Lifting Process Discovery and Conformance Checking to the Next Level: A General Approach to Object-Centric Process Mining

Wil M.P. van der Aalst[1,2]

[1]Process and Data Science, RWTH Aachen University, D-52074 Aachen, Germany
[2]Celonis, Theresienstraße 6, D-80333 München, Germany

Abstract

Traditional process mining approaches are case-centric, i.e., it is assumed that each event refers to a case, an activity, and a timestamp. However, in reality, events may refer to multiple objects of different types (instead of a single case). This simplifying assumption can be motivated by the fact that most process modeling notations are also case-centric, e.g., workflow nets, UML activity diagrams, BPMN models, and directly-follows graphs, all describe life-cycles of individual cases. However, as the process-mining discipline matures, we want to drop this assumption and better align event data and process models with the actual processes and the data stored in information systems. This explains the interest in Object-Centric Process Mining (OCPM). The significance of the transition from case-centric to process-centric is comparable to the transition from classical Petri nets to Colored Petri Nets (CPNs) and the transition from two-dimensional images (e.g., an X-ray) to three-dimensional images (e.g., a full-body MRI). This extended abstract shows how traditional techniques for process discovery and conformance checking can be lifted from case-centric to object-centric. We provide a generic framework that allows us to leverage traditional case-centric process mining techniques. This provides baseline approaches for object-centric process discovery and conformance checking.

Keywords

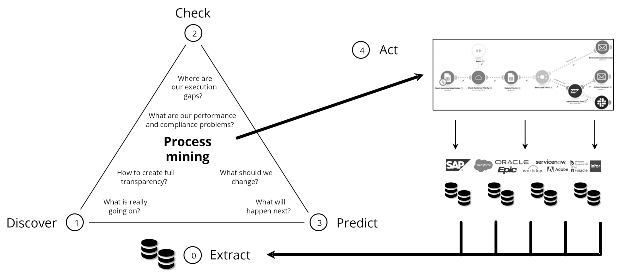
Process Mining, Object-Centric Process Mining, Process Discovery, Conformance Checking, Petri Nets, Object-Centric Event Data

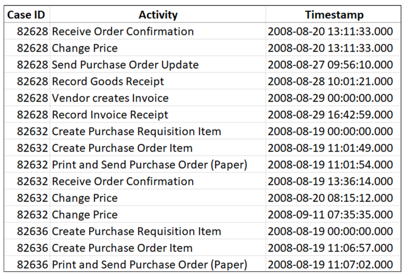
1. Introduction

Object-Centric Process Mining (OCPM) aims to discover and analyze processes starting from *Object-Centric Event Data* (OCED) [1, 2]. Traditional *case-centric* process mining allows for only one type of objects (called cases) and assumes that each event refers to precisely one object. In OCPM, there can be multiple object types, objects may be related, and one event may refer to any number of objects.

Figure 1 introduces basic process mining concepts. The right-hand side shows a small fragment of a traditional event log where each *event* refers to one *case*, an *activity*, and a *timestamp*. A case can be seen as a process instance consisting of events that are ordered using the timestamps and labeled using the activity attribute. Events may have many more attributes,

International Workshop on Petri Nets and Software Engineering (PNSE 2024)
wvdaalst@pads.rwth-aachen.de (Wil M.P. van der Aalst)
https://vdaalst.com/ (Wil M.P. van der Aalst)
0000-0002-0955-6940 (Wil M.P. van der Aalst)





Case ID	Activity	Timestamp
82628	Receive Order Confirmation	2008-08-20 13:11:33.000
82628	Change Price	2008-08-20 13:11:33.000
82628	Send Purchase Order Update	2008-08-27 09:56:10.000
82628	Record Goods Receipt	2008-08-28 10:01:21.000
82628	Vendor creates Invoice	2008-08-29 00:00:00.000
82628	Record Invoice Receipt	2008-08-29 16:42:59.000
82632	Create Purchase Requisition Item	2008-08-19 00:00:00.000
82632	Create Purchase Order Item	2008-08-19 11:01:49.000
82632	Print and Send Purchase Order (Paper)	2008-08-19 11:01:54.000
82632	Receive Order Confirmation	2008-08-19 13:36:14.000
82632	Change Price	2008-08-20 08:15:12.000
82632	Change Price	2008-09-11 07:35:35.000
82636	Create Purchase Requisition Item	2008-08-19 00:00:00.000
82636	Create Purchase Order Item	2008-08-19 11:06:57.000
82636	Print and Send Purchase Order (Paper)	2008-08-19 11:07:02.000

Figure 1: A traditional event log (right) and the different types of process mining: (0) extract, (1) discover, (2) check, (3) predict, and (4) act (left).

but the three attributes shown in Figure 1 are sufficient to discover case-centric process models, e.g., a classical Petri net (typically a workflow net with a clear start and end), a Directly-Follows Graph (DFG), a BPMN (Business Process Modeling Notation) model, or a UML activity diagram.

Currently, there are over 50 commercial process mining tools, all able to automatically discover process models using such event data [3, 4]. The high-end tools offer not only process discovery, but support all the tasks depicted in Figure 1 (left), i.e., also the extraction of event data from source systems such as SAP, Oracle, ServiceNow, SalesForce, etc., conformance checking to compare the real and modeled behavior, prediction of performance and conformance measures, and automatically triggering actions to improve processes based on process-mining diagnostics.

Although process mining is widely adopted (especially in Europe) and has proven to help organizations improve their processes, it is evident that the single-case assumption is severely limiting the scope of analysis and leads to distortions such as *convergence* and *divergence* [1, 2]. Most processes involve many interacting and related objects (e.g., orders, items, customers, suppliers, machines, etc.). Therefore, Object-Centric Process Mining (OCPM) is in focus, both in research and among tool vendors. For example, as we will show, the new Celonis process mining platform is completely based on OCPM.

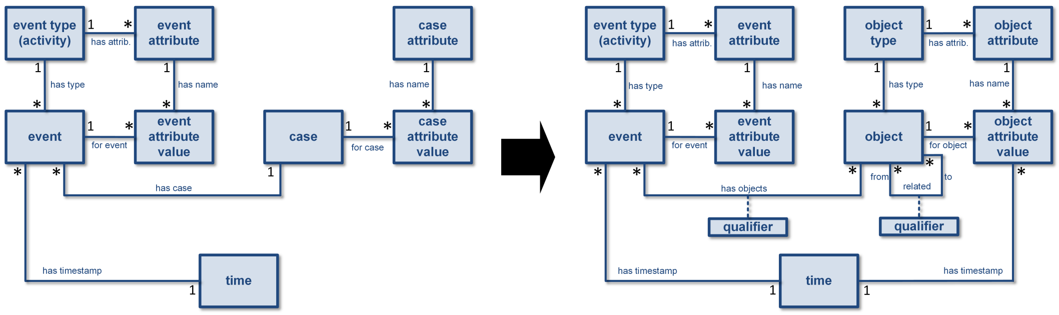


Figure 2: Case-centric event data (left) versus object-centric event data (right). The object-centric meta-model on the right adds Event-to-Object (E2O) and Object-to-Object (O2O) relations [5].

Figure 2 shows two meta-models compactly showing the differences between case-centric (left) and object-centric (right). The case-centric meta-model on the left represents the classical view that each event refers to precisely one case. The object-centric meta-model on the right uses objects instead of cases, and allows for arbitrary *Event-to-Object* (E2O) and *Object-to-Object*

(O2O) relations. These relations can also be qualified. Note that one event can have many objects and one object may be involved in many events. *Objects* and *events* are both *typed* and may have additional attributes. Often, we refer to an event type as the *activity*. In the remainder, we use the terms “event type” and “activity” interchangeably.

The rest of this extended abstract is organized as follows. Section 2 discusses Object-Centric Process Discovery (OCPD) and Section 3 discusses Object-Centric Conformance Checking (OCCC). Example implementations are briefly described in Section 4, followed by a discussion and conclusion (Section 5).

2. Object-Centric Process Discovery

Since the turn of the century, many process discovery techniques have been developed to automatically learn representations such as Petri nets, DFGs, and BPMN models from event data. This is challenging task because the input is just a sample of possible behaviors (i.e., only positive examples and incomplete). Process models with loops describe infinitely many possible traces, and even models without loops (but with concurrency) may have an exponential number of states and a factorial number of traces. Therefore, even for large event logs, one cannot assume that “what did not happen, cannot happen”.

Discovery approaches can be classified into two main categories: *bottom-up process discovery approaches*, including the Alpha algorithm and region-based techniques [6, 7, 8, 9, 10, 11, 12], and *top-down process discovery approaches*, such as inductive mining methods [13, 14, 15]. For a comprehensive review of process discovery techniques, refer to [16].

All of the mentioned approaches assume that each event refers to precisely one case. As a result, each case refers to a sequence of activities, and for process discovery event data can be reduced to a multiset of activity sequences (i.e., traces). We would like to leverage existing techniques for case-centric process discovery and discover object-centric process models. The only assumption that we make is that case-centric process discovery produces process models where each activity is unique, i.e., it is not allowed to have the same activity at two places in the process model. Very few process discovery techniques produce duplicate activities violating this assumption. An exception is process discovery using state-based regions with label splitting [10, 11]. We do allow for techniques that discover silent activities (i.e., skips) and gateways. This is not a problem for the approach described here, because the different object flows are only synchronized in events that correspond to unique activities. Therefore, we are able to reuse most of the existing discovery approaches.

Next, we describe a general approach to discovering object-centric process models.

1. **Selection**: Given object-centric event data according to the object-centric meta-model presented before (Figure 2), select the object types and event types that are in scope. It is also possible to make additional, more fine-grained selections for the selected object types. For example, select subsets of objects based on some filter criterion (e.g., remove all orders placed before a certain start date). It is also possible to use qualifiers or event-object-type combinations to filter E2O relations. The resulting selection of objects and events forms again object-centric event data in the sense of the meta-model on the right-hand-side in Figure 2.

2. **Flatten**: For each object type OT, create a traditional event log L_{OT} by flattening the event data. Given an object type OT, consider all events including at least one object of type OT. For each of these events, create an event in L_{OT} for *each* object of the selected type. For example, if a *place order* event refers to one *order* object and five *item* objects, then there will be one corresponding event in the event log for orders and five corresponding events in the event log for items. The result of this step is a traditional event log L_{OT} for each object type selected.
3. **Discover**: For each object type OT, use the event log L_{OT} and a traditional process discovery technique to discover a process model per object type. Any process discovery technique that produces unique activities can be used. The result is a process model M_{OT} per object type OT. Note that due to flattening, multiple process models may refer to the same activities, but the activity frequencies may be different.
4. **Correct**: One event in the original event log may refer to a variable number of events in the flattened event logs. This poses a problem when merging the models. Each process model M_{OT} needs to be "repaired" such that the frequency counts are correct. This can be achieved by "batching", i.e., if multiple events in the flattened event log correspond to the same event in the original event log, then the activity is assumed to handle all objects in the original event in a single step. This concept can be visualized using so-called *variable arcs* that do not show the flow of individual objects, but groups of objects.
5. **Merge**: The previous steps ensured that we have a model for each object type such that the activity names are unique and the frequencies are consistent (i.e., the activity frequencies match the frequencies in the original event logs before flattening). This means that the models are "in sync" and can be merged by fusing the activities with the same label.
6. **Enrich**: It is possible to enrich the merged process model with cardinality constraints learned from the original event log, e.g., activity *place order* involves one or more items and precisely one order. It is also possible to add descriptive statistics such as the minimal, maximal, and mean number of objects involved in an activity. Also timing information and frequencies can be added.

To illustrate the six steps, consider object-centric event data relating to 2000 orders, 6000 items, and 3000 packages. Assume that there are orders consisting of just one item, but also orders with over ten items. On average, orders consist of three items. Also, packages may contain a variable number of items. There are many packages with just one item, but also packages with five items. On average, a package has two items. Note that items belonging to the same order may end up in different packages and items in the same package may originate from different orders. Using case-centric process mining, one would need to focus on orders, items, or packages separately. However, this leads to partial models and misleading insights. Therefore, we apply the six steps mentioned before.

There may be many more object types and event types (i.e., activities), but assume we selected the object types *order*, *item*, and *package* (and the respective event types). Based on this selection, we flattened the object-centric event data into three traditional event logs. These flattened event logs are used to discover the three process models shown in Figure 3.

Figure 3 shows the life-cycles of the individual objects. However, the frequencies of the

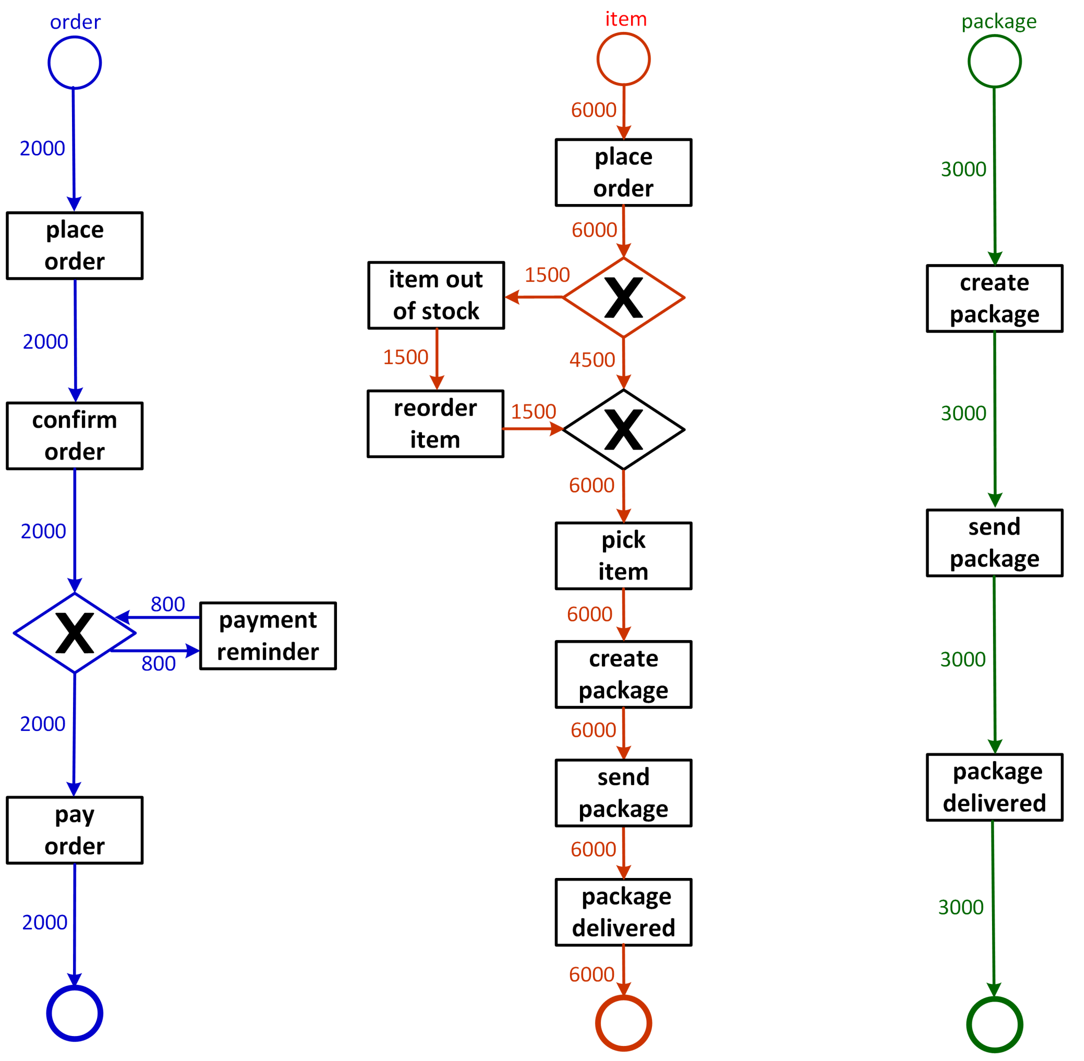


Figure 3: Three BPMN models discovered based on the three flattened event logs. Note that activity *place order* occurs 2000 times in the process model discovered for object type *order* and 6000 times in the process model discovered for object type *item*. Activity *create package* occurs 6000 times in the process model discovered for object type *item* and 3000 times in the process model discovered for object type *package*. These differences in frequencies need to be resolved in order to merge the different objects flows into a single model.

activities do not match. Note that *place order* occurs 2000 times in the *order* process, but 6000 times in the *item* process. There is also disagreement between the *item* and *package* processes, e.g., activity *create package* occurs 6000 times in the *item* process and 3000 times in the *package* process. These are the usual problems when flatting event data (see the convergence and divergence problems described in [1, 2]). Therefore, we apply the corrections mentioned in the fourth step. The result is shown in Figure 4. Activity *place order* now occurs 2000 times in each process and not 6000 times as suggested by the *item* process in Figure 3. Note that activities

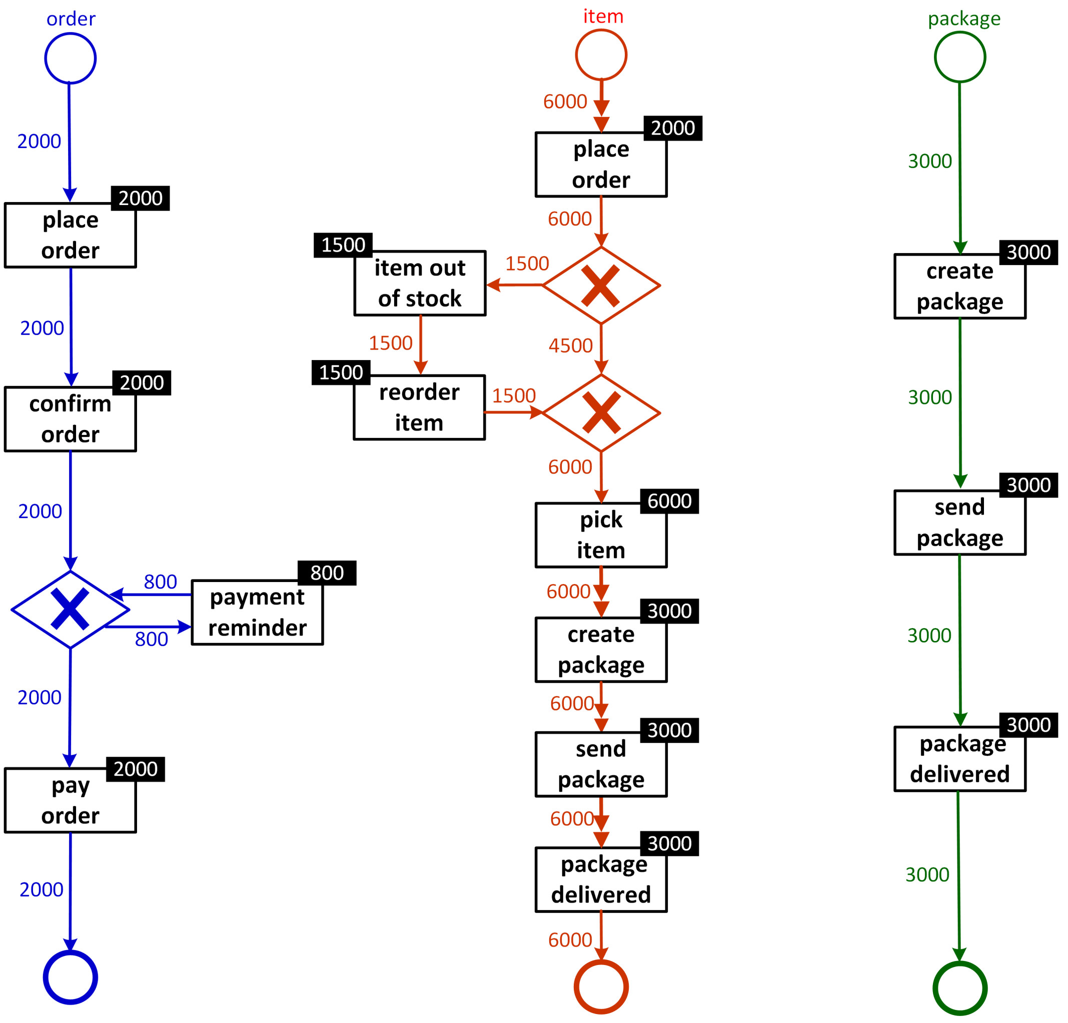


Figure 4: Three corrected BPMN models showing the original frequencies. This is achieved through so-called "variable arcs" (see the double-headed arcs going into the activities that may involve a variable number of objects of that type). In this example, only the model for the *item* process has to be corrected. Note that activity *place order* occurs 2000 times and, on average, handles three items. Note that activity *create package* occurs 3000 times and, on average, handles two items.

place order, *create package*, *send package*, and *package delivered* have "variable arcs" (represented by the double-headed arcs in Figure 4), and the correct frequencies are indicated. For example, *package delivered* occurs 3000 times and handles 6000 items.

Figure 5 shows the result of merging the three BPMN models from Figure 4. This step is trivial because activity labels are unique, and frequencies match. The resulting model shows the flow of three types of objects and the activities they are involved in. The object-centric BPMN model can be further extended with constraints and descriptive statistics related to cardinalities, frequencies, and times.

Note that the approach is generic and does not depend on a specific notation or a specific

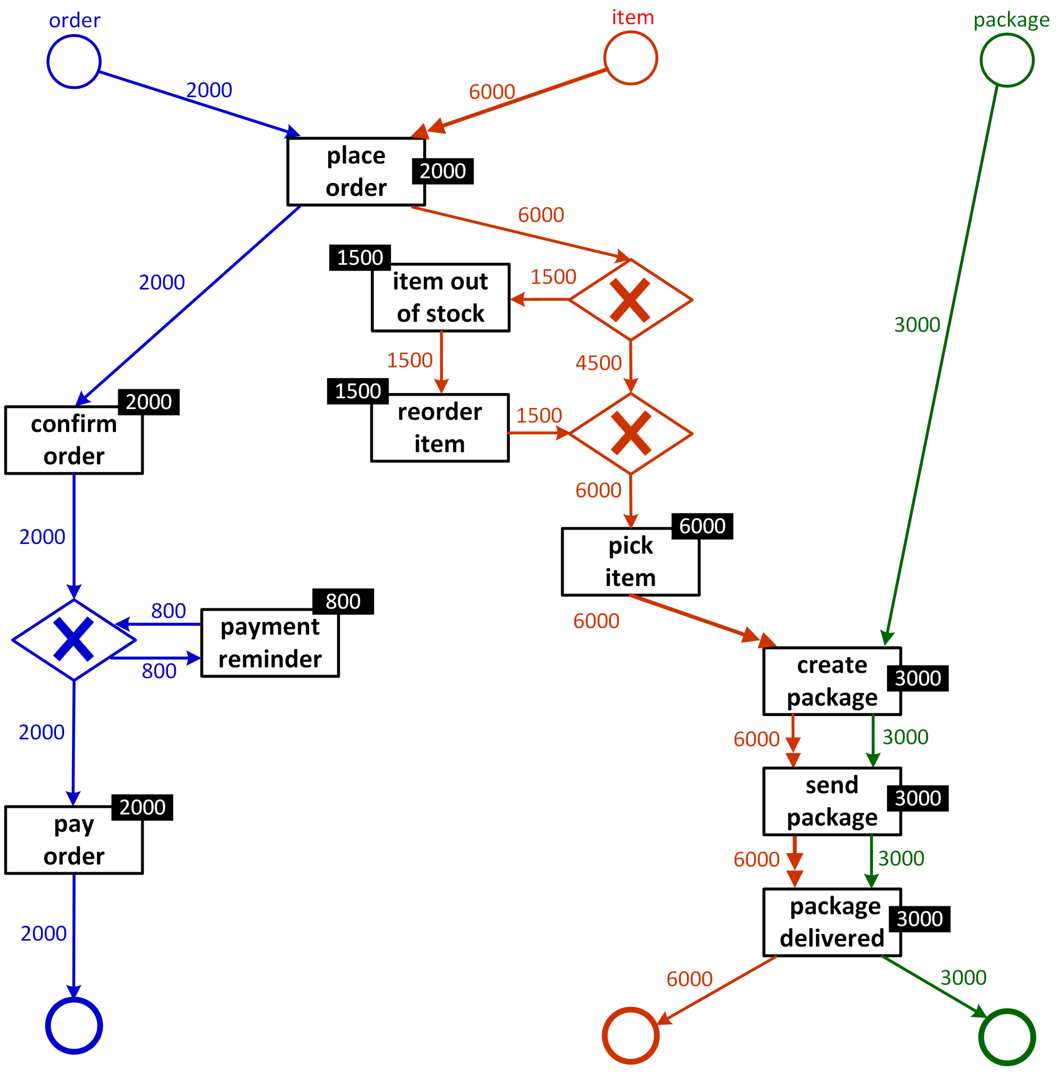


Figure 5: The merged object-centric BPMN model showing the object flows of three different object types.

discovery technique. The same ideas have already been applied to DFGs, Petri nets, process trees, etc. (see Section 4).

3. Object-Centric Conformance Checking

It is also possible to check the conformance of processes by comparing a process model with event data [6, 17]. The two most frequently used conformance-checking approaches are *token-based replay* [18] and *alignments* [19, 17]. Here, we can apply an approach similar to the one used for discovery in Section 2.

1. **Selection**: As input, we need object-centric event data and an object-centric process model. The assumption is that the scope of the data and model are the same. If not, further selection and alignment operations are needed.
2. **Flatten**: For each object type OT, create again a traditional event log L_{OT} by flattening the event data. Moreover, project the object-centric process model onto one model per object type OT. While flattening the model, replace the variable arcs with ordinary arcs, i.e., activities handle one object at a time.
3. **Check object flows**: For each object type OT, use the event log L_{OT} and the corresponding flattened model. Using this as input, traditional conformance-checking approaches can be used (e.g., token-based replay or alignment computations). This yields diagnostics per object type. Note that all deviations found per object type are also real deviations.
4. **Check cardinalities**: Using the object-centric event data it is also possible to check cardinality constraints (e.g., a *send package* event without items) and report deviations.

Note that the flattened event data and process models provide necessary but not sufficient conditions for conformance checking. Any deviation found in using the approach described before is a real deviation. However, some conformance problems may remain undetected. See [20] for a more detailed problem analysis. The object-centric process models used in OCPM tend to be underspecified. For example, *create package*, *send package*, and *package delivered* need to work on the same subsets of objects, but this is not clear from the model in Figure 4. To support this, we need to explicitly use O2O relations in our process models.

4. Implementation

The approaches for Object-Centric Process Discovery (OCPD) and Object-Centric Conformance Checking (OCCC) discussed in the previous sections should be seen as generic, baseline approaches. They illustrate that existing techniques for case-centric process mining can be reused and provide a good starting point. These approaches have been implemented in open-source tools such as OCPM [21, 22], OCPI [23], and OCPA [24] and are now making their way into commercial software products.

Celonis was the first process mining vendor to fully embrace OCPM. The Celonis process mining platform implemented the OCPM approaches described in this paper. In the new version of the platform, event data are stored as OCED and views (called perspectives) can be analyzed. All the types of process mining mentioned in Figure 1 have been adapted for the new setting. The Celonis *Multi-Object Process Explorer* (MOPE) is able to generate an object-centric DFG. The Celonis *Process Adherence Manager* (PAM) uses a variant of the Inductive Mining (IM) algorithm to discover object-centric BPMN models. These can be edited, stored, imported, and exported. Moreover, PAM supports conformance checking using alignments and allows for performance analysis across different object types. For example, it is easy to answer questions such as “What is the average time between placing an order and the last item being delivered?” and “How often do we send a payment reminder before the first item is delivered?”. Note that these questions involve multiple object types.

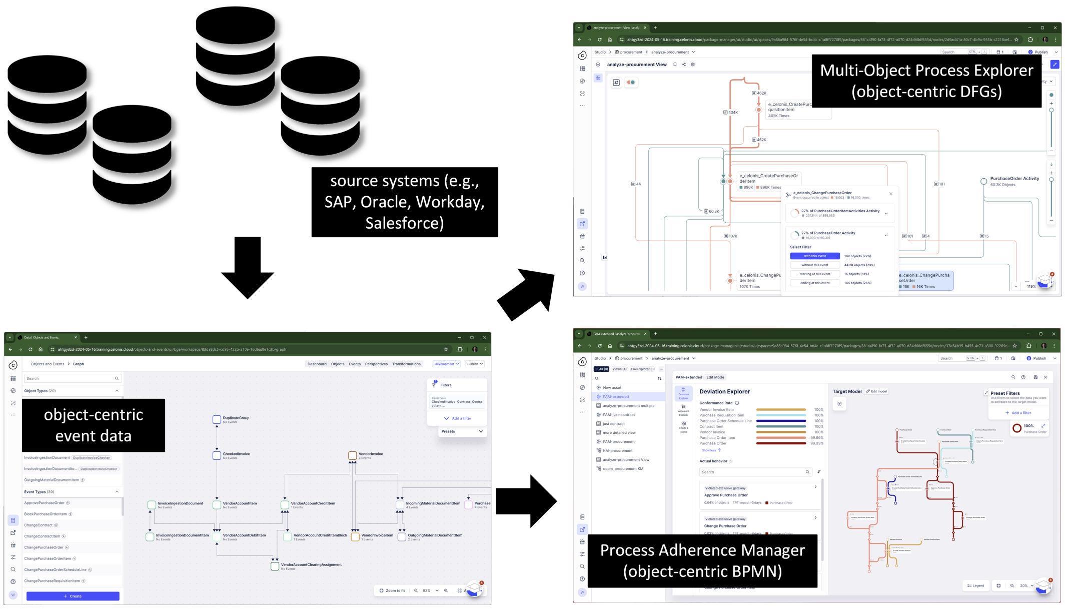

source systems (e.g., SAP, Oracle, Workday, Salesforce)
object-centric event data
Multi-Object Process Explorer (object-centric DFGs)
Process Adherence Manager (object-centric BPMN)

Figure 6: OCPM is also supported by the Celonis process mining platform. Data is stored using an object-centric data model. It is possible to discover object-centric DFGs and object-centric BPMN models. Also, conformance checking using alignments is supported for object-centric BPMN models.

5. Discussion and Conclusion

This extended abstract provides an overview of Object-Centric Process Mining (OCPM) and presents some of the key principles. The transition from case-centric to object-centric process mining is significant, akin to the shift from classical Petri nets to Colored Petri Nets (CPNs) and from two-dimensional images (like X-rays) to three-dimensional images (such as full-body MRIs). In classical Petri nets, tokens are indistinguishable. To represent cases, we need to assume that the Petri net models one case in isolation. This is not possible when moving to multiple object types. Moreover, we are interested in interactions between objects. Just like classical Petri nets and X-rays are still useful, also case-centric process mining is useful. However, as the field matures and is ready to tackle more ambitious questions, OCPM helps to take process mining to the next level.

There are three main reasons for using OCPM:

1. **Avoid repeatedly going back to your source systems.** Object-Centric Event Data (OCED) offers a single system-agnostic source of truth. This saves time and helps to capture real-life events and objects. Data extraction is decoupled from particular analysis questions. In traditional process mining, data is extracted using a specific case notion. This is not needed anymore, because it is possible to generate the views (often called perspectives) on demand.
2. **Avoid distortions due to the single-case assumption.** Squeezing reality into simple event logs creates distortions. This includes the unintentional replication of events (convergence) and loss of causal relations (divergence). In traditional process mining,

frequencies of activities and information on costs and delays highly depend on the way the data was flattened. Squeezing multiple object types into a single case notion also results in more complex process models where all structure is lost.

3. **See and understand the interactions between different object types.** Problems live at the intersections of processes and organizational entities. For example, low On-Time-In-Full (OTIF) scores may be caused by problems in sales, production, procurement, logistics, etc.

In this extended abstract, we limited ourselves to Object-Centric Process Discovery (OCPD) and Object-Centric Conformance Checking (OCCC). We showed how existing techniques can be used to create baseline OCPD and OCCC approaches. These approaches have been implemented in open-source tools and the widely used Celonis process mining platform. However, OCPM also extends to other tasks, such as predictive analytics and predictive and generative Artificial Intelligence (AI). For example, in [25] we show that using OCED helps to make more accurate predictions. This is unsurprising because exploiting the underlying structure relating objects and events and using more context provides a better basis for machine learning.

Despite these early successes, many improvements are possible. The object-centric process models used thus far in OCPM are rather underspecified compared to, for example, Colored Petri Nets (CPNs) with arc inscriptions and guards. The reason is that the relations are in the data and not in the model (e.g., which items belong to an order). It makes sense to consider Object-to-Object (O2O) relations more explicitly, instead of focusing mostly on Event-to-Object (E2O) relations. This will make process discovery and conformance checking more challenging. However, it will allow us to create “digital shadows” that are much closer to the actual processes.

Acknowledgments

Thanks to the PADS and Celonis teams for implementing the various OCPM techniques (including OCEL 2.0, OCPM, OCPI, OCPA, MOPE, Process Sphere, and PAM) and thanks the Alexander von Humboldt (AvH) Stiftung for supporting our research. The research is also funded by the Deutsche Forschungsgemeinschaft (DFG) under Germany’s Excellence Strategy, Internet of Production (390621612).

References

[1] W. van der Aalst, Object-Centric Process Mining: Unraveling the Fabric of Real Processes, Mathematics 11 (2023) 2691.

[2] W. van der Aalst, A. Berti, Discovering Object-Centric Petri Nets, Fundamenta Informaticae 175 (2020) 1–40.

[3] Process Mining, Process Mining Website, www.processmining.org, 2024.

[4] M. Kerremans, D. Sugden, N. Duffy, Magic Quadrant for Process Mining Platforms, Gartner Research Note G00790664, 2024. www.gartner.com.

[5] Object-Centric, Object-Centric Event Log Standard Website, www.ocel-standard.org, 2024.

[6] W. van der Aalst, Process Mining: Data Science in Action, Springer-Verlag, Berlin, 2016.

[7] W. van der Aalst, A. Weijters, L. Maruster, Workflow Mining: Discovering Process Models from Event Logs, IEEE Transactions on Knowledge and Data Engineering 16 (2004) 1128–1142.

[8] A. Augusto, R. Conforti, M. Marlon, M. La Rosa, A. Polyvyanyy, Split Miner: Automated Discovery of Accurate and Simple Business Process Models from Event Logs, Knowledge Information Systems 59 (2019) 251–284.

[9] R. Bergenthum, J. Desel, R. Lorenz, S. Mauser, Process Mining Based on Regions of Languages, in: G. Alonso, P. Dadam, M. Rosemann (Eds.), International Conference on Business Process Management (BPM 2007), volume 4714 of *Lecture Notes in Computer Science*, Springer-Verlag, Berlin, 2007, pp. 375–383.

[10] J. Carmona, J. Cortadella, M. Kishinevsky, A Region-Based Algorithm for Discovering Petri Nets from Event Logs, in: Business Process Management (BPM 2008), 2008, pp. 358–373.

[11] M. Solé, J. Carmona, Process Mining from a Basis of State Regions, in: J. Lilius, W. Penczek (Eds.), Applications and Theory of Petri Nets 2010, volume 6128 of *Lecture Notes in Computer Science*, Springer-Verlag, Berlin, 2010, pp. 226–245.

[12] J. Werf, B. Dongen, C. Hurkens, A. Serebrenik, Process Discovery using Integer Linear Programming, Fundamenta Informaticae 94 (2010) 387–412.

[13] S. Leemans, D. Fahland, W. van der Aalst, Discovering Block-structured Process Models from Event Logs: A Constructive Approach, in: J. Colom, J. Desel (Eds.), Applications and Theory of Petri Nets 2013, volume 7927 of *Lecture Notes in Computer Science*, Springer-Verlag, Berlin, 2013, pp. 311–329.

[14] S. Leemans, D. Fahland, W. van der Aalst, Discovering Block-Structured Process Models from Event Logs Containing Infrequent Behaviour, in: N. Lohmann, M. Song, P. Wohed (Eds.), Business Process Management Workshops, International Workshop on Business Process Intelligence (BPI 2013), volume 171 of *Lecture Notes in Business Information Processing*, Springer-Verlag, Berlin, 2014, pp. 66–78.

[15] S. Leemans, D. Fahland, W. van der Aalst, Scalable Process Discovery and Conformance Checking, Software and Systems Modeling 17 (2018) 599–631. doi:10.1007/s10270-016-0545-x.

[16] A. Augusto, R. Conforti, M. Dumas, M. La Rosa, F. Maggi, A. Marrella, M. Mecella, A. Soo, Automated Discovery of Process Models from Event Logs: Review and Benchmark, IEEE Transactions on Knowledge and Data Engineering 31 (2019) 686–705.

[17] J. Carmona, B. van Dongen, A. Solti, M. Weidlich, Conformance Checking: Relating Processes and Models, Springer-Verlag, Berlin, 2018.

[18] A. Rozinat, W. van der Aalst, Conformance Checking of Processes Based on Monitoring Real Behavior, Information Systems 33 (2008) 64–95.

[19] W. van der Aalst, A. Adriansyah, B. van Dongen, Replaying History on Process Models for Conformance Checking and Performance Analysis, WIREs Data Mining and Knowledge Discovery 2 (2012) 182–192.

[20] L. L. J. Adams, W. van der Aalst, Object-Centric Alignments, in: J. Almeida, J. Borbinha, G. Guizzardi, S. Link, J. Zdravkovic (Eds.), International Conference on Conceptual Modeling (ER 2023), volume 14320 of *Lecture Notes in Computer Science*, Springer-Verlag, Berlin, 2023, pp. 201–219.

[21] A. Berti, W. van der Aalst, OC-PM: Analyzing Object-Centric Event Logs and Process

Models, International Journal on Software Tools for Technology Transfer 25 (2023) 1–17.
[22] A. Berti, G. Park, M. Rafiei, W. van der Aalst, A Generic Approach To Extract Object-Centric Event Data From Databases Supporting SAP ERP, Journal of Intelligent Information Systems 61 (2023) 835–857.
[23] J. Adams, W. van der Aalst, OCpi: Object-Centric Process Insights, in: L. Bernardinello, L. Petrucci (Eds.), Application and Theory of Petri Nets and Concurrency (Petri Nets 2022), volume 13288 of *Lecture Notes in Computer Science*, 2022, pp. 139–150.
[24] J. Adams, G. Park, W. van der Aalst, ocpa: A Python Library for Object-Centric Process Analysis, Software Impacts 14 (2022) 100438.
[25] J. Adams, G. Park, W. van der Aalst, Preserving Complex Object-Centric Graph Structures to Improve Machine Learning Tasks in Process Mining, Engineering Applications of Artificial Intelligence 125 (2023) 106764.