Extraction of Object-Centric Event Logs through Virtual Knowledge Graphs

Extended Abstract

Jing Xiong1, Guohui Xiao2,3, Tahir Emre Kalayci4, Marco Montali1,3, Zhenzhen Gu1 and Diego Calvanese1,3,5

1KRDB Research Centre for Knowledge and Data, Free University of Bozen-Bolzano, 39100 Bolzano, Italy
2Department of Information Science and Media Studies, University of Bergen, 5007 Bergen, Norway
3Ontopic S.R.L., 39100 Bolzano, Italy
4Virtual Vehicle Research GmbH, 8010 Graz, Austria
5Department of Computing Science, Umeå University, 901 87 Umeå, Sweden

Abstract

Process mining is a family of techniques that supports the analysis of operational processes based on event logs. Among the existing event log formats, the IEEE standard eXtensible Event Stream (XES) is the most widely adopted. In XES, each event must be related to a single case object, which may lead to convergence and divergence problems. To solve such issues, object-centric approaches become promising, where objects are the central notion, and one event may refer to multiple objects. In particular, the Object-Centric Event Logs (OCEL) standard has been proposed recently. However, the crucial problem of extracting OCEL logs from external sources is still largely unexplored. In this paper, we try to fill this gap by leveraging the Virtual Knowledge Graph (VKG) approach to access data in relational databases. We have implemented this approach in the OnProm system, extending it from XES to OCEL support. The full version of this article has been submitted to an international conference.

Keywords

Process mining, object-centric event logs, virtual knowledge graph, ontology-based data access

1. Introduction

Process mining [1, 2] is a family of techniques relating the fields of data science and process management to support the analysis of operational processes based on event logs. To perform process mining, normally the algorithms and tools expect that the event logs are following certain standards. However, in reality, most IT systems in companies and organizations do not directly produce such logs, and the relevant information is spread in legacy systems, in particular, relational databases. Hence, event log extraction from legacy systems is a key enabler for process mining [3, 4, 5, 6].
There have been several proposals for the representation of event logs, e.g., eXtensible Event Stream (XES) [7], JSON Support for XES (JXES) [8], Open SQL Log Exchange (OpenSLEX) [9], and eXtensible Object-Centric (XOC) [10], where XES is the most adopted one, being the IEEE standard for interoperability in event logs [11]. In XES (and other similar proposals), each event is related to a single case object, which leads to problems with convergence (when an event is related to multiple cases and occurs repetitively) and divergence (when multiple events are in a single case and are hard to separate) [12]. To solve these issues, object-centric approaches become promising, where objects are the central notion, and one event may refer to multiple objects. In particular, along this direction, the Object-Centric Event Logs (OCEL) standard [13] has been proposed recently.

To the best of our knowledge, the crucial problem of extracting OCEL logs from external sources is still largely unexplored. The only exception is [14], where OCEL logs are extracted by identifying the so-called master and relevant tables in the underlying database and building a Graph of Relations (GoR). Though promising, this approach might be difficult to adopt when the underlying tables are complex and the GoR is hard to model, because it does not separate the storage level (i.e., the database) from the concept level (i.e., domain knowledge about events).

In this work, we try to fill this gap by leveraging the OnProm (http://onprom.inf.unibz.it/) framework [4, 5] for extracting event logs from legacy information systems. OnProm v1 was already relying on the technology of Virtual Knowledge Graphs (VKG) [15] to expose databases as Knowledge Graphs that conform to a conceptual model, and to query this conceptual model and eventually generate logs by using ontology and mapping-based query processing. It came with a toolchain to process the conceptual model, and to automatically extract XES event logs, by relying on the VKG system Ontop [16]. We present here OnProm v2, which we have modularized so that it becomes easier to extend, and in which we have implemented OCEL-specific features to extract OCEL logs.

2. The OnProm Framework for Event Log Extraction

We describe now the OnProm approach for event log extraction, as shown in Figure 1. To extract from a legacy information system $I = \langle R, D \rangle$, with relational schema $R$ and database $D$, event logs that conform to an event log standard $X$, OnProm works as follows:

(1) A domain ontology is a high-level abstraction of business logic concerned in a domain of interest. The user can design a domain ontology $T$ using the standard ontology language OWL 2 QL using any ontology editing tool, e.g., the Ontology Editor of the OnProm tool chain. Then the user creates a VKG mapping $M$ (using, e.g., the Ontop plugin for Protégé [17]) to declare how the instances of classes and properties in $T$ are populated from $I$. This step is only concerned with modeling the domain of interest and is agnostic to the event log standard.

(2) OnProm assumes that for the event log standard $X$, a specific (domain-independent) event ontology $E_X$ is available. The Annotation Editor of OnProm imports $E_X$, and allows the user to create annotations $L_X$, which are based on the classes in $E_X$, over the classes in $T$.

(3) OnProm assumes that for the standard $X$ also a set of SPARQL queries for extracting the log information is defined. By relying on a conceptual schema transformation approach [6]
and query reformulation of Ontop, using $\mathcal{L}_X$, $\mathcal{T}$, $\mathcal{M}$, and $\mathcal{R}$, these SPARQL queries are internally translated to SQL queries over $\mathcal{I}$. OnProm evaluates the generated SQL queries to construct corresponding Java objects and serialize them into log files compliant with $X$.

As mentioned, OnProm v1 only supported the XES standard. In this work, we have first modularized the system, by separating the above steps in different software components, so as to make it more extensible. Then we have introduced OCEL-specific features in Steps (2) and (3). Hence, OnProm v2 is now able to extract OCEL logs from relational databases.

Next we illustrate the functionality of OnProm for extracting OCEL logs through an example. The OCEL event ontology $\mathcal{E}_{\text{OCEL}}$ is a very simple ontology with only three classes: Object, Event, and Attribute. We consider Dolibarr [18] v14, a popular open source Enterprise Resource Planning (ERP) system. We have designed a Sale Orders domain ontology and the mapping in the Ontop system (Figure 2). We have then used the Annotation Editor of OnProm to annotate this ontology with $\mathcal{E}_{\text{OCEL}}$ classes (Figure 3). Based on the provided information, OnProm is able to extract OCEL logs automatically. Figure 4 shows a fragment of the extracted log in XML, and its graphical visualization.

**Acknowledgments**

This research has been supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation, by the Italian PRIN project HOPE, and by the EU H2020 project INODE, grant n. 863410.
Figure 2: Ontology and mappings shown in Ontop

Figure 3: OnProm Annotation Editor showing the annotated ontology of the Dolibarr ERP system

Figure 4: A fragment of the extracted OCEL log from the Dolibarr ERP system
References


