The onprom Toolchain for Extracting Business Process Logs using Ontology-based Data Access

Diego Calvanese, Tahir Emre Kalayci, Marco Montali, and Ario Santoso

KRDB Research Centre for Knowledge and Data
Free University of Bozen-Bolzano, Bolzano, Italy
{calvanese,tkalayci,montali,santoso}@inf.unibz.it

Abstract. Process mining techniques require the input data to be explicitly structured in the form of an event log. Unfortunately, in many real world settings, such event logs are not explicitly given, but they are implicitly stored in legacy information systems. Therefore, to enable process mining, there is a need to support the data preparation and the log extraction from legacy information systems. The onprom tool-chain aims at supporting users in the semi-automatic extraction of event logs from a legacy information system, reflecting different process-related views on the same data, and consequently facilitating multi-perspective process mining. The tool-chain is based on the ontology-based data access paradigm, and consists of three components, namely UML editor, annotation editor, and log extractor. Each component can be used both as a plug-in for the extensible process mining framework ProM, or within an integrated toolkit. The produced logs are fully compliant with the XES standard.

Keywords: data preparation for process mining, log extraction, event logs, XES format, ontology-based data access, conceptual modeling

1 Introduction

The emerging area of *process mining* has become increasingly popular both in academia and in industry to resolve the mismatch between process models and event data generated by process executions [1,2]. Process mining is a collection of techniques that combine model-based and data-oriented analysis to obtain useful insights on how business processes are executed in a real organizational environment in a synergic way. Through process mining, decision makers can *discover* process models from data, *compare* expected and actual behaviors, and *enrich* models with information obtained from their execution. The applicability of process mining depends on two crucial factors [2,5,6]: *(i)* the availability of high-quality event data, and of event logs containing correct and complete event data about which cases have been executed, which events occurred for each case, and when they took place; *(ii)* the representation of such data in a format that is understandable by process mining algorithms, such as the XML-based IEEE eXtensible Event Stream (XES) standard [8].

However, in many real world settings, the enterprise exploits functionalities offered by more general enterprise systems such as ERP, CRM, SCM, and other business suites.

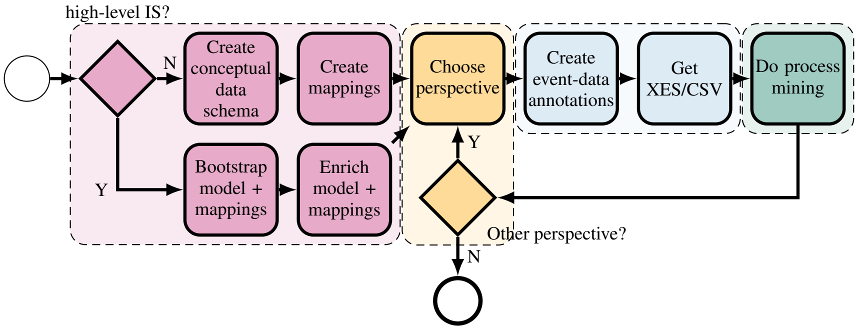

Fig. 1. The onprom methodology and its four phases [6,5]

In addition, such systems are typically configured for the specific needs of the company, and connected to domain-specific and other legacy information systems. In such a complex common setting, event data are not explicitly given, but have instead to be reconstructed by reconciling these different data sources and reflecting the perspective of interest. This is why there is a pressing need for techniques that are able to support data and process analysts in the data preparation phase, and in particular in the extraction of event data from legacy information systems [1]. Recently, we have proposed the onprom methodology [5,6,7] to support data and process analysts in the conceptual identification of event data, answering questions like: *(i)* Which are relevant concepts and relations? *(ii)* How do such concepts/relations map to the underlying information system? *(iii)* Which concepts/relations relate to the notion of case, event, and event attributes? Our methodology is backed up by a tool-chain that, once the aforementioned questions are answered, automatically extracts an event log conforming to the chosen perspective, and obtained by inspecting the data *residing at the sources*. This is done borrowing techniques from intelligent data management, and in particular by exploiting the ontology-based data access (OBDA) paradigm and tools [3,4,10].

A detailed account of the onprom framework is provided in [5].

2 The onprom methodology and tool-chain

The onprom methodology, illustrated in Figure 1, comprises four main phases. We assume the existence of a legacy information system $\mathcal{I} = \langle \mathcal{R}, \mathcal{D} \rangle$, with schema $\mathcal{R}$ and a set $\mathcal{D}$ of facts about the domain of interest. We consider the typical case where the information system is a relational database, hence $\mathcal{R}$ is a relational schema, and $\mathcal{D}$ is a relational database instance structured according to $\mathcal{R}$.

The first phase deals with the definition of an *OBDA model* towards understanding the data conceptually. In particular, the OBDA model serves a two-fold purpose [4,10]: *(i)* To provide a conceptual representation of the domain of interest, in terms of relevant concepts and relations that are understandable by domain experts. This is given in terms of a *conceptual data schema* $\mathcal{T}$; *(ii)* To explicitly link $\mathcal{I}$ to $\mathcal{T}$, by means of a declarative *mapping specification* $\mathcal{M}$, consequently allowing for querying $\mathcal{I}$ by using the vocabulary provided by $\mathcal{T}$. In onprom, we employ UML class diagrams as a concrete language for representing $\mathcal{T}$, and we rely on their logic-based encoding [4] in terms of the standard ontology language OWL 2 QL [9]. $\mathcal{M}$ consists of a set of logical assertions that map SQL queries over schema $\mathcal{R}$ to facts over $\mathcal{T}$.

Once the OBDA system is in place, onprom allows one to abstract away the information system. In this way, the analyst who is responsible for the data extraction can directly focus on $\mathcal{T}$, using the concepts and relations contained therein so as to concretely formulate which perspective has to be taken towards process mining. More specifically, this amounts to enrich $\mathcal{T}$ with annotations $\mathcal{L}$, each creating an implicit link between $\mathcal{T}$ and the core portion of the XES event log schema $\mathcal{E}$. In this light, each annotation expresses one of the following aspects: *(i) definition of a case*, indicating which class provides the basis to identify case objects, and which conditions have to be satisfied by instances of the selected class so as to classify them as case objects; *(ii) definition of an event*, indicating which class provides the basis to identify occurrences of such an event; *(iii) definition of an event attribute*, indicating which navigational route has to be followed within the diagram so as to fetch the value for such an attribute given an instance of the corresponding event. When all inputs are provided, event logs are extracted automatically by using the facilities provided by the OBDA paradigm, and in particular the query and mapping transformation techniques described in [7,5].

The onprom tool-chain supports the various phases of the methodology. It is open source and can be downloaded from `http://onprom.inf.unibz.it`. It is available as a stand-alone software, or as a set of plug-ins running inside the ProM process mining framework[1] using built-in data exchange mechanisms. It consists of the following components: *(i)* a *UML Editor* to model the conceptual data schema; *(ii)* an *Annotation Editor* to enrich the conceptual data schema with event-data annotations; *(iii)* a *Log Extractor* that extracts a XES event log from a given relational information system, exploiting the event-data annotations. Currently, the definition of the mapping specification is not covered within the onprom tool-chain, and we assume that it is realized manually or using third-party tools.

UML Editor. The UML Editor provides two main functionalities: modeling of a UML class diagram, and import/export from/to OWL 2 QL, leveraging the correspondence described in [4,5]. Currently, the editor makes some simplifying assumptions such as: *(i)* no support for *completeness* of UML generalization hierarchies (in line with the OBDA approach [4]); *(ii)* support for binary associations only, in line with Semantic Web languages; *(iii)* multiplicities in associations (resp., features) are restricted to be either *0* or *1*, allowing to express functionality and mandatory participation; *(iv)* no support for *IS-A* between associations.

The UML class diagrams produced with the UML Editor can be saved in a proprietary JSON format for further processing and as input for the Annotation Editor. It can also be exported as a standard OWL 2 QL ontology. The graphical layout information, which is not part of the OWL 2 QL language, is maintained in the form of OWL 2 annotations, thus resulting in an ontology fully compliant with the W3C standard. The editor also supports loading an existing conceptual data schema, if it is available as an

[1] `http://www.promtools.org`

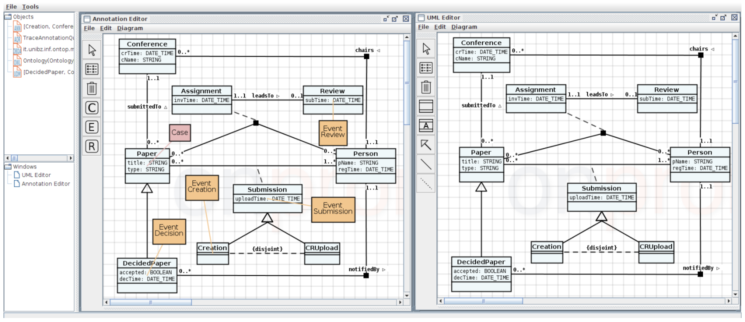

Fig. 2. The onprom toolkit screenshot showing five loaded objects and two open editors

OWL 2 QL ontology file. A UML Editor loaded with a conceptual data schema example is shown in the right part of the onprom toolkit screenshot in Figure 2.

Annotation Editor. The Annotation Editor supports analysts in the specification of event-data annotations on top of a UML class diagram. An annotated conceptual data schema loaded in the Annotation Editor is shown in the left part of Figure 2. To simplify the annotation task, the editor supports the following advanced operations. First, it allows one to choose properties and paths using navigational selections over the diagram via mouse-click operations. Second, it takes multiplicities on associations and attributes into account; when the user is selecting properties of the case and of events (in particular the timestamp), the editor enables only navigation paths along functional associations, thus guaranteeing that a unique data value is reached. The annotations are automatically translated into corresponding SPARQL queries by the editor. The annotated conceptual data schema can be exported using a proprietary JSON format, which can then be imported by the Log Extractor.

Log Extractor. The last component of the tool-chain implements the mapping synthesis technique as well as the XES log extraction functionalities, by leveraging the state-of-the-art OBDA system ontop[2] [3] and the OpenXES APIs[3]. To this end, the Log Extractor takes as input: *(i)* a conceptual data schema (generated via the UML Editor or represented as an OWL 2 QL file), *(ii)* an OBDA mapping specification, linking the conceptual data schema to the underlying relational database, and *(iii)* event-data annotations (created using the Annotation Editor). Given these inputs, the extractor works in two steps: *(i)* it automatically creates a *new mapping specification*, which establishes a direct correspondence between the database and the XES event log schema, and *(ii)* uses it to extracts a XES event log from the database. The extracted XES event log can be used with all the tools that are supporting the XES standard.

Experiments. We have evaluated the tool using a machine with processor Intel Core i5 2.4 GHz, 8 Gb of RAM, and Postgres as the DBMS. One of the experiments shows that it takes ∼11 minutes to produce a XES event log with 298 224 traces, 2 117 466 events, and 6 352 398 event attributes (∼450 MB for the whole XES log).

[2] http://ontop.inf.unibz.it
[3] http://www.xes-standard.org

3 The Demo

In our demo, we illustrate how to successfully employ the onprom tool-chain to tackle the challenging problem of data preparation for process mining on top of legacy databases. Specifically, *(i)* we start from a legacy relational database, *(ii)* use the UML Editor to conceptually model the data of interest, *(iii)* exploit the ontop mapping editor for Protégé[4] to link the resulting conceptual schema to legacy data via declarative mappings, and *(iv)* use the Annotation Editor to equip the schema with declarative annotations that indicate where cases, events, and their attributes are "located". In this way, we provide the necessary input to the Log Extractor. The screencasts of our demo are available at

`http://onprom.inf.unibz.it/index.php/screencasts`.

As mentioned, the onprom toolchain is open source and can be downloaded from

`http://onprom.inf.unibz.it`.

Acknowledgements. This research has been partially supported by the Euregio IPN12 *KAOS (Knowledge-Aware Operational Support)* project, which is funded by the "European Region Tyrol-South Tyrol-Trentino" (EGTC) under the first call for basic research projects, and by the UNIBZ internal project *OnProm (ONtology-driven PROcess Mining)*.

References

1. van der Aalst, W., et al.: Process mining manifesto. In: Proc. of the Business Process Management Int. Workshops. LNBIP, vol. 99, pp. 169–194. Springer (2012)
2. van der Aalst, W.M.P.: Process Mining - Data Science in Action. Springer, 2nd edn. (2016)
3. Calvanese, D., Cogrel, B., Komla-Ebri, S., Kontchakov, R., Lanti, D., Rezk, M., Rodriguez-Muro, M., Xiao, G.: Ontop: Answering SPARQL queries over relational databases. Semantic Web J. 8(3), 471–487 (2017)
4. Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Poggi, A., Rodriguez-Muro, M., Rosati, R.: Ontologies and databases: The *DL-Lite* approach. In: RW 2009 Tutorial Lectures. LNCS, vol. 5689, pp. 255–356. Springer (2009)
5. Calvanese, D., Kalayci, T.E., Montali, M., Santoso, A.: OBDA for log extraction in process mining. In: RW 2017 Tutorial Lectures, LNCS, vol. 10370, pp. 292–345. Springer (2017)
6. Calvanese, D., Kalayci, T.E., Montali, M., Tinella, S.: Ontology-based data access for extracting event logs from legacy data: The onprom tool and methodology. In: 20th Int. Conf. on Business Information Systems. LNBIP, vol. 288, pp. 220–236. Springer (2017)
7. Calvanese, D., Montali, M., Syamsiyah, A., van der Aalst, W.M.P.: Ontology-driven extraction of event logs from relational databases. In: Proc. of the 11th Int. Workshop on Business Process Intelligence. LNBIP, vol. 256, pp. 140–153. Springer (2016)
8. IEEE Computational Intelligence Society: IEEE Standard for eXtensible Event Stream (XES) for achieving interoperability in event logs and event streams. IEEE Std 1849-2016 (2016)
9. Motik, B., Cuenca Grau, B., Horrocks, I., Wu, Z., Fokoue, A., Lutz, C.: OWL 2 Web Ontology Language profiles (second edition). W3C Recommendation, W3C (Dec 2012), available at `http://www.w3.org/TR/owl2-profiles/`
10. Poggi, A., Lembo, D., Calvanese, D., De Giacomo, G., Lenzerini, M., Rosati, R.: Linking data to ontologies. J. on Data Semantics X, 133–173 (2008)

[4] `http://protege.stanford.edu`