The Social Process Mining Cockpit: A Collaboration Pattern Detection Tool for Enterprise Collaboration Systems

Jonas Blatt[1], Patrick Delfmann[1], Martin Just[1] and Petra Schubert[1]

[1] *University of Koblenz, Universitätsstraße 1, Koblenz, Germany*

Abstract

Social Process Mining (SPM) combines the research fields Process Mining (PM) and Social Collaboration Analytics. The SPM-Cockpit is a prototype that can detect and analyze Collaboration Patterns in event logs of Enterprise Collaboration Systems (ECS) semi-automatically. We use this tool to investigate activity patterns that occur while users work collaboratively in ECS. To do so, we adapt methods from Process Mining, Frequent Subgraph Mining, and Graph Clustering and bundle them in the SPM-Cockpit. The prototype is a first step towards investigating, understanding and categorizing collaboration activity in ECS automatically.

Keywords

Social Process Mining, Collaboration Pattern, Enterprise Collaboration Systems, Process Mining

1. Introduction

In recent years, many workers have shifted to a *remote workplace* [1]. At least the COVID-19 pandemic led companies to invest in collaborative software to stay competitive in such a remote work environment [2]. Employees had to work together in an efficient manner so that they still fulfilled their workload. Enterprise Collaboration Systems (ECS) provide functionality, such as wikis, forums, file sharing, or blog components, that improves collaborative work in an enterprise environment. Such components encourage a flexible digital workplace where collaboration is essential to the daily routine. In general, in an ECS, users work with *social documents* (e. g., wikis articles, blog entries, forums threads, or text files) [3].

Process Mining (PM) is an established research domain that analyzes event logs extracted from software systems [4]. Such event logs contain records that describe *what activities* happened at *what time* in the system during a particular *process instance* [5]. Additional event log information may describe the process context, for instance, the user who executed the activities, data used as in- or outputs for activities, and further environmental information. PM techniques use such event logs to discover process models, check the conformance of processes with prescribed behavior, or provide means for process enhancement [4]. As ECS also log event data, we can apply PM in principle. However, due to the inherent malleability of ECS, users use such systems in an ad-hoc manner, fulfill their tasks in an unstructured order, and are not bound to a governing process [3]. For instance, a user may first create a blog post and then update a wiki page, or the user does this the other way around. Thus, logs can be unstructured and complex, and applying PM algorithms to native ECS event data may result in so-called spaghetti models, which are hard to interpret, even for domain experts.

The SPM-Cockpit aims to detect patterns of collaborative activities of users in ECS. SPM is the acronym for *Social Process Mining*, which combines the research fields of *Process Mining* and *Social Collaboration Analytics* in ECS. As there are many steps and several methods involved in the

Proceedings of the Best BPM Dissertation Award, Doctoral Consortium, and Demonstrations & Resources Forum co-located with 21st International Conference on Business Process Management (BPM 2023), Utrecht, The Netherlands, September 11th to 15th, 2023.

jonasblatt@uni-koblenz.de (J. Blatt); delfmann@uni-koblenz.de (P. Delfmann); martinjust@uni-koblenz.de (M. Just); schubert@uni-koblenz.de (P. Schubert)

process of SPM, we need a tool that bundles these methods and guides the users in their application to ease access. Thus, the SPM-Cockpit provides tailored methods for preprocessing log data, for the discovery of process models, frequent subgraph mining and graph set clustering algorithms for the detection of collaboration patterns, and for creating collaboration pattern repositories (explained in the following section). Furthermore, the cockpit guides the users through the SPM process. The outcomes of the SPM-Cockpit are collections of *collaboration patterns* (CPs) that help to describe collaboration. CPs are subsections of process models that can be found frequently in process models mined from ECS logs. The primary target user for the SPM-Cockpit is an ECS analyst or a researcher who aims to improve the collaborative work processes of ECS. Thus, s/he can use the cockpit to mine CPs from ECS process models, which describe the collaborative interactions of different users with social documents. These outcomes illustrate the typical collaboration activities in ECS. Eventually, the CPs can be used as building blocks that can be embedded in Business Process Management environments to improve collaborative business processes and ECS.

In the remainder, we describe the SPM-Cockpit and its features in Section 2. Section 3 describes the maturity of the tool. Finally, Section 4 concludes the demo with an outlook to future work with further feasible improvements.

2. Tool Description & Features

The SPM-Cockpit is a web application that guides the user through the process of SPM. It is built with the *Django* web-framework[2] and uses the following complementary technologies: a relational database (MariaDB®) for storing configurations and user sessions, a *Redis*® database[3] for caching traces, a *Neo4j*®[4] graph database for storing the collaboration pattern repositories, and *Celery*[5] for distributing computationally expensive tasks with task queues. In addition, the application uses the process mining library *pm4py* [6] for handling event logs and applying the process discovery algorithms. We deploy the presented main components within docker containers. Figure 1 shows the basic architecture of the application with its containers and the data flow between them.

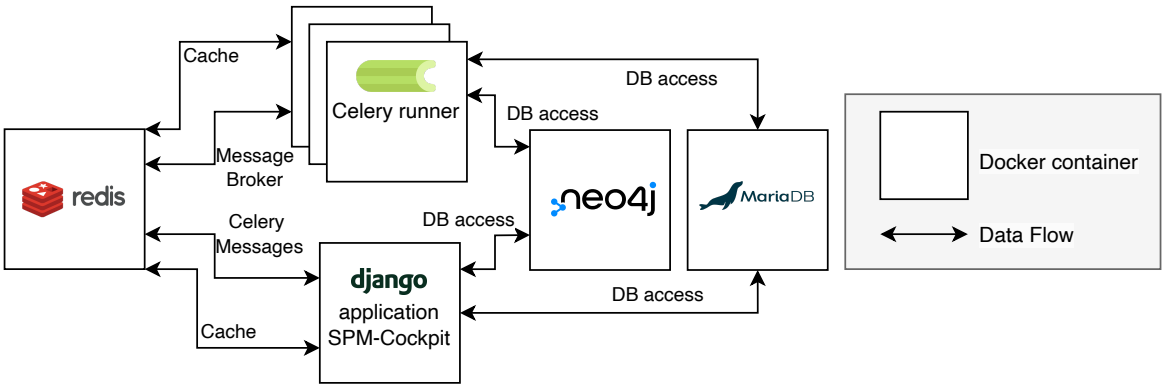

Figure 1: SPM-Cockpit Architecture

The BPMN model (Figure 2) shows the underlying process of the application with the SPM functionalities as activities and steps. If the results of an intermediate step do not satisfy the expected outcome, the user can step back and reconfigure the parameters and redo the corresponding activity (e. g., redo the process mining discovery with another algorithm). The SPM-Cockpit divides the proposed analysis into six main activities: 1. Importing an Event Log, 2. Preprocessing & Process Mining Configuration, 3. Exploring Process Models, 4. Frequent Subgraph Mining,

[2] https://www.djangoproject.com (last access: 24th of July, 2023)
[3] https://redis.io (last access: 24th of July, 2023)
[4] https://neo4j.com (last access: 24th of July, 2023)
[5] https://docs.celeryq.dev/en/stable (last access: 24th of July, 2023)

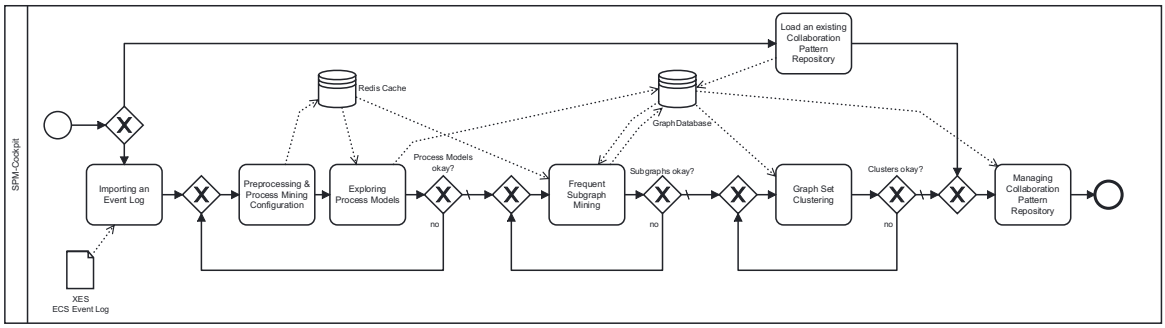

Figure 2: SPM-Cockpit Process

5. Graph Set Clustering, and 6. Managing Collaboration Pattern Repositories. The entry point of the SPM-Cockpit is the landing page with an overview of these SPM steps (Figure 3a). The following paragraphs explain each activity and describe their purpose, functions, and in- and outputs.

The first step is *Importing an Event Log*. ECS event logs have some special properties that are relevant for the further SPM process. We assume that the baseline case (the process instance) for such an event log is based on the so-called *workspace* (i. e., a particular working environment in an ECS that was created, for instance, for a project). For example, all events that were fired in one workspace belong to one trace in the event log. We further require the events to have the following attributes: the associated *social document*, the *executing user* (resource), and *the timestamp* when the event was fired. As *eXtensible Event Stream (XES)* [7] is the standard for event logs in the PM domain, we require this as input format.

After the upload is finished, the tool starts the second activity (preprocessing) with the default parameters. Thus, the second step, *Preprocessing & Process Mining Configuration*, triggers the preprocessing phase and defines how the process models should be mined. To detect which persons are working together on which social documents, the traces of the workspaces are further split into *sub-traces*. The intent is that we split each trace (the workspaces) into smaller traces based on those social documents that are jointly worked on by several users. Then, for each sub-trace, we identify ECS users who have worked in the same timeframe (of the trace) on the same documents and group those traces with direct or indirect overlaps in common users and timeframes. Thus, we create multiple sub-logs (the groups of sub-traces) based on the collaborative activities of the ECS users. Then, for each of these sub-logs, we can mine process models that express the collaboration. The process models are generated based on the PM configuration. For now, the cockpit user can decide whether to discover *Directly Follows Graphs* [8] or to discover process models using the *Heuristic Miner* [9]. However, further algorithms can be easily included in future versions of the SPM-Cockpit. The cockpit user can view the resulting process models in the next step.

The intermediate step, *Exploring Process Models*, lets the cockpit user investigate whether the generated process models are sufficient for further analysis. Therefore, the SPM-Cockpit provides

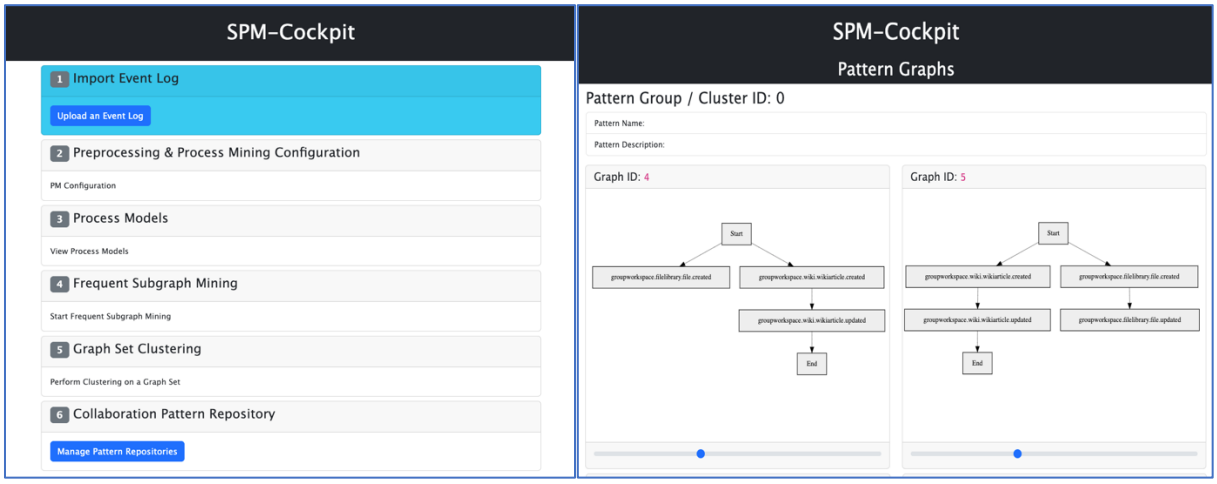

Figure 3: SPM-Cockpit Screenshots: a) Landing Page b) Patterns with Wiki and File activities

visualizations of the generated process models with basic navigation, pan and zoom functionality. The user can inspect the process models and step back if s/he has concerns regarding the PM configuration.

The *Frequent Subgraph Mining* step uses the previously generated process models and searches for patterns using Frequent Subgraph Mining (FSM) [10]. The idea is that through the discovery of frequent subgraphs, we identify patterns, which occur regularly when users work collaboratively on social documents in ECS. The result is a set of collaboration patterns. The current implementation of the FSM algorithm is *gSpan* [11], which searches for frequent substructures in graphs by building a depth-first search tree while adding forward and backward edges. This algorithm needs three parameters: The *min* and *max* number of vertices determine the min and max number of nodes (i. e., process activities in our context) and the *support* parameter defines how often the respective subgraph should occur to be present in the result set. The user defines these parameters and starts the FSM algorithm. Afterwards, s/he can inspect the resulting subgraphs and can step back if the subgraphs are too small, or the number of results is too small or too high. If this is the case, s/he can re-run the FSM algorithm with reconfigured parameters.

During the *Graph Set Clustering* step, similar patterns (subgraphs) are clustered into groups. Such groups then represent similar collaboration patterns rather than identical ones. The reason is that we assume that a typical collaboration scenario does not always look the same but rather similar. These groups/clusters can be viewed in the next step in the Collaboration Pattern Repository. The user can choose a *hierarchical* clustering algorithm or a *partitioning* clustering algorithm. Where the first requires the analyst to define the number of clusters, the latter finds the number of clusters based on the dissimilarities of the given graph set[6]. Again, based on the results, the user can redo the clustering process with a new configuration.

In the last step, *Managing Collaboration Pattern Repository*, the user can investigate the resulting clusters, which we store in a Collaboration Pattern Repository (CPR). Each cluster consists of a set of patterns (subgraphs) and can be visualized with a representative graph (e. g., the initial graphs from the clustering step). Additionally, the user can add a name and a description for the respective cluster as they represent similar collaboration patterns. The analyst can now use these patterns for further analysis that can provide insights into how the collaboration was conducted in the ECS. An example of two found patterns is illustrated in Figure 3b. These show the collaborative activities of users working on a wiki page and a file. As the CPRs are stored in a graph database (Neo4j), other existing CPRs can be loaded and inspected as well. Furthermore, the SPM-Cockpit provides the functionality for down-/uploading the CPRs.

3. Maturity of the Tool

We deployed one instance of the SPM-Cockpit[7] on a server. Furthermore, we published the related user documentation of the SPM-Cockpit[8] and the technical documentation of the underlining Python library[9]. We provide some ECS event logs, which one can select in the first step as sample data (instead of uploading an event log) directly in the SPM-Cockpit. We successfully applied the SPM-Cockpit with these event logs, which we demonstrate in the SPM-Cockpit documentation as a walkthrough video tutorial[10].

4. Conclusion and Future Work

The SPM-Cockpit is a tool for the analysis of ECS event data and provides a bundled toolset for the discovery and investigation of collaboration patterns. Thus, this tool extends the analysis of

[6] Additional clustering parameters can be defined by the user, but we refer here to the documentation.
[7] https://w3id.org/spm/cockpit (last access: 24th of July, 2023)
[8] https://w3id.org/spm/docs/cockpit (last access: 24th of July, 2023)
[9] https://w3id.org/spm/docs/colpadef (last access: 24th of July, 2023)
[10] https://w3id.org/spm/docs/cockpit/tutorial & https://youtu.be/4PYqSRKEPDw (last access: 24th of July, 2023)

unstructured processes in (social) document-orientated systems. For this purpose, we apply algorithms from the PM domain, graph-based algorithms, and clustering algorithms. For future improvements, we aim to:

- improve the preprocessing by including a parameter for temporal overlaps of the sub-traces to refine the sub-log generation.
- include further process discovery algorithms.
- extend subgraph discovery by implementing a *relaxed* Frequent Subgraph Mining algorithm that also finds subgraphs that are similar to the search pattern rather than being isomorphic. This way, similar subgraphs that would normally have a too low support in a "classic" FSM will not be missed.
- implement and evaluate further clustering algorithms to find the best one for SPM.
- apply existing CPRs to analyze unseen ECS event logs, while calculating metrics based on the patterns and the unexplored event log.

While we focus on ECS event data in SPM, further systems logs may be investigated with the help of the SPM-Cockpit in future research.

Acknowledgments

This work was partly funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – project number 445182359.

References

[1] S. P. Williams and S. Grams, "Remote Working Study 2022,", 2022, [Online]. Available: https://kola.opus.hbz-nrw.de/frontdoor/index/index/docId/2310

[2] Gartner, "Gartner Forecasts Worldwide Social Software and Collaboration Market to Grow 17% in 2021." 2021. [Online]. Available: https://www.gartner.com/en/newsroom/press-releases/2021-03-23-gartner-forecasts-worldwide-social-software-and-collaboration-market-to-grow-17-percent-in-2021

[3] S. P. Williams, J. Mosen, and P. Schubert, "The Structure of Social Documents," in *Proceedings of the Annual HICSS*, 2020, pp. 2825–2834.

[4] W. van der Aalst *et al.*, "Process Mining Manifesto," in *International conference on business process management*, 2012, pp. 169–194.

[5] J. De Weerdt and M. T. Wynn, "Foundations of Process Event Data," in *Process Mining Handbook*, Springer Science and Business Media Deutschland GmbH, 2022, pp. 193–211.

[6] A. Berti and S. J. van Zelst, "Process Mining for Python (PM4Py): Bridging the Gap Between Process- and Data Science," in *Proceedings of the ICPM Demo Track 2019*, Aachen, Germany, 2019, pp. 13–16.

[7] IEEE Computational Intelligence Society, *Standard for eXtensible Event Stream (XES) for Achieving Interoperability in Event Logs and Event Streams*. 2016. [Online]. Available: https://ieeexplore.ieee.org/servlet/opac?punumber=7740856

[8] W. M. P. van der Aalst, Ed., "Foundations of Process Discovery," in *Process Mining Handbook*, in Lecture Notes in Business Information Processing, vol. 448. Springer International Publishing, 2022, pp. 37–75.

[9] A. Weijters, W. M. van Der Aalst, and A. A. De Medeiros, "Process mining with the heuristics miner-algorithm," *Tech. Univ. Eindh. Tech Rep WP*, vol. 166, no. July 2017, pp. 1–34, 2006.

[10] C. Jiang, F. Coenen, and M. Zito, "A survey of frequent subgraph mining algorithms," *Knowl. Eng. Rev.*, vol. 28, no. 1, pp. 75–105, 2013

[11] X. Yan and J. Han, "gSpan: graph-based substructure pattern mining," in *2002 IEEE International Conference on Data Mining, 2002. Proceedings.*, IEEE Comput. Soc, pp. 721–724.