Business Processes and Their Participants: An Ontological Perspective

Greta Adamo[1,3(✉)], Stefano Borgo[2], Chiara Di Francescomarino[1], Chiara Ghidini[1], Nicola Guarino[2], and Emilio M. Sanfilippo[2]

[1] FBK-IRST, Via Sommarive 18, 38050 Trento, Italy
{adamo,dfmchiara,ghidini}@fbk.eu
[2] ISTC-CNR Laboratory for Applied Ontology, Trento, Italy
{stefano.borgo,nicola.guarino}@cnr.it, sanfilippo@loa.istc.cnr.it
[3] DIBRIS, University of Genova, via Dodecaneso 35, 16146 Genoa, Italy

Abstract. Business process modelling (BPM) notations, such as BPMN, UML-Activity Diagram (UML-AD), EPC and CMMN describe processes using a graphical representation of process-relevant entities and their interplay. Despite the wide literature on the comparison between different modelling languages, the BPM community still lacks an ontological characterisation of process elements, among which process participants, that is, the main entities involved in a business process. Purpose of this paper is to start filling this gap by providing an ontological analysis of business processes from the standpoint of process participants. In particular, by discussing participants common to languages such as BPMN, EPC, UML-AD, and CMMN we characterize them on the basis of their ontological properties.

1 Introduction

Business process modelling (BPM) notations describe processes using a graphical representation of process-relevant entities and their interplay. Examples include well known imperative languages such as BPMN, UML-Activity Diagram (UML-AD) and EPC together with declarative notations such as CMMN[1]. Despite the wide literature on both the investigation of execution semantics and the comparison between the graphical elements of different languages [9,11,13,22], the BPM community still lacks an ontological characterisation of process elements. While some efforts have been devoted to an ontological characterisation of specific modelling languages (see e.g., [18] for an investigation of the ontological commitments of activities and events in BPMN), this characterisation concerns only the behavioural component and neglects important structural entities which can

[1] Traditional process modelling notations rely on an imperative paradigm which aims at producing models that describe all allowed flows: every flow that is not specified in the model is implicitly disallowed. Recent declarative process modelling notations instead allow the production of more flexible models obtained by describing constraints on the allowed activity flows: all flows are allowed provided that they do not violate the specified constraints.

F. Esposito et al. (Eds.): AI*IA 2017, LNAI 10640, pp. 215–228, 2017.
https://doi.org/10.1007/978-3-319-70169-1_16

be modelled using the languages above, that is, process participants. As a result, process participants, such as actors or (data) objects, are exposed to a dichotomy: on the one hand they are among the main entities in a business process (diagram) and a fundamental component of an informative process model; on the other hand they are emblematically neglected when explaining or illustrating the very notion of process. In fact, for instance, what is the identity of a data object, i.e., whether different actors deal with the same or different data objects, or what is the status of a data object throughout the process execution, remain unclear.

The purpose of the paper is to provide an analysis of business processes from the standpoint of process participants. We first provide an illustration of different constructs used by imperative and declarative modelling notations (Sect. 2), and identify the ones that refer to process participants (Sect. 3). Then, by discussing the process participants common to the different notations, we dig into their ontological properties (Sect. 4). The analysis, and the subsequent characterisation of process participants within the different languages, provided in Sect. 5, is a first step toward the illustration of how an ontological analysis enlarged to process participants can support the interpretation of business process diagrams, the comparison between modelling notations, and the illustration of the different perspectives that BPM languages implicitly take on business processes.

2 Background

We briefly illustrate here the business process modelling languages taken into account throughout the paper. In order to aim for a general analysis and avoid possible biases due to the imperative/declarative nature of the modelling paradigms, we have selected three among the most popular languages that follow the imperative paradigm (BPMN, UML-AD, and EPC) and a notation that follows the declarative approach (CMMN). To support our brief description we make use of process diagrams illustrating a self explanatory scenario of a customer buying a flight ticket from a travel agency. Besides illustrating the scenario, the diagrams are "annotated" with speech balloons indicating the type of entity denoted by the graphical constructs.

BPMN. The BPMN[2] (Business Processing Modeling Notation) is a standard language, proposed by the Object Management Group (OMG) to design business processes. BPMN defines a Business Process Diagram (BPD) which includes a set of graphical elements divided in: (i) flow objects; (ii) data; (iii) connecting objects; (iv) swimlanes and (v) artefacts. The flow objects define the behaviour of a business process, as the one reported in Fig. 1. They are divided in *events*, *activities* and *gateways*. Events represent something that "happens" during the process and are divided in *start*, *intermediate* and *end* events. An activity is a generic term of work to be performed. An activity can be atomic (*task*) or compound (*sub-process*). A gateway determines the forking, merging and joining of

[2] http://www.omg.org/spec/BPMN/2.0/.

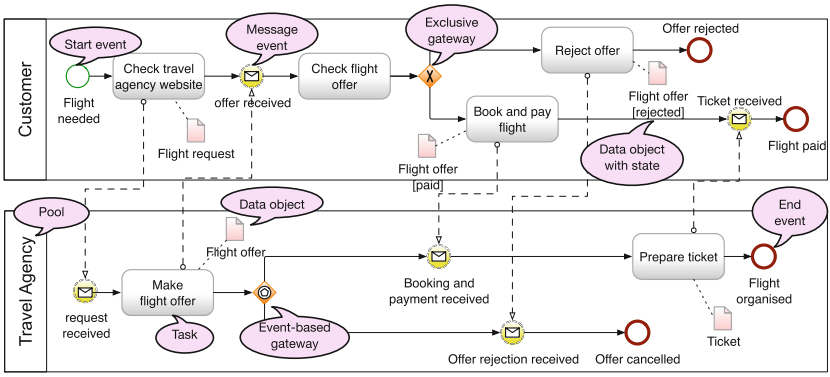

Fig. 1. A business process diagram in the BPMN language.

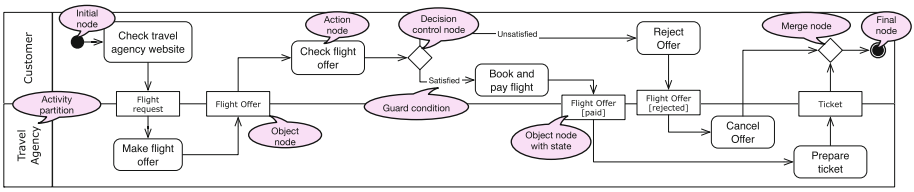

Fig. 2. A business process diagram in the UML AD language.

paths. In BPMN 2.0 information is represented through data, which include: *data objects*, *data inputs*, *data output* and *data stores*. The various flow objects are linked to each other through *connecting objects* which are not further discussed here. *Swimlanes* represent organisation units through pools and lanes, and they are usually used to answer the "who" question. BPMN provides further elements, called *artefacts*, to describe the context (or information) of the process. Artefacts are divided into groups and text annotations. Groups are useful to graphically cluster elements belonging to the same category; text annotations are used to specify additional textual information that can be valuable to the user of the diagram.

UML-ADs. UML Activity Diagrams[3] (ADs) are one of the diagram families of the OMG standardised language UML. Purpose of the activity diagrams, such as the one depicted in Fig. 2, is to describe the control and the data flow as a sequence of activity nodes connected by activity edges.

In detail, two main types of activity nodes are responsible of describing the control flow, i.e., the *action nodes* and the *control nodes*. While the former represent atomic steps within an activity, the latter allow for controlling the execution flow by means of the typical AND, OR and XOR logical operations. Two additional control flow nodes are used to depict the initial and final nodes. The

[3] http://www.omg.org/spec/UML/.

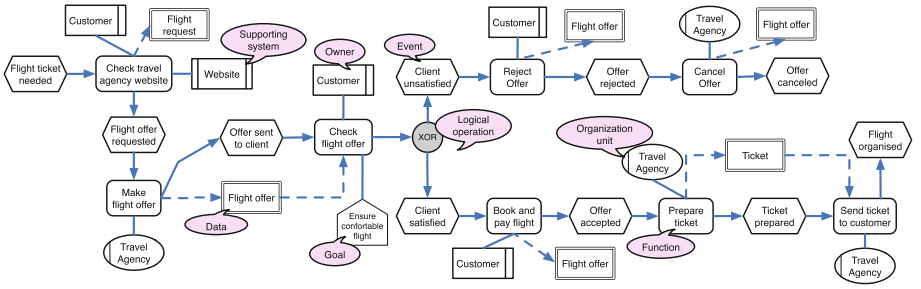

Fig. 3. A business process diagram in the EPC language.

intuitive semantics of AD can be explained in terms of *control flow tokens* flowing through the activity diagram, starting from the initial node and ending in the final node.

As for the data flow description, object nodes and object flows are, instead, the main ADs elements. Indeed, the object nodes represent objects at a given point of the flow and, as such, they can also have an associated *state*. Object flows are instead used for connecting the object nodes to the actions. In order to also capture the object flow semantics, besides the control flow tokens, object tokens are also introduced. They are similar to control flow tokens but also carry a reference to an object.

Furthermore, ADs provide a mechanism for grouping together activity nodes which have characteristics in common (*activity partitions*) mainly used as a means for defining organisational units. Finally, the notation allows for specifying activity pre- and post-conditions, for instance annotating activity edges with guards.

EPCs. Event-driven process chains (EPCs) are a workflow modelling language developed in the early 1990s as part of the Architecture of Integrated Information Systems (ARIS) framework [19].

In detail, three types of nodes are responsible of describing the control flow: the *function nodes*, the *event nodes* and the *logical operators nodes*[4] (see Fig. 3). While function nodes represent atomic activities and can thus be considered the active part of a control flow, event nodes represent the states in which the process happens to be, and can therefore be considered the passive part of the control flow. Functions and event alternate, capturing the intuition that states lead to activities (in a sort of pre-condition fashion), and activities generate states (in a sort of port-condition fashion). Finally, the XOR, AND and OR logical operators allow for controlling the execution flow.

Functions within the control flow can be connected to objects belonging to the other views of an ARIS model, namely the organisational, data, function

[4] The list of symbols of EPCs can vary, depending on the specific system implementation. The analysis and diagrams contained in this paper refer to the description provided in [20].

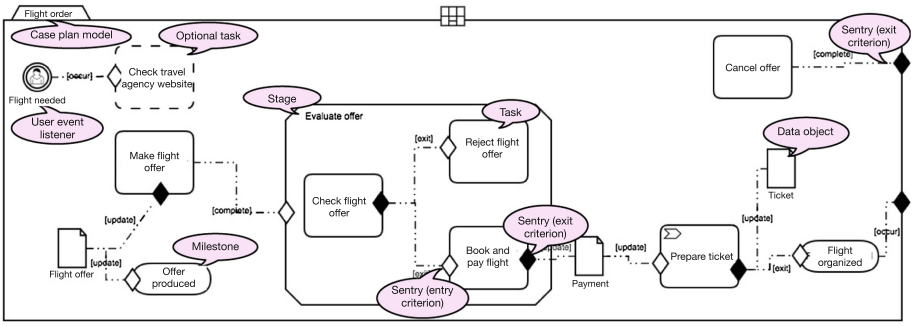

Fig. 4. A business process diagram in the CMMN language.

and product service views. While the number of objects differs from version to version, the core elements usually comprise: (i) input and output *data, material, services or resource objects* required or produced by a function; (ii) *owners* who are responsible for a specific function; (iii) *organisation units* (e.g., a department) responsible for a specific function; and (iv) *supporting systems* (e.g., a database) upon which the function acts. Depending on the version of the language, *goals*, denoted by house shaped pentagons, can be also connected to specific functions.

CMMN. The Case Management Model And Notation[5] (CMMN) language is a OMG standard notation for the declarative representation of process models. The main entity of CMMN is the *case*, which is described by a case diagram.

Differently from the previous languages, CMMN aims at capturing variable and flexible cases, following a declarative approach (see Fig. 4). Thus, rather than describing all the allowed flows of a process from start to end, it models cases as composed of process segments (*stages*) and *tasks*.

A case plan model, which is the component of CMMN devoted to the specification of the behaviour, contains: (possibly discretionary) *tasks*, *stages*, *milestones*, *event listeners*, *connectors*, and *sentries*. A task is a unit of work. Stages are plan fragments which can be composite or atomic. A milestone represents an accomplishment which occurs during the process of a case. Events can be related to: a case file (created, deleted, modified, and so on); tasks, stages and milestones (started, cancelled, finished, and so on); event listeners (timer and user event listener). Connectors are used to link different plan items. Finally sentries represent the entry and exit criteria for path items and can be used to direct the control flow using the AND and OR logical operators.

3 A Brief Comparison of Business Process Languages

We present here a short categorisation and comparative summary of the main elements of BPMN, UML-AD, EPC and CMMN. Modelling elements are grouped

[5] http://www.omg.org/spec/CMMN/1.1/.

Table 1. A comparison among modelling languages

		BPMN	UML-AD	EPC	CMMN
BEV	Func	Task Subprocess	Action node Activity	Function Process path	Task Stage
	Event	Start/End Intermediate Send/receive	Start/End node Accept event action Send signal action	–	Timer User Event Listener
	Flow	Gateway Sequence Flow Message Flow	Control node Control Flow Object Flow	Logical operators Control Flow Info Flow	Connector Sentry
	State	Guard on gateway	Guard on control node Pre- Post-condition on activity	Event Start/End event	Sentry Milestone
DT		Data input, data output, data store	Object node	Taxonomy of (I/O) data object	Case file item
ORG		Pool, Lane	Activity Partition	Organization Activity Owner	–

into the three basic categories of process modelling languages, namely the *behavioural* (BEV) category related to the control flow, the *data* (DT) category related to the data flow, and the *organisational* (ORG) category related to the "who question". Being the behavioural section the most articulated, we further describe it in terms of *Functional* (the executable pro-active actions), *Event* (what happens), *Flow* (how elements are connected and routed), and *State* (of the world) categories. The result of this grouping is summarised in Table 1.

First, we can observe that all the imperative languages, namely BPMN, EPC, and UML-AD, provide distinctive elements to indicate the start and the end of a process. CMMN, instead, is focused on the representation of flexible workflows and therefore does not force a specific start and a specific end, but only exiting conditions. Not surprisingly all four languages have graphical symbols for atomic activities. Similarly, subprocesses and generic groups of activities are foreseen in all languages but EPCs. Other common elements are routing nodes, connectors, and data objects. Routing nodes route the control flow using the typical XOR, OR and AND logical constructors. While CMMN does not have explicit graphical symbols for gateways, its specification indicates how to use sentries and connectors to represent them. Connectors are instead typically used to connect the various graphical elements indicating the flow of the process. BPMN and EPCs augment connectors with special symbols to denote connections different from the control flow, namely, the connection between actors (data) and an activity, or the messages exchanged between different activities. Also, the granularity and level of detail of data objects can vary, with EPCs particularly rich in defining a taxonomy of data objects to be used while modelling. Alternative (OR, XOR) routing nodes can incorporate guards, that is conditions that specify which branch to follow, in all the languages but EPCs (where this role can be taken by states). "Actors" and organisational entities who "own"/"perform" parts of a process are another rather common element, only absent in CMMN case plan model.[6]

[6] CMMN allows to associate organisational entities to cases during the run-time phase.

4 Process Participants: An Ontological Analysis

Roughly speaking, the agreement across the literature [13] about what a business process is boils down to this: given a goal, a process is a set of actions that, together, contribute to achieving that goal. Although in the literature one finds that the type and the token levels are often mixed,[7] the previous definition is about process types: the goal is a description of a desired state, e.g., that a certain product is assembled or an ordered item is shipped, whereas the actions are event types like sending a message or identifying a customer. The way actions contribute to the achievement of the goal is not commonly made explicit in notations like those considered in the previous sections,[8] but there is an implicit assumption that the process clarifies at least the sequence of actions to be performed, or the possible alternative sequences. The interpretation of the precedence relation in the sequence is however left open, and one can read it in terms of temporal, causal or dependence constraints, perhaps depending on context.

The above generality is common in application domains where large communities have to agree on a common language that mediates among different perspectives and interests. One consequence is that it is unclear what the identity and unity conditions for a process are, that is, when two process definitions actually define the same process, or when an action should be seen as part of a process. Although here we concentrate on process modelling, these problems are pretty general and affect also process mining tasks: in absence of unity and identity principles, it is hard to decide which actions should be registered in a single process log.

The strategy we propose to ground the unity and identity of business processes relies on their *participants*. The intuition is that if two processes have different participants, they are different processes. Also, we assume that if two actions have some participants in common, then—under suitable constraints—they may belong to the same process.

Consider, for example, a process type *pty* composed of two different actions act_1 and act_2 such that act_1 precedes act_2. Let us say that act_1 is *create form* and act_2 is *send form*. Even ignoring whether act_1 and act_2 belong to the same process type, or whether there is a precedence constraint between them, we may reconstruct this information by knowing that the two actions involve the same form, and that no form can be sent before it was created. Generalising from the example, the changes of (or in) a process participant may provide information to identify processes, establish the correct relations between actions and decide when different actions are part of the same process.

[7] 'Token' is hereby synonym of a process *occurrence* (an instance of a process type). While a process token occurs at a specific time, a process type is an abstract entity with no specific temporal location (see the distinction between `Activity` (type) and `ActivityOccurrence` (token) in the *Process Specification Language* (PSL) [7]).

[8] As a matter of fact, a language like BPMN does not force modellers to explicitly represent what changes in the (local) world are expected after an activity is performed.

To investigate business processes from this viewpoint, we firstly need to clarify what is an action and what are its participants. Recall that in the BPM literature actions (at the token level) refer to intentional transformations from some initial state (of the local world at stake) to some other state. Their participants are the entities that take part in the transformations. In the terminology of [2], action tokens are *events*,[9] while their participants are *objects*. As we shall see, the very same action may involve several types of objects as participants: physical objects (e.g., a knife used to cut a piece of bread); information objects (e.g., personal data involved in submitting a request); agents and/or organisations playing certain roles (e.g., an administrative employee receiving a form). Turning back to the four modelling languages described in Sects. 2 and 3, examples of participants are the entities denoted by means of the constructs classified under the data (DT) and the organisational (ORG) categories in Table 1.

From a general perspective, participants can be physical or non-physical: both exist in time, but only the former are located in the physical space. A person is an example of physical participant, whereas an *information object* such as the content of a person's ID (not its physical support) is a non-physical participant.

Information objects (a.k.a. *data objects*) are rather common in business processes and, as seen in the previous sections, modelling notations include different constructs for them. In applied ontology, only a few systems [1,12,14,21] have attempted a formal treatment of information objects. These ontologies agree in distinguishing between information objects and their physical carriers like paper sheets or computer files; also, the same information object may be encoded in multiple carriers while retaining its identity. For example, John's and Mary's copies of the *Divine Comedy* are two different carriers of the same information object. Generalising, we consider an information participant as a non-physical participant that is somehow 'manipulated' during a process. Additionally, we consider information participants as *dependent* entities that, in order to participate in a process, have to be encoded in at least one carrier. Note also that all the actions performed within a particular process occurrences are ultimately physical actions involving physical participants, so that information objects are actually *indirect* participants, which participate to the process by means of *information-bearing objects* containing their physical encodings.

Regarding physical participants, we may distinguish between material and immaterial entities. Material participants do have some physical body (e.g., a human body or a metallic frame), differently from immaterial objects like holes, which in some cases may still be considered as participants (e.g., in a process including a pin to be inserted in a hole). Holes belong to the broader class of *features*, which are dependent entities like information objects.

Another crucial distinction in BPM is between *agentive* and *non-agentive* participants. The former are indirectly represented in BPMN, whose pools and lanes refer to participants that are committed to and are responsible for the execution of the depicted process. Notoriously, the definition of agency is largely debated in AI. For our purposes, we take the view that an agent is an entity with

[9] 'Event' is the most general term used in [2] for entities occurring in time.

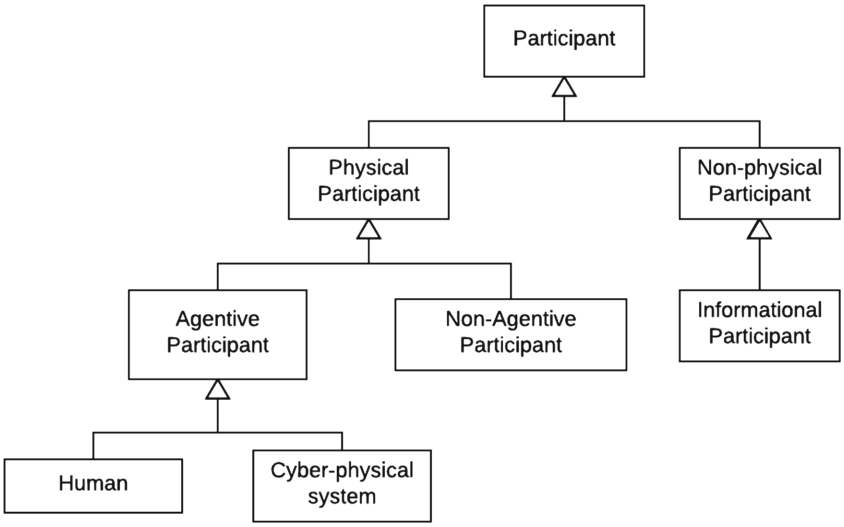

Fig. 5. Taxonomy of participants

sensors, actuators and the capability to act on itself or on the environment [17]. Human beings and organisations, such as those denoted with 'customer' and 'travel agency' in Fig. 2, are clearly agents. In a manufacturing domain, a lathe machine is an agent when, e.g., it has sensors by which it acquires data from the objects to be manufactured and acts upon them by elaborating these data through some software. So, in general, a cyber-physical system is an agent, while a traditional mechanical late is not an agent.

A minimal UML taxonomy of process participants based on some of the distinctions discussed so far is reported in Fig. 5. Notice that the agentive/non-agentive dichotomy only applies to physical participants, since we assume that non-physical lack the capability of interacting with the environment. Note also that some of the distinctions discussed above are orthogonal to those shown in the figure, so that they have not been reported explicitly. In particular, we assume that all agentive participants are material, while non-agentive participants may be either material or immaterial. Moreover, all physical participants may (or may not) be information-bearers.

Apart from the classification above, all participants can play *roles*. Non-agentive participants may be distinguished according to whether they undergo a change during an action. If so, they are called the *patients* of the action; otherwise, they may play the role of *instruments* or *resources*.

From an ontological perspective, roles correspond to properties that objects only contingently satisfy within the process context, like being used as a resource during a drilling process. In this sense, one object can loose or acquire a role while remaining the same entity. We assume that roles can be ascribed to any type of participant represented in Fig. 5, including information objects.

Finally, note that we rely on a general notion of participant covering any object that takes part in a process. One may however restrict this notion only to "relevant" objects. For instance, considering the flight purchase process of the previous sections, one may not want to consider as participants the computers that the customer and the travel agency use to perform their activities. In this sense, the relevant participants of a process are those that are directly related to the desired goal, which are typically common to multiple activities. This seems indeed to be the idea behind the notion of *business artifacts* [3], which may be intended as process participants that are passed by from an activity to another, somehow keeping track of what happens as long as the process goes on. In this spirit, the flight offer shown in Figs. 1 and 2 can be understood as a data object that can undergo different states (e.g., *requested, payed, rejected*) depending on how the process evolves.

5 Discussion

In this section, starting from the analysis of participants previously presented and looking at the diagrams in Figs. 1, 2, 3 and 4, we provide some insights on the modelling notations and the perspectives they take on the processes, focusing on the data and organisational constructs of Table 1.

The participants relevant in the flight purchase example are the customer, the travel agency and different information objects. The process thus includes different types of participants, material and immaterial ones.

By looking at the diagrams, we observe that no actors appear in CMMN and neither BPMN nor UML-AD specify whether 'customer' explicitly refers to a single individual (e.g., John) or to an organisation. In both cases, this individual is playing the *role* of a participant, who desires to book a flight ticket. This consideration reveals, besides the fact that CMMN lacks graphical constructs for specifying actors, the underspecification of both BPMN and UML-AD with respect to our analysis, since pools and activity partitions can't distinguish between different types of participants, nor between the participants themselves and the roles they play. Differently, the distinction between single actors and organisations can be explicitly conveyed in EPC, although the difference between participants and their roles is blurred.

In Figs. 1, 2 and 3, *Check travel agency website* results in a *Flight request* which is sent to the agency. On the basis of our analysis, such request is an *information-bearing object*, since what the customer sends to the agency is a (copy of a) physical object encoding an information object. The distinction between information objects and their physical support is not explicitly addressed by the languages we considered. On the one hand, it seems clear that when different agents share a data object, a certain file (support) is exchanged. On the other hand, it is implicitly assumed that the file displays some information.

According to the diagrams, when an instance of *Make flight offer* is accomplished, the new information object *Flight offer* is generated and sent to the

customer. We may wonder whether the customer and the agency handle the same flight offer. None of the graphical notations provide a means to address this issue.

When the flight offer is received by the customer, she checks the offer and decides either to reject, or to accept it and hence to proceed with the booking of the flight. This means that the offer—under the control of the customer—undergoes some update along with the execution of the tasks; e.g., it acquires new properties, namely, that of *having been accepted* or *rejected*. As we can see from the diagrams, indeed, the *Reject offer* and *Book and pay flight* tasks create further information objects, which can be understood as updates of the offer. At the same time, however, the flight offer handled by the agency remains "frozen", since it is updated with the customer's information only when the customer shares it with the agency. More precisely, during the time interval from the *Check flight order* to either *Reject offer* or *Book and pay flight*, the customer's offer and the agency's offer share some part in common but the former has more information than the latter. Once either the event *Booking and payment received* or *Offer rejection received* happens, the two information objects are again the same.The agency indeed receives the customer's order and updates its information with the customer's information.

Here we observe that BPMN (and in part UML-AD) offers the possibility to model the status of the data objects, e.g., the *Flight offer* being *paid* or *refused*. On the contrary, in EPC and CMMN this cannot be explicitly modelled, although it can be inferred by looking at the changes of the world. From a more general perspective, BPMN and EPC separate the data from the control flow (they have explicit different graphical notations for the two flows). Differently, in UML-AD and CMMN, data objects are represented in a unique flow with activities and control flow elements, so that the process execution cannot proceed unless the data object is processed/available. In this sense, data object participants play a fundamental role within the overall process, and it becomes necessary to properly identify which data objects the process manipulates.

To conclude, the analysis of participants needs to be extended to shed some light on the ontological characterisation of activity sequences, as well as to identify the different modelling approaches in the languages at hand. Once we recognise the changes that participants undergo in the context of a process, indeed, we can better understand how activities are related (e.g., via precedence constraints) in order for those changes to take place. This latter topic however deserves more attention and is left for future work.

6 Related Work

A number of works in the literature focused on the analysis and comparison of process languages and notations [11,22], as well as on the definition of a shared reference metamodel [9,13] for process description at an informal level. Several works at the intersection between knowledge engineering and business process modelling (e.g., [10,23]) focus on formal techniques aimed at verifying the consistency of process models, as well as their smooth execution.

Focussing on *ontology-based* business process modelling, which is the context of our paper, disparate ontologies have been proposed to semantically enrich process models. Among these, some ontologies axiomatise the properties that graphical elements satisfy according to modelling notations. In [16], for example, the authors present an OWL-based representation of BPMN that is used for reasoning on the consistency of the process models [5] and for the management of exceptional flows [6]. In a more general setting, De Nicola and colleagues [4] propose an upper-level ontology for business processes. In both these works, however, the authors do not attempt a clarification of the modelling notations on the basis of some reference ontology.Some initial works towards the analysis of BPMN based on foundational ontologies are presented in [8,15,18]. These however focus only on some modelling elements, i.e., activities and events, while leaving aside the analysis of participants, which is the focus of the presented work.

7 Conclusions and Future Work

In this work we focused on an ontology-based analysis of the properties characterising the process participants common to the main process modelling languages and notations. In the future we plan to extend such a preliminary analysis in order to deepen the investigation on the ontological commitments of process participants by further inspecting the different perspectives that BPM languages implicitly take on business processes, as well as providing modellers with guidelines to make an appropriate choice when selecting among different notations.

By observing Figs. 1, 2, 3 and 4 and Table 1, we can notice that differences exist also on the behavioural component. As an example, a key difference among the languages we considered concerns the representation of the (state of the) world in response to a process execution. Figure 3 emphasises this as one of the focuses of EPCs. UML-AD and CMMN lie in the middle by exploiting data objects and sentries for describing how the world is changed because of the process execution. For instance, in Fig. 4 the *Flight Offer* is the postcondition of the *Make flight offer*. BPMN, instead, only provides (optional) constructs for representing the state of data objects. From an ontological perspective, we would say that, differently from BPMN, EPC drives the modeller to explicitly represent the world's states affected by the designed process, while UML-AD and CMMN guide the modeller to implicitly represent the world's states through data objects and sentries. A further example concerns the relation between activities, and more in general the way the activities contribute to the achievement of the goal. For instance, assume that in a slight variation of the example of Sect. 2, the travel agency splits the activity *Make flight offer* in two subsequent steps *Send flight offer to customer* and *Archive offer* which, for purely organisational reasons, must be executed in this order. This would be a pure temporal relation between the activities in this specific setting. Instead the activity of *Paying for a flight* causes the *Preparation of the ticket*. Nonetheless these relations would be denoted by means of the same connector symbol. From an ontological perspective, we

would say that all the languages we considered here do not guide the modeller to represent different types of precedence relations. Recognising the changes that participants undergo in the context of a process, and connecting them to the way activities are related, is another topic that deserves specific attention and is left for future work.

Acknowledgments. This research has been partially carried out within the Euregio IPN12 KAOS, which is funded by the "European Region Tyrol-South Tyrol-Trentino" (EGTC) under the first call for basic research projects.

References

1. Bekiari, C., Doerr, M., Le Boeuf, P., Riva, P.: FRBR object-oriented definition and mapping from FRBRER, FRAD and FRSAD (version 2.4). International Working Group on FRBR and CIDOC CRM Harmonisation (2015)
2. Borgo, S., Masolo, C.: Foundational choices in DOLCE. In: Staab, S., Studer, R. (eds.) Handbook on Ontologies. IHIS, pp. 361–381. Springer, Heidelberg (2009). doi:10.1007/978-3-540-92673-3_16
3. Cohn, D., Hull, R.: Business artifacts: a data-centric approach to modeling business operations and processes. IEEE Data Eng. Bull. **32**(3), 3–9 (2009)
4. De Nicola, A., Lezoche, M., Missikoff, M.: An ontological approach to business process modeling. In: 2007 3th Indian International Conference on Artificial Intelligence (2007). ISBN 978-0-9727412-2-4
5. Di Francescomarino, C., Ghidini, C., Rospocher, M., Serafini, L., Tonella, P.: Semantically-aided business process modeling. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) ISWC 2009. LNCS, vol. 5823, pp. 114–129. Springer, Heidelberg (2009). doi:10.1007/978-3-642-04930-9_8
6. Ghidini, C., Di Francescomarino, C., Rospocher, M., Tonella, P., Serafini, L.: Semantics-based aspect-oriented management of exceptional flows in business processes. IEEE Trans. Syst. Man Cyber. Part **42**(1), 25–37 (2012)
7. Grüninger, M.: Using the PSL ontology. In: Staab, S., Studer, R. (eds.) Handbook on Ontologies. IHIS, pp. 423–443. Springer, Heidelberg (2009). doi:10.1007/978-3-540-92673-3_19
8. Guizzardi, G., Wagner, G.: Can BPMN be used for making simulation models? In: Barjis, J., Eldabi, T., Gupta, A. (eds.) EOMAS 2011. LNBIP, vol. 88, pp. 100–115. Springer, Heidelberg (2011). doi:10.1007/978-3-642-24175-8_8
9. Heidari, F., Loucopoulos, P., Brazier, F.M.T., Barjis, J.: A meta-meta-model for seven business process modeling languages. In: IEEE 15th Conference on Business Informatics, CBI 2013, pp. 216–221. IEEE Computer Society (2013)
10. Lam, V.S.W.: Formal analysis of BPMN models: a NUSMV-based approach. Int. J. Softw. Eng. Knowl. Eng. **20**(7), 987–1023 (2010)
11. List, B., Korherr, B.: An evaluation of conceptual business process modelling languages. In: Proceedings of the 2006 ACM Symposium on Applied Computing, pp. 1532–1539. SAC 2006. ACM, New York (2006)
12. Masolo, C., Vieu, L., Bottazzi, E., Catenacci, C., Ferrario, R., Gangemi, A., Guarino, N.: Social roles and their descriptions. In: Dubois, D., Welty, C., Williams, M. (eds.) Principles of Knowledge Representation and Reasoning, pp. 267–277. AAAI Press, Palo Alto (2004)

13. Mili, H., Tremblay, G., Jaoude, G.B., Lefebvre, E., Elabed, L., Boussaidi, G.E.: Business process modeling languages: sorting through the alphabet soup. ACM Comput. Surv. **43**(1), 4:1–4:56 (2010)
14. Mizoguchi, R.: Yamato: yet another more advanced top-level ontology. In: Proceedings of the Sixth Australasian Ontology Workshop, pp. 1–16 (2010)
15. Recker, J., Indulska, M., Rosemann, M., Green, P.: Do process modelling techniques get better? A comparative ontological analysis of BPMN. Australasian Chapter of the Association for Information Systems (2005)
16. Rospocher, M., Ghidini, C., Serafini, L.: An ontology for the business process modelling notation. In: Garbacz, P., Kutz, O. (eds.) Proceedings of the 8th International Conference on Formal Ontology in Information Systems (FOIS 2014). Frontiers in Artificial Intelligence and Applications, vol. 267, pp. 133–146. IOS Press, Amsterdam (2014)
17. Russell, S., Norvig, P.: Intelligence Artificial: A Modern Approach. Prentice-Hall, Egnlewood Cliffs (1995)
18. Sanfilippo, E.M., Borgo, S., Masolo, C.: Events and activities: is there an ontology behind BPMN?. In: Garbacz, P., Kutz, O. (eds.) Proceedings of the 8th International Conference on Formal Ontology in Information Systems (FOIS 2014). Frontiers in Artificial Intelligence and Applications, vol. 267, pp. 147–156. IOS Press, Amsterdam (2014)
19. Scheer, A.: ARIS - vom Geschaftsprozess zum Anwendungssystem. Springer, Berlin (2002). (4, durchges. aufl. edn.)
20. Scheer, A.W., Thomas, O., Adam, O.: Process modeling using event-driven process chains. In: Dumas, M., van der Aalst, W.M.P., ter Hofstede, A.H.M (eds.) Process-Aware Information Systems: Bridging People and Software Through Process Technology, pp. 119–146. Wiley, October 2005
21. Smith, B., Ceusters, W.: Aboutness: towards foundations for the information artifact ontology. In: Proceedings of the International Conference on Biomedical Ontology (ICBO) 2015 (2015)
22. Söderström, E., Andersson, B., Johannesson, P., Perjons, E., Wangler, B.: Towards a framework for comparing process modelling languages. In: Pidduck, A.B., Ozsu, M.T., Mylopoulos, J., Woo, C.C. (eds.) CAiSE 2002. LNCS, vol. 2348, pp. 600–611. Springer, Heidelberg (2002). doi:10.1007/3-540-47961-9_41
23. Wong, P., Gibbons, J.: Formalisations and applications of BPMN. Sci. Comput. Program. **76**(8), 633–650 (2011)