# Simulation-enhanced Action-oriented Process Mining in Production and Logistics

Felix Özkul[1*], Robin Sutherland[1], Sigrid Wenzel[1]

[1]Department Organization of Production and Factory Planning (pfp), University of Kassel, Kurt-Wolters-Str. 3, 34125 Kassel, Germany; **felix.oezkul@uni-kassel.de*

**Abstract.** Process mining is increasingly being used to gain insights into processes based on operational data. Recently, approaches have been researched as to how these findings can be automatically transferred into process-regulating actions during system operation to correct deviations between the actual and target process in real time. However, the implementation of such action-oriented process mining mechanisms requires sufficient testing of the implemented actions in the application to prevent undesirable side effects in the real system. This article explains how discrete-event simulation in production and logistics can be used to mitigate risks in the context of implementing action-oriented process mining through the use of an emulation model. For this purpose, we present simulation-enhanced action-oriented process mining as well as a proof-of-concept implementation based on a use case.

## Introduction

Discrete-event simulation (DES) has proven itself across many industries as a planning tool for production and logistics systems (e.g., see [1][2]). Its application enables users to analyze "what-if" or "how-to-achieve" questions by executing simulation experiments during the planning phase of a production or logistics system. The utilization of simulation during commissioning or system operation is less mature, but the potential benefits of using simulation during these system life cycle phases are also apparent (see [3]). In the following, a distinction is made between the levels of the technical system and the control system with respect to systems in production and logistics. This distinction can also be made with appropriate modeling in the simulation application, so that there is a real technical system, a real control system, a simulated technical system, and a simulated control system. In the context of this paper, *emulation* describes the use of a real control system being used in a simulated technical system (other types of coupling are described in [4]). The simulated technical system thus receives the same inputs that the emulated real technical system would receive, and the real control system can thus be tested simulatively.

DES can be used in combination with other digital methods such as process mining, for example, to reduce the effort involved in creating simulation models, or to simulatively generate synthetic input data for the application of process mining techniques [5]. The added value of the combined use of both methods has been presented in the pertinent scientific literature – also for the area of production and logistics [6].

Process mining is a research area at the intersection of data science and process science [7]. Process mining is mainly used to analyze processes based on past operational data (see [8]) and its use is favored by the increasing prevalence of information technology systems in production and logistics [6]. The classic types of process mining are process discovery, conformance checking (see Section 1), process enhancement, and more recently, performance analysis, comparative process mining, predictive process mining (machine learning-supported process mining), and action-oriented process mining (see Section 1; [7]). These types of process mining have some overlaps in terms of both methodology and application.

In combination with DES, the combined use of both methods can be used to gain insights into both the past and the future. Not part of this given time continuum is the present and the associated consideration of the combined use of methods during ongoing system operation. Input data for process mining techniques is conventionally stored in so-called event logs (see Section 1 for more details). The process information stored in these event logs can be used, for example, to extract models describing process behavior (process discovery), and to retrospectively identify deviations between the target and actual process (conformance checking). However, for process mining to be used in an action-oriented manner in

the current system operation, the process-related insight must be gained in *real time*, i.e., possibly before a process instance is completed and thus before the complete event log is generated. Streaming process mining methods (alternatively online process mining) are suitable for this purpose (see [9]). Streaming process mining analyzes so-called event streams (see Section 1) instead of conventional event logs [9], thereby gaining knowledge about process instances even before they are completed and, if necessary, enabling actions that regulate process instances.

Action-oriented process mining aims to generate these diagnostic-based actions (see [7]), thereby bridging the gap between insight and action that conventional process mining cannot. However, the implementation of such action-oriented process mining-based process control mechanisms has not yet been researched in detail in the area of production and logistics, and the risks associated with the implementation of action-oriented process mining are obvious. For example, in case of failure, testing on the real system may affect its operational performance (e.g., due to unforeseen system failures caused by actions of the action-oriented process mining mechanism). DES could, on the one hand, help mitigate these risks by generating event streams for different process scenarios, which are used as input resources for action-oriented process mining to analyze the different response actions (e.g., an online adjustment of machine parameters or job scheduling) as well as their effects. On the other hand, as an element of decision support, DES can help to determine the impact of non-conformity with regard to process instances to gain predictive insights into the necessary process control actions. This enables users to mitigate the risks associated with the implementation of action-oriented process mining and helps increasing its effectiveness in practice. Our approach uses simulation as an emulation model of an underlying technical system to test an action-oriented process mining mechanism.

The paper is structured as follows: Section 1 explains the key process mining terms. Section 2 specifies theoretical scenarios for *simulation-enhanced action-oriented process mining*. Section 3 presents a use case based on a proof-of-concept implementation. The paper concludes with a summary and a research outlook.

# 1 Process Mining Terms and Context

A *process* in the context of process mining refers to "a coherent series of changes that unfold over time and occur at multiple levels" ([10], p. 3). These changes are triggered by *events*. Event logs record the execution of processes based on events that start or end *activities*. Process and activity executions represent so-called instances. A process instance is also referred to as a *case*. Each case can be described by a sequence of events or activities, which is logged as a trace in the event log. In other words, event logs are a multiset of traces, with each trace being a sequence of events ([7], def. 3).

Events are assigned to their respective cases using a corresponding case identifier. Events and traces can be described in more detail using additional information (such as resources performing activities). In practical use, event logs log a finite number of events. Different types of process mining can then be applied on the basis of these logged events. Processes (as defined above) can be mapped and analyzed using process models (e.g., models in Petri net notation [11]).

Conformance checking can be used to check how well a process model is able to replay the process behavior observed in an event log. The scientific literature mentions various conformance checking methods, such as footprints, token-based replay (for Petri nets), and alignments [12]. The ability of a process model to reproduce recorded process behavior is referred to as its *fitness* [8][12] and is the most important indicator for describing process model quality (the other process model quality indicators are *precision*, *generalization*, and *simplicity*, see [12]). On the one hand, conformance checking can be used to quantify the quality of a modelled or extracted process model in relation to an event log (i.e., recorded process behavior). On the other hand, it is also possible to analyze the conformity of the logged cases in relation to a normative process model (i.e., a binding model specifying the target process). This approach of conformance checking is central to the implementation of simulation-enhanced action-oriented process mining.

## 1.1 Streaming Process Mining

The examination of cases using event logs enables an *a posteriori* determination of conformity in relation to a normative process model. However, for a process instance to be corrected during its execution in a production

and logistics system the conformity assessment must be carried out *a priori* with regard to the completion of the process instance. This requirement can be met by analyzing event streams.

Van Zelst et al. ([13], p. 7) define an *event stream* as "a continuous stream of events executed in context of an underlying business process". An event stream comprises a – potentially infinite – number of events (as defined above). The investigation of event streams is the subject of streaming process mining and includes streaming process discovery and online conformance checking [9]. Event stream analysis allows for the processing of very large event logs (like event logs too large to be stored in memory) or for continuous monitoring of processes. The latter is important for our contribution in the context of production and logistics. In principle, event streams can be analyzed using conventional process mining algorithms by combining events into batches and then processing them like event logs. However, there are also dedicated algorithms that enable online analyses (e.g., prefix alignments [14] or the analysis of behavioral patterns ([15], see Section 2) for conformance checking). Van Zelst et al. [13] present an architecture (S-BAR) for the application of common process discovery algorithms in online settings using abstract representations. These techniques focus on the investigation of activity relationships and control flow. In addition, Stertz et al. [16] present an approach that analyzes the temporal relationship between activities using a temporal profile. A temporal profile contains information about the (stochastically) expected durations of activities as well as their temporal distance and a normative process model can be infused with it to allow the application of temporal conformance checking (see Figure 1 "temporal items"). A temporal profile can, for example, be extracted from a normative process execution log (i.e., a log containing valid traces of a target process) which contains information about the start and end times of activities (i.e., start and end events) or on the basis of expert domain knowledge. New events are compared with a temporal profile and time-related outliers are detected by calculating their Z-score [17]. Temporal conformance can also be checked in the event stream in addition to the control flow view. The approach in this paper checks event streams based on behavioral conformance (see [15]) and temporal conformance [16] to monitor process execution online.

### 1.2 Action-oriented Process Mining

Streaming conformance checking based on event streams enables online monitoring of processes. Action-oriented process mining builds on this and uses streaming process mining to automatically generate actions based on detected deviations from target processes (i.e., non-conformities) that imply risks associated with the violation of process *constraints*. In the context of production and logistics, constraints can be, for example, case-specific production deadlines that are about to be missed (violation of temporal conformance) or the prescribed production sequences that are not being followed in the ongoing process (violation of behavioral conformance).

Park and Van der Aalst [18] provide a comprehensive framework for implementing action-oriented process mining for business processes with a focus on operational support (more in [8]). During the execution of business processes events are logged in a real information system and an event stream is generated which is then analyzed by the *constraint monitor* [18]. The constraint monitor analyzes the event stream given the abovementioned constraints (which, in our approach, are temporal and behavioral constraints), that are formalized in a constraint formula (e.g., in a production setting, one such formula could evaluate if a product is manufactured given a prescribed production sequence). The analysis of events given the constraint formulae yields constraint instances which are then sent as a constraint instance stream to the *action controller* [18]. The action controller evaluates the incoming constraint instances and generates an *action* based on the evaluation result and given an action formula [18]. The action formula takes in the constraint instance stream and a time window to produce a set of *transactions* (i.e., operations which the underlying information system can perform). An example of such an action in a production setting would be to increase an order's priority if production delays (i.e., temporal nonconformance) are diagnosed.

This paper builds on the framework of Park and Van der Aalst [18] and extends it with simulation-related components focusing on the implementation of action-oriented process mining in production and logistics.

Drieschner et al. [19] present an approach that focusses on simulation as a learning tool for action-oriented process mining. The similarity between this approach and our contribution is that both approaches use simulation as a data generator for process mining (see Section 2). However, the focus of Drieschner et al. [19] is pedagogical

and focuses on user interaction. Our approach focuses on the automated testing of an implemented action-oriented process mining mechanism. For this purpose, the simulation model emulates the real system and simulation runs are conducted.

## 2 Simulation-enhanced Action-oriented Process Mining

Based on the framework of Park and Van der Aalst [18], the following Figure 1 shows a possible architecture for simulation-enhanced action-oriented process mining. During the execution of a simulation model, a simulation-generated event stream is created whose events are recorded by an event monitor and evaluated with regard to their temporal and behavioral conformance. For this purpose, the approach of Stertz et al. [16] is applied on the basis of a normative temporal profile and the formulation of temporal constraints to identify temporal outliers within the event stream. Temporal constraints can be formulated, for example, by target dates that are set for the completion of an order. If, for example, a processing activity occurs too late or takes too long, this deviation is recognized via temporal conformance checking. In addition, the event monitor monitors the behavioral conformance of incoming events. The behavioral conformance is assessed by analyzing *behavioral patterns* (i.e., a set of activities and possible control flow relations [15]) and a reference behavioral model (expressed as the set of *prescribed* behavioral patterns which are expected for the underlying process (see [15]). This is shown as the reference behavioral model in Figure 1. This granular perspective on the control-flow allows an in-vivo analysis of singular observable behavioral patterns (which can be inferred from events but at a higher level of abstraction) during process execution and allows for the calculation of three distinct key indicators (see [15]): C*onformance* (indicating the correctness of the observed behavior), *completeness* (indicating case progression), and *confidence* (indicating the expected stability of the conformance).

Checked events are appended to the checked event stream and relayed to the simulation action controller. Depending on the event status (i.e., whether an event conforms or not), a suitable event routine is selected from the event routine space and communicated to the simulation model. Since actions in the (discrete-event) simulation model are generated by executing event routines, we use the term *event routine space* instead of the action space (cf. [18]). We refer to the *simulation action controller* as the component that fulfils the tasks of the action controller (see [18]). If specific behavioral or temporal non-conformity is detected for an event, a routine for handling these deviating events is automatically selected (if an event conforms, the default event routine is executed). The described workflow is shown in Figure 2.

The effects of the decisions by the simulation action controller can then be observed during the remainder of the simulation runs. Based on the simulation model configuration, event streams containing non-conforming events, i.e., *temporal or behavioral imperfections*, can be

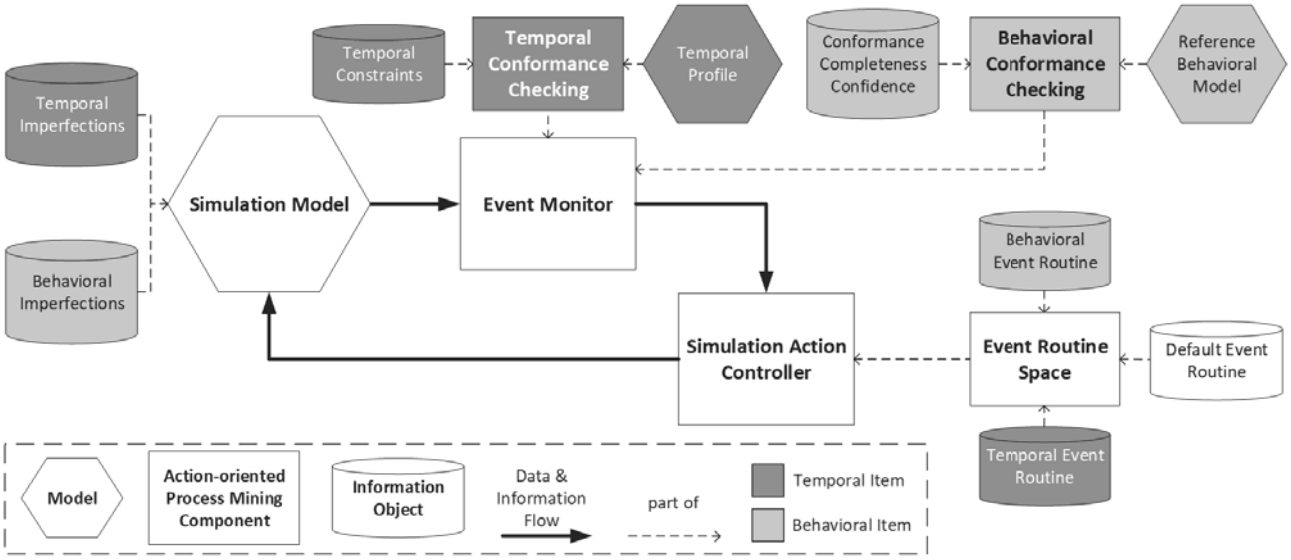


**Figure 1**: Architecture of the presented approach.

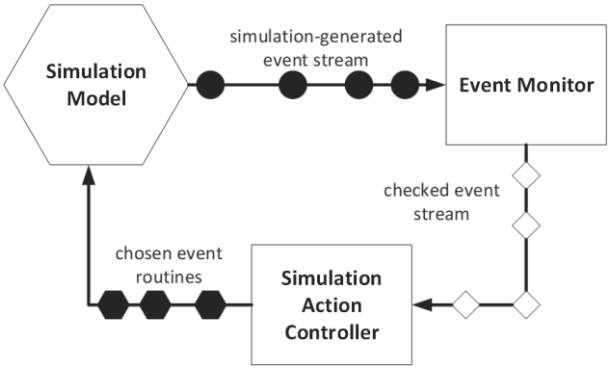


**Figure 2**: Information flow of the outlined approach.

generated during a simulation run to invoke automatic actions of the simulation action controller, the effects of which are assessed in the simulation.

This allows the action controller to be thoroughly tested simulatively before it is implemented in the real

system. Risks relating to the impairment of process sequences in the real system can be mitigated in this way. The results of the simulation application can be statistically validated through systematic experimental design and, in the context of the implementation of action-oriented process mining, should also capture rare events and faults that are difficult to observe in real system operation. The specific number of simulation runs depends on the specific use case and the number of different scenarios which have to be addressed by the simulation action controller. This ensures that the system behavior for these edge cases is also considered in the action controller of the real system. This is the qualitative added value of the proposed approach: Based on the execution of simulation experiments that cover the range of expected cases of process deviations in the real system, action-oriented process mining can be implemented without having to make any changes to the real system. Instead, the simulation is used as an emulation model for the real control system and risks associated with the action-oriented process mining implementation are mitigated.

# 3 Use Case

The following use case demonstrates the idea behind simulation-enhanced action-oriented process mining (see Section 1) using a practical case study and implements the architecture in Figure 1.

## 3.1 Application System

The application system is a conveyor system on a university laboratory scale, Figure 3 shows the corresponding simulation model.

Starting from the source conveyor, load units are fed onto the main conveyor. Load units carry objects that have an object type that determines the target production sequence on the machines M1-M6 and one load unit corresponds to one case. In addition, there are various stopping points (H1-H7) on the main conveyor, which can read and write to RFID tags attached to the load units. In addition to the conveyed object type, other load unit-related information (attributes) is also stored on the RFID tags, such as the priority of the order (high/low), the target time for completion of the order and the next machine in the object type-specific processing sequence. The ma-

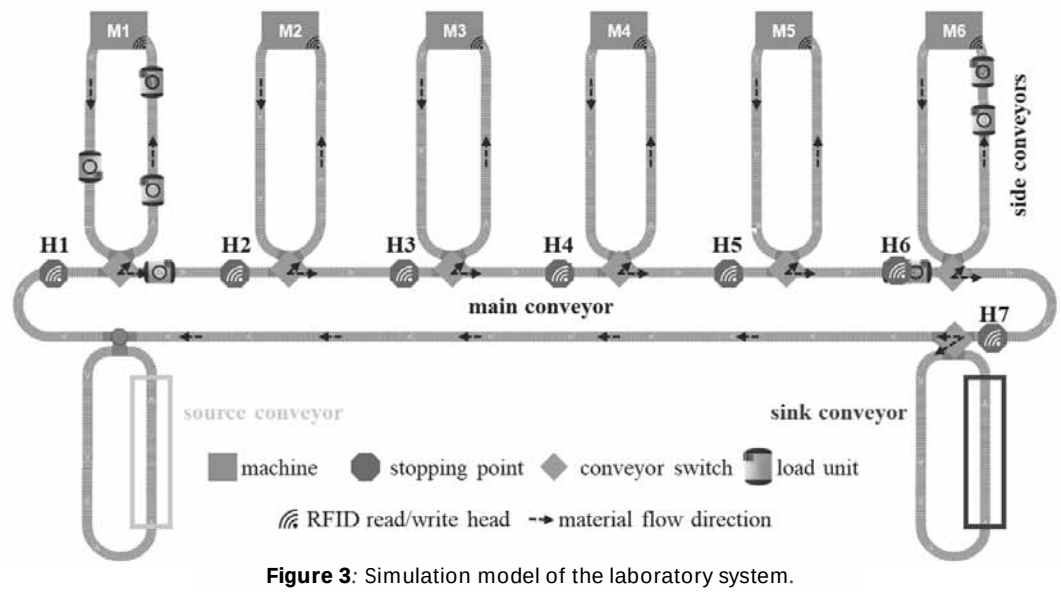


**Figure 3**: Simulation model of the laboratory system.

chines are located on side conveyors, which are connected to the main conveyor via conveyor switches. Side conveyors have a certain load unit capacity and load units can only be discharged onto a side conveyor if it can accommodate further load units. At the conveyor switches, load units with high priority have right of way; for load units with the same priority, first in first out (FIFO) applies.

We formalize a workflow net (a Petri net with certain properties, see [8]), which is a simplified process model for the behavior of the load units (cases) in the application system (Figure 4). The normative workflow net exhibits maximum fitness in relation to the underlying conveyor system. However, it enables additional behavior that is not possible in the real system due to the object type-specific processing sequences (i.e., its precision is low). To increase the precision of the model, the formalization of color sets and the introduction of transition guards would be appropriate. For simplicity, however, we decided not to include a colored Petri net of the load unit process. Starting from the source ('exit_source'), the various stopping points on the main conveyor are controlled and logging events are fired ('log_at_H*').

Depending on the conveyed object type on the load unit, its processing progress and the availability of side conveyor capacity, the load units are processed at the machines ('processing_on_M*') or conveyed in a waiting loop. Skips are modelled as silent transitions; load units that cannot be processed at their destination machine will still pass the subsequent stopping points on the main conveyor (i.e., it is not possible to skip logging activities, which is coherent with respect to the layout in Figure 3). Furthermore, the process model is infused with a temporal profile, which is based upon domain knowledge about the controls of the reference system. For clarity, we graphically omit the temporal distances between all activities and activity (transition) durations (examples for both are highlighted in blue in Figure 4).

### 3.2 Use Case

The goal of the use case is to illustrate how the approach can handle incorrect processing sequences (behavioral non-conformance) and processing delays (temporal non-conformance) and how automated actions can address them.

Due to the stochastic order loading of different object types, congestion of load units can occur in the system, which delays the processing of orders. Furthermore, it is assumed that the reading of the RFID tags – which are placed on the load units – is not always error-free, and that read and write errors can occur. A case is considered compliant if its processing sequence corresponds to the target processing sequence of its object type and the order is completed in time. If the target processing sequence is violated, the event monitor detects this deviation as behavioral non-conformance. Delays related to the processing of load units may lead to the detection of temporal non-conformance (see Section 2). In case of behavioral non-conformance, rerouting to the original target machine is defined as an automated action in the action space of the action controller.

In case of temporal non-conformance, the priority of

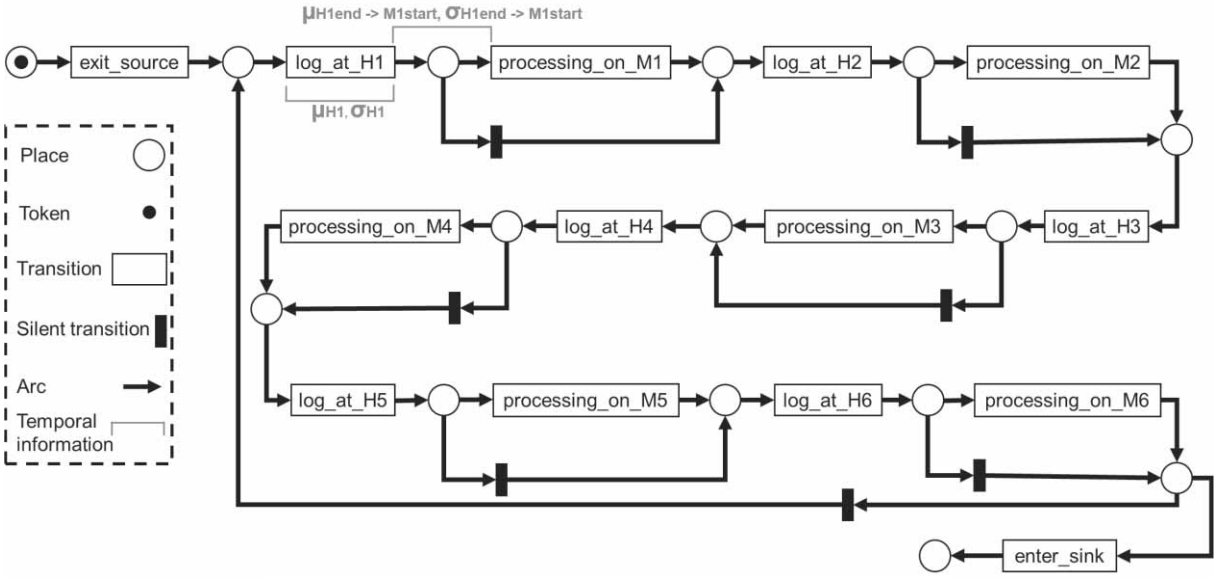


**Figure 4:** Workflow-net of the load unit process.

a load unit must be increased if there is a delay so that it can be routed to the conveyor switches more quickly if necessary. In the event of temporal non-conformance of a processing activity, it must also be assumed that a machine requires maintenance. A maintenance worker should therefore be sent out in this case to carry out the maintenance as quickly as possible.

### 3.3 Technical Implementation

We implement the architecture from Figure 1 using AnyLogic 8 and open-source process mining software systems. Figure 5 shows the proof-of-concept implementation.

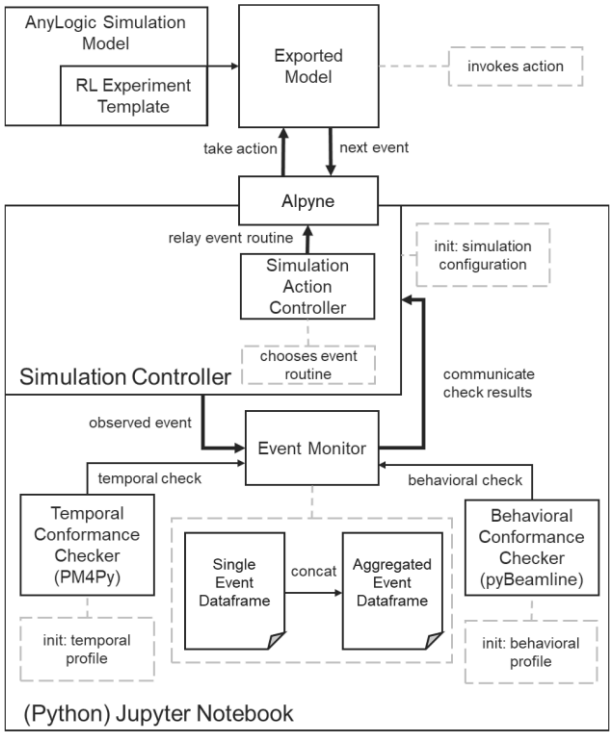


**Figure 5:** Structure of the proof-of-concept implementation.

First, a simulation model is built and an experiment template is implemented. The current AnyLogic software ecosystem provides a template for Reinforcement Learning (RL) (i.e., the possibility to define observations, actions and configurations) which can be adapted because – similarly to RL – our approach utilizes the observation and action capabilities in the simulation context.

Therefore, we repurpose the RL pipeline to implement simulation-enhanced action-oriented process mining. The simulation model containing the experiment template is exported to a standalone (Java) model, which is bi-directionally linked to a Python Jupyter Notebook using the Alpyne software library [20]. The exported model is configured to invoke actions based on certain key events, those being the logging events shown in Figure 4 at the stopping points (H1-H6). Whenever a logging event occurs, the simulation engine pauses the simulation run and passes the events to the Python simulation controller, which implements Alpyne as its simulation interface. The observed event is then relayed to the event monitor which transforms and logs the observed event in a Pandas (https://pandas.pydata.org/docs/) Dataframe object (‘Single Event Dataframe’) to perform temporal and behavioral conformance checking. Software systems for the implementation of streaming conformance checking (the central component of the event monitor) are currently only available to a limited extent (see [21]). The comprehensive Python-based open-source framework PM4Py [22] implements the temporal conformance checking approach of Stertz et al. [16]. Burattin [21] provides an open-source streaming process mining software named Beamline, which is available for Java and Python (pyBeamline). Our work implements pyBeamline’s behavioral conformance checking [15][21] and provides a compact implementation. Based on a reference event log (which can be generated, for example, by simulating the workflow net shown in Figure 4 or by exporting traces from a valid simulation model after the warm-up phase) a normative behavioral model is initially mined with pyBeamline and new event instances are checked against it. In order to assess the behavioral conformance of a running case, we implement a second Dataframe object to which each event is appended (technically, at each iteration the ‘Single Event Dataframe’ and the ‘Aggregated Event Dataframe’ are concatenated). This step is computationally costly but necessary to contextualize events with regard to the previous events of their running case. Furthermore, each event is checked against the temporal profile of the normative process which is created using PM4Py (the computation of a temporal profile requires start and end timestamps for activities). The PM4Py implementation additionally requires floating point numbers as timestamps whereas other algorithms often work based on datetime formatted timestamps. After conformance checking, the results of the check are relayed to the simulation action controller which then chooses an appropriate event routine (action). If the event monitor does not register non-conformance, the default routing logic is applied to the load unit (see Section 3.1). If, for example, the event monitor detects that a case is being processed late (temporal non-conformance), the simulation action controller increases its priority to enable faster transport

to the processing machines. After selecting a suitable action, it is communicated to the simulation model via Alpyne and the simulation engine continues the simulation run. The control logic of the simulation can be modified by the simulation controller (with appropriate implementation in the simulation model) to specifically generate non-conforming events, for example by generating misreads at the stopping points with a predetermined probability, which ensures that load units are incorrectly ejected or not ejected. In this way, the components of action-oriented process mining can be specifically checked within the framework of simulation experiments with regard to their ability to detect and recommend and thus their suitability as operational support before they are implemented in the real system (see Section 2). This avoids risks associated with undesirable side effects in reality.

# 4 Summary and Outlook

This article presents a combination approach of discrete-event simulation and process mining, in which a simulation model is used as an emulation model for the implementation of action-oriented process mining. The qualitative added value lies in the mitigation of risks associated with the automated process-regulating actions in production and logistics systems. Furthermore, the use of simulation allows the introduction of targeted imperfections to improve the testing of action-oriented process mining mechanisms. However, the current results can only be seen as a starting point for subsequent research challenges. These concern, among other things, the incorporation of a more dynamic action space and the explicit consideration of data and information quality. The current implementation uses a rules-based mechanism for generating event routines because the underlying use case is of lower complexity than industrial systems. Herein lies a limitation of the proposed approach. Possible research directions in this context include the exploration of action spaces using RL (since the current software implementation is already predestined for it). The consideration of data and information quality is crucial for the successful combined use of simulation and process mining and is therefore a subject for future research as well. Furthermore, the quantitative added value of the presented approach has yet to be shown in more complex production and logistics settings in subsequent studies.

## Literature


[1] Gutenschwager K, Rabe M, Spieckermann S, Wenzel S. *Simulation in Produktion und Logistik*. 1st Edition. Berlin, Heidelberg: Springer; 2017. 290 p.

[2] Verein Deutscher Ingenieure. VDI 3633 Blatt 1 – Simulation von Logistik-, Materialfluss- und Produktionssystemen – Grundlagen. 2014, Berlin: DIN Media.

[3] Bicalho-Hoch A L, Özkul F, Wittine N, Wenzel S. A Tool-Based Approach to Assess Simulation Worthiness and Specify Sponsor Needs for SMEs. In Feng B, Pedrielli G, Peng Y, Shashaani S, Song E, Corlu C G, Lee L H, Chew E P, Roeder T, Lendermann P, editors. Proceedings of the 2022 Winter Simulation Conference. *2022 Winter Simulation Conference (WSC)*; 2022 Dec; Singapore. Institute of Electrical and Electronics Engineers: Piscataway, New Jersey. 1818-1829. doi: https://doi.org/10.1109/WSC57314.2022.10015373.

[4] Follert G, Trautmann A. Emulation intralogistischer Systeme. In Wenzel S, editor. Simulation in Produktion und Logistik. 12. ASIM-Fachtagung; 2023 Sep; Kassel. San Diego, Erlangen: Society for Modeling and Simulation International SCS Publishing House e.V. 521-530.

[5] Özkul F, Sutherland R, Wenzel S, Jessen U, Spieckermann S. Verknüpfung von ereignisdiskreter Simulation und Process Mining in Produktion und Logistik. In Breitenecker F, Deatcu C, Durak U, Körner A, Pawletta T, editors. ASIM SST 2022 Proceedings Langbeiträge. 26. ASIM Symposium Simulationstechnik; 2022 Jul; Vienna. ARGESIM Publisher: Vienna. 39-48. doi: https://doi.org/10.11128/arep.20.a2035.

[6] Spieckermann S, Stöhr N, Mayer G, Özkul F, Wenzel S. Fallbeispiele aus Produktion und Logistik für die Verknüpfung von ereignisdiskreter Simulation und Process-Mining. In Bergmann S, Feldkamp N, Souren R, Straßburger S, editors. Simulation in Produktion und Logistik. 20. ASIM-Fachtagung; 2023 Sep; Ilmenau. Ilmenau: Universitätsverlag Ilmenau. 155-165, doi: https://doi.org/10.22032/dbt.57785.

[7] Van der Aalst W M P. Process Mining: A 360 Degree Overview. In: Van der Aalst W M P, Carmona J, editors. *Process Mining Handbook*. 1st edition. Cham: Springer International Publishing; 2022. p 3-34.

[8] Van der Aalst W M P. *Process mining. Data science in action*. 2nd Edition. Berlin, Heidelberg: Springer Publishing; 2016. 467 p.

[9] Burattin A. Streaming Process Mining. In: Van der Aalst W M P, Carmona J, editors. *Process Mining Handbook*. 1st edition. Cham: Springer Publishing; 2022. p 349-372.

[10] vom Brocke J, Van der Aalst W M P, Grisold T, Kremser W, Mendling J, Pentland B, Recker J, Roeglinger M, Rosemann M, Weber B. Process Science: The Interdisciplinary Study of Continuous Change. *SSRN Electronic Journal*. 2021; 1-9. doi: https://dx.doi.org/10.2139/ssrn.3916817.

[11] Reisig W. *Understanding Petri Nets*. 1st Edition, Berlin, Heidelberg: Springer Publishing; 2013. 236 p.

[12] Carmona J, Van Dongen B, Solti A, Weidlich M. *Conformance checking. Relating processes and models*. 1st Edition, Cham: Springer Nature; 2018. 285 p.

[13] Van Zelst S J, Van Dongen B F, Van der Aalst W M P. Event stream-based process discovery using abstract representations. *Knowledge and Information Systems*. 2017; 54: 407-435. doi: https://doi.org/10.1007/s10115-017-1060-2.

[14] Van Zelst S J, Bolt A, Hassani M, Van Dongen B F, Van der Aalst W M P. Online conformance checking: relating event streams to process models using prefix-alignments. *International Journal of Data Science and Analytics.* 2019; 8: 269-284. doi: https://doi.org/10.1007/s41060-017-0078-6.

[15] Burattin A, Van Zelst S J, Armas-Cervantes A, Van Dongen B F, Carmona J. Online Conformance Checking Using Behavioural Patterns. In Weske M, Montali M, Weber I, vom Brocke J, editors. Business Process Management. *16th International Conference on Business Process Management*; 2018 Sep; Sydney, Australia. Cham: Springer Publishing. 250-267. doi: https://doi.org/10.1007/978-3-319-98648-7_15.

[16] Stertz F, Mangler J, Rinderle-Ma S. Temporal Conformance Checking at Runtime based on Time-infused Process Models. 2020; doi: https://doi.org/10.48550/arXiv.2008.07262.

[17] Crocker L, Algina J. *Introduction to classical and modern test theory*. 2nd Edition, New York: Holt, Rinehart and Winston; 1986. 527 p.

[18] Park G, Van der Aalst W M P. Action-oriented process mining: bridging the gap between insights and actions. *Progress in Artificial Intelligence*. 2022; 1-22. doi: https://doi.org/10.1007/s13748-022-00281-7.

[19] Drieschner C, Heckl A, Krcmar H. Simulation Tool for Learning Action-Oriented Process Mining. In Castro M, El-Abd M, Zeid A, editors. EDUCON 2023 Conference Proceedings. *IEEE Global Engineering Education Conference 2023*, 2023 May; Kuwait. Institute of Electrical and Electronics Engineers: Piscataway, New Jersey. 1-8. doi: https://doi.org/10.1109/EDUCON54358.2023.10125248.

[20] Wolfe-Adam T, Alpyne - A Python library for executing AnyLogic models, Anylogic. https://github.com/t-wolfeadam/Alpyne.

[21] Burattin A. Beamline: A comprehensive toolkit for research and development of streaming process mining. *Software Impacts.* 2023; 17(100551):1-3. doi: https://doi.org/10.1016/j.simpa.2023.100551.

[22] Berti A, Van Zelst S, Schuster D. PM4Py: A process mining library for Python. *Software Impacts*. 2023 17(100556):1-7. doi: https://doi.org/10.1016/j.simpa.2023.100556.