Towards Object-centric BPMN Process Models

Anjo Seidel[1,*], Maximilian König[1] and Mathias Weske[1]

[1]Hasso Plattner Institute, University of Potsdam, Prof.-Dr.-Helmert-Str. 2-3, 14482 Potsdam, Germany

Abstract

BPMN, the de facto standard for business process modeling, is designed to model activities and control flow for a specific case notion. The new paradigm of object-centric process mining proposes to discover process models that are not bound to one specific case notion, but rather define a process as the combined behavior of the involved data objects. In that context, different process modeling approaches aim to depict object-centric behavior. However, the object-centric capabilities of BPMN have not yet been evaluated. Therefore, this paper investigates to what extent BPMN supports object-centric modeling features from literature. We show that the semantics of BPMN process models are insufficient to depict object-centric processes. To overcome the identified limitations, we rely on concepts from prior work on case management to propose the new notion of object-centric BPMN process models.

Keywords

Business process modeling, Object-centric processes, BPMN 2.0, fragment-based Case Management, Object-centric Petri nets with Identifiers

1. Introduction

The OMG standard Business Process Model and Notation (BPMN) is well-established in practice and the most widely used process modeling language [1]. Like most traditional activity-centric process modeling approaches, BPMN relies on a single case notion [2], i.e., a specific perspective on the process, to define control flow relations between activities that are relevant for reaching business goals [3]. To represent the interaction of activities with business objects, data objects can be modeled.

The new trend of object-centric process mining proposes a more comprehensive perspective on business processes by identifying relevant business objects and their individual behavior [2]. Based thereon, the different objects can be correlated through activities that operate on sets of objects. Currently, most approaches to define and visualize "object-centric process models" are based on Petri nets [4], while more high-level process modeling approaches have been introduced under the term "data-centric process models". An overview of the latter is presented by Steinau et al. [5]. Van der Aalst [6] already motivates the refinement of BPMN to better capture object-centric behavior. Yet, the current capabilities of BPMN are not analyzed, and possible adjustments are not specified.

This paper analyzes object-centric capabilities of BPMN by referring to requirements and characteristics from literature. Our investigation shows that BPMN lacks flexibility and precise semantics for identifying and correlating data objects. This indicates the need for object-centric BPMN process models (OC-BPMN); an approach that capitalizes on BPMN's comprehensibility and the holistic behavior represented in contemporary object-centric process modeling approaches. To model OC-BPMN process models, this paper proposes a graphical modeling language combining BPMN process fragments with a data model. It bases its concepts on prior work on fragment-based case management [7].

In the remainder of this paper, we first discuss related work in section 2. We analyze the object-centric capabilities of the status quo of BPMN in section 3, before introducing the novel OC-BPMN process models and informally describing their semantics in section 4. Finally, section 5 discusses the proposed approach and concludes the paper.

Proceedings of the Best BPM Dissertation Award, Doctoral Consortium, and Demonstrations & Resources Forum co-located with 22nd International Conference on Business Process Management (BPM 2024), Krakow, Poland, September 1st to 6th, 2024.

*Corresponding author.
anjo.seidel@hpi.de (A. Seidel); maximilian.koenig@hpi.de (M. König); mathias.weske@hpi.de (M. Weske)
0000-0002-9652-5340 (A. Seidel); 0000-0002-2244-1179 (M. König); 0000-0002-3346-2442 (M. Weske)

2. Related Work

The object management group defines the standard BPMN 2.0 [8], which is one of the most widely used process modeling approaches [1]. To provide clear execution semantics, Dijkman et al. [9] provide translational semantics of control flow structures to Petri nets without considering data objects at all. Building thereon, mappings to Petri nets considering data objects and states [10] and BPMN input and output sets [11] have been proposed. Making use of the additional expressiveness of colored Petri nets, Ramadan et al. cover more complex BPMN constructs such as subprocesses and multi-instance behavior, but mostly neglect complex data interactions with these elements [12]. Extending on BPMN, Meyer et al. [13] introduce the concept of foreign keys between data objects and translate data pre- and postconditions to SQL queries, thus assigning meaning to data objects and data associations. König and Weske provide formal execution semantics for multi-instance behavior on data [14].

Building on the notation of BPMN, Hewelt et al. propose fragment-based case management [7], a hybrid modeling approach that combines imperative control flow between activities and declarative data flow between process fragments. The semantics are formalized as colored Petri nets by Haarmann et al. [15].

Existing high-level process modeling approaches already aim to represent object-centric behavior. Steinau et al. [5] provide an overview and a framework to systematically analyze the capabilities of data-centric process modeling approaches. For instance, Philharmonic Flows [16] define object-centric behavior, which can be transformed into BPMN 2.0 [17].

More recent process modeling approaches extend on Petri nets to represent object-centric processes. Gianola et al. [4] provide an analysis identifying object-centric modeling features. Synchronous Proclets, as presented by Fahland [18], allow describing the behavior of different object types. Transitions referring to events that interact with different object types are associated. Evolving from ν-Petri nets [19], Petri nets with identifiers (PNID) [20] support the identification of individual objects and their links. In the context of process mining, van der Aalst and Berti propose object-centric Petri nets (OCPN) [21] that define variable arcs to transfer a variable number of tokens at once but lack object identification. Combining the concepts of OCPNs and PNIDs, Gianola et al. [4] propose object-centric Petri nets with identifiers (OPID) that overcome the limitations of the prior Petri net approaches. Analog to PNIDs, they represent object identities and object relations, while variable arcs can transfer a variable number of object identities.

3. Object-centricity in BPMN process models

The previous section outlines contemporary object-centric process modeling approaches. To investigate the capabilities of BPMN, we refer to a set of modeling features for object-centric process models as described by Gianola et al. [4]. The results of comparing OCPN [21], PNIDs [20], OPIDs [4], BPMN [8] and the novel OC-BPMN are presented in Table 1. Furthermore, the issues of convergence and divergence [2] are discussed from a process modeling perspective.

3.1. Object-centric process modeling features

Object-centric process models need to allow for *object creation* and *object deletion*, e.g., in the order management process, it should be possible to create new orders and also to delete them. The existing Petri net-based data semantics for BPMN [10, 11] support object creation. Object deletion can be modeled implicitly through final data states. An order can be considered *completed* and is therefore unavailable for further processing. The modeling feature *multi-object spawning and transfer* describes the capability to create and operate on lists of data objects. As described by König et al. [14], BPMN is capable of iteratively creating items for an order. They are added to a list, just like in OPIDs.

BPMN is not yet able to represent *object references* to distinguish individual object instances. In existing formalizations, objects are represented as anonymous tokens. In the example, orders and items cannot be identified via IDs. Hence, their links cannot be identified either and *object relations* are not supported. *Synchronization* describes the ability to remember the object relations throughout the process. Items of different orders should not be mixed up. As BPMN cannot represent links between objects, there are no means to correlate objects at different points of the process.

Concurrent object flows describe the ability to process objects concurrently and independently of each other. BPMN supports this feature only partially through multi-instance subprocesses. Given a list data object, each element can be processed by a different subprocess instance concurrently. Yet, the parent process needs to wait for the termination of all subprocess instances.

Table 1
Overview of supported (✓), unsupported (✗) and partially supported (~) modeling features for object-centric process modeling approaches (adapted from [4]).

	object creation	object removal	concur. object flows	multi-object transfer	object relations	synchro-nization	object reference
OCPN	✓	✓	✓	✓	✗	✗	~
PNID	✓	✓	✓	~	✓	~	✓
OPID	✓	✓	✓	✓	✓	✓	✓
BPMN	✓	✓	~	~	✗	✗	✗
OC-BPMN	✓	✓	✓	✓	✓	✓	✓

3.2. Convergence and Divergence in BPMN

In addition to the elaborated modeling features, we investigate the issues of convergence and divergence, as described by van der Aalst [2] in the context of process mining.

Divergence occurs in one-to-many relations between the case object, which determines the case notion, and other related objects. As illustrated in Figure 1 (left), the case notion of the order requires a multi-instance behavior for all related items.

Convergence, on the other hand, describes the issue that a case notion is the subject of a many-to-one relation. BPMN cannot concisely represent multiple process instances that share an activity. In the case notion of an item in Figure 1 (right) the activity *place order* is executed for every item, but from an order perspective, it processes multiple items at once. Traditionally, batch processing [22] aims to address this issue. These limitations stemming from case-centric process models prevent truly concurrent object flows.

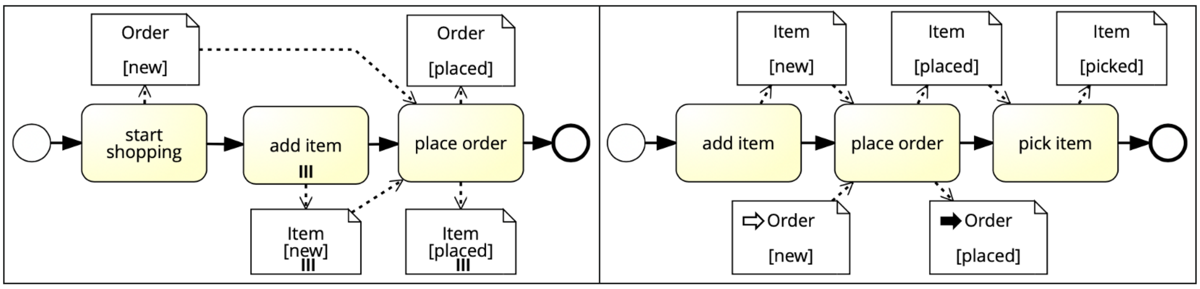

Figure 1: An excerpt of an order management process as BPMN process diagrams in the case notion of an order (left) and an item (right).

To overcome the limitations of traditional BPMN, we propose object-centric BPMN (OC-BPMN), which bases its concepts on fragment-based case management (fCM) [7]. In this approach, the traditional monolithic control-flow structure of BPMN is split into process fragments that each represent an individual object perspective. As later shown in Figure 3, the order management process can be modeled as three process fragments, one for the case notion of an order, one for items, and one for routes. During execution, these fragments can run repeatedly and concurrently and are only constrained by the availability of data objects. The use of fragments also overcomes the need for sub-processes to handle divergence and batch activities for convergence.

4. Object-centric BPMN Process Models

To overcome the limitations of traditional BPMN as discussed in section 3, we propose object-centric BPMN (OC-BPMN), which bases its concepts on fragment-based case management (fCM) [7]. In this approach, the traditional monolithic control-flow structure of BPMN is split into process fragments that each represent an individual object perspective. For fCM, a translational semantics to colored Petri nets defines object identities, and their correlation [15]. As shown in Figure 3, the order management process can be modeled as three process fragments, one for the case notion of an order, one for items, and one for routes. During execution, these fragments can run repeatedly and concurrently and are only constrained by the availability of data objects. OC-BPMN combines four modeling artifacts: (i) a data model defining the available object types and their associations, (ii) object lifecycles for each object type that describe the states and state transitions of data objects, (iii) process fragments with a BPMN syntax, and (iv) an object diagram as a representation of the current data state of a running OC-BPMN.

4.1. Data Model

Analog to OPIDs and OCELs, OC-BPMN defines a set of possible data object types as a data model. For the example order management process, the UML class diagram [23] on the left-hand side of Figure 2 describes the classes for orders, items, and routes. Additionally, it defines associations between classes. An order can have many items, and many items can be correlated to one route.

In line with OPIDs [4], we restrict our class diagram to binary associations and prohibit many-to-many relations. Many-to-many relations can be represented using association classes and reification. Note that orders, for example, have an implicit many-to-many relation to routes.

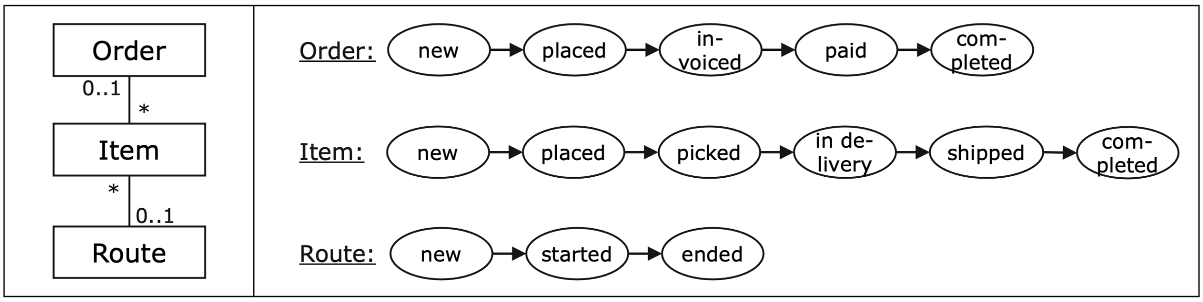

Figure 2: The UML class diagram (left) and the respective object lifecycles for each class (right) for the order management process.

4.2. Object Lifecycles

Analog to BPMN [13] and fCM [7], OC-BPMN specifies an object lifecycle (OLC) for each data class. An OLC defines a set of states as an abstraction of concrete attributes or processing states. Also, state transitions are defined. According to the OLCs in Figure 2, an order can be *new*, then *placed*, *invoiced* before being *paid*, and finally *completed*. In the same manner, the available states and transitions are defined for items and routes. An item can initially be *new*, *placed*, *picked*, *in delivery*, *shipped*, and *completed*. A route is either *new*, *started*, or *ended*.

4.3. Process Fragments

In OC-BPMN, concurrently running process fragments jointly define the object-centric process. As illustrated in Figure 3, the order management process can be expressed as three fragments. Within a fragment, activities can be connected via traditional control flow, which orders their execution. Additionally, activities read and write data object nodes and list data object nodes that specify a class from the class diagram and a state from its corresponding object lifecycle. Following established BPMN semantics [24], activities can be executed if they are both control flow (cf) and data flow (df) enabled. An activity is cf-enabled without ongoing cf-arcs or if at least one has been enabled by previous activities. It is df-enabled in an execution state if data objects in the specified states exist with the required links.

In the order fragment, *start shopping* is always enabled and creates a new object instance of *Order* in state *new*. Given a set of new items, *place order* is then enabled for items and the new order. It will change the items and the order into the state *placed* respectively. Furthermore, the transferred items are all linked to the order to reference them later. Then the order can be invoiced and paid by the fragment. Concurrently, the individual items are processed in the item fragment, and can be correlated to routes and processed further in the routes fragment.

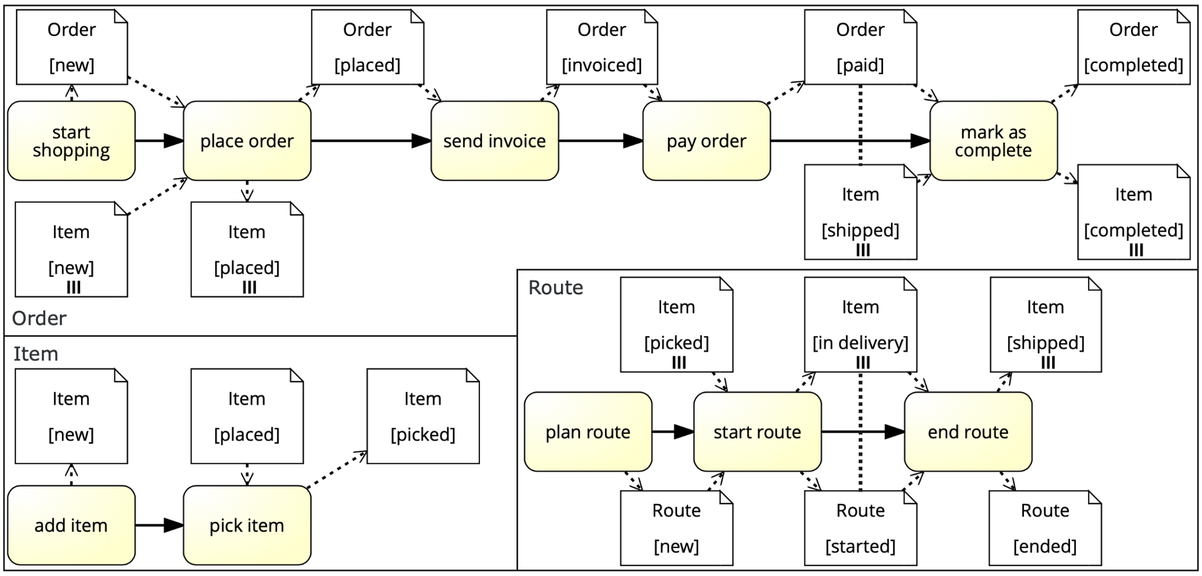

Figure 3: The object-centric BPMN fragments representing the order management process.

In OC-BPMN, we introduce a new notation to specify the required synchronization of related objects. Object nodes can be associated to indicate that the execution of the following activity requires a synchronization of the referenced objects at runtime. For instance, the node for *Order[paid]* is associated with the list data object node for *Item[shipped]*. The activity *mark as complete* can, therefore, only be executed for orders and items in the respective states and only if they have been linked before. After the execution, the links will be preserved. Analogously, *end route* synchronizes a route with its items.

4.4. Data State

With OC-BPMN, we also propose a notation for the instance level of an OC-BPMN model. The data state is an instance of the class diagram. It represents objects as class instances with a state attribute of their respective OLC. Objects can be linked, such that each link is an association instance.

In Figure 4, a UML object diagram [23] depicts the data state containing the paid order $o1$ and placed order $o2$, the items $i1$ (shipped), $i2$ (picked), and $i3$ (placed), and the ended route $r1$ and new route $r2$. Also note that $o1$ is linked to multiple items, i.e., $i1$ and $i2$, and $o2$ is linked to $i3$. In the current state, only $r1$ is linked to $o1$. In a future data state, $r2$ could be linked to $i2$ and $i3$, constituting an implicit many-to-many relation between order and route.

An object diagram can be useful to visualize the current state of a running process or to configure an initial data configuration for simulating or executing the OC-BPMN.

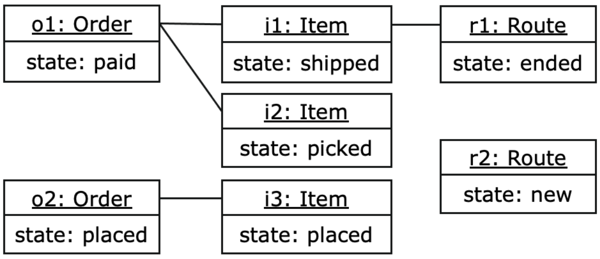

Figure 4: The UML object diagram depicting an instance of the class diagram (cf. Figure 2) for the order management process.

5. Discussion & Conclusion

This paper investigates the capabilities of BPMN to model object-centric processes. In our analysis of modeling features, we identify limitations for BPMN regarding object identification and missing link semantics. Also, the case notion of BPMN process models leads to the problems of convergence and divergence. To overcome these limitations, we build on concepts from case management to propose the novel modeling paradigm of object-centric BPMN (OC-BPMN).

OC-BPMN models can flexibly combine process fragments that are modeled with different case notions, i.e., object perspectives, allowing to overcome convergence and divergence in process modeling. In the future, we will provide precise execution semantics for OC-BPMN. Object-centric Petri nets with identifiers by Gianola et al. [4] promise a suitable goal formalism, which can be extended with exact synchronization as proposed for Proclets [18]. The resulting formalism will allow for precise correlation mechanisms of linked data objects throughout the process execution.

So far, we only analyzed BPMN and not other contemporary object-centric process modeling approaches that could also provide useful visualizations for object-centric processes. Future studies should investigate their capabilities as well and compare their comprehensibility for users empirically.

In the future, contemporary discovery algorithms for OCPNs can be extended to also allow for discovering the proposed OC-BPMN models. This includes a discovery of the creation of links between objects and their synchronization behavior. On the other hand, OC-BPMN models could also be used and configured to model and refine object-centric simulation models, allowing for the creation of object-centric event logs.

In summary, we propose a novel, object-centric dialect of BPMN with execution semantics based on OPIDs. We believe that OC-BPMN offers a more well-known notation for object-centric processes, making the object-centric paradigm more accessible in practice.

References

[1] M. Dumas, D. Pfahl, Modeling software processes using BPMN: When and when not?, in: Managing Software Process Evolution, Springer, 2016.

[2] W. M. P. van der Aalst, Object-centric process mining: Dealing with divergence and convergence in event data, in: Software Engineering and Formal Methods - 17th International Conference, SEFM 2019, Proceedings, volume 11724 of *Lecture Notes in Computer Science*, Springer, 2019.

[3] M. Weske, Business Process Management - Concepts, Languages, Architectures, 4th Edition., Springer, 2024.

[4] A. Gianola, M. Montali, S. Winkler, Object-centric conformance alignments with synchronization, in: International Conference on Advanced Information Systems Engineering, Springer, 2024.
[5] S. Steinau, A. Marrella, K. Andrews, F. Leotta, M. Mecella, M. Reichert, DALEC: a framework for the systematic evaluation of data-centric approaches to process management software, Softw. Syst. Model. 18 (2019).
[6] W. M. van der Aalst, Object-centric process mining: unraveling the fabric of real processes, Mathematics 11 (2023) 2691.
[7] M. Hewelt, M. Weske, A hybrid approach for flexible case modeling and execution, in: Business Process Management Forum: BPM Forum 2016, Rio de Janeiro, Brazil, September 18-22, 2016, Proceedings 14, Springer, 2016.
[8] OMG, Business Process Model and Notation (BPMN), Version 2.0.2, Technical Report, Object Management Group, 2014. https://www.omg.org/spec/BPMN/2.0.2/PDF.
[9] R. M. Dijkman, M. Dumas, C. Ouyang, Semantics and analysis of business process models in bpmn, Information and Software technology 50 (2008).
[10] A. Awad, G. Decker, N. Lohmann, Diagnosing and Repairing Data Anomalies in Process Models, in: Business Process Management Workshops, BPM 2009 International Workshops, Ulm, Germany, September 7, 2009. Revised Papers, Lecture Notes in Business Information Processing, Springer, 2009.
[11] S. von Stackelberg, S. Putze, J. Mülle, K. Böhm, Detecting Data-Flow Errors in BPMN 2.0, Open Journal of Information Systems (OJIS) 1 (2014).
[12] M. Ramadan, H. G. Elmongui, R. Hassan, BPMN formalisation using coloured petri nets, in: Proceedings of the 2nd GSTF annual international conference on software engineering & applications (SEA 2011), 2011.
[13] A. Meyer, L. Pufahl, D. Fahland, M. Weske, Modeling and Enacting Complex Data Dependencies in Business Processes, in: Business Process Management - 11th International Conference, Lecture Notes in Computer Science, Springer, 2013.
[14] M. König, M. Weske, Multi-instance data behavior in BPMN, in: Companion Proceedings of the 42nd International Conference on Conceptual Modeling: ER Forum co-located with ER 2023, CEUR Workshop Proceedings, 2023.
[15] S. Haarmann, M. Montali, M. Weske, Refining case models using cardinality constraints, in: International Conference on Advanced Information Systems Engineering, Springer, 2021.
[16] V. Künzle, M. Reichert, PHILharmonicFlows: towards a framework for object-aware process management, J. Softw. Maintenance Res. Pract. 23 (2011).
[17] M. Breitmayer, L. Arnold, M. Pejic, M. Reichert, Transforming object-centric process models into bpmn 2.0 models in the philharmonicflows framework, in: Modellierung 2024, Gesellschaft für Informatik eV, 2024.
[18] D. Fahland, Describing behavior of processes with many-to-many interactions, in: Application and Theory of Petri Nets and Concurrency: 40th International Conference, PETRI NETS 2019, Proceedings, Springer, 2019.
[19] F. Rosa-Velardo, D. de Frutos-Escrig, Decidability problems in petri nets with names and replication, Fundam. Informaticae 105 (2010).
[20] S. Ghilardi, A. Gianola, M. Montali, A. Rivkin, Petri net-based object-centric processes with read-only data, Inf. Syst. 107 (2022).
[21] W. M. P. van der Aalst, A. Berti, Discovering object-centric petri nets, Fundam. Informaticae 175 (2020).
[22] L. Pufahl, M. Weske, Batch activity: enhancing business process modeling and enactment with batch processing, Computing 101 (2019).
[23] OMG, Unified Modeling Language (OMG UML), Version 2.5.1, Technical Report, Object Management Group, 2017. https://www.omg.org/spec/UML/2.5.1/PDF.
[24] M. Weske, Business Process Management - Concepts, Languages, Architectures, 3rd Edition., Springer, 2019.