PyStack't: Real-Life Data for Object-Centric Process Mining

Lien Bosmans[1], Jari Peeperkorn[2] and Johannes De Smedt[2]

[1]Independent researcher, Herent (Belgium)
[2]Research Center for Information Systems Engineering (LIRIS), KU Leuven, Leuven (Belgium)

Abstract

The availability of representative event logs is a prerequisite for algorithmic design and evaluation of novel (object-centric) process mining techniques. This work presents PyStack't, a Python package that supports data preparation for object-centric processing mining. It provides predefined data transformations that extract process data from publicly available APIs (GitHub) and export it to different OCED formats (OCEL 2.0, EKG). In addition, it includes summary statistics and interactive graph visualizations for data exploration. By tailoring to newcomers in the field and focusing on integrations with other open-source tools, this contribution aims to strengthen the emerging OCPM (tool) ecosystem.

Keywords

object-centric process mining, event logs, event log pre-processing, process visualisation

1. Introduction

Process mining research heavily relies on frequently re-occurring event logs used for algorithmic design and evaluation. While useful for benchmarking, this recycling of event logs is also done out of necessity. Most process data must be kept confidential, and the pre-processing needed to create a novel event log requires a considerable investment of time and effort. Object-centricity adds further complexity; storing object-centric event data (OCED) cannot be done efficiently within a single CSV or related tabular file, and OCED formats are less widespread than their case-centric counterpart XES. This could act as a barrier of entry to the field and potentially slow down progress. PyStack't is a Python package that extracts real-life process data from APIs and stores it as OCED in a database file. In the current version, this is limited to collaborative processes in GitHub code repositories. After extraction, the process data can be exported to popular OCED formats, such as OCEL 2.0[1] and event knowledge graphs (EKG). To support preliminary analysis, data exploration functionality is included as well.

To our best knowledge, PyStack't is the only open-source tool that can generate novel object-centric event logs from real-life data without requiring any data mapping from users. While a number of commercial tools offer predefined data transformations for popular data sources such as SAP or Salesforce, open-source process mining tools generally only accept input data stored in a compatible (object-centric) event log. This leaves it up to users to either invest significant effort to do the required data transformations themselves, or to limit the choice to available datasets only. By providing export functionality to different OCED formats, PyStack't offers integration with other (open-source) tools that focus on the subsequent steps of object-centric process mining (OCPM), strengthening the emerging ecosystem.

The design of PyStack't is modular, enabled by the Stack't relational schema, to ensure that new features are automatically compatible with existing functionality [1]. User friendliness is also a key consideration, illustrated by the choice for simple function calls and the inclusion of extensive documentation. This tailors the tool towards people who want to get started with object-centric process

Proceedings of the Best BPM Dissertation Award, Doctoral Consortium, and Demonstrations & Resources Forum co-located with 23rd International Conference on Business Process Management (BPM 2025), Seville, Spain, August 31st to September 5th, 2025.
*This work was supported in part by the Research Foundation Flanders (FWO) under Project 1294325N.
lienbosmans@live.com (L. Bosmans); jari.peeperkorn@kuleuven.be (J. Peeperkorn); johannes.desmedt@kuleuven.be (J. De Smedt)
https://github.com/LienBosmans/ (L. Bosmans); https://jaripeeperkorn.github.io/ (J. Peeperkorn)
0009-0007-5624-3975 (L. Bosmans); 0000-0003-4644-4881 (J. Peeperkorn); 0000-0003-0389-0275 (J. De Smedt)

[1]https://www.ocel-standard.org/

mining, such as students or new practitioners. However, we hope that the permissive license also motivates more experienced people to adapt the tool to their specific needs.

Motivating the study of collaborative software development with OCPM

Open-source software projects that reach a certain maturity and size are often developed and maintained by a small core team together with a broad group of contributors. This community effort is supported by various processes. Some of those are described in publicly available contribution guidelines.[2] However, a large part remains invisible, buried deep in the activity logs of numerous issues, pull requests, and releases.

We believe this data could provide an interesting source to study collaborative processes with process mining, for example but not limited to: resolution time for reported bugs, consistency and quality of the review process, or retention of contributors and how their contributions to the project (e.g., bug reports, documentation improvements, code development) evolve over time. We consider OCPM a well-aligned choice because of the connections between issues (bug reports and requests for new or improved functionality), pull requests (submitting new contributions for review), the core team of maintainers, the larger community of contributors, the code inside the repository, and possible dependencies on other software.

We expect some learnings from successful open-source projects could be transferred to business environments, since the collaboration dynamic of a small group of payroll employees supported by various contract developers can resemble that of an an open-source community project.

Currently, PyStack't can only extract activity data from GitHub code repositories. Choosing GitHub as a first data source is motivated by multiple reasons; its API is well documented, its issue tracker creates sufficient digital traces to be studied with object-centric processing mining, and GitHub hosts a wide variety of substantial open-source code repositories.

2. Features

PyStack't is published on PyPi: https://pypi.org/project/pystackt/. The features of the current version 0.1.0 can be divided into three categories, as visualized in figure 1. A video that demonstrates the different functionalities is available at https://youtu.be/AS8wI90wRM8.

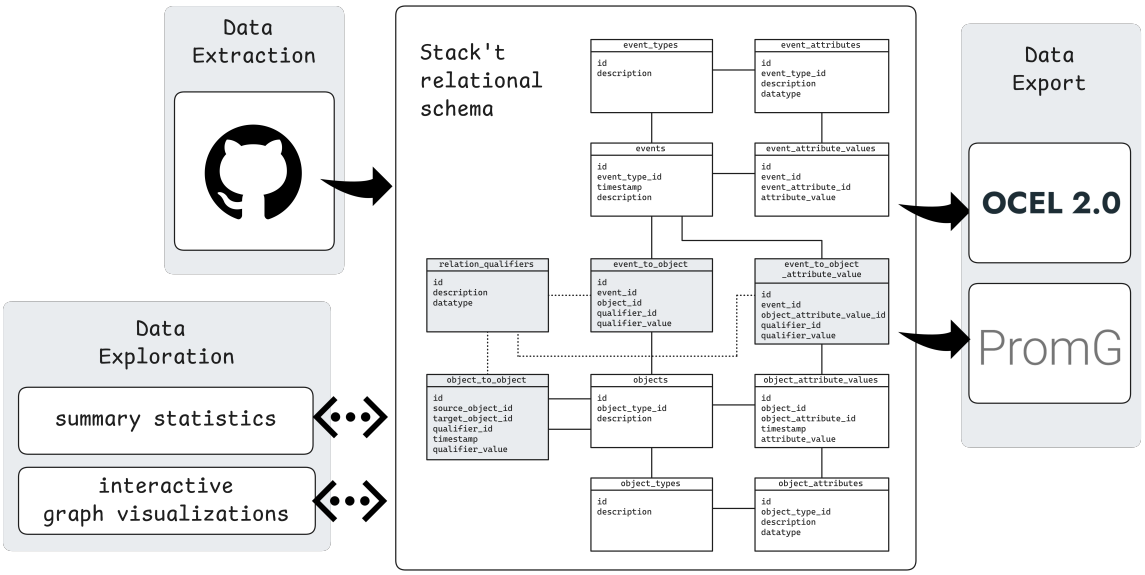

Figure 1: The design of PyStack't is modular, enabled by the Stack't relational schema [1].

[2] An example is the pandas contributing guide (https://pandas.pydata.org/docs/dev/development/contributing.html).

Data Extraction

- `get_github_log`: Extracts activity data linked to a code repository using the GitHub API. Includes predefined data mapping for multiple API responses[3] to the Stack't relational schema, an object-centric event data format. Output is stored in a DuckDB[4] database file.

Data Export

- `export_to_ocel2`: Maps object-centric event data to the OCEL 2.0 format [2]. The result is stored in a SQLite database file compatible with tools such as Ocelot[5] and OCPQ [3].
- `export_to_promg`: Generates a folder structure consisting of CSV and JSON files that can be ingested by PromG [4] to build an event knowledge graph.

Data Exploration

- PyStack't offers a local interactive data visualization app. Users can view and interact with event traces for any combination of objects with a filter on included event and object types.
- `create_statistics_views`: Supports initial analysis with predefined views that contain summary statistics.

3. Stability and Coverage

Feature	Stability	Scope
Extract OCED from GitHub repository	Reliable: includes error handling, tested for large datasets, no known bugs	Supports all GitHub repos, limited customization
Export to OCEL 2.0	Reliable: validated compatibility with other tools, tested for large datasets, no known bugs	Any process data in Stack't relational schema can be exported
Export to PromG	Experimental: first version, output requires user validation	Any process data in Stack't relational schema can be exported
Generate summary statistics	Reliable: tested for large datasets, no known bugs	Only includes basic statistics
Interactive data visualization	Usable for small to medium sized datasets	Attributes are not yet included, limited customization

4. Documentation

Extensive documentation is hosted at: https://lienbosmans.github.io/pystackt/. Each feature is described in a separate page, including:

- an example code snippet;
- descriptions of input parameters and expected function behavior;
- additional instructions, e.g., how to generate a GitHub access token or view data stored in a DuckDB database file;
- overview of extracted data, including descriptions of event/object types, relations, and attributes (if applicable);
- links to relevant information, such as GitHub data policies.

[3] Examples of such API responses can be found at https://api.github.com/repos/LienBosmans/stack-t/issues/33, https://api.github.com/repos/LienBosmans/stack-t/issues/33/timeline and https://api.github.com/user/6475031.

[4] https://duckdb.org/

[5] https://ocelot.pm/

5. Use Case

To demonstrate PyStack't, the `pandas` repository (github.com/pandas-dev/pandas) was used as a data source. During the data extraction, intermittent save functionality mitigated the risk of forced system restarts and GitHub API outages. The activity data of 57,806 GitHub issues could be extracted. Two issues were skipped due to a 404 status message, indicated by a warning message in the log.

The output is a DuckDB database file containing 1,151,801 events (37 types) with 370,529 event attributes values (37 attributes), 253,857 objects (4 types) with 763,455 object attribute values (15 attributes), 2,484,082 event-to-object relations, and 68,796 object-to-object relations.

To generate a smaller dataset for additional testing, the `pm4py` repository (github.com/process-intelligence-solutions/pm4py) was used. Activity data for all 523 issues could be extracted. The output file contains 3,919 events (21 types) with 1,559 event attributes values (21 attributes), 1,673 objects (4 types) with 5,107 object attribute values (15 attributes), 8,685 event-to-object relations, and 559 object-to-object relations.

5.1. Approximate run times

Function	Approximate run time[6]	Comment
`get_github_log`	29 hours, 10 minutes (`pandas`), 10 minutes (`pm4py`)	Limited by GitHub API rate limits. Outputs DuckDB file of 87.7 MB (`pandas`), 3.5 MB (`pm4py`)
`export_to_ocel2`	20 seconds (`pandas`), 5 seconds (`pm4py`)	Outputs SQLite file of 227 MB (`pandas`), 1 MB (`pm4py`). Ocelot accepts `pm4py` but fails with *Out of Memory* error for `pandas`. OCPQ can load both.
`export_to_promg`	22 seconds (`pandas`), 3 seconds (`pm4py`)	Outputs folder structure of 268 MB (`pandas`), 1 MB (`pm4py`)
`create_statistics_views`	< 1 second (both)	Data size does not affect the creation of a database view.
`prepare_graph_data`	10 seconds (`pandas`), < 1 second (`pm4py`)	Needed once before running visualization app.
`start_visualization_app`	5 seconds initial load time (`pm4py`)	App freezes when attempting to load `pandas` dataset.

5.2. Interactive data exploration

The application generates interactive graph visualizations for the selected objects. Objects can be searched, sorted and selected in the table at the top. Users can opt to only include a subset of event types and object types using the check boxes on the left. A detailed description of all components is available in the documentation.

6. Conclusion

This work presents PyStack't, a Python package that supports data preparation for object-centric process mining. We demonstrated its ability to generate novel OCED logs in different formats by extracting activity data from the GitHub repositories of `pandas` and `pm4py`. An interactive application for data exploration was presented as well. Given the need for more real-life datasets, we believe this to be a valuable addition to the (open-source) OCPM tool ecosystem.

[6] Measured on laptop with Intel(R) Core(TM) i7-8565U processor and 16 GB RAM.

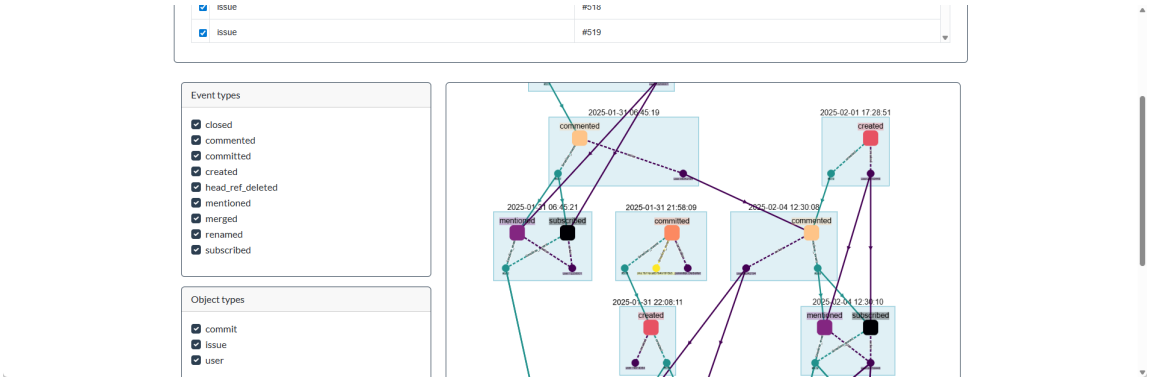

Figure 2: Screenshot of interactive data exploration with `pm4py` dataset.

Maturity PyStack't is a relatively new Python package, first released in February 2025, that can reliably create OCED logs with over a million events. Not all features have the same level of maturity; a detailed overview can be found in section 3.

Future Roadmap We are motivated to extend PyStack't with additional tool integrations and improved support for creating real-life OCED datasets. Concretely, we are working on below features.

- Improved PromG integration.
- Functionality to manipulate datasets, e.g., create filtered dataset, combine different datasets, rename types.
- More responsive and user-friendly UI for interactive visualizations.
- Research additional data sources to include.

Declaration on Generative AI

During the preparation of this work, ChatGPT was used to generate a list of writing prompts. After using this service, the authors answered these prompts and combined the replies into a first draft.

References

[1] L. Bosmans, J. Peeperkorn, A. Goossens, G. Lugaresi, J. De Smedt, J. De Weerdt, Dynamic and scalable data preparation for object-centric process mining, arXiv preprint arXiv:2410.00596 (2024). URL: https://arxiv.org/abs/2410.00596.

[2] A. Berti, I. Koren, J. N. Adams, G. Park, B. Knopp, N. Graves, M. Rafiei, L. Liß, L. T. G. Unterberg, Y. Zhang, et al., Ocel (object-centric event log) 2.0 specification, arXiv preprint arXiv:2403.01975 (2024).

[3] A. Küsters, W. M. van der Aalst, Ocpq: Object-centric process querying and constraints, in: International Conference on Research Challenges in Information Science, Springer, 2025, pp. 383–400.

[4] A. Swevels, E. L. Klijn, D. Fahland, Object-centric process mining (and more) using a graph-based approach with promg., in: ICPM Doctoral Consortium/Demo, 2023.