

An Open-Source Integration of Process Mining Features into the Camunda Workflow Engine: Data Extraction and Challenges

Alessandro Berti*, Wil van der Aalst*, David Zang†, Magdalena Lang†

*Process and Data Science Department, RWTH Aachen University
Process and Data Science department, Lehrstuhl für Informatik 9 52074 Aachen, Germany
Emails: a.berti@pads.rwth-aachen.de, wvdaalst@pads.rwth-aachen.de

†viadee Unternehmensberatung AG
Konrad-Adenauer-Ufer 7, 50668

Emails: david.zang@viadee.de, magdalena.lang@viadee.de

Abstract—Process mining provides techniques to improve the performance and compliance of operational processes. Although sometimes the term “workflow mining” is used, the application in the context of Workflow Management (WFM) and Business Process Management (BPM) systems is limited. The main reason is that WFM/BPM systems control the process, leaving less room for flexibility and the corresponding deviations. However, as this paper shows, it is easy to extract event data from systems like Camunda, one of the leading open-source WFM/BPM systems. Moreover, although the respective process engines control the process flow, process mining is still able to provide valuable insights, such as the analysis of the performance of the paths and the mining of the decision rules. This demo paper presents a process mining connector to Camunda that extracts event logs and process models, allowing for the application of existing process mining tools. We also analyzed the added value of different process mining techniques in the context of Camunda. We discuss a subset of process mining techniques that nicely complements the process intelligence capabilities of Camunda. Through this demo paper, we hope to boost the use of process mining among Camunda users.

Index Terms—Process Mining; Workflow Management; Data Extraction and Preprocessing; Process Engine

I. INTRODUCTION

The vast majority of business processes (including enterprise resource planning, customer relationship management, document management) are nowadays supported by information systems. These systems manage (but not always regulate) the execution of a business process, and record *event data* with fine detail about each step of the process. In this context, process mining [1] allows to improve operational processes by exploiting the event data recorded by such systems. An *event log* can be extracted from an information system’s database in order to apply the process mining algorithms. An event log contains event data of multiple executions of the business process. Process mining techniques include: *process discovery*, i.e., the automated discovery of a process model from event data; *conformance checking*, i.e., the comparison between a process model and the event data; *model enhancement*, i.e., the enrichment of the model with additional perspectives (for example, execution guards [2]), prediction and simulation algorithms. Open-source software supporting process mining includes ProM, APROMORE and PM4Py.

Process mining techniques have been applied to Workflow Management (WFM) and Business Process Management (BPM) systems. There are connectors to the YAWL WFM system [3] and some other WFM/BPM systems, including Signavio, Bizagi, and Bonita, that allow to extract the event data and operate on the process models contained in such systems. This paper focuses on Camunda and is a result of a collaborative project between the RWTH Aachen University and *viadee Unternehmensberatung AG*. Before, there was no open-source connector to extract logs useful for process mining purposes from Camunda, although Camunda is open-source and holds detailed event data. Therefore, we developed and evaluated such a connector, and *viadee* has integrated event log extraction techniques in its software stack.

Camunda is widely used, e.g., by Deutsche Telekom, Warner Music, Allianz, DB, Zalando and Generali. Camunda uses the BPMN 2.0 notation for modeling. Among the main selling points of Camunda are high throughput and collaboration and integration possibilities. Camunda can be easily integrated with different information systems, business intelligence and big data systems such as QlikView, Apache Spark and Kafka. This explains our goal to provide process mining for the large Camunda user base.

In this demo paper, (A) we present our implementation of a process mining extractor for Camunda, that is able to extract a set of event logs for the processes executed by Camunda, and (B) we discuss the existing process mining techniques that complement the business intelligence capabilities of Camunda. Figure 1 provides an overview of the approach implemented in the paper. The extractor is publicly available. For demonstrative purposes, it is integrated with a graphical interface based on PM4Py that offers basic process mining functions. Moreover, the techniques analyzed in this paper are available in open-source software.

The remainder of this demo paper is organized as follows. Section II describes the basic structure of the Camunda database, along with a methodology of extraction of event logs and process models from the Camunda database. Section III discusses the added value of process mining techniques for Camunda users. Section IV describes the set-up of the tool.

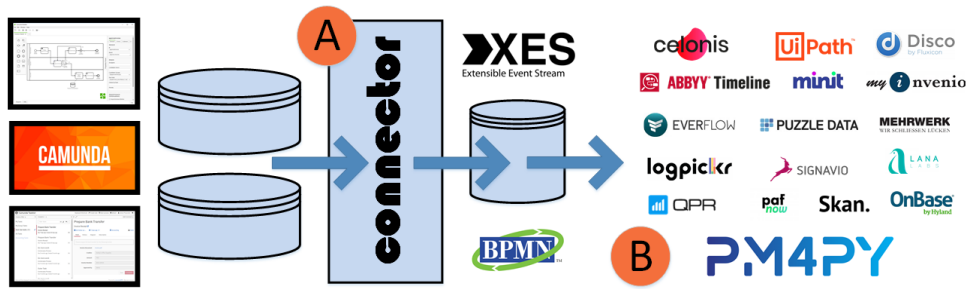


Fig. 1. Overview of the toolchain supporting process mining in the context of Camunda. In this paper, we provide (A) a connector to Camunda that is able to extract event logs and the BPMN diagrams modeling the processes. (B) an overview on the most valuable process mining techniques complementing Camunda. Although the connector is generic, we showcase the integration using PM4Py.

Finally, Section V concludes the paper.

II. EXTRACTING EVENT LOGS AND PROCESS MODELS FROM CAMUNDA

In this section, we will focus on how to extract logs containing the historical executions of the processes supported by Camunda, and how to extract the process models of such processes.

1) *Extracting Event Logs*: The extraction is done directly at the database level. The Camunda workflow engine supports different relational databases (e.g., PostgreSQL, Oracle, MySQL). We will focus exclusively on the completed executions, and ignore ongoing executions.

The table containing the historical executions of the processes is the `ACTI_HI_ACTINST` table. The rows of this table are the events thrown by Camunda. The table contains all the basic information that is needed to extract event logs:

- The identifier of the process that is executed is stored in the `proc_def_key_` column. This column contains as many different values as processes are executed via the Camunda process engine.
- The identifier of the process execution (case ID) is stored inside the `proc_inst_id_` column.
- The name of the BPMN element that are executed via the Camunda process engine is stored inside the `act_name_` column.
- The type of the BPMN element is stored inside the `act_type_` column.
- The start and end timestamps are stored inside the `start_time_` and the `end_time_` columns, respectively.
- The identifier of the resource that performs the event is stored inside the `assignee_` column.

Basically, an event log is created for each distinct value of the `proc_def_key_` column. The resulting table for an individual process is enough to analyze the control flow of the process and its bottlenecks. Other attributes at the event level can be obtained by merging the `ACT_HI_ACTINST` table with the `ACT_HI_DETAIL` table. The latter contains a row for each distinct attribute that is associated with an event.

These attributes can be useful to investigate the process more thoroughly, also for predictive analyses.

An important point is that also the traversal of gateways and internal or boundary events are included in the event log. So, not only the tasks are recorded, but the exact path of the model. This can simplify the frequency or performance decoration of the process model: performing token-based replay or alignments to find the path that is followed is not necessary. A postprocessing activity is only necessary when the execution of tasks needs to be analysed.

We will present the implementation of an connector in Section IV. A property of the connector is that it is incremental: the first extraction extracts all the events from the beginning of the time, while the following extractions extract only the events that are inserted since the previous extraction. This permits to keep the log updated, keeping the workload low.

2) *Extracting Process Models*: Aside from event logs, we can also extract the BPMN models of the processes supported by the workflow engine. In a Tomcat distribution of Camunda, each process supported by Camunda has its own folder in `PATH-TO-CAMUNDA-SERVER/webapps/`. As example, if a process has name `invoice`, its corresponding folder is `PATH-TO-CAMUNDA-SERVER/webapps/invoice`. To extract the BPMN model associated to the invoice process, the content of the `PATH-TO-CAMUNDA-SERVER/webapps/invoice/WEB-INF/classes` folder should be taken. As another possibility, we could refer to querying the REST API for the process diagram.

The extracted BPMN models can be imported in different process mining tools. In order to perform analyses such as decision mining and conformance checking (see Section III), the BPMN model should be converted to a Petri net model. This is difficult for many constructs (for example, OR-joins and OR-splits, swimlanes, subprocesses) and thus can lead to problems for complex real-life processes. An overview of the problematics of conversion from a BPMN model to a Petri net is found in [4].

III. PROCESS MINING ON TOP OF CAMUNDA

In the previous section, we have described an approach to extract event logs for the different processes supported

TABLE I

ANALYSIS OF THE PROS AND CONS OF THE APPLICATION OF SEVERAL PROCESS MINING TECHNIQUES IN THE CONTEXT OF THE CAMUNDA ENGINE. MANY OBSERVATIONS HERE HOLD GENERALLY FOR ANY WFM/BPM SYSTEM.

Technique	Pros	Cons
Process Discovery	It is possible to show the frequent paths in processes. Moreover, it becomes visible when people bypass the system.	The process model underlying the event data is already contained in Camunda and probably not surprising.
Conformance Checking	It is possible to measure the precision of the process model, in order to understand how much extra behavior is allowed.	It is expected that the event data already follows the model. Hence, some measures such as the calculation of fitness are not useful.
Decision Mining	It is possible to enrich the BPMN model with guards that describe and regulate the behavior of the process at the decision points.	Many execution guards are already inserted in the BPMN diagrams during the design phase. The discovered guards might be trivial or overfit the data.
Concept Drift Analysis	Process mining can be used to detect process changes. Possible reasons include changes of the process model or day-night shifts.	Many of the possible change points are known or deliberate.
Prediction of the Remaining Time	The technique provides an estimation of the remaining time for the process instances, in order to detect possible service level agreements violations.	The quality of the predictions performed by state-of-the-art approaches on real datasets must be improved.
Social Network Analysis	The collaboration between the resources can be analyzed from different angles (e.g., to see the effect on performance).	Roles are often set and controlled by the system.
Model Enhancement	It is possible to identify the bottlenecks of the process, and the most frequent paths.	Basic performance measurements are already provided by the WFM/BPM system.

by Camunda. This enables the application of several process mining tools and techniques. In this section, we want to analyze which process mining techniques are most useful in the context of the Camunda processes. The techniques are implemented and released as open-source software, including the one based on PM4Py presented in Section IV. Table I provides an overview of the approaches, along with their pros and cons.

1) *Process Discovery and Conformance Checking*: The two most popular process mining disciplines are process discovery and conformance checking. The scope of application of process discovery is pretty limited, since the event data contained in the database is regulated by the process models inserted in Camunda. A BPMN model is also a formal model that enables the application of conformance checking techniques. For less regulated processes, the goal of conformance checking is to identify deviations in the process model, and the executions of the process are evaluated by their *fitness* according to the process model. For WFM/BPM systems, we can expect to have perfect fitness for all the process executions. However, another application of conformance checking is the measurement of *precision*. A model is precise when it does not allow for extra behavior, i.e., behavior that does not appear in the event data. Models can have low precision when they flexibly allow the execution sequence of activities. Hence, measuring precision can provide a measure for the “flexibility” of the process model. A popular measure for precision is proposed in [5].

2) *Decision Mining*: The application of a *decision mining* technique allows to enrich the model with *execution guards* that are extracted automatically from the event data. These are conditions that are required in order to execute a path in the model. Hence, decision mining helps to reduce the amount of behavior allowed in the process model by an adaptation of the model towards the guards that are discovered by the technique. A mature approach for decision mining is proposed in [2]. On the other hand, BPMN models are often already decorated

with execution guards that are defined in the design phase. Hence, decision mining could end up finding exactly the same guards without adding anything new. Another problem is the discovery of trivial guards, or guards that overfit the data. A careful selection of the guards is necessary after performing the analysis.

3) *Concept Drift Analysis*: A *concept drift analysis* allows to identify the points in time where the execution of the process changes. Different types of concept drifts exist: sudden drifts (where the process becomes immediately significantly different), gradual drifts and seasonal drifts. An approach for the detection of concept drifts is described in [6]. While the technique is interesting, many concept drift points are already known in the context of WfMSs as Camunda: for example, the underlying BPMN schema changes, or there are differences in the execution of a process between day and night.

4) *Prediction*: Given an incomplete process execution, it may be useful to estimate the remaining execution time based on historical executions. Several approaches have been proposed [7], [8], however in our experiments the quality of predictions on top of real-life datasets is still not completely satisfying. Moreover, this paper only covers the extraction of complete (historical) process instances.

5) *Other Analyses*: In this category, we include *social network analysis* [9], that with different metrics (such as the handover of work, the working together, the similar activities metric) calculates the collaboration between the different organizational resources. *Model enhancement* with frequency and performance metrics is particularly important to identify the bottlenecks of the process (from a performance point of view) and the most frequent paths.

IV. SET-UP OF THE CONNECTOR

The connector presented in this section is completely open-source and is available at

<https://github.com/Javert899/incremental-camunda-parquet-exporter>. A completely working demo environment can be easily obtained by using *docker-compose* inside the folder of the project¹. In the prepared environment, there are:

- A PostgreSQL relational database that is supporting the Camunda BPM engine and is exposed at port 5432.
- The Camunda BPM engine, that is running at port 8080. The Camunda interface can be reached at <http://localhost:8080/camunda-welcome/index.html>. The installation contains some demonstrative models and event data that can be extracted by the connector and used for process mining analysis.
- The connector, that is written in the Python language and is configured to reach the PostgreSQL database (for the extraction of the event data) and the Camunda BPM docker container (to extract the BPMN models).
- An open-source process mining solution [10], [11], with its own event logs database, that is reachable at port 80, providing *admin/admin* as access credentials for the interface. The services and the web interface are offering the logs for all the processes contained in Camunda. In the demo interface, a single process is offered to the user.

The processes contained in Camunda are offered, through the connector, in the web interface, graphically allowing for the following operations:

- Process discovery of a directly-follows graph, and of a process tree or Petri net discovered using the inductive miner algorithm. While the model itself is already known, the frequency and performance information are important to understand which parts of the process are more critical for key performance indicators and service level agreements.
- Cases Exploration: understanding which cases have longer duration, and which are the events of such cases.
- Social Network Analysis: shows the interaction between the organizational resources using the Camunda BPM engine through some classic metrics.

The deployment of the connector through *docker-compose* is integrated with the process mining tool, but the event log is available for usage in other process mining platforms. An example log that is extracted by the technique is available at <http://www.alessandroberti.it/invoice.xes>.

V. CONCLUSION

In this paper, we presented a tool to extract event logs and process models from the Camunda system and analyzed the applicability of process mining tools on such models and logs. Thereby, process mining comes into reach of all organizations using Camunda with almost no effort. The extractor we implemented was also integrated with the open-source process mining tool PM4Py, along with instructions

¹The command *docker-compose up* starts the docker containers that are referred in the *docker-compose.yml* file

on how to deploy a complete environment that contains Camunda supported by the PostgreSQL database, the process mining tool, and the extractor. The deployment shows how Camunda can be extended with process mining capabilities in a straightforward way. While the PM4Py deployment is for demonstrative purposes, the extractor can be used on real-life deployments of Camunda and combined with any process mining tool. An example use case for our tool is proposed at http://www.alessandroberti.it/only_appendix.pdf.

Next to process discovery and conformance checking, our integration allows to discover the bottlenecks of the processes and to identify execution guards that are not contained in the process model, but implicitly assumed by the resources performing the process. Thereby, the analysis can help to improve the quality of the BPMN model. Other analysis, such as the detection of concept drifts, and the prediction of the remaining time, can also be useful.

As a result of this project, process mining techniques have been integrated in the *viadee Unternehmensberatung AG* software stack. The project located at <https://github.com/viadee/camunda-kafka-polling-client> proposes an implementation of a polling client on top of the Apache Kafka event streaming platform to poll data from Camunda, and the project located at <https://github.com/viadee/bpmn.ai> proposes a data preparation pipeline for such process data.

REFERENCES

- [1] W. van der Aalst, *Process mining: discovery, conformance and enhancement of business processes*. Springer, 2011, vol. 2.
- [2] M. De Leoni and W. van der Aalst, "Data-aware process mining: discovering decisions in processes using alignments," in *Proceedings of the 28th annual ACM symposium on applied computing*. ACM, 2013, pp. 1454–1461.
- [3] A. Rozinat, M. Wynn, W. van der Aalst, A. Ter Hofstede, and C. Fidge, "Workflow simulation for operational decision support using yawl and prom," *BPM Center Report BPM-08-04*, *BPMcenter.org*, vol. 298, pp. 302–306, 2008.
- [4] R. M. Dijkman, M. Dumas, and C. Ouyang, "Semantics and analysis of business process models in bpmn," *Information and Software technology*, vol. 50, no. 12, pp. 1281–1294, 2008.
- [5] J. Muñoz-Gama and J. Carmona, "A fresh look at precision in process conformance," in *International Conference on Business Process Management*. Springer, 2010, pp. 211–226.
- [6] R. J. C. Bose, W. van der Aalst, I. Žliobaitė, and M. Pechenizkiy, "Handling concept drift in process mining," in *International Conference on Advanced Information Systems Engineering*. Springer, 2011, pp. 391–405.
- [7] M. Polato, A. Sperduti, A. Burattin, and M. de Leoni, "Time and activity sequence prediction of business process instances," *Computing*, vol. 100, no. 9, pp. 1005–1031, 2018.
- [8] N. Tax, I. Verenich, M. La Rosa, and M. Dumas, "Predictive business process monitoring with lstm neural networks," in *International Conference on Advanced Information Systems Engineering*. Springer, 2017, pp. 477–492.
- [9] W. van der Aalst, H. A. Reijers, and M. Song, "Discovering social networks from event logs," *Computer Supported Cooperative Work (CSCW)*, vol. 14, no. 6, pp. 549–593, 2005.
- [10] A. Berti, S. J. van Zelst, and W. van der Aalst, "Process Mining for Python (PM4Py): Bridging the Gap Between Process-and Data Science," in *ICPM Demo Track (CEUR 2374)*, 2019, p. 1316.
- [11] A. Berti, S. van Zelst, and W. van der Aalst, "PM4Py Web Services: Easy Development, Integration and Deployment of Process Mining Features in any Application Stack," in *BPM Demo Track*, 2019.