Discovering Process Models of Different Granularity from Legacy Software Systems

Marius Breitmayer[1,*,†], Lisa Arnold[1,†], Stephan La Rocca[2,†] and Manfred Reichert[1,†]

[1]Institute of Databases and Information Systems, Ulm University, Germany
[2]PITSS GmbH Stuttgart, Germany

Abstract

Processes models shall enable a better understanding and management of business processes as well as the information systems implementing these processes. Usually, different stakeholders of various enterprise levels are interested in process models which raises specific requirements concerning process model abstraction. While business managers are interested in high-level (i.e., abstract) process views, process participants need more fine-grained process views during process enactment. This also applies to many legacy software systems that implement business processes, but were originally not designed to provide process model details. In this paper, we present an approach for preprocessing event logs obtained from legacy software systems such that process models of different granularity levels can be discovered from them.

Keywords

process mining, event log, preprocessing, process model abstraction

1. Introduction

The organizational structure of an enterprise may be considered as a set of methods through which the organization is logically divided into distinct sets of business functions. The latter need to be harmonized across multiple granularity levels (e.g., different hierarchy levels of the organisational structure) to achieve enterprise goals [1, 2]. Moreover, the required tasks are often structured in terms of process models, which are used by various stakeholders from different enterprise hierarchy levels (e.g., managers, executives, or employees). Usually, these stakeholders have different expectations concerning the level of granularity in which a process model shall be displayed. For example, specific data required in the context of a particular process task might not be relevant for managers, but are crucial for process participants to correctly perform their tasks at runtime. Consequently, process models of different granularity levels need to be discovered to meet those expectations.

Process mining and process discovery, respectively, leverage recorded audit or workflow data (i.e., event logs) to reconstruct the business processes implemented by enterprise information

15th Central European Workshop on Services and their Composition, February 16–17, 2023, Hannover, Germany
*Corresponding author.
†These authors contributed equally.
marius.breitmayer@uni-ulm.de (M. Breitmayer); lisa.arnold@uni-ulm.de (L. Arnold); slarocca@pitss.com (S. La Rocca); manfred.reichert@uni-ulm.de (M. Reichert)
0000-0003-1572-4573 (M. Breitmayer); 0000-0002-2358-2571 (L. Arnold); 0000-0003-2536-4153 (M. Reichert)

S. Böhm and D. Lübke (Eds.): 15th ZEUS Workshop, ZEUS 2023, Hannover, Germany, 16–17 February 2023, published at http://ceur-ws.org

systems [3, 4]. In general, the discovered process models are strongly correlated with both the existence of suitable event logs and their quality [5]. Although there exist mature process discovery algorithms, which are able to deal with noise [6] and incompleteness [7], in the context of legacy software applications their application is still limited, as process-related event logs are usually not readily available in these systems [8].

In previous work [9, 10], we have presented a possible solution to this challenge. Basically, our approach observes the interactions of process participants with the legacy software system and records these interactions in a fine-grained way, which shall also allow obtaining event logs that are suitable for process discovery algorithms. This paper presents an approach to abstract the fine-grained information recorded in corresponding event logs to enable the discovery of process models suitable of different granularity levels.

The remainder of this paper is structured as follows: Section 2 describes the proposed solution and shows how it allows creating event logs constituting the basis for discovering process models of different levels of granularity. Section 3 describes the discovery algorithm applied to these logs as well as the proposed hierarchy levels for resulting process models. In Section 4, we evaluate the approach using a real-world event log. Section 5 discusses related work. Finally, Section 6 provides a summary and outlook.

2. Solution Approach

In the context of legacy software systems, we define a process as a sequence of related interactions the users have with the system [10]. Consequently, an event log derived from a legacy software system is described by entries of which each corresponds to an individual user interaction. In general, user interactions are driven by the structure and layout of the user forms (e.g., to check an invoice or to process an order) of the legacy software system. In an Oracle legacy system, for example, a graphical user interface (GUI) is located on a canvas (see L3 in Fig. 1). The latter, in turn, may comprise several screens or canvases (see L2 in Fig. 1) which again contain form fields and buttons (see L1 in Fig. 1). Consequently, the events recorded in an event log contain information like the respective form field filled or button clicked by a user as well as the form or screen in which the user interaction took place. An event log obtained from an Oracle legacy software system, therefore, comprises event data, which are characterized by a user story (i.e., a case identifier), timestamps (e.g., entering or leaving a field), additional information based on user specification, and the actual user interaction (e.g., button 'X' clicked or field 'Y' filled). A user interaction is structured in the way *ScreenName.EventName*, and examples include ORDERS.DATE_CONTROL_BLOCK.MONTH_PLUS1.WHEN-BUTTON-PRESSED (i.e., in the order screen a button to increase a date by one month was pressed) and ORDERS.MAIN_CANVAS.BUTTON_SAVE.WHEN_BUTTON_PRESSED (i.e., in the main canvas of the order screen the save button was pressed). In other words, the event log documents user interactions at the finest granularity level. Discovering process models based on such fine-grained event logs, however, might lead to complex process models that are hard to read and understand [11]. To tackle this challenge, we preprocess the event log stored form legacy software systems, while considering hierarchies and the event log structure. Note that this shall allow for the discovery of multiple process models with different granularity based on the same initial event log.

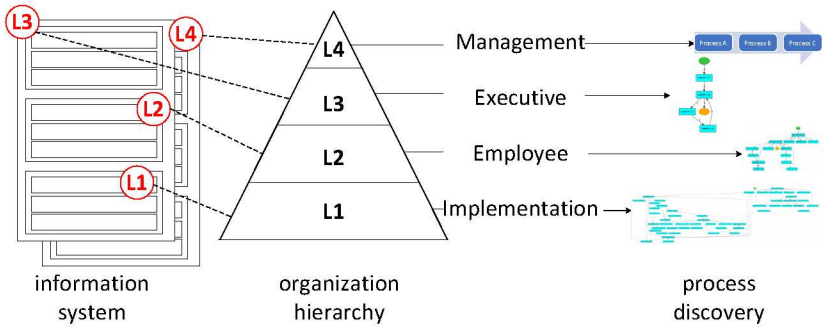

Figure 1: Proposed approach for event log preprocessing with hierarchies

2.1. Event Log Hierarchies

The different stakeholders intended in process models require different levels of abstraction. By preprocessing event logs accordingly, we are also able to discover process models of different granularity levels. For this purpose, we extract corresponding hierarchical information from the individual user interactions taking the structure of the form elements the respective user has interacted with into account. The resulting event log and event log entries respectively allow differentiating between four different granularity levels:

The **Management Level** shall refer to multiple processes and show how these processes are connected and coordinated (see L4 in Fig. 1).

The **Executive Level** deals with a high-level view of the processes – it solely considers the events related to the completion of forms during process execution (see L3 in Fig. 1). Example: ORDERS

The **Employee Level** considers more detailed information on the form screens required to complete a process (see L2 in Fig. 1). Example: ORDERS.MAIN_CANVAS

The **Implementation Level** covers the most fine-grained event log representation. It also includes specific form fields and buttons (see L1 in Fig. 1). Example: ORDERS.MAIN_CANVAS.BUTTON_SAVE.WHEN_BUTTON_PRESSED

When abstracting the event log to a higher granularity level (e.g., from the employee level to the executive level), we remove consecutive identical event log entries (e.g., multiple interactions in the same form are recorded). This reduces the complexity of process models of higher hierarchical levels.

Finally, the fine-grained event logs enable a mapping between the event label (e.g., BUTTON_SAVE.WHEN_BUTTON_PRESSED) and a less technical representation of the event label (e.g., Button Save Pressed). A less technical language representation of labels fosters the comprehension of the fine-grained process model, and, therefore, increases the comprehension of discovered process models.

Table 1
Domain Expert Recognition (N=13) [10].

	Inductive (Tree)	Inductive (BPMN)	DFG	Heuristic (thold=0.75)	Heuristic (thold=0.9)	Heuristic (thold=0.95)
Mean (SD)	3.08 (1.07)	3.08 (0.73)	2.31 (1.2)	4.15 (0.77)	4.46 (0.63)	3.38 (1.27)

3. Process Discovery

3.1. Algorithm Selection

Prior to applying process discovery algorithms to the preprocessed event logs, we conducted a Delphi study with domain experts in order to identify which process discovery algorithm yields the most appropriate results [10]. Study participants (N=13) were asked to evaluate to which degree they could recognize the legacy software system based on the resulting process models on a scale from 1 (not at all) to 5 (completely). Table 1 presents the results.

The process models generated by the Heuristic Miner yielded the best results regarding recognition. Therefore, we apply the Heuristic Miner for the discovery of process models for individual hierarchy levels [12].

3.2. Granularity Levels

The **implementation level (L1)** reflects to the most fine-grained representation of the recorded user interactions. It represents the individual interactions the user(s) had with the legacy software system. Corresponding process models foster the migration of a legacy software systems to modern technologies as the event log documents all user interactions that occurred at runtime, including exceptions that occurred in the legacy software system.

The **employee level (L2)** deals with the screens and canvases of a form filled by users when interacting with the legacy software system. Usually, corresponding screens and canvases contain form fields dealing with the same topic, e.g., contact data of a supplier or the items of an order. Consequently, the discovery of processes on the employee-level provides employees with an overview of process-relevant blocks that need to be completed in order to execute a process. This, in turn, enables a better understanding as well as acceptance of executed processes for employees.

While the employee level is concerned with the screens and canvases of a form, the **executive level (L3)** further considers the order in which users usually interact with the forms. Process models of the executive level document in which way the users navigate through the variety of forms (i.e., in which order different forms are filled in). This allows for an abstracted view on the processes implemented in the legacy system, while at the same time enabling a high-level view on the forms to be completed during process execution.

The **management level (L4)**, does not relate different forms as it relates multiple processes. A model discovered on this level shows how different processes are executed in order to achieve a goal. In this setting, different processes implemented by the legacy software system are related to each of them and, consequently, discovered.

Table 2
Comparison of the Resulting Process Models and their Granularity Levels

	Implementation Level	Employee Level	Executive Level
Number of Activities	52	15	4
Reduction (%)	0.00%	-71.15%	-92.31%

4. Evaluation

We applied the presented approach for preprocessing event logs to an event log we generated through the dedicated recording of the sessions a particular user had with an Oracle legacy software system in an industrial setting [9, 10]. The user could provide supplementary information for the recorded business process (e.g., description and context of the process). In total, the considered legacy software system comprises 589 database tables with 9977 columns. More than 8000 different database statements (60712 statements in total) were implemented in more than 5 million lines of code. Finally, the legacy software system comprises 1285 forms and 6243 different screens.

We preprocessed the event log to provide the information specific to the hierarchical levels introduced in Section. 2. Subsequently, we applied the heuristic miner to discover the process models for each hierarchy level. Note that we used the heuristic miner with default configuration (i.e., dependency_threshold = 0.5, and_threshold = 0.65, loop_two_threshold = 0.5) to discover the process models. Fine-tuning of these parameters might further improve the quality of generated process models. Figures 2 - 4 depict the resulting process models for hierarchy levels L1 to L3. As the collected event log solely contains the interactions of a single process (i.e., a single user story), we were unable to discover a process model for the management level. The event log as well as the process models are provided in an anonymous version[1]. When juxtaposing the process models discovered by the heuristic miner, the different levels of granularity for each organizational level become evident. In case of the three process models discovered from the legacy system event log (see Figs. 2 - 4), the number of activities are described in Table 2. In the given event log, the overall number of activities was reduced by 71.15% (implementation vs. employee level), and by 92.31% (implementation vs. executive level). A reduction of 73.33% in terms of the number of activities can be observed when comparing employee and executive level.

The size (i.e., the number of elements) of a process model does effect both its understandability [13] as well as the likelihood of errors [14, 15]. According to the 7 Process Modelling Guidelines [16], larger process models are more difficult to understand, and have a higher error probability. In our approach, the number of discovered activities decreases for higher hierarchy levels. Consequently, our approach for preprocessing and discovering process models from legacy software systems is able to provide suitable process models for various stakeholders as the granularity decreases for higher hierarchy levels. To be more precise, process models suitable for of higher level hierarchies (i.e., L2-L4) allow for the abstraction of fine-grained event data.

[1] https://cloudstore.uni-ulm.de/s/iHyJtA9riioyJEK

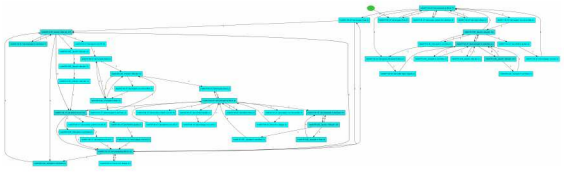

Figure 2: Implementation level

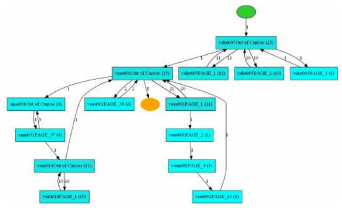

Figure 3: Employee level

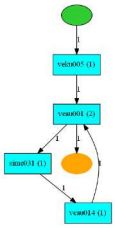

Figure 4: Executive level

5. Related Work

This work is related to the research areas of event log preprocessing [17] and event log abstraction [11] in process mining [3].

Event log preprocessing is concerned with tasks that substantially improve the performance of process mining algorithms by detecting and removing noise as well as traces and activities that contain undesired behavior [17]. Preprocessing is either concerned with transformation or detection techniques. Transformation techniques filter the event log based on a likelihood and, consequently, remove unlikely event log entries [18], or they delete erroneous event log entries [19]. Other approaches consider temporal aspects during event log preprocessing [20]. Detection techniques try to identify those events that are problematic for the quality of an event log based on patterns and clustering techniques [17]. Approaches for event clustering usually use some sort of internal representation for event logs, derive corresponding clusters, and use them to aggregate the event log [21, 22]. As opposed to existing approaches for event log transformation and event log detection, our approach leverages information from the structure of an information system (i.e., how users interact with the system and how corresponding user forms are organized), to provide process models suitable for readers of different granularities.

Regarding event log abstraction, supervised techniques use time intervals [23], manual mappings [24], views on the process model [25, 26] or reference models [27] to abstract behavior in the event log, whereas unsupervised techniques aim to identify re-occurring patterns [28, 29]. In contrast, our approach leverages domain knowledge about the event log automatically derived from the structure of the legacy software system to aggregate event log entries accordingly.

6. Conclusion and Outlook

This paper presents an approach to leverage information from (legacy) software systems to preprocess event data, while considering various hierarchies to provide suitable process models for different stakeholders. This not only enables non-domain experts to better understand a legacy software system, but the discovered process models also serve as process documentation for the legacy systems, facilitating software migration projects. Additionally, process models for higher organizational levels (e.g., employee, executive or management) are less complex and, therefore, more suitable for humans to understand especially. This is helpful for stakeholders to understand the process models, and to identify improvement potential. In future work, we apply the presented approach to additional event logs obtained from legacy software systems, as well as event logs that are not collected from a dedicated recording.

References

[1] H. Mintzberg, The structuring of organizations: a synthesis of the research, Prentice-Hall Englewood Cliffs, N.J, 1979.

[2] Q. Tran, Y. Tian, Organizational structure: Influencing factors and impact on a firm, American Journal of Industrial and Business Management 03No.02 (2013) 8.

[3] W. M. P. van der Aalst, Process Mining: Data Science in Action, Springer, 2016.

[4] W. M. P. van der Aalst, Process discovery: Capturing the invisible, IEEE Computational Intelligence Magazine 5 (2010) 28–41.

[5] W. M. P. van der Aalst, et al., Process mining manifesto, in: Int'l Conf on BPM'11, 2011, pp. 169–194.

[6] A. Weijters, J. Ribeiro, Flexible heuristics miner (fhm), in: Symp on CIDM'11, 2011, pp. 310–317.

[7] A. K. A. de Medeiros, A. J. Weijters, W. M. P. van der Aalst, Genetic process mining: an experimental evaluation, Data Mining and Knowledge Discovery 14 (2007) 245–304.

[8] W. M. P. van der Aalst, Object-centric process mining: Dealing with divergence and convergence in event data, in: Software Engineering and Formal Methods, Springer International Publishing, Cham, 2019, pp. 3–25.

[9] M. Breitmayer, L. Arnold, M. Reichert, Towards retrograde process analysis in running legacy applications, in: 14th Central European Workshop on Services and their Composition (ZEUS 2022), number 3113 in CEUR Workshop Proceedings, 2022.

[10] M. Breitmayer, L. Arnold, S. L. Rocca, M. Reichert, Deriving event logs from legacy software systems, in: 4th International Conference on Process Mining (ICPM 2022), ICPM 2022 Workshops, Springer Nature Switzerland AG, 2022.

[11] S. J. van Zelst, F. Mannhardt, M. de Leoni, A. Koschmider, Event abstraction in process mining: literature review and taxonomy, Granular Computing 6 (2021) 719–736.

[12] A. J. M. M. Weijters, W. M. P. van der Aalst, A. K. A. de Medeiros, Process mining with the heuristicsminer algorithm, 2006.

[13] J. Mendling, H. A. Reijers, J. Cardoso, What makes process models understandable?, in: Business Process Management, Springer Berlin Heidelberg, Berlin, Heidelberg, 2007, pp. 48–63.

[14] J. Mendling, G. Neumann, W. M. P. van der Aalst, Understanding the occurrence of errors in process models based on metrics, in: On the Move to Meaningful Internet Systems 2007: CoopIS, DOA, ODBASE, GADA, and IS, Springer Berlin Heidelberg, Berlin, Heidelberg, 2007, pp. 113–130.

[15] J. Mendling, H. Verbeek, B. van Dongen, W. van der Aalst, G. Neumann, Detection and prediction of errors in epcs of the sap reference model, Data Knowledge Engineering 64 (2008) 312–329. Fourth International Conference on Business Process Management (BPM 2006) 8th International Conference on Enterprise Information Systems (ICEIS' 2006).

[16] J. Mendling, H. A. Reijers, W. M. P. van der Aalst, Seven process modeling guidelines (7pmg), Inf. Softw. Technol. 52 (2010) 127–136.

[17] H. M. Marin-Castro, E. Tello-Leal, Event log preprocessing for process mining: A review, Applied Sciences 11 (2021).

[18] R. Conforti, M. L. Rosa, A. H. t. Hofstede, Filtering out infrequent behavior from business

process event logs, IEEE Transactions on Knowledge and Data Engineering 29 (2017) 300–314.

[19] N. Tax, N. Sidorova, W. M. P. van der Aalst, Discovering more precise process models from event logs by filtering out chaotic activities, Journal of Intelligent Information Systems 52 (2019) 107–139.

[20] S. Song, Y. Cao, J. Wang, Cleaning timestamps with temporal constraints 9 (2016) 708–719.

[21] R. Jagadeesh Chandra Bose, W. Aalst, van der, Context aware trace clustering : towards improving process mining results, in: Proceedings of the Ninth SIAM International Conference on Data Mining (SDM 2009, Sparks NV, USA, April 30-May 2, 2009), Society for Industrial and Applied Mathematics (SIAM), 2009, pp. 401–412.

[22] M. Boltenhagen, T. Chatain, J. Carmona, Generalized alignment-based trace clustering of process behavior, in: Application and Theory of Petri Nets and Concurrency, Springer International Publishing, Cham, 2019, pp. 237–257.

[23] M. L. van Eck, N. Sidorova, W. M. P. van der Aalst, Enabling process mining on sensor data from smart products, in: 2016 IEEE Tenth International Conference on Research Challenges in Information Science (RCIS), 2016, pp. 1–12.

[24] A. Begicheva, I. Lomazova, Discovering high-level process models from event logs, Modeling and Analysis of Information Systems 24 (2017) 125–140.

[25] J. Kolb, M. Reichert, A flexible approach for abstracting and personalizing large business process models, Applied Computing Review 13 (2013) 6–17.

[26] R. Bobrik, M. Reichert, T. Bauer, View-based process visualization, in: 5th Int'l Conf. on Business Process Management (BPM'07), number 4714 in LNCS, Springer, 2007, pp. 88–95.

[27] M. de Leoni, S. Dündar, Event-log abstraction using batch session identification and clustering, in: Proceedings of the 35th Annual ACM Symposium on Applied Computing, SAC 2020, 2020, pp. 36–44.

[28] R. P. Jagadeesh Chandra Bose, W. M. P. van der Aalst, Abstractions in process mining: A taxonomy of patterns, in: Business Process Management, 2009, pp. 159–175.

[29] C. W. Günther, A. Rozinat, W. M. P. van der Aalst, Activity mining by global trace segmentation, in: Business Process Management Workshops, 2010, pp. 128–139.