a comparison is deemed useful, by the distinction in the DOLCE foundational ontology [13].

3. Ontological analysis of BPMN events and activities

Events and activities in BPMN are connected to other events and activities in the same (in a different) pool by solid (dashed) arrows; these arrows mark execution precedence and thus temporal dependences. This reveals the temporal nature of events and activities in BPMN. Among the approaches in Section 1, [10] and [14] represent events and activities as (types of) *occurrents*, *perdurants* in the DOLCE terminology, i.e. entities that extend in time by accumulating temporal parts, while [19] models them as (sequences of) state transitions. In addition, [10] assumes that tasks and messages in BPMN are *actions*, i.e., occurrents intentionally performed by (physical or institutional) agents corresponding to BPMN-pools. This choice comes from an implicit assumption in most (if not all) BPMN descriptions, i.e., that a pool's label denotes more or less explicitly an agent which is interested or active in (and sometimes even responsible of) the process in the pool.[5]

In the following we use the DOLCE taxonomy of perdurants—mainly the concepts of Achievement, Accomplishment, State, and Process—to discuss and ontologically characterize the difference between Activities and Events, and between Catch- and Throw-events. While we find helpful to use a foundational ontology like DOLCE, we remark that the analysis could be based on other ontological systems provided they include a taxonomy of perdurants.

3.1. Activities and tasks

We have seen that a difference between activities and events, pointed out in the BPMN Quick Guide, regards their temporal extension: events are instantaneous (punctual) while

[4] http://www.bpmn.org

[5] These agents are then participants in the process and one could look at the ontological nature of the participants and the kind of participation relation to distinguish types of activities.

activities take time to be executed, they extend in time. In addition, BPMN explicitly considers tasks, like the perdurant *Request quotes* in the *Purchaser* pool of Figure 1, to be atomic. The relationship between *being instantaneous* and *being atomic* is not trivial given that a BPMN task, an *atomic* perdurant, can have a positive temporal extension in some temporal framework.

Four-dimensionalim [20] assumes that perdurants extend in time by having different temporal slices at different times. This would rule out tasks because, by extending in time, they necessarily have (temporal) proper parts, i.e., in the DOLCE terminology, they are *anti-atomic* perdurants. These observations lead to the suggestion that BPMN is modeling the world at some *granularity*: tasks are *considered to be atomic* (in the context of the BPMN model) even though in the actual world they have temporal parts. For example, one could assume that the *Request Quotes* instances in the world (but not in the model) are composed by a first step of type *Prepare the data for a request*, followed by a step of type *Collect the addresses*, and that none of these types fall under *Request quotes* type. From an ontological perspective this means that *Request Quotes* is *anti-atomic* and *anti-homeomeric*, i.e., *necessarily* (where 'necessity' is here used in the ontological sense) all the instances have parts that do not belong to *Request Quotes*.[6] (The anti-homeomericity is evident from BPMN sub-processes by which the structure of the whole perdurant is explicitly included in the BPMN-model.) One could also weaken *anti*-homeomericity to *non*-homeomericity, i.e., to claim that it is possible to have activities without proper components of type different from the one of the whole.[7]

A weaker way to understand tasks relies on a conceptual, as opposed to ontological, modality. Let us leave four-dimensionalism aside and assume that extended perdurants can be atomic, a position compatible with DOLCE. In this case, tasks could be understood neither as necessarily non-atomic nor as necessarily atomic: it is always possible to conceive a task as atomic or non atomic depending on the context, granularity, or perspective on the world.

Secondly, we observe that the mereological sum of two instances of a task like *Sign the contract* (in the same business process) is not an entity of type *Sign the contract*. This is consistent with the assumption that BPMN activities represent units of work, which are identified by the purpose of achieving given business goals; they have a structure and *culminate* with the achievement of the goal. This implies the *anti-cumulativity* of activities: the sum of two instances of a given activity A is *necessarily* not of type A.

We can then conclude that by considering a strict ontological-modality activities are anti-atomic and anti-cumulative, i.e., they can be mapped to DOLCE *accomplishments*. Vice versa, by assuming a conceptual-modality, only sub-processes may be mapped to accomplishments. More generally tasks would be mapped to DOLCE *events*, i.e., tasks are anti-cumulative types of perdurants with no commitment on atomicity and homeomericity.

3.2. Catch events

We saw that events are instantaneous, consequently they are temporally atomic. This reading reflects the idea of temporal atomicity as a 'punctual occurrence' in Mourelatos

[6] In DOLCE anti-homeomericity implies anti-atomicity.

[7] Truly homeomeric perdurants, like *being sitting* or *being open*, cannot be conceptualized has having steps of different types.

sense [16], that is, an entity that cannot occur (extend) over or throughout a temporal stretch. *Catch* events like the reception of a message, are in general exogenous, i.e., their happening is outside of the control of the pool they belong to or, at least, of the branch of the pool process at stake.[8] In this perspective *None* start events, e.g. the start event for *Purchaser* in Figure 1, could be understood as 'the system is turned-on'. In addition, being culminating perdurants the catch events are anti-cumulative. Anti-cumulativity and atomicity charactrize the subcategory of achievements in DOLCE.

In Figure 1, the process of *Service Provider A* cannot proceed unless a trigger occurs, i.e., unless a message is received. What are the implications from the perspective of the service provider? If the system of this service provider is 'turned-off', the message will never be received. Thus, behind a catch event there is the assumption that the process is waiting to be triggered, i.e., the system is on a receiving mode. Differently from activities, these kinds of perdurants (e.g., *waiting*) are *homeomeric*—i.e.,the temporal slices of waiting-instances (if any) are themselves waiting-instances—and *cumulative*—i.e., the mereological sum of two (immediately consecutive) waiting-instances is still a waiting-instance. Homeomeric and cumulative perdurants are called *states* in DOLCE. For instance, assume that the process is present but not evolving; this implies that at least one of the process participants needs to be in a position to catch the trigger. One can imagine a scenario in which a message-trigger is sent to a target pool but the latter is not expecting a message and misses it. In order for the receiver to become aware of an arriving message, it has to be in a receiving state. Of course, it is irrelevant whether the receiver is aware of this or else: the simple fact of having the phone turned on, or to be in the office at working hours makes the relevant agent a receiver. From an agent-based reading of BPMN, there seems to be a commitment towards the existence of certain states in which some participant is in a waiting status for a certain trigger. These, however, are not modeled in BPMN and this could be problematic. For example, Figure 1 indicates that *Service providers A, B* and *C* are (by default) in a waiting status for receiving messages. Thus, a catch event identifies (perhaps implicitly) a state and it further indicates that the pool is committed towards a specific trigger to occur. We think that making explicit these hidden states could solve some problems linked to the way the reception of a message (or signal) are managed.

One could go further and, following [19], consider catch events as state-transitions. For example, the reception of messages can be understood by referring to two states: the state of 'waiting for the message' and the state of 'message received', where the latter is the pre-condition for executing the successive task. The trigger thus enacts a state transition and, in turn, the starting of the new state enables the process to perform its subsequent tasks. In this case the graphical element of BPMN for an event would implicitly comprise these two states since it focuses on the transition itself.[9] In the case of *None* catch (start, intermediate) the trigger that is holding the process is not specified. From our viewpoint, there are at least two possible views regarding the semantics of this modelling construct. It might be a placeholder for the initial temporal boundary of the process that in our example corresponds, as a logical constraint, to 'there are no parts of the process that precede the *Request Quote* task'. In this case, the *None* catch bears no further ontological commitment. On the other side, one can return to the idea of a

[8] BPMN is quite permissive and some cases should be analyzed in detail; for instance, at least in principle, it is possible to have BPMN-models where an agent sends a message to herself.

[9] Also, the analysis of the causal dependencies among triggers, events and tasks could be very informative.

(hidden) waiting state. The latter case seems to be incompatible with the interpretation of the start event as 'the system is turned-on'.

3.3. Throw events

Similarly to catch events, throw (intermediate- or end-) events are instantaneous—then temporally atomic—and anti-cumulative, i.e., in DOLCE they are classified as achievements. Differently from *catch* events, throw events tend to be endogenous: actions under the control of the pool they belong to like the sending of a message. Note that even though tasks are atomic (assuming a given granularity), tasks are extended in time and therefore could be conceived as structured perdurants. Vice versa throw events are punctual, thus are intrinsically unstructured, a sort of state transitions.

Figure 1 includes end throw events only. The *None* end events in the *Supplier A, B, C* pools mean that these processes end without any defined result (aka throw trigger). In the *Purchaser* pool, the process ends with a *Terminate* event so that any running task in the process flow is ended, that is, interrupted and dismissed whenever (some part of) the execution flow reaches the *Terminate* event. This type of events can be understood as the achievement of the whole process; it indicates that all needed tasks have been executed and that those running are now irrelevant and must be abandoned, thus the process is completed. In a fashion similar to start events, the *None* end event can be interpreted either as an ontologically neutral placeholder in the model, as a logical constraint, or as an (ontologically committed) achievement. Note that the end events marked with a specific trigger icon, like *Message*, *Terminate*, *Signal* and *Error*, indicate an achievement as well but now the culmination point is qualified: the triggers that are specified in these cases (message, signal, termination and error) are amongst the participating entities of the achievement.

We conclude with some considerations on the reading of catch events as achievements since it seems more comprehensive and could explain why start, intermediate and end events are introduced and discussed simultaneously in BPMN. Consider the diagram in Figure 2. It shows two interacting pools, the *Book's author* and the *Editor*. Once the former has written a book and chosen an editor, it sends the book to the *Editor* by a *Message* event (throw intermediate event) labeled *Send Book* in the *Book's author* pool. This message is caught by the *Message* event (start event) in the pool *Editor*. The receipt of this message-trigger enables the *Review Book* task. After the *Editor* assesses the book, it notifies to the *Book's author* whether the book is accepted or rejected.[10] Once the *Editor* has sent the notification by an intermediate throw event, namely *Send acceptance* (*rejection*, resp.ly) *notification*, the *Editor* process ends. The *Book's author* receives the notification via a catch event, performs the *Read Notification* task, and terminates its process. From the processes' flow it is clear how the DOLCE achievement grasps the ontological nature also of the throw intermediate event.

4. Conclusions and future work

We focused on the ontological analysis of the BPMN notions of activity (task) and throw/catch event, and classified them within the DOLCE account of perdurant entities, in

[10] In this case, the choice is modelled by an exclusive gateway (diamond with *X*): only one of the two outgoing paths can be executed.

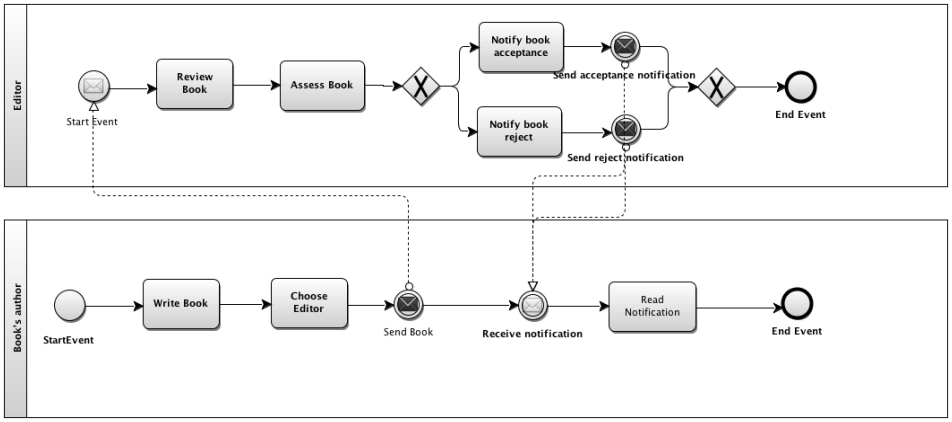

Figure 2. BPMN process diagram example 2

particular exploiting the distinction between accomplishments and achievements. BPMN activities are neither homeomeric nor cumulative. Their unifying criteria is the realization of specific business goals. Furthermore assuming an ontological-modality they are anti-atomic—in this case they can be mapped to accomplishments in the DOLCE framework—while assuming a conceptual-modality they are neither atomic nor anti-atomic—in this case they can be mapped to events in the DOLCE framework. BPMN throw events are atomic, anti-cumulative, and culminating perdurants, i.e., DOLCE achievements. BPMN catch events can be considered logical (or ontologically neutral) elements and indicate the existence of related cumulative and homeomeric perdurants, i.e. states in DOLCE. The results of our analysis are still preliminary. Yet the hope is that it can help to reach a new and deeper understanding of the system; to develop homogeneous tools to support business analysis; and to develop sound BPMN-driven ontologies.

In the future, we shall expand this first analysis and develop a formalization capturing our results on BPMN. Additionally, from the study of the BPMN metamodel, we believe that the concept of trigger has a relevant role in the standard: several BPMN event types are classified depending on this modelling construct. From the BPMN official specification [18], it seems that a trigger is an entity with causal powers on the process flow. Nevertheless, its meaning is not clear and some BPMN practitioners indeed either avoid the concept altogether [2], or do not characterise the notion beyond the BPMN definition [11]. Causality is a notoriously difficult relation to understand and model, so it is not surprising that the standard makes no further attempt to clarify what a trigger is. Some approaches to causality have been proposed in the context of applied ontology (e.g. [12,8]) and their direct application does not seem trivial. More work is required to analyze the notion of trigger, to investigate whether BPMN commits to causality and in which sense, and finally to understand whether existing ontological approaches to causality can clarify this commitment.

Acknowledgements: This research is in part supported by the Gecko and Pro2Evo projects of the "Fabbrica del Futuro" (funding: Ministero dell'Istruzione, dell'Università e della Ricerca) and by the IZSVE RC 09/12 "Il monitoraggio del TAT (Turnaround time) come strumento per migliorare l'efficienza complessiva del laboratorio" (funding:

Ministero del Lavoro, della Salute e delle Politiche Sociali). We thank Nicola Guarino and Nicola Zeni for fruitful discussions.

References

[1] W. Abramowicz, A. Filipowska, M. Kaczmarek, T. Kaczmarek Semantically enhanced Business Process Modelling Notation. In M. Hepp et al. (Eds.), *Semantic Business Process and Product Lifecycle Management. Proceedings of the Workshop SBPM 2007*, Innsbruck, April 7, 2007

[2] BPMN Modeling Reference. http://camunda.org/bpmn/reference.html. Last access March 2014

[3] J. de Bruijn, D. Fensel, U. Keller, M. Kifer, H. Lausen, R. Krummenacher, A. Polleres, L. Predoiu Web Service Modeling Language (WSML). W3C Member Submission 3 June 2005 Available at: http://www.w3.org/Submission/WSML. Last access March 2014

[4] A. De Nicola, M. Missikoff, and F. Smith. Towards a method for business process and informal business rules compliance. *Journal of Software: Evolution and Process*, 24(3):341–360, 2011.

[5] C. Di Francescomarino, C. Ghidini, M. Rospocher, L. Serafini, P. Tonella Reasoning on semantically annotated processes. Service-Oriented Computing, ICSOC 2008. Springer, Berlin Heidelberg, 2008.

[6] Remco M. Dijkman, Marlon Dumas, and Chun Ouyang. Semantics and analysis of business process models in BPMN. *Information and Software Technology*, 50(12):1281–1294, 2008.

[7] D.R. Ferreira *Enterprise Systems Integration. A Process-Oriented Approach*. Springer-Verlag Berlin Heidelberg, 2013

[8] A. Galton States, Processes and Events, and the Ontology of Causal Relations. In M.Donnelly, G., Guizzardi (Eds.), *Formal Ontology in Information Systems*. IOS Press, Amsterdam, 2012

[9] C. Ghidini, M. Rospocher, L. Serafini A Formalisation of BPMN in Description Logics. Technical Report TR 2008-06-004, FBK-irst, 2008.

[10] G. Guizzardi, G. Wagner Can BPMN Be Used for Making Simulation Models? 7th International Workshop on Enterprise & Organizational Modeling and Simulation (EOMAS 2011), together with the 23rd International Conference on Advanced Information System Engineering (CAiSE'11), London, UK.

[11] IBM Terminology. Available at: http://www-01.ibm.com/software/globalization/terminology. Last access March 2014

[12] J.Lehmann, S.Borgo, C.Masolo, A.Gangemi Causality and Causation in DOLCE. In A.C.Varzi, L.View (Eds.), *Formal Ontology in Information Systems*, IOS Press, Amsterdam, 2004

[13] C. Masolo, S. Borgo, A. Gangemi, N. Guarino, A. Oltramari. WonderWeb Deliverable D18. Ontology Library. Available at: http://wonderweb.man.ac.uk/deliverables.shtml, 2002. Accessed March 2014

[14] M. Missikoff, M. Proietti, and F. Smith. Linking ontologies to business process schemas. Technical Report 10-20, Istituto di Analisi dei Sistemi ed Informatica del CNR, 2010. ISSN: 1128 - 3378.

[15] D.L. McGuinnes, F. van Harmelen OWL Web Ontology Language. Overview. W3C Recommendation 10 February 2004. Available at: http://www.w3.org/TR/owl-features. Last access March 2014

[16] A.P.D. Mourelatos Events, Processes and States. In P.Tedeschi and A. Zaenen (Eds.), *Tense and Aspect*, pp.191-212. Academic Press, New York, 1981

[17] C. Natschlaeger. Towards a BPMN 2.0 Ontology. In R. Dijkman et al. (Eds.), *Business Process Model and Notation. 3rd Int. Workshop BPMN 2011*, Springer Heidelberg Dordrecht London New York, 2011

[18] Object Management Group (OMG) Business Process Model and Notation (BPMN). Version 2.0, 2011 Available at: http://www.bpmn.org Last access March 2014

[19] J. Recker, M. Indulska, M. Rosemann, P. Green. Do Process Modelling Techniques Get Better? A Comparative Ontological Analysis of BPMN. Australasian Conf. on Information Systems. Sydney, 2005

[20] T. Sider, *Four-Dimensionalism. An Ontology of Persistence and Time*. Oxford: Clarendon Press, 2001.

[21] W.M.P van der Aalst, K. van Hee *Workflow Management. Models, Methods and Systems*. The MIT Press, Cambridge, Massachusetts, London, England, 2002

[22] M. Weske *Business Process Management. Concepts, Languages, Architectures*. Springer Verlag, Berlin, Heidelberg, New York, 2007

[23] Workflow Management Coalition Terminology and Glossary. WFMC-TC-1011 - Issue 3.0, Feb. 1999. Available at: http://www.aiai.ed.ac.uk/project/wfmc/ARCHIVE/DOCS/glossary/glossary.html Last access March 2014