Multiple Case Notion in Log Extraction

Rachel Brabender and **Oliver Clasen**

Hasso Plattner Institute, Department of Business Process Technology
Prof.-Dr.-Helmert-Straße 2-3, 14482 Potsdam, Germany
{rachel.brabender, oliver.clasen}@student.hpi.de

Abstract. Current process mining techniques are taking eXtensible Event Stream (XES) logs in order to analyze process flows. Nevertheless, companies store information about their executed activities within relational databases by using in-table versioning. In order to be able to use the common process mining techniques, one has to extract such XES logs from relational databases at first. The eXtensible Object-Centric (XOC) modelling notation allows to describe in contradiction to the XES standard the interactions between multiple object types without discarding information. In this paper, we present the approach "Multiple Object Type Projection" as a new way to derive case notions from XOC logs. The resulting XES logs provide a specific view on the process and the interaction between multiple object types. By this, we are able to reduce the number of edges and getting a more clear view on complex process flows.

Keywords: Object-Centric Modelling · Convergence · Divergence · Process Discovery

1 Introduction

Process Mining is becoming more relevant in the digital era. It describes the application of different techniques to analyze and visualize e.g. operational processes [1]. Industries are dealing with the big amount of data and are interested in applying these techniques for investigating bottlenecks, discrepancies, optimization possibilities and so on. A common approach is to extract an event log which has an *eXtensible Event Stream*[1] (XES) format from the stored data in a database (or data warehouse). Each event has an activity name, a timestamp and is assigned to one case identifier [2]. Additional information might be stored in form of attributes. However, event data of a process isn't always directly provided in the XES modelling notation. Sometimes, it needs to be extracted from object-centric information systems which may be build on top of a relational database.

This raises many challenges to deal with. At first, the extraction of an event stream can be very time consuming, especially if you don't have any domain knowledge. This is reasoned by the increasing number of logged events resulting

[1] IEEE 1849-2016 XES Standard: https://xes-standard.org/

in larger databases. Without proper documentation, one can quickly lose the overview. In addition, one needs to identify how to correlate events within the XES logs (called case notion). A case notion narrows the perspective on a process and thereby may discard events or other information such as the interaction between multiple process instances from the resulting log. At the same time, convergence (one event is related to multiple cases) and divergence (independent, repeated executions of the same activity type within a single case) problems are appearing and lead to inadvertently duplicated events which could result in incorrect conclusions. [9]

In general, there are existing two different ways on how to extract event logs from relational data sources, based on redo logs (implicitly recorded process) or derived from in-table versioning (explicitly recorded process). Within a redo log each operation in the database is saved with a timestamp while in in-table versioning the primary key is extended with a timestamp column. [9]

In this paper, we are proposing an algorithm for extracting XES logs from an object-centric information system by applying a Multiple Object Type Projection. It is an enhancement of an approach from [3], where ours provides the possibility to consider the process flow of multiple object types at the same time. Thereby, it tackles the problem of discarding information. Furthermore, we are evaluating the presented approach by investigating the sequence of activities and the appearance of convergence and divergence problems.

First, we look at the related work regarding log extraction (XES) from the database (Section 2). In Section 3, we picture a process flow of a rental shop for filling a database. Subsequently, we will explain the *eXtensible Object-Centric* (XOC) notation as an approach to store and visualize object-centric data and compare it with the XES notation (Section 4). For deriving XES logs, we present a baseline approach and demonstrate our enhancement for extracting data from object-centric information systems. Lastly, we analyze the resulting Directly-Follow-Graphs (DFG) and discuss shortcomings in Section 5 and 6. The implementations related to our evaluation can be found on GitHub[2].

2 Related Work

In the field of in-table versioning, extracting event logs could be done based on the ontology-based data access paradigm [5][6] or by correlating events by using the similarity of their attributes [11]. Lu et al. presents an artifact-centric approach which is based on i.e., business entities [10]. In order to simplify the extraction process, an analysis system for defining events, resources and their relations is presented in [7].

In this paper, we are focusing on deriving process-centric logs (XES) from object-centric data where we reduce the relations of the selected object type perspective. In order to describe object-centric data, [9] proposes a way on how to extract, transform and store data using the eXtensible Object-Centric (XOC)

[2] `https://github.com/brrrachel/MultCaseNotionLogExtr`

modelling notation. It solves the challenge of identifying a case notion and enables new possibilities to analyse the data such as data-aware process model discovery [8], and new conformance checking techniques [4].

Previous mentioned approaches mostly concentrated on how to extract XES logs. Exceedingly few have focused the topic of convergence and divergence. To address this problem, in [3] a more faithful approach is presented for deriving different case notions based on object types and combine them into a Directly-Follows Multigraph (DFM) to get a more holistic view on the process.

We use the presented object type projection from [3] and enhance it. Furthermore, we test both approaches on a dataset from an own simulation. Finally, we analyse how the convergence and divergence problems behave in this scenario.

3 Database Simulation

Since the most process mining techniques are using the standard XES modelling notation, the research landscape does not provide sophisticating datasets for testing new object-centric related approaches. In the following, we present a simulation process involving six objects within nine different activities for filling a database.

We are simulating a database of a rental shop where do-it-yourselfers can create requests for renting inventories such as machines or essential tools. Each of our two stakeholders, the customer and the store, can perform specific activities. Six different objects may be involved at different stages: the *customer*s create *rental*s involving at least one *inventory*, while the *staff* members, creates *invoice*s for the rental requests as well as *inspection* reports for the returned inventories. In Figure 1, we outline how the different object types (in form of triangles) are involved in the process for each activity.

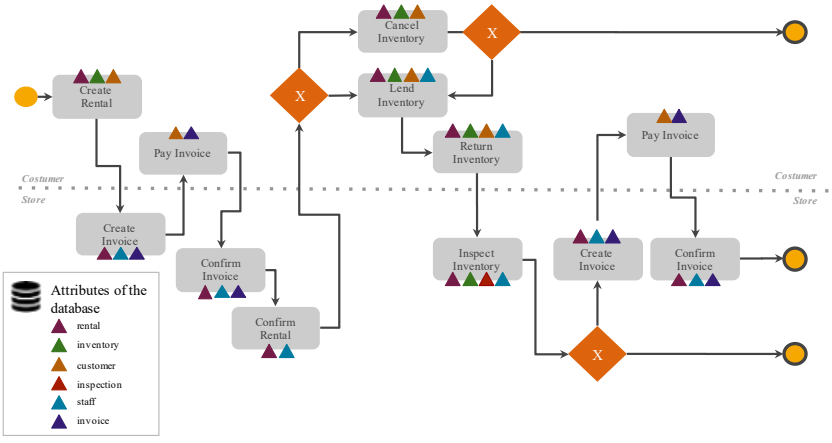


Fig. 1: Model of our Process in an Online Rental Shop

Before a customer can pick up an inventory he/she has to create a corresponding rental request (using the online presence of the rental shop). This request has to contain at least one inventory, whatever is available and he/she needs. In our simulation, a customer randomly selects up to two of all available inventories to rent. Available means, that the inventory is available in the store and already inspected at this time. After the creation of the rental request, a staff member of the shop creates an invoice. The customer will pay the invoice and the staff confirms the payment and the correctness of the placed order. The customer can now pick up the machine(s) he/she reserved or cancel the reservation for a specific inventory. After its usage at home, he/she has to return them to the shop. The customer does not have to return them all at once, maybe he/she needs one of the machines/tools a little bit longer than the others. So he/she probably returns them one by one. All returned inventories are then inspected by the staff for possible damages. If it turns out that an inventory is damaged, the customer has to pay an additional fee.

An implementation for simulating the described scenario is realized as follows: we iterate through each time interval in our simulation time until all rentals are finished (the entire stock of inventories are returned) and all invoices are payed and confirmed. In each interval, at first the customer and then the store can choose between their actions to take. The probability of choosing an activity is shown in Table 1 and Table 2. When the stakeholders are choosing an action, it is not assumed that there is something to do. Such as for example, if the customer decides to pay its invoices, probably there are no open invoices. In this case, he/she is passing this turn. Furthermore, we assume that the stakeholder will work through the entire batch within an interval, which means e.g. that the customer will pay for all open invoices or will pick up all requested inventories at the same time. Simultaneously, the store will e.g. create invoices for all recently created rental requests or inspects all recently returned inventories within the same interval.

The execution of each activity is recorded within the database by using the current time stamp and the involved objects. In our simulation scenario, we have 79 customers almost equally distributed over seven stores with 200 randomly distributed inventories. We generated 17115 events from *1st January 2020 8:00:00 AM* to *2nd February 2020 0:55:48 AM* with a time period of $i = 3600sec.$

Probability	Activity
10%	Create Rental
25%	Pay Invoices
30%	Cancel Inventory
35%	Return Inventories and Lend Inventories

Table 1: Probabilities of the customer's actions

Probability	Activity
30%	Create Invoices
20%	Confirm Invoices
20%	Confirm Rentals
30%	Inspect Inventories and Create Invoices

Table 2: Probabilities of the store's actions

When one customer or store has performed an activity, we proceed the time t_x by $t_{x+1} = t_x + \frac{i}{C+S}$, where C and S are the respective total numbers of customers c and stores s. We ignored possible opening times of the stores.

4 Modelling Notations

Since the current process mining techniques are based on XES logs, Li et al. [9] are presenting an approach on how to visualize the behavior between object-centric data without *flattening* them into the XES log notation. In the following, we summarize the idea of the XOC modelling notation and then compare it with the XES notation. Finally, we present approaches to derive XES Logs from the XOC modelling notation.

4.1 The eXtensible Object-Centric (XOC) Modelling Notation

The aim of the modelling notation is to describe the behavior of objects within an object-centric system. In such a system, a database table is seen as object type or class. A record within a database table is considered as an object. As for example, the first row of the database table *rental* is seen as an object from type *rental* and can be identified by its primary key. A relationship between the objects from different types is created, when the id of an object is used as an foreign key within an other table. A *object model* describes a snapshot of the database at a certain point in time. Thereby it "consists of objects, object relations and two functions indicating the class and attributes of each object" [9, p. 4].

Transitions from one *object model* to another are evolved by performing activities such as "create rental" (also called *event type*). An *event type* summarizes the execution of multiple *change type*s, where each of them is the usage of a SQL statement within the database. Thereby, a XOC log consists of events, event types, object references, object models and an event ordering. [9]

Table 3 shows a fragment of an XOC event log from our simulation. Each row represents an event, where each event has an id, an activity name, timestamp and shows the involved objects. An object is involved, when it is created, used within an activity or even if the staff or the customer as objects are executing an activity.

4.2 Comparison between the XOC and the XES Modelling Notation

Since the XOC modelling notation presents a new concept for describing the process flow using records of a database, we want to compare this approach with the XES modelling notation. The main findings are summarized in Table 4.

In the XOC modelling notation, events are not related to cases such as using the XES modelling notion. It dissociate oneself from a strict assignment of events to cases and establishes relations between the involved objects and the events. In contrast, the XES modelling notation abstracts the process into a certain

event id	activity	timestamp	rental	inventory	customer	staff	inspection	invoice
297	Create Rental	2020-01-02 00:52:19	{120}	{41, 101}	{28}	{}	{}	{}
300	Create Invoice	2020-01-02 00:56:30	{106, 107, 116, 117, 120}	{}	{}	{11}	{}	{109, 110, 111, 112, 113}
383	Pay Invoice	2020-01-02 04:11:09	{}	{}	{28}	{}	{}	{106, 109, 113}
493	Confirm Invoice	2020-01-02 09:57:54	{}	{}	{}	{11}	{}	{..., 106, 109, 110, 111, 112, 113, ...}
834	Confirm Rental	2020-01-02 22:57:12	{..., 117, 120, 123, ...}	{}	{}	{14}	{}	{}
1418	Return Inventory	2020-01-04 06:53:01	{120}	{101}	{28}	{17}	{}	{}

Table 3: A fragment of an event log from the simulation.

perspective, where we consider the different traces before having a look deeper on its events.

The advantage of one modelling notation is the disadvantage of the other. Since the XES modelling notation is describing a certain perspective on the process, there is a clear affiliation of the events to possible process flows (cases). Deriving such information from XOC logs requires thoughtful discovering techniques. Since the abstraction provides comprehensibility, the XES notation on databases discards information, as for example which objects are involved in which events/cases.

By comparing both modelling notations, we can recognize that multiple challenges arise when we want to derive XES logs from object-centric information systems. This topic has also been evaluated in [9]. Since object-centric data can be correlated in many different ways, each correlation creates a specific perspective on the process and thereby **affects the quality of the data**. These transformations introduce problems such as data convergence and divergence which will be explained in more detail in Section 5.2. Furthermore, **process instances are considered in isolation**. If objects are involved into multiple events belonging to different cases such information isn't easy to derive anymore. Information about involved objects is stored within events as **attributes, but is not directly connected with each other**. So, it depends on the case notion which determines the cases and thereby creates a specific perspective. Since the **case notation is not naturally given** or even always clear, someone has to define it at first.

	eXtensible Event Stream (XES)	**eXtensible Object-Centric (XOC)**
Perspective	Inheritance Hierarchy Structure (Top-Down Approach)	Object-Centric Data (e.g. database tables)
Log Entries	Single Case Notion: events belonging to cases	Multiple Case Notions: events describing interaction between objects
Relations	1 case to 1...* events and 1 event to 1 case	1 object to 1...* events and 1 event to 1..* objects
Drawback	the decision for a perspective discards information	tend to become complex and difficult to understand

Table 4: A comparison of the XES with the XOC modelling notation

4.3 Deriving Case Notions

In the following, we want to explain the baseline approach from [3] for extracting XES logs from object-centric information systems. Furthermore, we want to present an enhancement to generate a specific view using two object types.

The approach from [3] called "Object Type Projection" assumes that every object type could be used as case notion. That means (compare Listings 1.1), each object from a selected object type describes a specific perspective on the process and thereby creates a case. The case id is derived from the object identifier. One collects all events in which the object from the selected type is involved and adds the information about the involved objects from other object types as attributes. As an example (consider Table 3), if we chose to derive a case notion from the object type "rental", a case with id 120 will be created involving inter alia events with id 297, 300, 834 and 1418.

We enhance this approach by offering the opportunity to derive a case id based on the unique combination of objects from different types (compare Listings 1.2. We call this approach "Multiple Object Type Projection". As for example (consider Table 3), if we choose to derive a XES log from the object types "rental" and "inventory", we pick all rental ids and inventory ids. Taking rental 120 and the inventories 41 and 101 from Table 3, we are forming two case ids (case id 1 with rental 120 and inventory 41 as well as case id 2 with rental 120 and inventory 101). In case with id 2, also event with id 1418 is collected since both objects are involved here.

5 Discovering

After extracting and creating the XES logs with presented approaches from Section 4.3, we apply process mining techniques. At first, we analyse the differences between the extraction methods by having a look at the chronological sequence

```
Select an object type
For each object of the object type:
    Create trace with the id of the object
    Collect all events where object is involved
    For each event:
        Include other involved objects as attributes
```

Listing 1.1: Object Type Projection

```
Select two object types
Derive for each unique combinations of ids of both object
    types a new_id
For each new_id:
    Create trace
    Select all events where both object instances are involved
    For each event:
        Include other involved objects as attributes
```

Listing 1.2: Multiple Object Type Projection

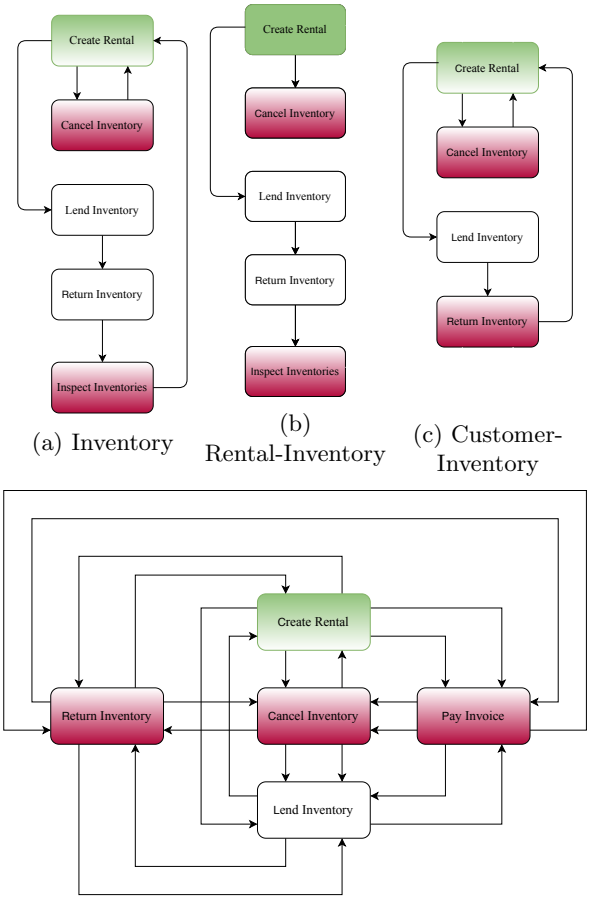
of activities for different case notions. Secondly, we investigate them on convergence and divergence problems.

5.1 Sequence of Activities

The order of activities within a process can be visualized in different ways. One possible approach is using the Directly-Follow-Graph (DFG) representation. To create the DFGs, we use the "Convert log to directly follows graph" plugin within ProM 6.9[3]. In the following, we will discuss exemplary the four different case notions: *Inventory*, *Customer*, *Rental-Inventory* and *Customer-Rental*. The resulting DFGs are visible within Figure 2, where the green color indicates a starting point of a process and the red color a possible end point.

The DFG for the case notion *Inventory* (compare Figure 2a) shows how an inventory behaves in the complete process. We can identify two possible process flows: a reserved inventory is directly canceled or an inventory is inspected after it was lent and returned. The loops are indicating that an inventory can be requested multiple times. In comparison, the DFG for the case notion *Customer* in Figure 2d shows a more complex process flow. The customer is involved in five activities and has many different process flows. This is reasonable by the fact, that one customer can have multiple rental orders running at the same time. Therefore, almost every sequence of activities is possible. A DFG with this amount of edges on a relatively small number of activities is very confusing and hard to read. Thereby, we can only identify the activities and potential starting

[3] http://www.promtools.org/doku.php



(a) Inventory (b) Rental-Inventory (c) Customer-Inventory

(d) Customer

Fig. 2: Directly Follow Graphs for different case notions

and end points of the process. A possible way to reduce the number of edges (and getting a clear understanding of the main process flow) is the application of a threshold. However, these edge cases can reveal interesting uncertainties within the examination.

An application of the Multiple Object Type Projection based on the object types customer and rental results in a more precise and clear DFG (compare Figure 2c). It shows the interactions of a customer with a specific rental. Thereby we don't lose edge cases within that correlation. Furthermore, combining object types (such as rental and inventory) and assigning each of them a case id, can show an unique process flow. This is also visually confirmed by Figure 2b since it does not contain a loop e.g. from "Inspect Inventories" to "Create Rental".

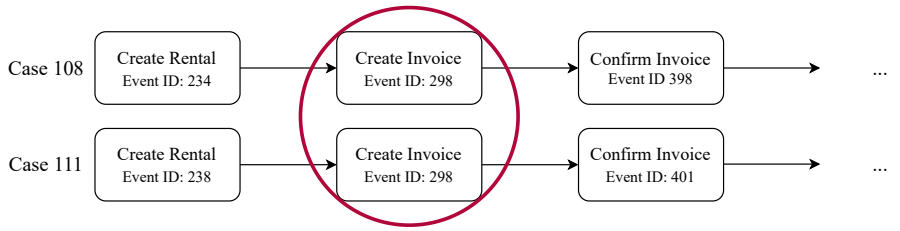
To sum it up, the Multiple Object Type Projection provides an approach to get more specific insights on processes regarding the relations between the involved objects. Additionally, the resulting DFGs contain less trace variants so an application of a threshold isn't absolutely necessary and thereby we don't lose information.

5.2 Convergence and Divergence Problem

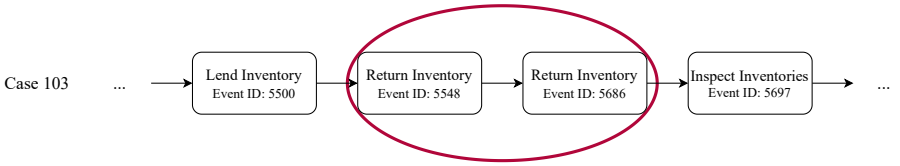
The convergence and divergence problem always comes along when choosing a specific perspective on a process. Convergence means that one execution of an activity in the real world leads to an duplication of the same event listed in different cases. Therefore, these events are redundant but necessary to display the process in the chosen perspective correctly. At the same time, divergence refers to a multiple execution of the same event type within one case. The resulting DFG can be miss leading, especially when one hasn't comprehensive knowledge about the underlying process. [3]

In the example of the *Rental* case notion, both problems are emerging. Within the activity "Create Invoice" multiple invoices can be generated. Accordingly, this activity instance will appear multiple times in different cases of the resulting XES log (convergence). At the same time, each returned inventory leads to multiple instances of "Return Inventory" activity within one case (divergence).

When having a look at the Multiple Object Type Projection, we are using the case notion *Customer-Inventory* as an example. Both (convergence and divergence) problems are emerging here, too. A customer can create a rental with multiple inventories. This event of creating the rental will then appear in

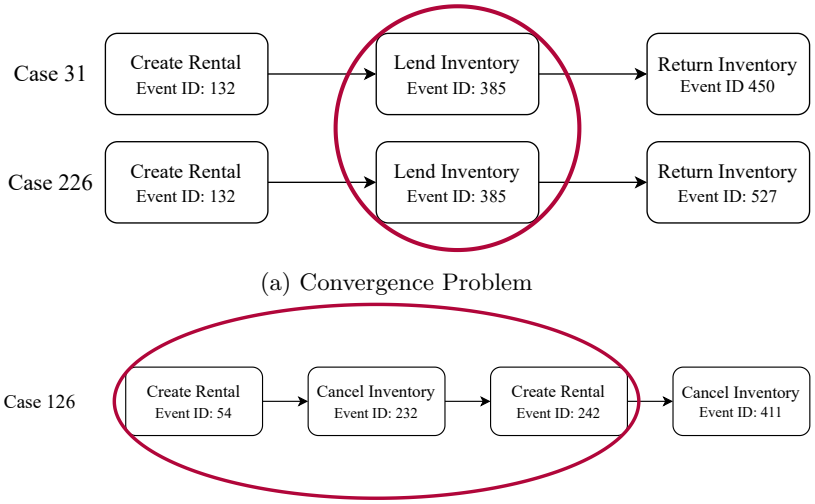


(a) Convergence Problem



(b) Divergence Problem

Fig. 3: Exemplary visualization of the convergence and divergence problem within the *Rental* case notion



(a) Convergence Problem

(b) Divergence Problem

Fig. 4: Exemplary visualization of the convergence and divergence problem within the *Customer-Inventory* case notion

multiple cases (convergence, compare Figure 4a). At the same time, a customer creates multiple rentals for the same inventory (divergence, compare Figure 4b).

The investigation of these case notions show, that the convergence problem within the derived XES logs are results of the duplication of events for each instance of the selected object type. The divergence problem is a result of the real world process (where in our case some inventories are returned at different points in time). Whenever the perspective on a process is reduced, one should be aware of these both problems.

6 Summary and Further Work

In this work, we examined the different possibilities of describing behavior within object-centric information systems and discussed the XOC modelling notation as a possible method. While the XES modelling notation is still the current standard as input for common process mining techniques, we presented the Object Type Projection from [3]. Furthermore, we enhanced this approach for considering multiple object types for the derivation of a case notion. Finally, we examined the results regarding the sequence of activities and the convergence/divergence problem.

We were able to show, that our approach Multiple Object Type Projection is capable of visualizing the interactions between different object types. The new case notion allows us to create DFGs with less edges, and thereby we get a better overview of the process (e.g. the *Customer* vs. *Customer-Inventory* case

notion). An application of a threshold is no longer absolutely necessary. The loss of information is solved in this way, that we explicitly consider multiple object types when deriving a XES log.

Our approach could be extended by testing the Multiple Object Type Projection on a larger process/database, since our exemplary process flow is still very small and easily to access. Here, it would be interesting to involve more than two object types into the projection. Another idea is to evaluate the so called *weak* Multiple Object Types Projection approach, where not all considered object types need to be involved into an activity at the same time so it is picked for the case. This could lead to an extended perspective and may include more activities than using the Object Type Projection from [3].

References

1. van der Aalst, W.M.P.: Process Mining: Discovery, Conformance and Enhancement of Business Processes. Springer-Verlag Berlin Heidelberg (2011)
2. van der Aalst, W.M.P.: Data Mining: Science in Action. Springer-Verlag Berlin Heidelberg (2016)
3. van der Aalst, W.M.P.: Object-centric process mining: Dealing with divergence and convergence in event data. Software Engineering and Formal Methods (2019)
4. van der Aalst, W.M.P., Li, G., Montali, M.: Object-centric behavioral constraints. ArXiv **abs/1703.05740** (2017)
5. Calvanese, D., Kalayci, T.E., Montali, E.M., Tinella, S.: Ontology-based data access for extracting event logs from legacy data: The onprom tool and methodology. BIS 2017: Business Information Systems pp. 220–236 (2017)
6. Calvanese, D., Montali, M., Syamsiyah, A., van der Aalst, W.M.P.: Ontology-driven extraction of event logs from relational databases **256**, 140–153 (2016)
7. Ingvaldsen, J.E., Gulla, J.A.: Preprocessing support for large scale process mining of sap transactions. Business Process Management Workshops pp. 30–41 (2007)
8. Li, G., , de Carvalho, R.M., van der Aalst, W.M.P.: Automatic discovery of object-centric behavioral constraint models. BIS 2017 p. 43–58 (2017)
9. Li, G., de Murillas, E.G.L., de Carvalho, R.M., van der Aalst, W.M.P.: Extracting object-centric event logs to support process mining on databases. Information Systems in the Big Data Era pp. 182–199 (2018)
10. Lu, X., Nagelkerke, M., van de Wiel, D., Fahland, D.: Discovering interacting artifacts from erp systems. IEEE Transactions on Services Computing pp. 1–15 (2015)
11. Pérez-Castillo, R., Weber, B., de Guzmán, I.G.R., Pinggera, M.P..J.: Assessing event correlation in non-process-aware information systems. Software Systems Modeling p. 1117–1139 (2014)