# Ont-OCEL: An Ontology-based Representation for OCEL

Mahmood Soltani [1], Mohsen Kahani[1] and Behshid Behkamal [1]

[1] *Ferdowsi University of Mashhad, Mashhad, Iran*

**Abstract**

Process mining is an emerging field of research that helps organizations gain deeper insights into their business processes. The process mining approach begins with an event log, which is extracted from an information system. Various standards, such as XES, exist for representing event logs. However, the Object-Centric Event Log (OCEL) format is currently considered state-of-the-art because it can capture multiple case notions for events' behavior, unlike other formats that rely on a single case notion. In this paper, we propose a new ontology-based representational model for the OCEL format. One of the advantages of this model is that it allows for easy enrichment of the event log with domain ontologies, enabling knowledge extraction. Additionally, SPARQL can be used to query and filter event logs, which enables the creation of flattened event data (a classical event log), extraction of statistical information, and building of a simple process map. Based on this representation, all event logs of an organization can be merged into an RDF-based knowledge graph, which can be extended progressively as new information becomes available or stream logging is required. To demonstrate the practical application of this work, we used a real object-centric event log and created an ontology to represent it. We also developed some SPARQL query templates to filter and extract useful information.

**Keywords**

Event log, ontology, object-centric event log, process mining, SPARQL

## 1. Introduction

Process mining is a technique that involves analyzing event data recorded in information systems to gain insights into the behavior and performance of business processes. By using process mining, organizations can identify process bottlenecks, inefficiencies, and opportunities for improvement. It comprised four main areas: process discovery, conformance checking, process reengineering, and operational support [1]. Event logs are the primary input data for process mining, and they provide a comprehensive record of the execution of business processes. At its most basic level, an event log contains information related to a single process, including CaseID, activity name, and timestamp.

Several methods and models have been proposed to standardize the format of event logs [2], [3]. These standardized formats serve as a data transport mechanism for process mining tools. However, storing event information in traditional models and standard formats, such as XES, can cause issues like divergence and convergence [4]. These problems can have negative impacts on the results of process mining tasks, including process discovery, and lead to undesired outcomes.

To address these issues, the Object-Centric Event Log (OCEL) format was proposed [5]. OCEL extends the traditional event log format by including additional information about the objects involved in the process, such as their attributes, states, and relationships. By using OCEL, it is

*TEXT2KG 2023: Second International Workshop on Knowledge Graph Generation from Text, May 28 - Jun 1, 2023, co-located with Extended Semantic Web Conference (ESWC), Hersonissos, Greece.*
Proceedings Name, Month XX–XX, YYYY, City, Country
EMAIL: mahmood.soltani@mail.um.ac.ir (A. 1); kahani@um.ac.ir (A. 2); behkamal@um.ac.ir (A. 3)
ORCID: 0009-0008-4859-3278 (A. 1); 0000-0002-2603-6066 (A. 2); 0000-0003-3151-1885 (A. 3)

possible to overcome the problem of a single case notion and prevent performance issues. In the previous object-centric formats, such as XOC, the size of the log would increase significantly with an increasing number of events. However, there are some challenges when it comes to incorporating external data sources that are not addressed in OCEL. Here, we are going to briefly present the problems we have practically encountered.

- ***Challenge 1. Domain knowledge cannot be directly applied in an event log***: Since the type of domain knowledge and event log representation are not the same, it is not possible to directly use domain knowledge in process mining tasks. Nevertheless, there are some formats where it is possible to connect event labels with concepts in an ontology using textual references. By using textual references to link event labels with ontology concepts, it may be possible to leverage some domain knowledge to improve the accuracy and effectiveness of process mining. However, this approach is limited in scope and may not be suitable for all types of process mining tasks.
- ***Challenge 2. Semantic process mining*****:** Process mining is a technique that focuses on extracting valuable information from event logs, which are often represented as string data. However, by extracting the underlying semantics of the event log data, instead of relying solely on the string-based labels, process mining tasks can be better accomplished.
- ***Challenge 3. In online process mining, the event log cannot be updated optimally when processes are running continuously*****:** Most process mining tools require an event log file to be uploaded before performing process mining tasks. One of the challenges of using event log files in process mining is that they are typically static and do not update in real-time. This means that if new events occur during the process execution, they are not immediately available for process mining tools to analyze. Instead, a new updated event log file needs to be created and uploaded into the process mining tool, which can cause a delay in the availability of the latest data for analysis.
- ***Challenge 4. Storing the log of each process in a standalone file:*** One of the challenges of process mining is that process information is often stored separately in individual files or within a single file with no explicit connections between the processes. This lack of integration and coherence can make it difficult to analyze the data effectively and derive insights from the process mining tasks. However, in order to improve the accuracy and effectiveness of process mining, it may be necessary to leverage shared information between processes. By identifying commonalities across multiple processes, such as common subprocesses, similar activity patterns, or shared resources, analysts can gain a deeper understanding of process performance and identify opportunities for improvement.
- ***Challenge 5. There are different separated representational models for event log and process model.*** As previously stated, event logs and process models represent distinct artifacts that are stored in different formats. While event logs are typically stored as text files, process models come in various formats, including BPMN. As a consequence, analyzing and mining these artifacts requires aligning and comparing them. If there were no fundamental differences in the representational models of event logs and process models, the analysis and mining of these artifacts would be much easier.

In this paper, we propose an ontology-based model to represent object-centric event logs as the state-of-the-art event log format. An ontology is a formal definition of concepts and their relationships in a specific domain with class and relation components defined based on a semantic language such as OWL. Therefore, the main contributions of our proposed model are:

1. ***Based on the proposed model, event log can be enriched by various domain ontologies and gain comprehensive knowledge.*** To facilitate the association between domain knowledge and event logs, one approach is to establish a *same-as* statement linking the concepts in the domain ontology and the event log ontology. This enables a seamless connection between the two, allowing for the efficient utilization of domain-specific information in log analysis. Additionally, it simplifies the task of interpreting event logs by

providing a clear understanding of the underlying concepts, leading to improved decision-making and problem-solving capabilities.

2. ***Semantic process mining.*** The proposed model, which is based on ontology, can contribute to performing semantic process mining tasks in two ways. First, by combining domain ontology and mapping common concepts, and second, by adding relationships between concepts and inferring knowledge.

3. ***Event log can be extended when data is generated on a continuous basis.*** Event logs play a crucial role in capturing and analyzing data from various sources. However, with the increasing volume of data generated on a continuous basis, it becomes imperative to extend the capabilities of event logs to accommodate this data. By doing so, organizations can gain deeper insights into their operations and make data-driven decisions in real-time. One way to achieve this is by extending event logs to capture data in real-time. This involves defining RDF triples that allow for seamless integration of new data into the existing event log. With this approach, any time a new event is recorded, the relevant information can be easily stored and used immediately. This capability not only facilitates real-time decision making but also ensures that the event log remains up-to-date and relevant.

4. ***It is possible to integrate all of an organization's process event logs into an ontology-based model.*** In this proposed model, each process is represented as a distinct log object, which can be defined using the appropriate ontology. Additionally, all the events associated with each process are linked to its corresponding log object using the *has_event* object property. The *same_As* relations between each pair of concepts in two separate logs enable the easy comparison and analysis of two distinct yet similar processes.

5. ***SPARQL can be used for querying and inferring new knowledge.*** The proposed representation model is stored in RDF format, which allows for the use of SPARQL to query and extract valuable information from the ontology. One possible application of this is to flatten the OCEL based on a particular object type, retrieve activity relations, and generate a straightforward process map. With the help of SPARQL, we can easily analyze the ontology and gain insights into the relationships between different components.

The rest of the paper is summarized as follows: The next section reviews the related work. Our proposed model is presented in Section 3. In section 4, we put our model into practice and explain how SPARQL can be used to query the event log. Finally, section 5 concludes the paper and discusses future work.

## 2. Related work

Previous works related to the topic can be classified into two main categories. The first category includes works that focus on the standardization and representation of event logs, which are the starting point for process mining. The second category includes works that aim to use ontology in process mining tasks, such as domain ontologies. These works address challenges related to enriching event logs with additional information and enhancing the accuracy of process mining results.

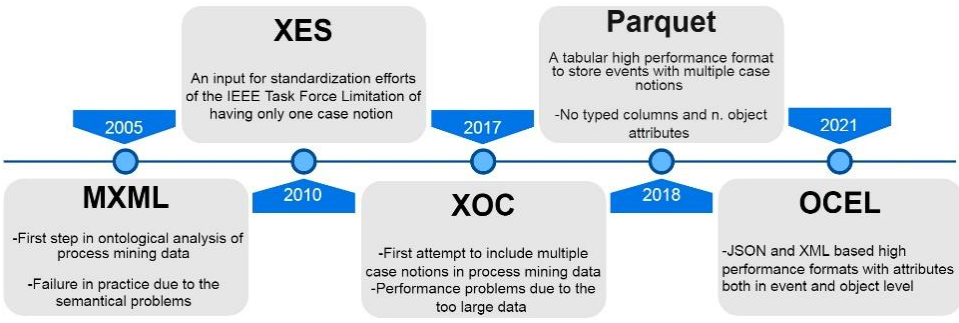


**Figure 1:** Event log storing standards over time.

## 1.1. Event log representation

Event logs are the primary input for process mining techniques. As shown in Figure 1, various standards have been proposed for storing event logs that have evolved over time [2], [3], [5]. Among all, XES is the most well-known format which has been accepted as IEEE standard in 2014.

Despite its popularity, the XES standard has a limitation in that it lacks support for multiple case notions, leading to problems such as divergence (an event linking to multiple cases) and convergence (an activity being executed repeatedly with the exact case notion). To address these issues, several models have been proposed. For example, the OpenSLEX model, proposed in [6], can generate different views from the database flexibly. Additionally, the XOC model has been suggested for storing object-centric event logs [7]. However, to address some of the problems of these models, such as performance, the OCEL model has been presented recently. This model focuses on serialization in JSON and XML formats, as well as tool support [5]. Graphs have been used as a structure for storing event log in some studies. Fahland D. proposed the event knowledge graph in [8] as a means of representing object-centric event logs through the use of *labeled property graphs*. The event knowledge graph model provides a powerful tool for visualizing the relationships between events, objects, and their properties, enabling a deeper understanding of complex event logs.

## 1.2. Ontology and process mining

Ontology has been applied in many areas of process mining so far. Semantic process mining is the primary context in which ontology is used. Ontology gives meaning to process mining and changes the process mining techniques from a label-based level to a concept-based level. In other words, the semantic process mining methods extract a model at the conceptual level, and some inference is possible because the event log is linked to the ontology [9], [10]. In their paper [9], Alves de Medeiros and van der Aalst discussed different scenarios that are relevant to semantic process mining. These scenarios include log inspection, log cleaning, model discovery, and conformance analysis.

The work by Okoye et al. [11] proposed a semantic framework to improve the outcomes of process mining techniques. In other words, their framework introduces an approach that uses activity semantics to make inferences.

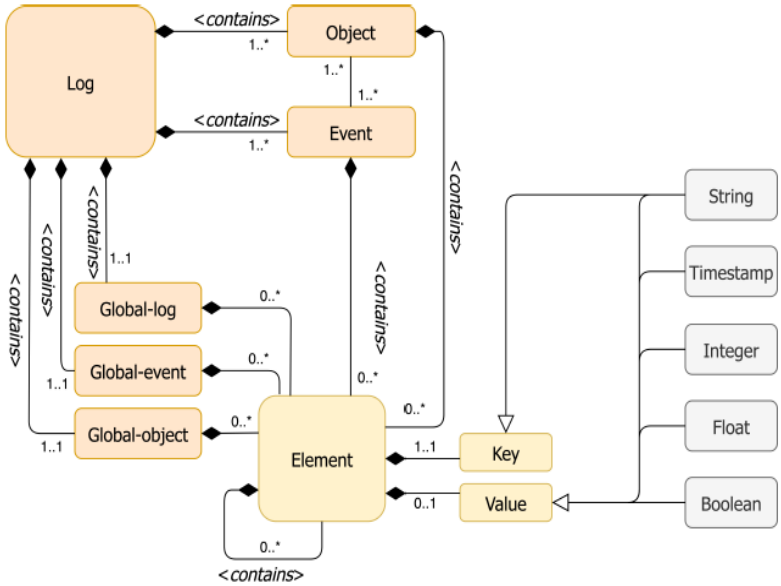


**Figure 2:** The UML class diagram for OCEL [5].

Some standard event log formats support semantic annotations by connecting event log labels to ontology semantic concepts. For example, *ModelReference* is an optional extra attribute in the SA-MXML format that links to a list of concepts in ontologies [12].

In addition, there is a lot of research that has used ontologies for other purposes. For example, Calvanese D. et al. in [13], introduced a technique for extracting event logs in XES format from relational databases through the use of an ontology. The authors in [14] and [15] utilized ontology to enhance the quality of the event log and repair activity labels. Another study conducted in [16] performs the segmentation of the process model with the help of an ontology by removing the semantic ambiguity of the log entities. Some other researchers utilized ontology to represent the process model. In [17], Leida M. et al. developed a representational model for real-time processes using the RDF language and its graph. In [18], [19], the authors proposed ontology-based business process representations that focus on the web service domain.

## 3. Ont_OCEL

In the real world, processes deal with various case notions, and considering them based on a single case notion can lead to problems during analysis [5]. For example, in a part of an O2C process, the two case notions of *order* and *item* are considered along with three activities (place order, check item, pack item). If we want to use process mining techniques, we can use two case notions. In an object-centric event log, we can store multiple case notions and capture complete process information.

Figure 2 shows the UML class diagram of the OCEL standard format. Log, events, and objects are the three main structures of this model. Each log instance includes events and objects set. Global entities (log, event, and object) indicate some default values for elements of the corresponding class. An event indicates an activity execution instance that contains an *identifier*, *activity name*, *timestamp*, and *related objects*. It may have some optional features, like a *resource*. Finally, object class has object instance information such as type (required) and color (optional)

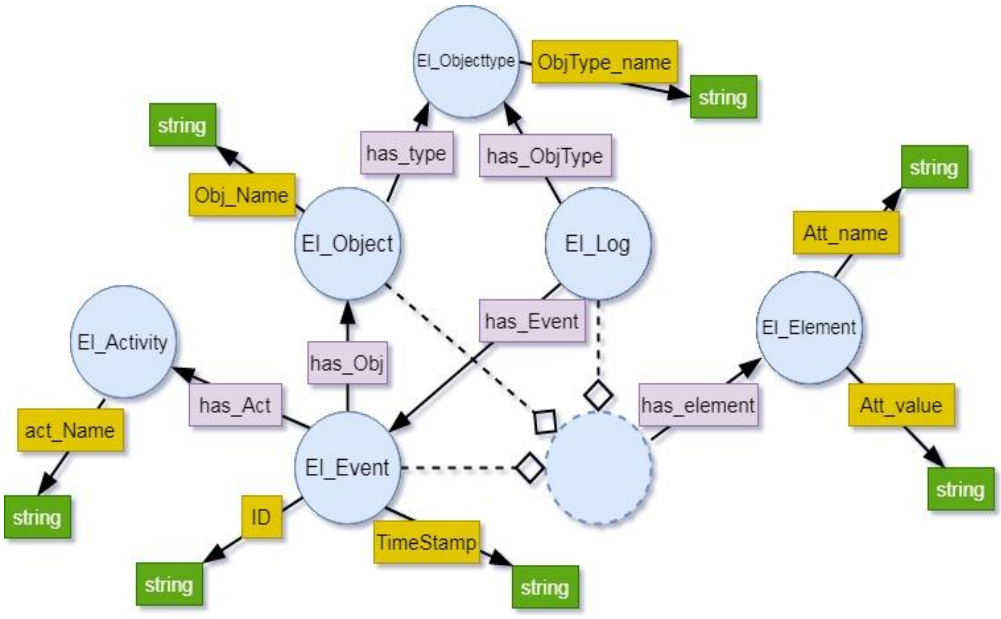


**Figure 3:** Proposed ontology schema to represent OCEL data(Ont-OCEL)

features.

The primary aim of this research is to propose a novel structure that can effectively capture and represent event log information, while also addressing the various challenges outlined in the introduction section. the use of the OCEL format in the presented model is a way to ensure that

the model is built on a solid and state-of-the-art foundation, and that it is capable of capturing the complexities of real-world processes. Moreover, we have leveraged the powerful capabilities of ontology as the foundational component of our proposed model, enabling us to effectively capture and represent the OCEL data. These capabilities include:

6. Improved data integration: Ontology provides a shared and standardized vocabulary and semantics for different domains, facilitating data integration across different systems and applications.
7. Enhanced data quality: By defining a standard vocabulary and semantics, ontology helps to reduce ambiguity, inconsistency, and redundancy in data, leading to improved data quality.
8. Improved search and retrieval: Ontology enables more accurate and efficient search and retrieval of information by providing a standardized vocabulary and semantics that enhances machine understanding of data.
9. Enabling automated reasoning: ontologies also facilitate inference, which is the process of deriving new knowledge from existing knowledge. By defining the relationships between different concepts within a domain, ontologies enable automated reasoning engines to infer new knowledge based on the existing knowledge represented in the ontology.

The OWL language can be used in cases where the information in documents must be processed by a machine [20]. OWL explicitly represents the meaning of terms and relationships between them, which is called ontology. The remainder of this section introduces the proposed ontology schema, which serves as the foundation for the event log representation. We present the key features and components of the ontology schema, along with the process of its development and construction.

The proposed ontology, as shown in Figure 3, has six classes:

10. **Log**: The log class is a collection of events and objects, each object can have a type. In the proposed model, different processes can be stored in an integrated manner, each of which belongs to an instance of the log class.
11. **Event**: An event represents part of a business process and is an activity that is done at a specified time and belongs to a log instance.
12. **Object**: As mentioned, each event in the object-centric event log model can be associated with more than one object so we have a collection of object types and their instances.
13. **Object type**: Any object has a type. We can filter event log based on these types and flatten the object-centric event log to have a classic event log based on one case notion.
14. **Activity**: As the performed activity in an event is highly important, therefore in the proposed model, we have considered it as an independent class.
15. **Element**: In the OCEL model, each feature is defined by a key-value pair in the element class, which is also defined in the proposed model and can be connected with other classes.

**Table 1**
The Object-properties and datatype-properties of the proposed ontology

| Property Name | Type | Domain | Range |
|---|---|---|---|
| has_event | object | el_log | el_event |
| has_element | object | el_object, el_event, el_log | el_element |
| has_type | object | el_object | el_objecttype |
| has_object | object | el_event | el_object |
| has_objecttype | object | el_log | el_objecttype |
| has_activity | object | el_event | el_activity |
| act_name | Data type | el_activity | xsd:string |
| ev_timestamp | Data type | el_event | xsd:dateTimeStamp |
| ev_id | Data type | el_event | xsd:string |
| obj_name | Data type | el_object | xsd:string |
| objtype_name | Data type | el_objecttype | xsd:string |

| | | | |
|---|---|---|---|
| att_key | Data type | el_attribute | xsd:string |
| att_value | Data type | el_attribute | xsd:string |

In the proposed model, event log optional attributes are represented as the element class but because of the importance of the default attribute, such as *Timestamp* for an event, we have considered them independently. The relationships between classes are implemented using object-property and use datatype-property to relate an individual to a value that has a data type. The complete list is given in Table 1.

After creating the ontology schema, all instances within the event log are incorporated into the ontology. As the event logs contain a large amount of information, this process should be automated. A program has been developed using the Python language which takes in the OCEL as input and adds the log instances to the defined ontology. The OCEL library [5], [21] has been utilized to import OCEL files and query log elements. Additionally, the OwlReady2 library [22], [23] has been utilized to construct and manipulate the ontology. To demonstrate the applicability of the proposed model, a real object-centric event log [21] with 11522 object instances and 22367 events has been employed.

## 4. Ont-OCEL in practice

In this section, we will demonstrate the application of our model and explains how SPARQL, a semantic query language, can be utilized to extract information from the event log. SPARQL is a query language that is specifically designed for querying data in RDF format [24]. With the help of SPARQL, we can extract relationships that are not explicitly stated in the event log, such as 'followed_by' and 'preceded_by', and utilize them to construct a process map. Moreover, to avoid the complexity of a spaghetti process map, we can apply filters based on the frequency of these relationships' occurrence.

**Query template 1:**

```
select fn:concat( (str(?obj_name) ,',', (str(?act_name )),',',?time)
where {
?event    syntax:type OCEL:el_event;
          OCEL: has_activity  ?act;
          OCEL: ev_timestamp ?time;
          OCEL:has_object  ?obj.
?act      OCEL:act_name  ?act_name.
?obj      OCEL:has_type ?Objtype;
          OCEL:obj_name ?obj_name.
?Objtype  OCEL:objtype_name "_OBJ_".{
```

**Table 2**
Selected Subset of Query Output Displaying flattened OCEL based on "orders" object type

| OrderID | Activity Name | TimeStamp |
|---|---|---|
| 990001 | place order | 5/20/2019 9:07 |
| 990001 | confirm order | 5/20/2019 11:13 |
| 990001 | pick item | 5/20/2019 11:20 |
| 990001 | item out of stock | 5/20/2019 13:54 |
| 990001 | reorder item | 5/21/2019 10:03 |

| | | |
|---|---|---|
| 990001 | item out of stock | 5/23/2019 10:40 |
| 990001 | pick item | 5/23/2019 11:00 |
| 990001 | pay order | 5/23/2019 11:27 |
| | . . . | |
| 990002 | place order | 5/20/2019 10:35 |
| 990002 | pick item | 5/20/2019 10:38 |
| 990002 | pick item | 5/20/2019 17:08 |
| 990002 | pick item | 5/20/2019 18:15 |
| 990002 | confirm order | 5/21/2019 12:19 |
| | . . . | |

In the proposed model, SPARQL queries have a range of applications, one of which is converting OCEL into a classical flattened event log. This can be achieved by creating an event log with a single case notion for each object type present in OCEL. To extract a flattened event log and generate a comma-separated values (CSV) file, we can utilize Query Template 1 by substituting "_Obj_" with the object type of our choice. For instance, in Query Template 1, we substituted the "_Obj_" parameter with the value "*orders*" and executed it on the running example, resulting in a flattened event log based on the "*orders*" object type. Table 2 displays a portion of the output.

As mentioned before, by utilizing SPARQL, it is possible to extract all activities that were performed immediately before or after a particular activity. This means that we can obtain the "followed_by" or "preceded_by" relations for all activities. Moreover, these extracted relations can be employed to query other significant relations such as parallelism, causality, and choice relations between two activities. To illustrate, in Query template 2, we can generate a "choice" relation triple for every pair of choice activity.

**Query template 2:**

```
Construct{ ?after_1  OCEL:Is_Choice ?after_2 }
Where {
?first OCEL: followed_by ?after_1;
OCEL: followed_by ?after_2.
FILTER NOT EXISTS
{{?after_1 OCEL:followed_by ?after_2.}
  UNION
 {?after_2 OCEL:followed_by ?after_1}}
FILTER (?after_1  != ?after_2)}
```

## 5. Conclusions

In this article, we introduced Ont_OCEL, a model for representing OCEL format based on ontology. Our proposed model has several advantages, including the ability to enhance the event log using domain ontology for knowledge inference. Furthermore, SPARQL can be utilized to create flattened event data, extract statistical data, extract activity relations, and build simple process maps. By adopting this approach, all event logs of an organization can be merged into a single ontology and updated with additional information as needed. We tested Ont_OCEL using a real-world object-centric event log and created an ontology to showcase its applicability.

Moving forward, the proposed model can be employed as a storage model in ontology-based process mining frameworks, enabling process mining tasks such as log quality enhancement, conformance checking, and process discovery. This paper provides a foundation for future research in this area and demonstrates the potential for Ont_OCEL to be used in process mining applications.

## 6. References


[1] W. van der Aalst, "Data Science in Action," in Process Mining: Data Science in Action, W. van der Aalst, Ed. Berlin, Heidelberg: Springer, 2016, pp. 3–23. doi: 10.1007/978-3-662-49851-4_1.

[2] H. M. W. Verbeek, J. C. A. M. Buijs, B. F. van Dongen, and W. M. P. van der Aalst, "XES, XESame, and ProM 6," in Information Systems Evolution, Berlin, Heidelberg, 2011, pp. 60–75. doi: 10.1007/978-3-642-17722-4_5.

[3] B. V. Dongen and W. M. P. van der Aalst, "A Meta Model for Process Mining Data," in EMOI-INTEROP, 2005.

[4] W. M. P. van der Aalst, "Object-Centric Process Mining: Dealing with Divergence and Convergence in Event Data," in Software Engineering and Formal Methods, Cham, 2019, pp. 3–25. doi: 10.1007/978-3-030-30446-1_1.

[5] A. Ghahfarokhi, G. Park, A. Berti, and W. Aalst, "OCEL: A Standard for Object-Centric Event Logs," Jul. 2021, pp. 169–175. doi: 10.1007/978-3-030-85082-1_16.

[6] E. Gonzalez Lopez de Murillas, "Process mining on databases: extracting event data from real-life data sources," Phd Thesis 1 (Research TU/e / Graduation TU/e), Technische Universiteit Eindhoven, Eindhoven, 2019.

[7] A. Berti and W. van D. Aalst, "Extracting Multiple Viewpoint Models from Relational Databases," presented at the 8th International Symposium on Data-Driven Process Discovery and Analysis (SIMPDA), Dec. 2018, vol. LNBIP-379, p. 24. doi: 10.1007/978-3-030-46633-6_2.

[8] D. Fahland, "Process Mining over Multiple Behavioral Dimensions with Event Knowledge Graphs," in Process Mining Handbook, W. M. P. van der Aalst and J. Carmona, Eds. Cham: Springer International Publishing, 2022, pp. 274–319. doi: 10.1007/978-3-031-08848-3_9.

[9] A. K. Alves de Medeiros and W. M. P. van der Aalst, "Process Mining towards Semantics," in Advances in Web Semantics I: Ontologies, Web Services and Applied Semantic Web, T. S. Dillon, E. Chang, R. Meersman, and K. Sycara, Eds. Berlin, Heidelberg: Springer, 2009, pp. 35–80. doi: 10.1007/978-3-540-89784-2_3.

[10] K. Okoye, A. R. H. Tawil, U. Naeem, and E. Lamine, "Semantic Process Mining Towards Discovery and Enhancement of Learning Model Analysis," in 2015 IEEE 17th International Conference on High Performance Computing and Communications, 2015 IEEE 7th International Symposium on Cyberspace Safety and Security, and 2015 IEEE 12th International Conference on Embedded Software and Systems, Aug. 2015, pp. 363–370. doi: 10.1109/HPCC-CSS-ICESS.2015.164.

[11] K. Okoye, S. Islam, U. Naeem, and S. Sharif, "Semantic-based Process Mining Technique for Annotation and Modelling of Domain Processes," International journal of innovative computing, information & control: IJICIC, vol. 16, pp. 899–921, Jan. 2020, doi: 10.24507/ijicic.16.03.899.

[12] A. K. A. de Medeiros, W. Van der Aalst, and C. Pedrinaci, "Semantic process mining tools: Core building blocks," 2008.

[13] D. Calvanese, M. Montali, A. Syamsiyah, and W. M. P. van der Aalst, "Ontology-Driven Extraction of Event Logs from Relational Databases," in Business Process Management Workshops, 2015. doi: 10.1007/978-3-319-42887-1_12.

[14] S. Sadeghianasl, A. H. Ter Hofstede, M. T. Wynn, S. Turkay, and T. Myers, "Process Activity Ontology Learning From Event Logs Through Gamification," IEEE Access, vol. 9, pp. 165865–165880, 2021.

[15] S. Ghalibafan, B. Behkamal, M. Kahani, and M. Allahbakhsh, “An ontology-based method for improving the quality of process event logs using database bin logs,” International Journal of Metadata, Semantics and Ontologies, vol. 14, no. 4, pp. 279–289, Jan. 2020, doi: 10.1504/IJMSO.2020.115436.
[16] A. Pourmasoumi, M. Kahani, E. Bagheri, and M. Asadi, “Process Fragmentation: An Ontological Perspective,” in Enterprise, Business-Process and Information Systems Modeling, Cham, 2015, pp. 184–199. doi: 10.1007/978-3-319-19237-6_12.
[17] M. Leida, B. Majeed, M. Colombo, and A. Chu, “A Lightweight RDF Data Model for Business Process Analysis,” Jan. 2013, vol. 162, pp. 1–23. doi: 10.1007/978-3-642-40919-6_1.
[18] D. Roman et al., “Web service modeling ontology,” Applied ontology, vol. 1, no. 1, pp. 77–106, 2005.
[19] I. Weber, J. Hoffmann, and J. Mendling, “Semantic business process validation,” in Proceedings of the 3rd International Workshop on Semantic Business Process Management (SBPM’08), CEUR-WS Proceedings, 2008, vol. 472.
[20] D. L. McGuinness, F. Van Harmelen, and others, “OWL web ontology language overview,” W3C recommendation, vol. 10, no. 10, p. 2004, 2004.
[21] “OCEL standard.” http://ocel-standard.org/ (accessed Mar. 14, 2022).
[22] J.-B. Lamy, “Owlready: Ontology-oriented programming in Python with automatic classification and high level constructs for biomedical ontologies,” Artificial Intelligence in Medicine, vol. 80, pp. 11–28, Jul. 2017, doi: 10.1016/j.artmed.2017.07.002.
[23] “Welcome to Owlready2’s documentation! — Owlready2 0.36 documentation.” https://owlready2.readthedocs.io/en/v0.36/ (accessed Mar. 14, 2022).
[24] E. D. Valle and S. Ceri, “Querying the Semantic Web: SPARQL,” in Handbook of Semantic Web Technologies, J. Domingue, D. Fensel, and J. A. Hendler, Eds. Berlin, Heidelberg: Springer, 2011, pp. 299–363. doi: 10.1007/978-3-540-92913-0_8.