

Highlights

Analyzing Structural, Operational, and Performance Impact of Process-Aware Information System Updates

Gyunam Park, Marco Comuzzi, Wil M. P. van der Aalst

- Novel data-driven method to assess structural, operational, and performance impacts of PAIS updates
- Digital Twin Interface Model leveraging Object-Centric Petri Nets for comprehensive system modeling
- Four-step approach identifying changes and analyzing impacts at multiple organizational levels
- Practical implementation through web-based software artifact and evaluation with real-life cases

Analyzing Structural, Operational, and Performance Impact of Process-Aware Information System Updates

Gyunam Park^a, Marco Comuzzi^{b,*}, Wil M. P. van der Aalst^c

^a*Fraunhofer Institute for Applied Information Technology FIT, Sankt Augustin, 53757, Germany*

^b*Department of Industrial Engineering, Ulsan National Institute of Science Technology, Ulsan, 44919, Republic of Korea*

^c*Process and Data Science Group, RWTH Aachen university, Aachen, 52074, NRW, Germany*

Abstract

Digital transformation often entails small-scale updates to the business functions exposed by enterprise systems, which support the execution of organizational business processes. These changes may increase operational friction in process execution, decreasing the process performance, increasing the complexity of maintenance activities, and making system documentation and training obsolete. Detecting and understanding their impact on process execution is therefore crucial to maintain smooth operations. Previous research contributions concerning the tracking and impact analysis of information systems updates are limited in scope, functionality, and tool support. In this paper, we propose a data-driven approach that uses the recently developed Digital Twins of Organizations (DTOs) to assess the impact of (process-aware) information systems updates. We model the updates

*Corresponding author

Email addresses: gyunam.park@fit.fraunhofer.de (Gyunam Park), mcomuzzi@unist.ac.kr (Marco Comuzzi), wvdaalst@pads.rwth-aachen.de (Wil M. P. van der Aalst)

using the configuration of DTOs and the newly introduced concept of actions in DTOs. The impact of the application of actions in a given configuration is quantitatively assessed by different types of impacts: structural, operational, and performance-related. We show the feasibility of the proposed approach by discussing a software prototype implementing it, a case study involving artificially-generated data from a standard ERP procure-to-pay process, and a case study using real-world data from a loan application process.

Keywords: Business Process, System Update, Impact, Digital Twin

1. Introduction

Process-aware Information Systems (PAIS), such as ERP and CRM, play a key role in modern organizations, underpinning the execution of business processes [1]. Modern digital transformation often involves frequent and small-scale changes, or *updates*, to such systems, which are vital to enable continuous adaptation to dynamic business environments [2]. The need for such capabilities is illustrated by the measures needed to handle the Covid-19 pandemic. During the pandemic, organizations had to continuously adapt their systems to varying degrees of home-working workforce, or to variations of sales and orders that deviated extensively and unexpectedly from the usual seasonality.

PAIS updates may deeply affect an organization. They can increase operational friction during the execution of business processes, thus negatively impacting organizational efficiency. They may also negatively affect the management and maintenance of PAIS if not tracked correctly, decreasing the overall organizational effectiveness.

Due to the complex design and deep operational pervasiveness of modern PAIS, it is challenging to assess the impact of PAIS updates. Existing approaches focus on process schema evolution facilities and maintaining the formal correctness of process models [3] or mainly take a model-driven view on impact analysis [4, 5, 6]. Model-driven approaches are suitable to assess the structural impact of information system updates, like the number of business functions that an update may affect, while they are limited in addressing the impacts that concern the run time process execution, like the impact on specific process performance measures.

In fact, understanding the operational and performance impact of a PAIS update is crucial for the effective management of the update. At the operational level, the impact of a PAIS update concerns the running instances of business processes involving the business objects and functions subject to change. For instance, removing the need to collect some customer information for privacy reasons may affect those customers for whom the information has been collected already. At the performance level, PAIS updates may impact on diagnostic measures of business processes: for instance, decreasing the level of tolerated mismatch between invoices and payments may slow down the processing of procurement cases, which in turn decreases the throughput of the procurement process.

To illustrate the problem more concretely, consider a Procure-To-Pay (P2P) process supported by an ERP system. The process involves business functions such as *create purchase requisition*, *create purchase order*, *create goods receipt*, *verify material*, *create invoice receipt*, and *clear invoice*, which manipulate business objects such as *purchase orders* and *materials*. Suppose

that, to alleviate a bottleneck in handling purchased goods, the organization decides to update the constraint rule of the *create purchase order* function by increasing the minimum material quantity required to create a purchase order from 5 to 10 units. This seemingly simple update has repercussions along all three impact dimensions. From a *structural* perspective, the change propagates upstream to the *create purchase requisition* function, since some requisitions will no longer lead to a purchase order, and downstream to all subsequent functions (goods receipt, material verification, invoicing, and invoice clearing), whose workload and input characteristics may change. From an *operational* perspective, at the time of the update, dozens of purchase requisitions and materials are already in-flight in the process and may be affected by the new rule. From a *performance* perspective, the update may reduce the average sojourn time of the *create purchase order* function due to a shorter queue, while also reducing the total number of purchase orders processed during the change period. Assessing such multi-dimensional impacts in a systematic and data-driven manner is the core challenge that this paper addresses.

More precisely, the problem addressed in this paper is: given a PAIS and an update to the business rules of a business function, how can we systematically identify and quantify the structural, operational, and performance impacts of that update, leveraging data recorded during system execution?

In [7], the authors introduced a conceptual framework to examine the structural, operational, and performance impacts using Digital Twins of Organizations (DTOs) [8]. A DTO is a digital replication of an organization's operations, providing a transparent view to such operations. Specifically,

the framework utilizes *Digital Twin Interface Model* (DT-IM) that models (i) a PAIS based on Object-Centric Petri Nets (OCPNs) [9] and (ii) PAIS updates using the notion of *actions*. The DT-IM facilitates the evaluation of a PAIS update’s impact by analyzing the modified elements of the OCPN (structural impacts), the operational states affected by the update (operational impacts), and the changes in diagnostic measures after the update (performance impacts).

This paper presents a data-driven method that implements the aforementioned conceptual framework. Our proposed method unfolds in four steps. Initially, we identify the *changes* caused by a PAIS update, where a change is defined as a modification in the business rules due to an action. Subsequent to identifying changes, we embark on three distinct impact analyses: firstly, assessing structural impacts via the DT-IM elements; secondly, evaluating operational impacts by leveraging event logs recorded by the system; and thirdly, examining performance impacts by computing performance measures from the event logs.

In this paper, we take a design science perspective to tackle the problem of modeling and assessing the impact of process-aware information systems. The practical relevance of the problem that we address lies with the increasing relevance of digital transformation for modern enterprises [2]. The rigor of the design process is ensured by extending a sound conceptual model of DTOs already published in the literature, i.e., DT-IMs [10]. The design search process is inspired by the existing literature on ERP post-implementation change management [5], from which we draw ideas to model system updates and evaluate their impact. The design science artifact produced by this work

is the impact analysis method and its prototype implementation. For the evaluation, coherently with the guidelines for design-oriented IS research by Österle et al. [11], we consider the development of a pilot application prototype and its demonstration through scenario-based case studies. To this aim, we have built a Web-based impact analysis software artifact. Moreover, we have conducted a case study based on a procure-to-pay process resembling the standard one of the SAP ERP system (using the same business objects, functions, and constraint and operation rules found in a real SAP ERP environment) and a case study using real-life event log data from a loan application process in a Dutch financial institute.

The remainder is organized as follows. We discuss the related work in Section 2. Then, we present the preliminaries in Section 3. Next, we introduce the DT-IM in Section 4 and our approach to impact analysis based on the DT-IM in Section 5. Section 6 presents the evaluation of the proposed approach. Section 7 discusses insights and limitations. Finally, Section 8 concludes the paper.

2. Related Work

We divide the discussion of related work into three parts. Section 2.1 reviews how PAIS and their updates have been conceptualised in the literature. Section 2.2 reviews approaches in the literature to manage business process specifications and service evolution. Finally, Section 2.3 discusses more specifically proposals to build digital twins of business processes.

2.1. Process-Aware Information Systems

PAIS have been conceptualised as a class of information systems that manage the execution of business processes, orchestrating people and other resources to achieve tangible business goals [1, 12]. A rich body of work on PAIS addresses dynamic change and evolution at the process schema, instance, organisational, and operational levels [13, 14], identifying both desirable change patterns and the technical mechanisms required to support them. The ADEPT project [3] is a seminal line of work that develops workflow management technology supporting both ad-hoc changes at the instance level and structural evolution at the process type level, while preserving formal correctness properties such as absence of deadlocks, proper completion, and compliance with temporal constraints. Its evolution ADEPT2 extends these ideas with systematic schema-evolution facilities and on-the-fly migration of running instances [15, 14]. In addition to control-flow evolution, ADEPT approaches explicitly address changes in the organisational model (e.g., restructuring organisational entities or updating access-control and authorization rules) and in the operational layer (e.g., updating application services bound to process activities) [16, 17], analyzing the impact of such changes on structural correctness, instance states, and compliance constraints. From this perspective, the present work contributes to this stream of research by integrating the management of PAIS changes with both structural and run-time impact analysis enabled by digital-twin-based models.

Enterprise Resource Planning (ERP) systems are a class of PAIS ubiquitously adopted by organizations and extensively studied in the literature [1, 18]. Post-implementation is a critical phase for ERP systems, as companies

often struggle to realize the benefits of large investments in ERP implementation and configuration [18]. In this phase, ERP systems require continuous updates to face changing business requirements, volatile markets, and technological advances [19]. The information system updates considered in this paper correspond to functional upgrades and enhancements in ERP systems [20], which modify or extend existing business functionality—business functions or processes—to better meet organisational requirements.

2.2. Workflow and Service Evolution

Early work on workflow evolution [21] defines a minimal and consistent set of primitives for modifying workflows—such as declaring variables, adding or merging tasks—and techniques for handling running instances, including migration or ad-hoc changes. Flexibility can be handled at the schema level or via ad-hoc interventions on running instances [13]; at the schema level, it can be achieved through configuration and adaptation [22]: the former assumes known evolution options that could be modelled using configurable notations [23, 24], whereas adaptation allows on-the-fly changes and is better suited to unexpected situations. Runtime flexibility is also achieved by underspecifying parts of processes [25] or repairing them when they interfere at execution time [26].

In the context of ERP systems, Soffer et al. [4, 27] propose an ERP-modelling language and alignment methodology that, however, does not explicitly represent changes or their impact. Lin et al.’s [28] proposal also supports users to manage ERP post-implementation changes, but without impact evaluation. Parhizkar and Comuzzi [5] propose methods and tool support for impact analysis of ERP changes inspired by engineering change management, pro-

viding a bridge between product-lifecycle management and IS-change management [29]. These works, together with PAIS-oriented change-support frameworks [14], highlight that managing updates in process-aware systems requires not only syntactic transformation rules but also explicit reasoning about correctness, compliance, and operational continuity.

In service-oriented computing, many works address service evolution and updates, including transformation frameworks for evolving service-oriented architectures through model-driven and pattern-based techniques [30, 31]. Such frameworks define transformation layers in which pattern-based or rule-based changes are composed to preserve consistency under structural and interface changes. Complementary approaches adopt MAPE-K-style feedback loops for data-driven adaptation [32], where Monitoring, Analysis, Planning, and Execution use a shared Knowledge base to derive reconfiguration actions such as service re-routing, scaling, or substitution [33]. These instantiate autonomic-computing principles in service-oriented and cloud environments [32], and provide a conceptual basis for integrating process-aware control with data-driven adaptation. Our work aligns with this direction by proposing digital-twin-enabled impact analysis of PAIS updates over running process executions.

2.3. Digital twins of business processes

The impact of process model changes on process run-time performance can be analyzed using business process simulation [34] and, more recently, using digital twins of business processes, which can be extended into full-fledged digital twins of organizations (DTOs) [35, 36]. DTOs can be broadly defined as software models that leverage organizational data to dynamically

monitor, analyze, and improve organizational activities over time. While there is growing practical interest in these models, research on DTOs is still relatively limited and only a few concrete implementations exist [35]. Compared to DTOs, traditional process simulation lacks continuous integration with the information systems generating the process execution data, as well as a structured means to characterize precisely the business rules associated with the functions used by a business process [35].

Fornari et al. [37] propose a research manifesto on digital twins of business processes, clarifying the relationship between process models and digital twins and outlining open research challenges around data integration, synchronization, multi-modeling, and data-driven simulation. They distinguish between a real-world business process, its process model, its digital shadow, and its digital twin, and emphasize that digital process twins should combine dynamic process models with real-time operational data and simulation capabilities to support monitoring, optimization, and what-if analysis. As far as concrete implementations are concerned, Compagnucci et al. [38] propose a conceptual architecture of a digital twin for process performance what-if analysis, which is instantiated on a synthetic event log on port logistics. This approach does not provide a taxonomy of process changes and only uses simulation to quantify changes on process KPIs based on process models changes.

3. A Meta-Model of Process-Aware Information Systems

Figure 1 introduces a meta-model of PAIS entities, updates, and impacts of updates considered in this work. This is inspired mainly by the work of Parhizkar and Comuzzi [5] in the context of ERP systems, which we extend

considering business rules associated to activities and explicitating different types of impacts of business function updates.

The meta-model is deliberately designed to capture a *minimum viable* set of entities sufficient to formalize the impact analysis of PAIS updates, rather than providing an exhaustive ontology of all PAIS aspects. The core entities, i.e., business processes, business functions, business objects, and business rules (constraint and operation rules), represent the essential elements through which PAIS updates propagate to affect system operations, consistently with standard PAIS conceptualizations [1, 12]. We acknowledge that several relevant PAIS aspects are not covered by this meta-model, including the organizational model (e.g., roles, access rights), fine-grained data-level dependencies beyond object attributes, cross-process interactions, and the presentation layer. These aspects are discussed as limitations in Section 7. We consider this scoping decision as a deliberate design choice: starting with a minimal yet well-founded set of entities allows us to demonstrate the feasibility and utility of the impact analysis approach, while providing a foundation that can be extended in future work.

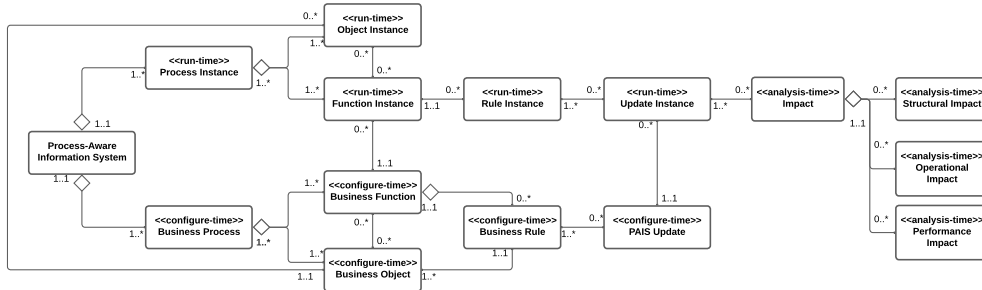


Figure 1: A meta-model of PAISs as a *UML 2.0 class diagram*

PAIS Entities. The meta-model separates entities concerned with the configure-time from entities related to the run-time. At configure-time, a PAIS is constituted by a set of business processes. A process involves business functions (e.g., *create purchase order*, *confirm order*, etc.) that manipulate business objects (e.g., *purchase order*, *material*, etc.). A business function is constrained by business rules defined over business objects. Generally, business rules translate organisational knowledge, such as compliance requirements or access rules to subscription services, into explicit statements, usually in the form if-then-else, that can be associated to individual business functions [39, 40]. The Business Rule Group [41] defines four categories of business rules: definition of business terms, facts relating business terms to each other, constraints, and derivations (operations). In this work, we focus on constraint rules and operation rules, which directly affect the operations embedded into business processes. Operation rules describe the functionality of business functions (i.e., how to manipulate business objects), while constraint rules restrict the execution of business functions. At run-time, a PAIS instantiates instances of processes, functions, and objects.

PAIS Updates. A PAIS may be modified by creating, updating, or deleting existing entities, i.e., business processes, objects, and business functions. In this work, we focus on updates to business functions (i.e., updates on the business rules of business functions). Creating a new entity extends the functionality of a system and only applies to new instances of business functions or objects. Therefore, it has no impact on business operations. Deleting an entity can be seen as a special case of updating it.

A PAIS update changes the business rules associated with a business

function. For instance, a *place order* function in an ERP system can be updated by changing the minimum price to create orders (i.e., a constraint rule) or changing the set of attributes that the function updates in a business object (i.e., an operation rule). At run time, a PAIS update is instantiated as an update instance. An update instance modifies rule instances, which in turn changes the operation and constraint of the associated function instance.

Impacts of PAIS Updates. Any update instance results in an impact on the system. In this work, we consider a general definition of impact from the change management literature [42], where the impact of a change can be defined as any qualitative consideration or qualitative measure that helps assessing the design feasibility and cost of a change request. Practically, we extend the notion of impact of a business function change defined by Comuzzi and Parhizkar [5], which is limited to counting the number of function, processes (static impact) and running process instances (runtime impact) affected by a change, by considering also the impact of function updates on the value of typical performance indicators. Therefore, in this work, we consider three types of impacts:

- *structural impacts*, which concern configure-time business functions, e.g., the number of impacted business functions. These are equivalent to the static impact defined in [29, 5];
- *operational impacts*, which concern run-time object instances, e.g., the number of impacted object instances. These are equivalent to the run-time impact defined in [29, 5];
- *performance impacts*, which concern changes in diagnostic measures

of business objects and functions, e.g., the difference in the average throughput time of a business object before/after updates or the difference in the average waiting time for a business function before/after updates.

4. Modeling PAISs: Digital Twin Interface Model

In this work, we use a *digital twin interface model* to model PAIS entities along with PAIS updates. The DT-IM uses Object-Centric Petri Nets (OCPNs) as its core formalism. This choice is motivated by the following considerations. First, real-world PAIS processes involve multiple interacting object types, e.g., in a P2P process, a purchase order may contain multiple materials, each linked to goods receipts and invoices. Traditional single-case process models cannot natively represent such many-to-many relationships, making an object-centric representation essential. Second, Petri nets provide formal execution semantics based on markings and transition firings, which naturally support both structural analysis (through the partial order on transitions) and operational analysis (through token-based replay on event logs). Third, OCPNs extend Petri nets with object types associated to places and variable arcs, enabling the modeling of object-centric processes while maintaining a direct connection to Object-Centric Event Logs (OCELs) [9]. This creates a coherent link between the process model, the event data, and the impact analysis that is lacking in alternative formalisms, e.g., colored Petri nets can model multiple object types, but lack the direct connection to OCELs, why BPMN extensions provide graphical expressiveness, but lack the formal replay semantics needed for operational impact analysis. OCPNs

offer the best balance between formal rigor and practical applicability for our data-driven approach.

Before defining the model, we introduce required background knowledge on OCPNs and OCELs. Note that the formal definitions provided throughout this and the following sections are necessary for precision and reproducibility; they also directly guide the prototype implementation discussed in Section 6.

4.1. Preliminaries: Object-Centric Petri Nets (OCPNs) and Object-Centric Event Logs (OCELs)

First, a Petri net is a directed bipartite graph of places and transitions. A labeled Petri net is a Petri net with labeled transitions.

Definition 1 (Labeled Petri Net). *Let $\mathbb{U}_{act}$ be the universe of activity names. A labeled Petri net is a tuple $N=(P, T, F, l)$ with P the set of places, T the set of transitions, $P \cap T = \emptyset$, $F \subseteq (P \times T) \cup (T \times P)$ the flow relation, and $l \in T \rightarrow \mathbb{U}_{act}$ a labeling function.*

An OCPN is a labeled Petri net in which each place is associated with an object type to represent interactions among different object types. It may also include *variable* arcs, representing the consumption/production of a variable number of tokens in one single transition firing.

Definition 2 (Object-Centric Petri Net). *Let $\mathbb{U}_{ot}$ be the universe of object types. An object-centric Petri net is a tuple $ON=(N, OT, pt, F_{var})$ where $N=(P, T, F, l)$ is a labeled Petri net, $OT \subseteq \mathbb{U}_{ot}$ is a set of object types, $pt \in P \rightarrow OT$ maps places to object types, and $F_{var} \subseteq F$ is the subset of variable arcs. $\preceq_{ON}$ is a partial order on the set of transitions T , i.e., for any*

two transitions $t_1, t_2 \in T$, if $t_1 \preceq_{ON} t_2$ and $t_2 \preceq_{ON} t_1$, then $t_1 = t_2$, and if $t_1 \preceq_{ON} t_2$ and $t_2 \preceq_{ON} t_3$, then $t_1 \preceq_{ON} t_3$.

Figure 2 shows an OCPN describing a part of a Procure-To-Pay process. There are two types of places associated with two object types, i.e., *purchase order* (denoted in brown) and *material* (blue). For instance, $pt(p1)=pt(p3)=\textit{purchase order}$, $pt(p2)=pt(p4)=\textit{material}$, and $(p2, t1)$, $(t1, p4)$, and $(p4, t2)$ are variable arcs and, as an example, $(p2, t1)$ models that a purchase order includes one or more materials. Note that we assume a partial order relation on the transitions, e.g., $t1 \preceq_{ON} t2$, $t2 \preceq_{ON} t3$, etc.

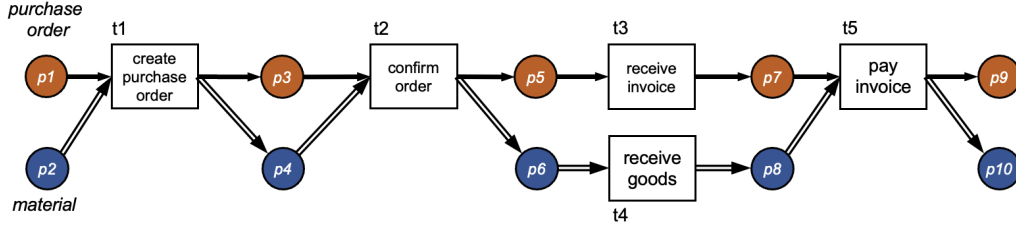


Figure 2: An example of object-centric Petri nets

Definition 3 (Notations). Let $ON=(N, OT, pt, F_{var})$ be an object-centric Petri net with $N=(P, T, F, l)$.

- $\bullet t = \{p \in P \mid (p, t) \in F\}$ is the set of input places of $t \in T$.
- $t \bullet = \{p \in P \mid (t, p) \in F\}$ is the set of output places of $t \in T$.

For instance, in Figure 2, $\bullet t2=\{p3, p4\}$ and $t2 \bullet =\{p5, p6\}$.

Definition 4 (Marking). Let $\mathbb{U}_{oi}$ be the universe of object identifiers. Let $type \in \mathbb{U}_{oi} \rightarrow \mathbb{U}_{ot}$ be a function mapping object identifiers to object types. Let

$ON=(N, OT, pt, F_{var})$ be an object-centric Petri net, where $N=(P, T, F, l)$. $Q_{ON}=\{(p, oi) \in P \times \mathbb{U}_{oi} \mid type(oi)=pt(p)\}$ is the set of possible tokens. A marking M of ON is a multiset of tokens, i.e., $M \in \mathcal{B}(Q_{ON})$.

For instance, the marking $M_1=[(p3, po1), (p4, m1), (p4, m2)]$ in Figure 2 denotes three tokens where place $p3$ has one token referring to object $po1$ of type *purchase order* and $p4$ has two tokens referring to objects $m1$ and $m2$ of type *material*.

We define the semantics of an OCPN using the notion of binding. A binding describes the execution of a transition consuming objects from its input places and producing objects in its output places.

Definition 5 (Binding). Let $\mathbb{U}_{omap}=\{omap \in \mathbb{U}_{ot} \not\rightarrow \mathcal{P}(\mathbb{U}_{oi}) \mid \forall_{ot \in dom(omap)} \forall_{oi \in omap(ot)} type(oi)=ot\}$ be the universe of all object mappings indicating which object identifiers are included per type. Let $ON=(N, OT, pt, F_{var})$ be an object-centric Petri net with $N=(P, T, F, l)$. A binding $(t, b) \in T \times \mathbb{U}_{omap}$ is a tuple of transition t and object mapping b specifying the objects consumed from the input places of t and produced to the output places of t .

For instance, $(t2, b1)$ in Figure 2 describes the execution of transition $t2$ with $b1$ where $b1(purchase\ order)=\{po1\}$ and $b1(material)=\{m1, m2\}$.

A binding (t, b) is *enabled* in marking M if all the objects specified by b exist in the input places of t . It is *fired* by consuming objects from the input places of transition t and producing the corresponding objects for the output places as specified by b .

Definition 6 (Binding Execution). Let $ON=(N, OT, pt, F_{var})$ be an object-centric Petri net with $N=(P, T, F, l)$. Let (t, b) be a binding. $cons(t, b)=[(p, oi) \in$

$Q_{ON} \mid p \in \bullet t \wedge oi \in b(pt(p))$] is the multiset of tokens to be consumed and $prod(t, b) = [(p, oi) \in Q_{ON} \mid p \in t \bullet \wedge oi \in b(pt(p))]$ is the multiset of tokens to be produced. Binding (t, b) is enabled in marking $M \in \mathcal{B}(Q_{ON})$ if $cons(t, b) \leq M$. The occurrence of an enabled binding (t, b) in marking M leads to the new marking $M' = M - cons(t, b) + prod(t, b)$, denoted as $M \xrightarrow{(t, b)} M'$.

For instance, $(t2, b1)$ in Figure 2 is enabled in marking M_1 since $po1$, $m1$, and $m2$ exist in its input places, i.e., $p3$ and $p4$. By executing $(t2, b1)$, $po1$ is removed from $p3$ and added to $p5$, while $m1$ and $m2$ are removed from $p4$ and added to $p6$, leading to the new marking $M' = [(p5, po1), (p6, m1), (p6, m2)]$.

An object-centric event log records the execution of processes supported by a PAIS.

Definition 7 (Object-Centric Event Log (OCEL)). Let $\mathbb{U}_{ei}$ be the universe of event identifiers, $\mathbb{U}_{attr}$ denotes the universe of attributes, and $\mathbb{U}_{val}$ denotes the universe of values. Let $\mathbb{U}_{map} = \mathbb{U}_{attr} \rightarrow \mathbb{U}_{val}$ be the set of all possible attribute-value mappings. An object-centric event log, denoted by L , is a quadruple $L = (O, E, R, \mu)$, where

- $O \subseteq \mathbb{U}_{oi}$ represents a collection of objects (i.e., object instances),
- $E \subseteq \mathbb{U}_{ei}$ represents a collection of events (i.e., function instances),
- $R \subseteq O \times E$ represents relationships between the events and objects, and
- $\mu \in (E \rightarrow \mathbb{U}_{map}) \cup (O \rightarrow \mathbb{U}_{map})$ represents a mapping function.

Table 1: A fragment of an event log with purchase requisition (pr), item (i), and purchase order (po).

event id	activity	timestamp	pr	i	po
e_{93}	<i>create purchase requisition</i>	25-05-2023:09.35	$\{r_1\}$	$\{i_1\}$	$\emptyset$
e_{94}	<i>approve item</i>	25-05-2023:13.35	$\emptyset$	$\{i_1\}$	$\emptyset$
e_{95}	<i>create purchase order</i>	25-05-2023:15.35	$\emptyset$	$\{i_2, i_3\}$	$\{o_2\}$
...	...	...	...	...	...

The set of all possible logs is denoted by $\mathbb{U}_L$.

Table 1 depicts a subset of an event log $L_1 = (O_1, E_1, R_1, \mu_1)$ with $O_1 = \{r_1, i_1, i_2, i_3, o_2, \dots\}$, $E_1 = \{e_{93}, e_{94}, \dots\}$, $R_1 = \{(r_1, e_{93}), (i_1, e_{93}), \dots\}$, $\mu(e_{93})(act) = \text{create purchase requisition}$, $\mu(e_{93})(time) = 25-05-2023:09.35$, etc.

4.2. Modeling PAIS Entities

A digital twin interface model represents a PAIS by 1) describing the process logic of a PAIS using an OCPN and 2) describing business rules, including constraint rules and operation rules, using linear rule expressions.

Definition 8 (Linear Rule Expression). Let $V \subseteq \mathbb{U}_{attr}$ be a set of variables. We denote the universe of all boolean expressions over attributes V as $EXPR(V)$. A linear rule expression is a boolean formula $expr \in EXPR(V)$ that evaluates to true or false. An evaluation function $eval_V \in EXPR(V) \times (V \twoheadrightarrow \mathbb{U}_{val}) \rightarrow \{\text{true}, \text{false}\}$ evaluates the expression to either true or false.

For instanc, given $V_1 = \{\text{quantity}, \text{product}\} \subseteq \mathbb{U}_{attr}$, we can define various linear rule expressions, e.g., $[\text{quantity} < 10] \in EXPR(V_1)$. Assume

that $vmap_1 \in V_1 \rightarrow \mathbb{U}_{val}$ describes a value assignment of an item with five webcams, each of which weighs 200g, i.e., $vmap_1(\mathbf{product}) = \mathit{webcam}$ and $vmap_1(\mathbf{quantity}) = 5$. Then, $eval_{V_1}([\mathbf{quantity} < 10], vmap_1) = \mathit{true}$ since $vmap_1(\mathbf{quantity}) < 10$.

A digital twin interface model consists of an OCPN and a collection of business rules, such as constraint rules and operation rules. The transitions of the OCPN represent *business functions*, whereas the object types indicate *business objects*. A constraint rule specifies a condition to enable the execution, whereas an operation rule describes a functionality, i.e., the derivation of new values based on the current values. The former is defined over variable names, and the latter is defined over variable names and their corresponding prime variables. A prime variable v' is used to distinguish the new data value that is written by a transition to variable $v \in \mathbb{U}_{attr}$ from its old value before the transition is executed (e.g., $\mathbf{price} \in \mathbb{U}_{attr}$ refers to an old price before changing the price, whereas $\mathbf{price}'$ refers to a new price after the change).

Definition 9 (Digital Twin Interface Models (DT-IMs)). *A digital twin interface model, denoted as DT , is a tuple (ON, V, CR, OR) where*

- $ON = (N, OT, pt, F_{var})$ is an OCPN where $N = (P, T, F, l)$,
- $V \subseteq \mathbb{U}_{attr}$ is a set of variable names,
- $CR \subseteq OT \times \mathit{EXPR}(V)$ is a set of constraint rules, and
- $OR \subseteq OT \times \mathit{EXPR}(V \cup V')$ is a set of operation rules where $V' = \{v' \mid v \in V\}$ is a set of prime variables.

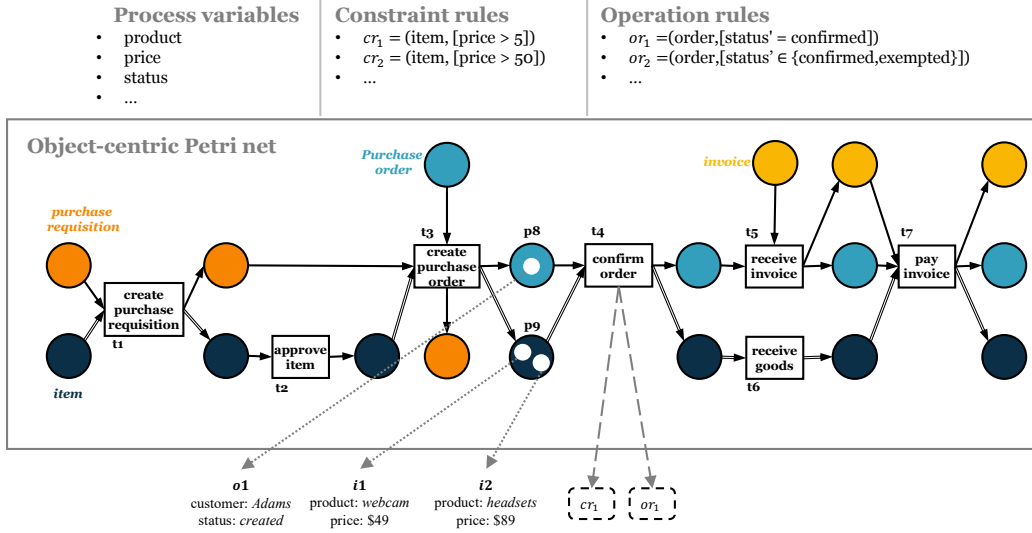


Figure 3: An example of digital twin interface model.

Figure 3 shows a DT-IM, $DT_1 = (ON_1, V_1, CR_1, OR_1)$, representing a PAIS supporting a simple Purchase-To-Procure (P2P) process. The set of variable names (V_1) includes **product**, **price**, **status**, etc. The set of constraint rules (CR_1) includes

- $cr_1 = (item, [price > 5])$, which represents that the price of an *item* should be higher than 5,
- $cr_2 = (item, [price > 50])$, which represents that the price of an *item* should be higher than 50.

The set of operation rules (OR_1) includes

- $or_1 = (order, [status' = confirmed])$, which represents that the status of an *order* should be updated to confirmed, and
- $or_2 = (order, [status' \in \{confirmed, exempted\}])$, which represents

that the status of an *order* should be updated to either confirmed or exempted.

A configuration of a DT-IM specifies the assignment of constraint and operation rules to transitions.

Definition 10 (Configuration). *Let $DT = (ON, V, CR, OR)$ be a DT-IM with $ON = (N, OT, pt, F_{var})$ and $N = (P, T, F, l)$. A configuration of DT is a tuple $(cmap, ormap)$ where $cmap \in T \rightarrow \mathcal{P}(CR)$ is a constraint rule assignment and $ormap \in T \rightarrow \mathcal{P}(OR)$ is an operation rule assignment. We denote C_{DT} to be the set of all possible configurations of DT .*

The gray dashed arrows in Figure 3 describe a configuration of DT_1 w.r.t. *confirm order*, i.e., $conf_1 = (cmap_1, ormap_1)$ such that $cmap_1(t4) = \{cr_1\}$ and $ormap_1(t4) = \{or_1\}$.

An operational state describes the status of a business process, i.e., which objects reside in which parts of the process, using the marking of OCPNs.

Definition 11 (Operational State of a DT-IM). *Let $DT = (ON, V, CR, OR)$ be a DT-IM with $ON = (N, OT, pt, F_{var})$ and $N = (P, T, F, l)$. An operational state of DT is a tuple $os_{DT} = (M, ovmap)$ where $M \in \mathcal{P}(Q_{ON})$ is a marking of ON and $ovmap \in \mathbb{U}_{oi} \rightarrow (V \rightarrow \mathbb{U}_{val})$ is an object value assignment s.t. $dom(ovmap) = \{oi \in \mathbb{U}_{oi} \mid \exists_{p \in P} (p, oi) \in M\}$. We denote OS_{DT} to be the set of all possible operational states of DT .*

The gray dotted arrows in Figure 3 describe an operational state of DT_1 , $os_1 = (M_1, ovmap_1) \in OS_{DT_1}$, where $M_1 = [(p8, o1), (p9, i1), (p9, i2)]$ and $dom(ovmap_1) = \{o1, i1, i2\}$. Purchase order *o1* is from *Adams*, i.e.,

$ovmap_1(o1)(customer) = Adams$. It contains two items: Item $i1$ is a \$49 webcam, i.e., $ovmap_1(i1)(product) = webcam$, and $ovmap_1(i1)(price) = 49$ and $i2$ is a \$89 headsets, i.e., $ovmap_1(i2)(product) = headsets$ and $ovmap_1(i2)(price) = 89$.

We define the semantics of DT-IMs using the concept of *interface bindings*, which is extended from the binding of an OCPN such that a binding is *enabled* with an additional condition: All the rules associated with the binding's transition should be satisfied with the configuration and operational state of the DT-IM.

Definition 12 (Interface Binding). *Let $DT = (ON, V, CR, OR)$ be a DT-IM with $ON = (N, OT, pt, F_{var})$ and $N = (P, T, F, l)$. An interface binding is a tuple (t, b, w) where (t, b) is a binding of ON and $w \in (\mathbb{U}_{oi} \rightarrow (V \rightarrow \mathbb{U}_{val}))$ is a function that writes objects' attribute values. An interface binding (t, b, w) is enabled with a configuration $(cmap, ormap) \in C_{DT}$ and an operational state $(M, ovmap) \in OS_{DT}$ if the following conditions are satisfied:*

- (t, b) is enabled at M ,
- $\forall_{(ot, cr) \in cmap(t), oi \in b(ot)} eval_V(cr, ovmap(oi)) = true$, and
- $\forall_{(ot, or) \in ormap(t), oi \in b(ot)} eval_V(or, ovmap'(oi)) = true$, where $ovmap'(oi) = ovmap(oi) \oplus^1 \{(v', w(oi)(v)) \mid v \in dom(w(oi))\}$.

For instance, an interface binding $(t4, b1, w1)$ describes the execution of transition $t4$ with $b1$ and $w1$ where $b1(purchase\ order) = \{o1\}$, $b1(item) =$

¹ $A \oplus B$ represents the update of mappings. Let $A \in X \rightarrow Y$ and $B \in X \rightarrow Y$ be two mappings from a set X to a set Y . The operation $A \oplus B$ is defined as a new mapping C

$\{i1, i2\}$, and $w1(status) = \text{confirmed}$. The interface binding is *enabled* with $conf_1 \in C_{DT_1}$ and $os_1 \in OS_{DT_1}$ since 1) $(t4, b1)$ is enabled at M_1 , 2) $(item, [\text{price} > 5])$ evaluates to *true* for all items with $ovmap_1(i1)(price) = 49$ and $ovmap_1(i2)(price) = 89$, and 3) $or_1 = (purchase\ order, [\text{status}' = \text{confirmed}])$ evaluates to *true* for $o1$, i.e., $ovmap'_1(o1)(status) = \text{confirmed}$.

The execution of an enabled binding leads to a new operational state with a new marking.

Definition 13 (Executing Interface Binding). *Let $DT = (ON, V, CR, OR)$ be a DT-IM with $ON = (N, OT, pt, F_{var})$ and $N = (P, T, F, l)$. The occurrence of an interface binding (t, b, w) , enabled with $conf \in C_{DT}$ and $(M, ovmap) \in OS_{DT}$, leads to the new marking M' s.t. $M \xrightarrow{(t,b)} M'$ and new object value mapping $ovmap' = ovmap \oplus w$.*

For instance, the execution of $(t4, b1, w1)$ leads to $(M_2, ovmap_2)$, where $M_2 = [(p10, o1), (p11, i1), (p11, i2)]$, $ovmap_2(o1) = ovmap_1(o1) \oplus \{(status, \text{confirmed})\}$.

4.3. Modeling PAIS Updates

As introduced in Section 3, a PAIS update modifies the business rules associated with business functions. An *action* in DT-IMs represents a PAIS update as the modification of configurations.

from X to Y , where $C(x)$ is given by:

$$(A \oplus B)(x) = \begin{cases} B(x) & \text{if } x \in \text{dom}(B), \\ A(x) & \text{if } x \notin \text{dom}(B) \text{ and } x \in \text{dom}(A). \end{cases}$$

Definition 14 (Action). Let DT be a DT -IM. An action $\alpha \in C_{DT} \rightarrow C_{DT}$ maps an old configuration to a new configuration. A_{DT} is the set of all possible actions defined over DT .

As shown in Figure 4, $\alpha_1 \in A_{DT_1}$ describes an action that changes the lower bound of the price of items to execute “confirm order” from 5 to 50, i.e., $\alpha_1(conf_1) = (crmap', ormap)$ where $crmap'(t4) = \{cr_2\}$. The action $\alpha_2 \in A_{DT_1}$ allows the exemption of order confirmations, i.e., $\alpha_2(conf_1) = (crmap, ormap')$ where $ormap'(t4) = \{or_2\}$.

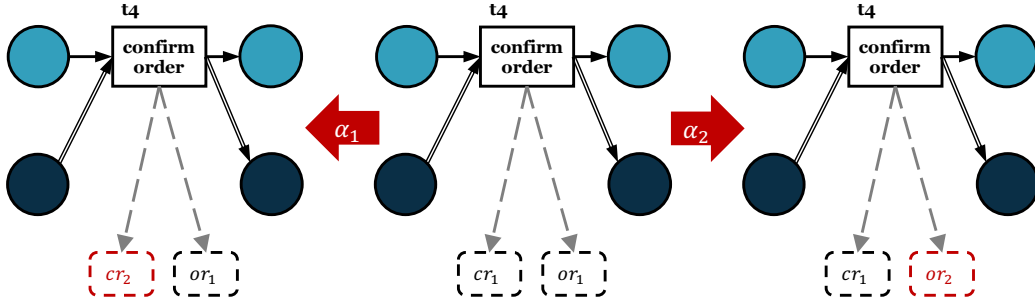


Figure 4: An example of actions

An action instance describes the enactment of an action. An action is applied at a certain start time, and, in principle, the new configurations that it entails can remain in place for the foreseeable future until a condition, e.g., on performance measures, is met or until a specific end time. For simplicity, in this work, we consider only the latter case, emphasizing the usefulness of temporal management of PAIS updates.

Definition 15 (Action Instance). Let DT be a DT -IM. Let $\mathbb{U}_{time}$ be the universe of timestamps. $\mathbb{U}_{tw} = \{(st, ct) \in \mathbb{U}_{time} \times \mathbb{U}_{time} \mid st < ct\}$ denotes the set of all possible time windows. An action instance $ai \in A_{DT} \times \mathbb{U}_{tw}$ is a

pair of an action and a time window. AI_{DT} is the set of all possible action instances defined over DT .

For instance, $ai_1 = (\alpha_1, (01.\text{Jan.}2024, 31.\text{Jan.}2024)) \in AI_{DT_1}$ describes the application of α_1 during the month of January 2024. Note that the action instance is permanent if the completion time is unbound.

5. Impact Analysis

This section introduces an approach to analyzing the structural, operational, and performance impacts of action instances using a DT-IM. Our proposed approach consists of four steps, as shown in Figure 5. First, we detect *changes* caused by action instances. A change refers to the transitions where the business rules are changed due to an action. Based on the changes, we conduct three impact analyses. First, we analyze structural impacts using the elements of the DT-IM. Second, we analyze operational impacts using the operational states of the DT-IM. Finally, we analyze performance impacts using performance measures associated with the transitions and object types of the DT-IM.

5.1. Change Detection

The concept of an action instance, as described in Subsection 4.3, refers to the execution of an action that changes configurations within a specific time window. A *change* in this context is understood as the transitions (i.e., business functions) affected by the action within the given time window. *Change detection* identifies the specific changes caused by an action instance under a particular configuration.

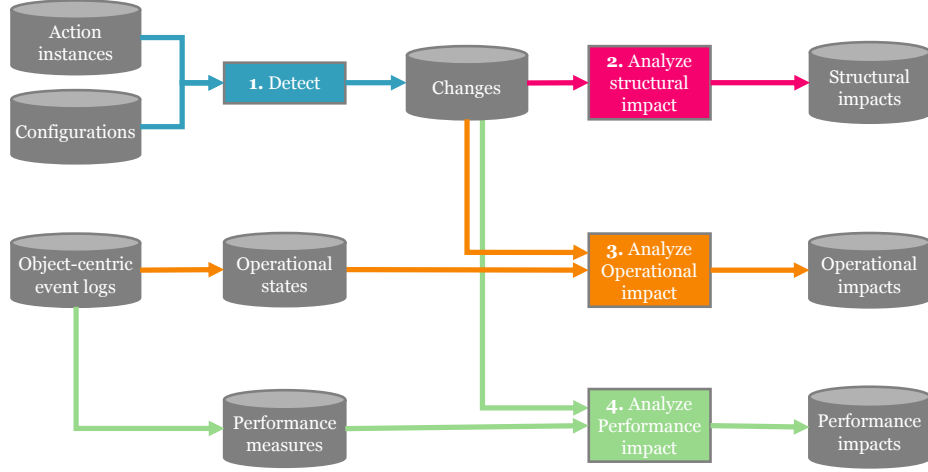


Figure 5: An overview of the proposed impact analysis using digital twin interface models.

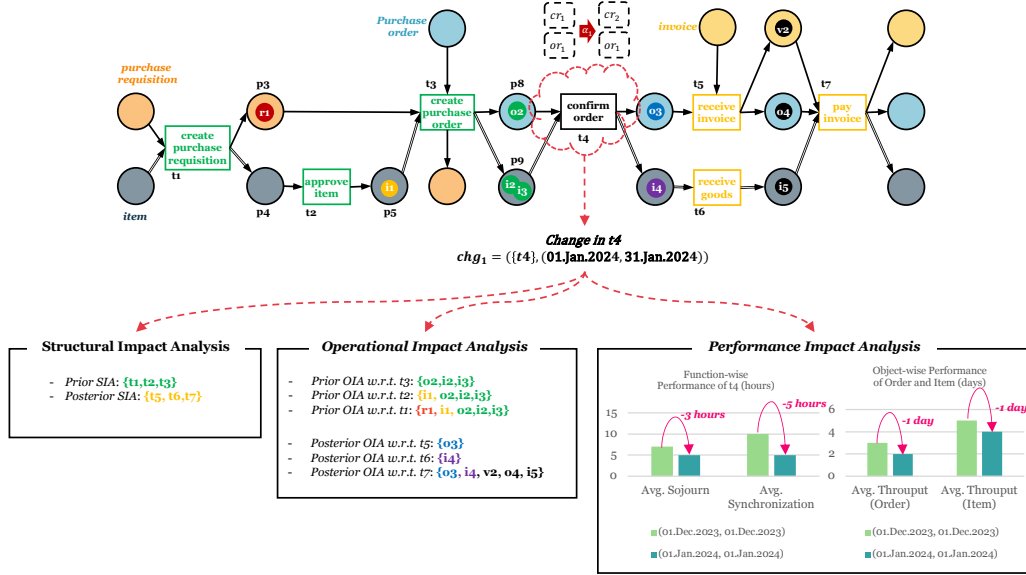


Figure 6: An example of the proposed impact analysis

Definition 16 (Change Detection). Let $DT = (ON, V, CR, OR)$ be a DT-IM with $ON = (N, OT, pt, F_{var})$ and $N = (P, T, F, l)$. Let $CHG_{DT} \subseteq \mathcal{P}(T) \times \mathbb{U}_{tw}$ denote the set of all the possible changes of DT. A change

detection $detect_{DT} \in (AI_{DT} \times C_{DT}) \rightarrow CHG_{DT}$ maps an action instance and a configuration to a change.

For instance, as illustrated in Figure 6, the change caused by $ai_1 \in AI_{DT}$ is identified as $t4$, occurring from 01.Jan.2024 to 31.Jan.2024. Formally, this is represented as $chg_1 = detect_{DT}(ai_1, conf_1) = (\{t4\}, (01.Jan.2024, 31.Jan.2024)) \in CHG_{DT}$. For any change $chg = (T, tw) \in CHG_{DT}$, the transition set is denoted by $\pi_T(chg) = T$ and the time window by $\pi_{tw}(chg) = tw$.

5.2. Structural Impact Analysis

After detecting changes, Structural Impact Analysis (SIA) is conducted to assess the structural impact of the changes. In this work, we introduce two SIAs: *prior SIA* and *posterior SIA*. The former focuses on analyzing transitions that occur prior to the identified changes.

Definition 17 (Prior SIA). Let $DT = (ON, V, CR, OR)$ be a DT-IM with $ON = (N, OT, pt, F_{var})$ and $N = (P, T, F, l)$. For any $chg \in CHG_{DT}$, a prior structural impact is defined as $sa_{DT}^{prior}(chg) = \{t \in T \mid t' \in \pi_T(chg) \wedge t \preceq_{ON} t' \wedge l(t) \neq \tau\}$.

As depicted in Figure 6 with the green-colored transitions, $sa_{DT}^{prior}(chg_1) = \{t1, t2, t3\}$ since $t1 \preceq_{ON} t4$, $t2 \preceq_{ON} t4$, and $t3 \preceq_{ON} t4$.

Next, the posterior SIA analyzes the transitions that follow the changed transitions.

Definition 18 (Posterior SIA). Let $DT = (ON, V, CR, OR)$ be a DT-IM with $ON = (N, OT, pt, F_{var})$ and $N = (P, T, F, l)$. For any $chg \in CHG_{DT}$, a posterior structural impact is defined as $sa_{DT}^{post}(chg) = \{t \in T \mid t' \in \pi_T(chg) \wedge t' \preceq_{ON} t \wedge l(t) \neq \tau\}$.

As depicted in Figure 6 with the yellow-colored transitions, $sa_{DT}^{post}(chg_1) = \{t5, t6, t7\}$ since $t4 \preceq_{ON} t5$, $t4 \preceq_{ON} t6$, and $t4 \preceq_{ON} t7$.

5.3. Operational Impact Analysis

Operational Impact Analysis (OIA) is designed to evaluate the impact of specific actions on the operational state within DT-IMs. We start OIA by *replaying* object-centric event logs to deduce the operational state from the event log. For example, replaying event e_{93} from the event log depicted in Table 1 results in producing tokens r_1 and i_1 at places $p3$ and $p4$, respectively. Following this, replaying event e_{94} leads to the consumption of token i_1 from $p4$ and its production at $p5$.

Definition 19 (Replay). *Let $DT = (ON, V, CR, OR)$ be a DT-IM with $ON = (N, OT, pt, F_{var})$ and $N = (P, T, F, l)$. $replay_{DT} \in \mathbb{U}_L \rightarrow OS_{DT}$ maps event logs to the operational states of DT.*

The process of replaying object-centric event logs on Object-Centric Petri Nets (OCPNs) is detailed in [9]. Note that we deliberately do not specify the update of object value assignments as they are not utilized in the subsequent OIA. Instead, we focus on updating the marking of the OCPNs.

Using the up-to-date operational states computed from object-centric event logs, we conduct two operational impact analyses: *prior OIA* and *posterior OIA*. First, the prior OIA analyzes the objects that will (possibly) be processed by the changed transitions. Given a change, the up-to-date operational state, and a transition, it returns the objects that were processed by the transition and will (possibly) be processed by changed transitions by analyzing the marking of the operational state.

Definition 20 (Prior OIA). Let $DT = (ON, V, CR, OR)$ be a DT-IM with $ON = (N, OT, pt, F_{var})$ and $N = (P, T, F, l)$. For $t, t' \in T$, $\Delta(t, t') = \{x \in T \mid t \preceq_{ON} x \wedge x \preceq_{ON} t'\}$ denotes the set of transitions between t and t' . For any $chg \in CHG_{DT}$, $os \in OS_{DT}$, and $t \in sa_{DT}^{prior}(chg)$, a prior operational impact is defined as $oia_{DT}^{prior}(chg, os, t) = \{oi \in \mathbb{U}_{oi} \mid \exists p \in P, t' \in \pi_T(chg), x \in \Delta(t, t')((p, x) \in F \vee (x, p) \in F) \wedge (p, oi) \in \pi_M(os)\}$.

Figure 6 describes the prior OIA for os_1 and chg_1 . As depicted with green tokens, $oia_{DT}^{prior}(chg_1, os_1, t3) = \{o2, i2, i3\}$, i.e., the change in $t4$ affects objects $o2$, $i2$, and $i3$ that were processed by $t3$ in the past and will be processed by $t4$ in the future. As depicted with green and yellow tokens, $oia_{DT}^{prior}(chg_1, os_1, t2) = \{i1, o2, i2, i3\}$, i.e., the change in $t4$ affects objects $i1$, $o2$, $i2$, and $i3$ that were processed by $t2$ in the past and will be processed by $t4$ in the future. As depicted with green, yellow, and red tokens, $oia_{DT}^{prior}(chg_1, os_1, t1) = \{r1, i1, o2, i2, i3\}$, i.e., the change in $t4$ affects objects $r1$, $i1$, $o2$, $i2$, and $i3$ that were processed by $t1$ in the past and will be processed by $t4$ in the future.

Next, the posterior OIA analyzes the objects that were (possibly) processed by the changed transitions. Given a change, the up-to-date operational state, and a transition, it returns the objects that can be processed by the transition and were (possibly) processed by changed transitions by analyzing the operational state.

Definition 21 (Posterior OIA). Let $DT = (ON, V, CR, OR)$ be a DT-IM with $ON = (N, OT, pt, F_{var})$ and $N = (P, T, F, l)$. For $t, t' \in T$, $\Delta(t, t') = \{x \in T \mid t \preceq_{ON} x \wedge x \preceq_{ON} t'\}$ denotes the set of transitions between

t and t' . For any $chg \in CHG_{DT}$, $os \in OS_{DT}$, and $t \in sa_{DT}^{post}(chg)$, a posterior operational impact is defined as $oia_{DT}^{post}(chg, os, t) = \{oi \in \mathbb{U}_{oi} \mid \exists_{p \in P, t' \in \pi_T(chg), x \in \Delta(t', t)}((p, x) \in F \vee (x, p) \in F) \wedge (p, oi) \in \pi_M(os)\}$.

Figure 6 describes the posterior OIA for os_1 and chg_1 . As depicted with blue tokens, $oia_{DT}^{post}(chg_1, os_1, t5) = \{o3\}$, i.e., the change in $t4$ affects purchase order $o3$ that was processed by $t4$ in the past and will be processed by $t5$ in the future. As depicted with purple tokens, $oia_{DT}^{post}(chg_1, os_1, t6) = \{i4\}$, i.e., the change in $t4$ affects item $i4$ that was processed by $t4$ in the past and will be processed by $t6$ in the future. As depicted with blue, purple, and black tokens, $oia_{DT}^{post}(chg_1, os_1, t7) = \{o3, i4, v2, o4, i5\}$, i.e., the change in $t4$ affects objects $o3$, $i4$, $v2$, $o4$, and $i5$ that were processed by $t4$ in the past and will be processed by $t7$ in the future.

5.4. Performance Impact Analysis

Performance impact analysis aims to analyze the performance impact of an action instance using specific performance measures. We categorize these measures into two types: *function-wise* and *object-wise* performance measures. Function-wise performance measures assess the efficiency of individual business functions, e.g., their average service time. These measures are quantified using transitions in DT-IMs. For a given transition and time window, a function-wise measure yields a performance measure specific to that transition within the specified period.

Definition 22 (Function-wise Performance Measures). *Given a DT-IM $DT = (ON, V, CR, OR)$ with $ON = (N, OT, pt, F_{var})$ and $N = (P, T, F, l)$,*

and a performance measure name fm , the function $fperf_{DT}^{fm} \in T \times \mathbb{U}_{tw} \rightarrow \mathbb{R}$ maps a transition and a time window to a performance value.

For example, $fperf_{DT}^{sojourn}(t4, (01.Dec.2023, 31.Dec.2023)) = 7$ (*hours*) denotes the average sojourn time for transition $t4$ between the specified dates.

The function-wise performance measure function can be implemented following the approach in [43], which entails coupling events from an object-centric event log to an OCPN and computing various performance measures *flow time*, *sojourn time*, *synchronization time*, *pooling time*, and *lagging time*.

In contrast, object-wise performance measures focus on the efficiency of instantiated business objects, like the average throughput time. These measures depend on the object type and a comparison time window, providing a performance measure specific to the object type within that period.

Definition 23 (Object-wise Performance Measures). *Given a DT-IM $DT = (ON, V, CR, OR)$ with $ON = (N, OT, pt, F_{var})$ and $N = (P, T, F, l)$, and an object-wise performance measure name om , the function $operf_{DT}^{om} \in OT \times \mathbb{U}_{tw} \rightarrow \mathbb{R}$ maps an object type and a time window to a performance value.*

For instance, $operf_{DT}^{throughput}(\text{purchase order}, (01.Dec.2023, 31.Dec.2023)) = 3$ (*days*) signifies the average throughput time of orders during the specified period.

The object-wise performance measure function can be implemented using the approach outlined in [44], which involves calculating various object-wise performance metrics such as *relevant events*, *throughput*, *work-in-process*, etc.

Performance impact analysis aims to evaluate the effects of an action instance on these performance measures. Function-wise Performance Impact

Analysis (FPA) examines the impact of an action instance on function-wise performance. It computes the difference in a performance measure between the change time window and a comparison time window for each transition involved in the change.

Definition 24 (Function-wise Performance Impact Analysis). *Given a DT-IM $DT = (ON, V, CR, OR)$ with $ON = (N, OT, pt, F_{var})$ and $N = (P, T, F, l)$, and a performance measure name fm , for any change $chg \in CHG_{DT}$ and comparison time window $tw_{comp} \in \mathbb{U}_{tw}$, the function-wise performance impact analysis is defined as $fpa_{DT}^{fm}(chg, tw_{comp}) = \{(t, diff) \in T \times \mathbb{R} \mid t \in \pi_T(chg) \wedge diff = fperf_{DT}^{fm}(t, \pi_{tw}(chg)) - fperf_{DT}^{fm}(t, tw_{comp})\}$.*

Figure 6 illustrates the FPA with respect to sojourn time and synchronization time for change chg_1 in comparison time window $tw_{comp}^1 = (01.Dec.2023, 31.Dec.2023)$. The sojourn time averages 7 hours in the comparison window and 4 hours during the change, leading to $fpa_{DT}^{sojourn}(chg_1, tw_{comp}^1)(t4) = -3$ hours, indicating a reduction in sojourn time by 3 hours for *confirm order*. Similarly, $fpa_{DT}^{sync}(chg_1, tw_{comp}^1)(t4) = -5$ hours, showing a reduction in synchronization time by 5 hours.

Object-wise Performance Impact Analysis (OPA) assesses the impact of an action instance on the performance of business objects. Like FPA, OPA calculates the difference in performance measures between the change duration and the comparison time window.

Definition 25 (Object-wise Performance Impact Analysis). *Given a DT-IM $DT = (ON, V, CR, OR)$ with $ON = (N, OT, pt, F_{var})$ and $N = (P, T, F, l)$, and an object-wise performance measure name om , for any change*

$chg \in CHG_{DT}$ and comparison time window $tw_{comp} \in \mathbb{U}_{tw}$, the object-wise performance impact analysis is defined as $opa_{DT}^{om}(chg, tw_{comp}) = \{(ot, diff) \in \mathbb{U}_{ot} \times \mathbb{R} \mid \exists_{t \in \pi_T(chg), p \in \bullet t} ot = pt(p) \wedge diff = operf_{DT}^{om}(ot, \pi_{tw}(chg)) - operf_{DT}^{om}(ot, tw_{comp})\}$.

Figure 6 also shows the OPA regarding throughput time for change chg_1 and comparison window tw_{comp}^1 . The throughput time of *purchase orders* averages 3 days in the comparison window and 2 days during the change. Therefore, $opa_{DT}^{throughput}(chg_1, tw_{comp}^1)(purchase\ order) = -1$ day, signifying a reduction in throughput time of *purchase orders* by 1 day on average. Additionally, $opa_{DT}^{throughput}(chg_1, tw_{comp}^1)(item) = -1$ day, indicating a reduction in average throughput time of *items* by 1 day.

6. Evaluation

In this section, we evaluate the approach presented in this paper. Coherently with our choice of following a design science approach [11], we discuss a multi-method evaluation involving (i) the development of a prototype system implementing the approach, (ii) a case study involving the simulation of a pseudo-real case using the implemented prototype, and (iii) a case study in which we use the implemented prototype with data generated by the execution of a real-life business process. The first evaluation demonstrates the feasibility of the proposed approach, whereas the last two show its utility in practical settings. Notably, the simulation-based case study is constructed to mirror a real SAP ERP system, using the same business objects, functions, and business rules (both constraint and operation rules) found in a standard SAP P2P environment, thus closely reflecting the conditions of a real-world

PAIS deployment. The real-life case study uses production event log data from an actual financial institution, demonstrating the applicability of the approach to real organizational data.

6.1. Prototype Implementation

We have implemented the proposed approach as a web-based application named *ProAct: Impact Analysis*. This tool consists of four integral functional modules: *impact analysis setup*, *structural impact analysis*, *operational impact analysis*, and *performance impact analysis*. The source code for this implementation has been made publicly accessible and can be found at <https://anonymous.4open.science/r/proact-IS/>. In addition, a comprehensive user manual detailing deployment procedures and usage guidelines is available at https://proact.readthedocs.io/en/latest/impact_analysis/.

First, the impact analysis setup component facilitates the creation of a DT-IM, relying on user interaction to enable the subsequent impact analysis. The OCPN is discovered using the technique suggested in [9], while constraint and operation rules are provided by the user. The initial configuration of the DT-IM is user-defined, where they assign specific rules to transitions. For that, the user assigns the rules to transitions. To determine the operational state of the DT-IM, we utilize the replay technique proposed in [9].

Second, the structural impact analysis component provides interactive bar charts for analyzing the structural impact of a selected action instance. As shown in Figure 7, the visual interface analyzes impacted activities and enables a comparative evaluation across various object types.

Third, the operational impact analysis component employs bar charts for both Prior and Posterior Operational Impact Analysis (Prior OIA and

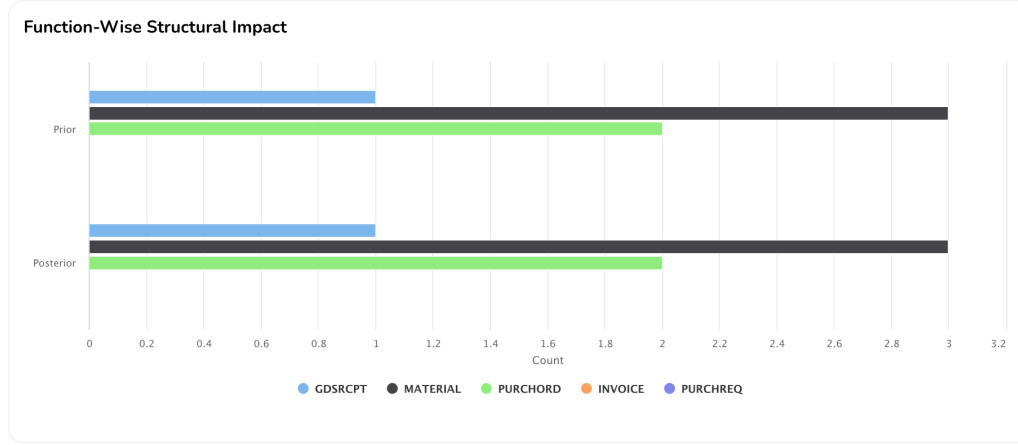


Figure 7: Visualization of the structural impact analysis. This illustrates the prior and posterior structural impacts of a change on five object types: *GDSRCPT*, *MATERIAL*, *PURCHORD*, *INVOICE*, and *PURCHREQ*. The horizontal bars represent the count of prior activities affected before (Prior) and after (Posterior) the implementation of the change.

Posterior OIA). As exemplified in Figure 8, the interface for Prior OIA reveals the action instance’s impact on various object types processed after specific activities and before the changed activity.

Finally, the performance impact analysis component is designed to offer both Function-wise Performance Impact Analysis (FPA) and Object-wise Performance Impact Analysis (OPA). The interface for FPA, as demonstrated in Figure 9, supports analyzing the function-wise performance impact of the selected action instance. The impact is quantified and color-coded, offering an in-depth analysis across different object types.

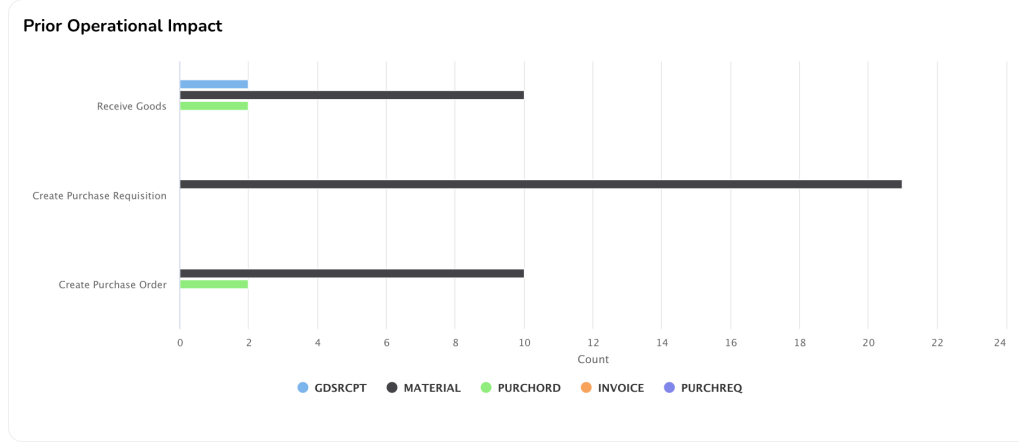


Figure 8: Visualization of the prior operational impact analysis. This displays the prior operational impact of a change in relation to the three activities in the process: *Receive Goods*, *Create Purchase Requisition*, and *Create Purchase Order*. The horizontal bars represent the count of impacted objects for each activity, categorized by different object types: *GDSRCPT*, *MATERIAL*, *PURCHORD*, *INVOICE*, and *PURCHREQ*.

6.2. Simulation-based Case Study

Using the implemented prototype, we have conducted a case study on an artificially-constructed PAIS that supports a Procure-To-Pay (P2P) process. The information system reflects a real-life SAP ERP system supporting a P2P process, using the same business objects and functions (operations and constraints) found in a SAP ERP system. Using the artificial PAIS, we have simulated the P2P process, considering 24 human resources with different capacities and performances. Purchase orders are created following an exponential distribution; the business hours are set as 9-17 from Monday to Friday; work assignments are scheduled using the *First-in-First-out* rule.

Figure 10 shows the DT-IM representing the PAIS (DT_{p2p}). The process involves five object types. First, a purchase requisition is created with

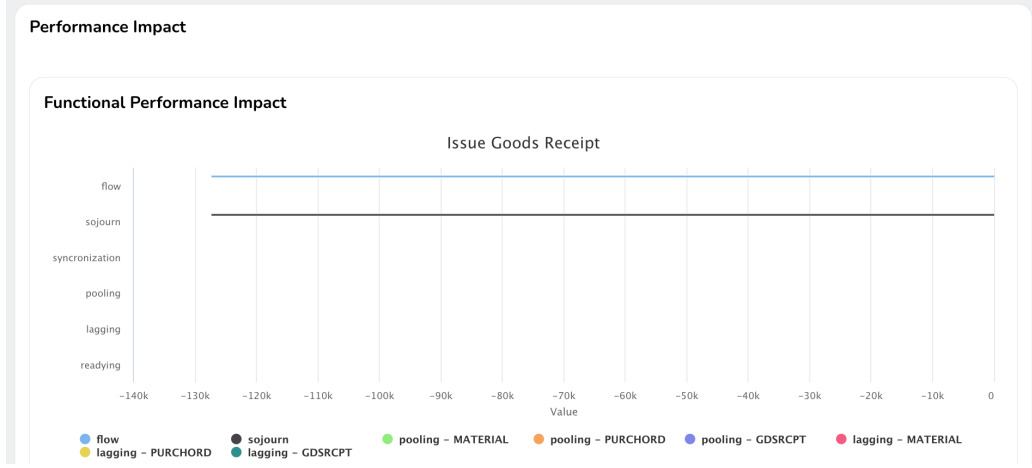


Figure 9: Visualization of the function-wise performance impact analysis. This illustrates the performance impact analysis for *Issue Goods Receipt*, showing how various performance metrics are affected by the changes. The x-axis represents the magnitude of the performance impact (measured in seconds), and the y-axis lists the performance measures: flow, sojourn, synchronization, pooling, lagging, and readying. Each bar represents the impact on a specific performance measure. Negative values indicate a performance improvement.

multiple materials. Next, a purchase order is created based on the purchase requisition and the materials. A goods receipt is produced after receiving the materials of the purchase order. Then, the material is verified and issued for various purposes, and, concurrently, the invoice for the purchase order is received. Finally, the invoice is cleared.

The process involves various constraint rules and operation rules. The dotted lines in Figure 10 show the default configuration of the system, $conf_{p2p}^1 = (crmap_{p2p}^1, ormap_{p2p}^1) \in C_{DT_{p2p}}$. For instance, *create purchase requisition* is associated with *cr1*, i.e., the planned delivery day of material should be higher than the minimum feasible delivery day (i.e., ten days). Moreover, it is associated with three operation rules *or1*, *or2*, and *or3*, i.e., the quantity,

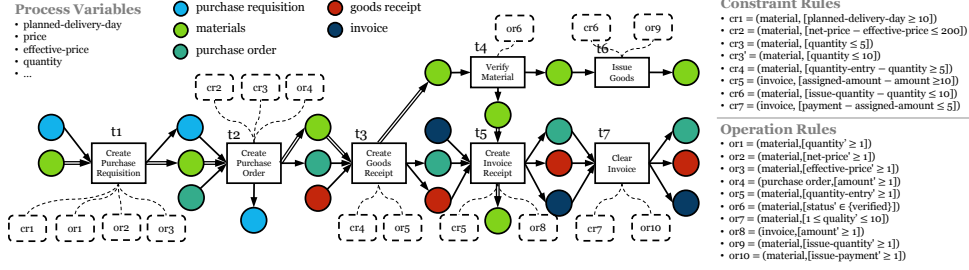


Figure 10: A DT-IM representing a P2P process. The dotted lines represent the default configuration $conf_{p2p}^1 \in C_{DT_{p2p}}$.

net price, and effective price of each material should be more than or equal to 1.

Action Instance. The process suffers from bottlenecks in handling purchased goods, e.g., creating goods receipts, verifying materials, and clearing invoices. The root cause of the inefficiency in the process is an excessive number of purchase orders received, which cannot be handled in the later steps of the process. To tackle the problem, we reduce the number of purchase orders by prioritizing purchase requisitions that contain a higher quantity of materials and postponing the creation of purchase orders for purchase requisitions involving smaller quantities of materials.

More concretely, we implement an action $\alpha_{p2p} \in A_{DT_{p2p}}$ that changes the constraint rule assigned to $t2$ such that a higher quantity of material is required to create a purchase order, i.e., $\alpha_{p2p}(conf_{p2p}^1) = (crmap_{p2p}^{1'}, ormap_{p2p}^1)$ where $crmap_{p2p}^{1'} = crmap_{p2p}^1 \oplus (t2, \{cr3'\})$. The minimum quantity of each material should be higher than or equal to 10 (5 more than in the current configuration). We execute α_{p2p} for five days from 09:00 21-12-2021 to 09:00 25-12-2021, represented as an action instance $ai_1 = (\alpha_{p2p}, (09:00\ 21-12-2021, 09:00\ 25-12-2021)) \in$

$AI_{DT_{p2p}}$. After the period, the criteria are set back according to the original rule, i.e., $cr3$, processing the purchase orders with a lower quantity of material.

Change Detection. The change caused by the action instance is $t2$ from 09:00 21-12-2021 to 09:00 25-12-2021, i.e., $chg_{p2p}^1 = detect(ai_1, conf_{p2p}^1) = (\{t2\}, (09:00\ 21-12-2021, 09:00\ 25-12-2021))$. The structural, operational, and performance impact of the change are analyzed next.

Structural Impact Analysis. The prior structural impact refers to *create purchase requisition*, i.e., $sa_{DT_{p2p}}^{prior}(chg_{p2p}^1) = \{t1\}$. The design of *create purchase requisition* may be affected since certain requisitions will not be transformed into purchase orders by design. The posterior structural impact is more diverse, i.e., $sa_{DT_{p2p}}^{post}(chg_{p2p}^1) = \{t3, t4, t5, t6, t7\}$. For instance, the design of *create goods receipt*, i.e., $t3$, may need to be enhanced due to the increased variety of materials in a single order.

Operational Impact Analysis. Figure 11(a) shows the prior operational impact of the action instance on ongoing purchase requisitions and materials. 37 purchase requisitions and 95 materials are affected after *create purchase requisition* ($t1$). Since we limit the execution of the activity, some of the purchase requisitions will not proceed in the process. Figure 11(b-f) show the posterior operational impact of the action instance in different phases of the process. First, 8 purchase orders and 23 materials are affected before *create goods receipt* ($t3$) and 58 materials are affected before *verify material* ($t4$). Moreover, before *create invoice receipt* ($t5$), 24 purchase orders are affected, while 83 materials and 16 goods receipts are affected. Also, 78 materials are affected before *issue goods* ($t6$). Finally, 32 purchase orders,

24 goods receipts, and 10 invoices are affected before *clear invoice* (t_7).

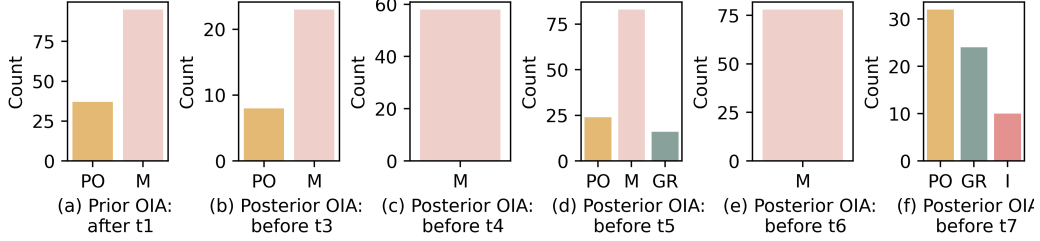


Figure 11: Prior and posterior operational impacts of action instance ai_1

Performance Impact Analysis. More importantly, the action instance has several function-wise and object-wise performance impacts. In terms of function-wise performance impacts, the average sojourn time of *create purchase order* has been reduced by 9 minutes. This is due to the decrease in the queue for the activity, resulting from the new business rule setting the higher minimum quantity for creating purchase orders. In terms of object-wise performance impacts, the total number of purchase orders has been reduced by 13 during the execution of the action instance. Moreover, the average throughput time of the purchase orders has been improved by 4 minutes.

6.3. Real-life Case Study

We present a case study where a PAIS is simulated using real-life event data from a loan application process of a Dutch financial institute [45]. First, we construct the DT-IM that represents the process from the event data. Figure 12 shows the object-centric Petri net of the DT-IM. Initially, a customer creates an application by visiting the bank or using an online system. While

the application is completed and accepted, the bank offers loans to the customer by sending offers to the customer and making a call. The offer is either accepted or refused. In case no further offers are created, the application is canceled. Note that we only partially constructed the DT-IM with a focus on the control flow, as we had limited information to derive the constraint and operation rules of the process.

6.3.1. Action on Canceling Loan Applications

This scenario deals with an action that changes the business rules associated with activity *Cancel Application (CA)* of the loan application process, e.g., modifying the conditions for automatic cancellation due to inactivity, updating the eligibility criteria for loan cancellation based on credit score, and altering the documentation requirements for justifying a cancellation request. The action is applied in five different periods between 01. Oct. 2016 and 09. Dec. 2016. Below are the corresponding action instances:

- *A-CA-1* refers to the execution of the action starting on 01. Oct. 2016 and ending on 14. Oct. 2016.
- *A-CA-2* refers to the execution of the action starting on 15. Oct. 2016 and ending on 28. Oct. 2016.
- *A-CA-3* refers to the execution of the action starting on 29. Oct. 2016 and ending on 11. Nov. 2016.
- *A-CA-4* refers to the execution of the action starting on 12. Nov. 2016 and ending on 25. Nov. 2016.

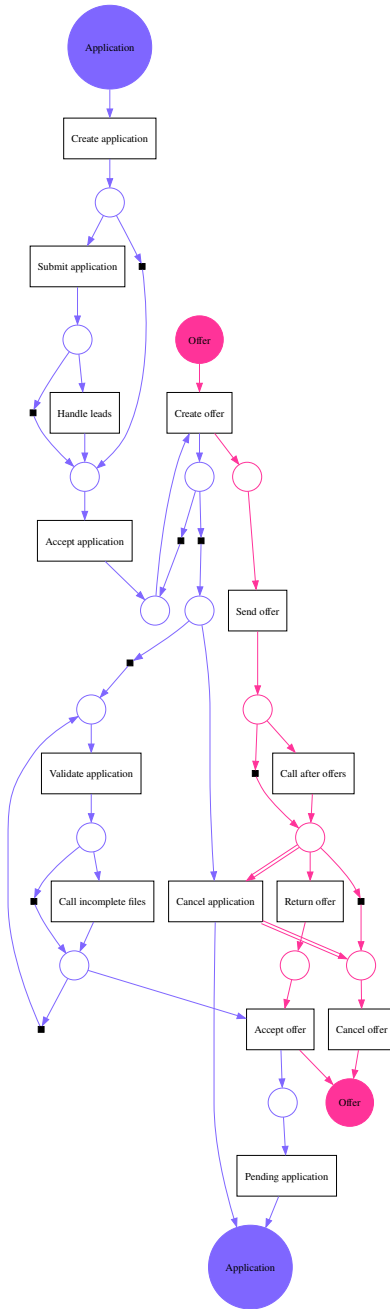


Figure 12: The object-centric Petri net representing the loan application process

- *A-CA-5* refers to the execution of the action starting on 26. Nov. 2016 and ending on 9. Dec. 2016.

Structural Impact Analysis All the action instances, i.e., *A-CA-1*, *A-CA-2*, ..., have the same structural impact since they change the business rules of the same activity, i.e., *cancel application*. For instance, the prior structural impact in terms of *application* refers to five activities in the process, including *create offer*, *accept application*, *handle leads*, *submit application*, and *create application*. Moreover, the prior structural impact w.r.t. *offer* refers to three activities, including *call after offers*, *send offer*, and *create offer*. Since *cancel application* is the end of the process, there are no posterior impacts.

Operational Impact Analysis Figure 13 shows the prior operational impact of the action instances on ongoing applications. First, *A-CA-1* affects around 547 applications that were created and can be canceled. A slightly less number of applications, i.e., 534 applications, are affected by *A-CA-2*. In the subsequent action instances starting on 29-Oct., 12-Nov., and 26-Nov., the impact on the created applications decreases, i.e., 529, 489, and 410 applications are affected, respectively. The same trend is observed for the applications in other stages, i.e., for the applications after *submit application*, the applications after *handle leads*, the application after *create offer*, the applications after *complete*, the applications after *accept application*, and the applications after *call*.

There is almost no difference between the number of applications that are affected after *create application* and the number of applications that are affected after *submit application*. For instance, for *A-CA-1*, the number of

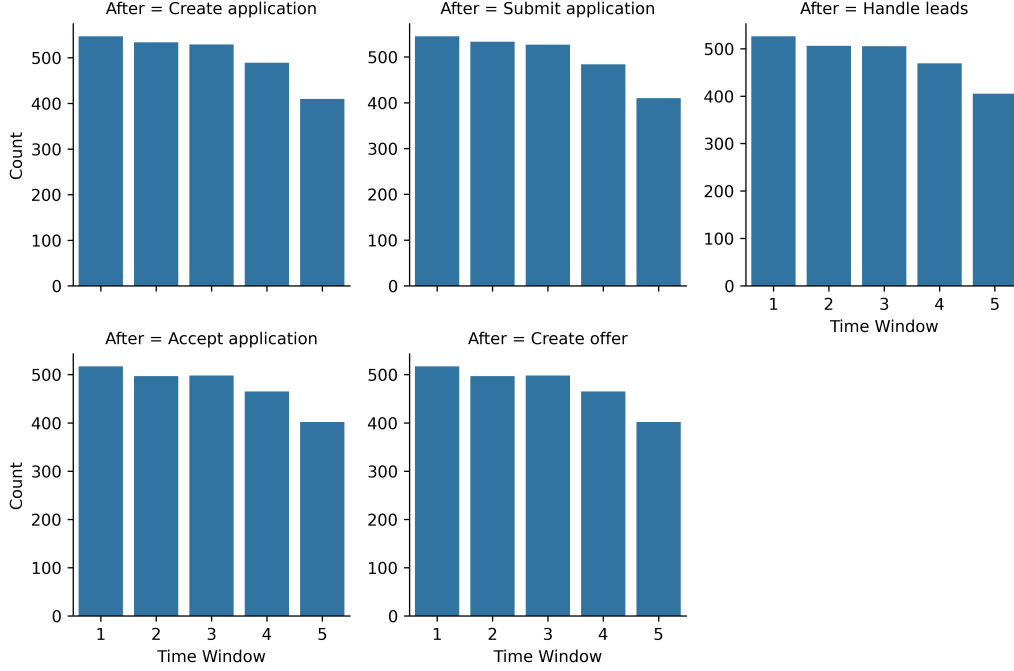


Figure 13: Prior operational impact of $A-CA-1$, $A-CA-2$, $A-CA-3$, $A-CA-4$, and $A-CA-5$ on ongoing applications

applications affected after *create application* is 547, while the number of applications affected after *submit application* is 545. This is because applications are submitted shortly after the creation. Moreover, almost no difference exists between the number of the affected applications after *accept application* and the number of the affected applications after *create offer*. For instance, $A-CA-1$ affects 517 accepted applications and 517 offered applications, $A-CA-2$ affects 497 accepted applications and 497 offered applications, etc. This is because offers are made shortly after the acceptance of applications.

Figure 14 shows the prior operational impact of the action instances to ongoing offers. $A-CA-1$ affects 795 offers after *create offer* and $A-CA-2$

affects slightly more offers, i.e., 812 offers, after *create offer*. However, the number of the affected offers decreases for the later action instances due to the lower number of offers in the process. The same trend is observed for the number of the affected offers after *send offer* and the affected offers after *call after offers*. A significantly lower number of offers are affected after *call after offers* activity, compared to the affected offers after *create offer* and the ones after *send offer*. For instance, *A-CA-1* affects 321 offers after *call after offers*, which is 474 lower than the number of the affected offers after *create offer* and 407 lower than the number of the affected offers after *send offer*.

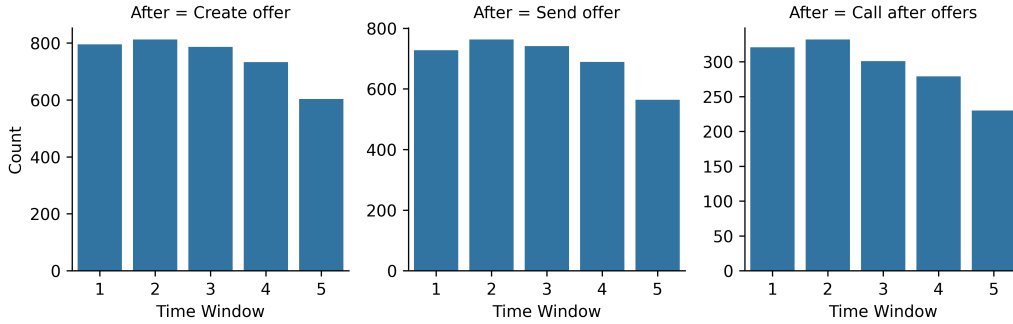


Figure 14: Prior operational impact of *A-CA-1*, *A-CA-2*, *A-CA-3*, *A-CA-4*, and *A-CA-5* on ongoing offers

6.3.2. Action on Calling Loan Applicants

This scenario deals with an action that changes the business rules associated with calling loan applicants (i.e., activity *call after offers*), e.g., introducing a priority system for applicants with higher credit scores and implementing a time-sensitive follow-up protocol for unanswered calls. The action is applied in five different periods between 01. Oct. 2016 and 09. Dec. 2016. Below are the corresponding action instances:

- *A-call-1* refers to the execution of the action starting on 01. Oct. 2016 and ending on 14. Oct. 2016.
- *A-call-2* refers to the execution of the action starting on 15. Oct. 2016 and ending on 28. Oct. 2016.
- *A-call-3* refers to the execution of the action starting on 29. Oct. 2016 and ending on 11. Nov. 2016.
- *A-call-4* refers to the execution of the action starting on 12. Nov. 2016 and ending on 25. Nov. 2016.
- *A-call-5* refers to the execution of the action starting on 26. Nov. 2016 and ending on 9. Dec. 2016.

Structural Impact Analysis. All the action instances, i.e., *A-call-1*, *A-call-2*, ..., have the same structural impact since they changes the same activity, i.e., *call after offers*. The prior structural impact in terms of *application* refers to five activities in the process, including *create offer*, *accept application*, *handle leads*, *submit application*, and *create application*, whereas the one w.r.t. *offer* refers to *send offer* and *create offer*. The posterior structural impact in terms of *application* refers to *cancel application* and *pending application*. The posterior structural impact in terms of *offer* refers to *accept offer*, *return offer*, and *cancel application*.

Operational Impact Analysis. Figure 15 shows the prior operational impact of the action instances on ongoing offers. First, *A-call-1* affects 474 offers that were created and might be called. The number of the affected offers slightly increases to 481 for *A-call-2* and to 485 for *A-call-3*. However,

for *A-call-4* and *A-call-5*, the number of the affected offers decreases to 454 and 374, respectively. The prior operational impact to the ongoing offers after *send offer* shows a similar trend. The number of the affected offers for *A-call-1* is 407, while the ones for *A-call-2* and *A-call-3* are 431 and 440, respectively, showing an increasing trend by time. However, the number drops to 410 for *A-call-4* and to 334 for *A-call-5*.

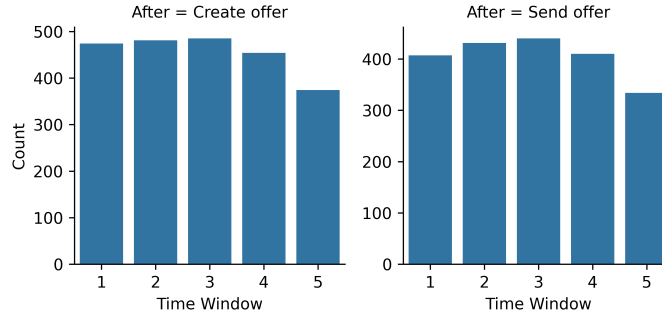


Figure 15: Prior operational impact of *A-call-1*, *A-call-2*, *A-call-3*, *A-call-4*, and *A-call-5* on ongoing offers

Figure 16 shows the posterior operational impact of the action instances to ongoing offers. The number of the affected offers that will (possibly) be returned is equal to the number of the affected offers whose application will (possibly) be canceled. For instance, *A-call-1* affects 321 offers for both *return offer* and *cancel application*. This is because, after creating an offer, we either return the offer for further processing or cancel the corresponding offer, i.e., the equal number of offers is waiting for *return offer* and *cancel application*.

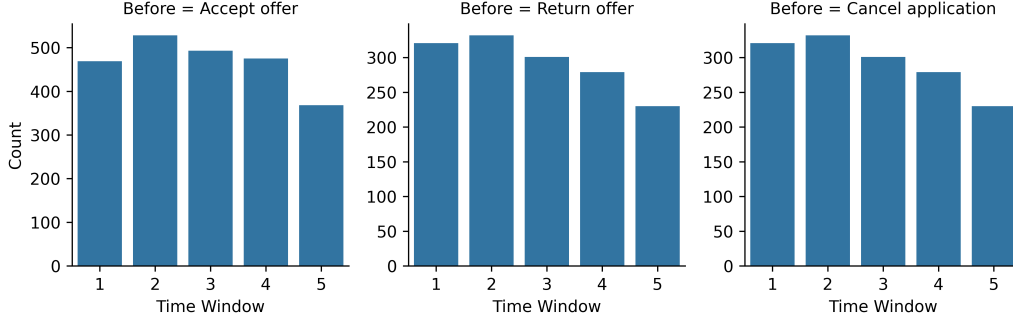


Figure 16: Posterior operational impact of *A-call-1*, *A-call-2*, *A-call-3*, *A-call-4*, and *A-call-5* on ongoing offers

6.4. Threats to Validity

We discuss the threats to validity of the evaluation presented in this section.

Regarding *construct validity*, the simulation-based case study uses artificially generated event data. While the process structure, business objects, and business rules are designed to mirror a real SAP ERP P2P process, the simulated data may not capture all real-world complexities, such as exception handling, ad-hoc process deviations, or the full variability of human behavior. Consequently, the quantitative results (e.g., the specific number of affected objects or the magnitude of performance changes) should be interpreted as illustrative rather than representative of any specific real-world deployment.

Regarding *internal validity*, in the real-life loan application case study, the DT-IM was constructed considering the control flow only, without explicitly specifying constraint and operation rules. This limits the demonstration of the full approach, particularly regarding the change detection step, which relies on rule specifications. Moreover, the accuracy of the operational impact

analysis depends on the alignment between the DT-IM and the event log; misalignment may introduce noise into the results, as discussed further in Section 7.

Regarding *external validity*, two case studies are presented: one simulated and one based on real-life data. While these cover different process types (procurement and loan application) and data characteristics (synthetic and real), further case studies in other domains (e.g., healthcare, manufacturing) are needed to assess the generalizability of the approach. Moreover, a full deployment of the prototype within an operational PAIS change management workflow, where impact analysis results directly inform update decisions, constitutes a natural next step for evaluating the practical utility of the approach.

Finally, regarding *reliability*, the prototype implementation and its source code are publicly available, enabling independent replication of the reported results.

7. Discussion

This section presents a discussion of the insights from the real-life case study, limitations, and future work.

7.1. Insights from Real-life Case Study

The impact analysis for the real-life case study on the loan application process offers various insights that can be used in managing the changes caused by the actions. First, the structural impact analysis for the action on canceling loan applications helps analysts anticipate which upstream activities from the prior structural impact, e.g., *create offer*, *accept application*, and

handle leads, they need to proactively manage to avoid potential disruptions caused by changes to the cancellation criteria. For instance, if stricter cancellation rules are implemented, analysts can foresee that fewer applications will progress to these later steps and allocate resources more efficiently. Staff who manage *cancel application* can be reassigned to handle earlier steps such as *create offer* or *handle leads*. Additionally, this foresight enables analysts to inform customers about the new requirements in the earlier stage.

Moreover, the operational impact analysis builds on the insights from the structural impact analysis by providing precise data on the number of applications or offers affected by the changes. For example, with detailed information about how many applications are at risk in each step (e.g., 517 applications at the *create offer* step), analysts can better plan for fluctuations in workload across the process. Since a large portion of the affected applications (517 out of 547 in the first time window) is concentrated after the *create offer* step, resources reserved in anticipation of the disruptions caused by the cancellation policy changes can be allocated to this step to minimize the impact. Once the initial surge of cancellations has been managed in the *create offer* step, resources can then be reallocated to upstream activities such as *accept application* or *handle leads*.

Finally, the operational impact analysis offers valuable insights into customer satisfaction by revealing how many customers are affected by the policy changes. A significant difference exists between impacting 10 applications versus 500 applications. When hundreds of applications are affected, the likelihood of customer dissatisfaction increases, and analysts can use this data to gauge the severity of the potential backlash. By understanding the scale of

the impact, analysts can also determine the most suitable time to implement the changes, perhaps choosing a period where customer interactions are less intense.

7.2. Limitations

The proposed approach has several limitations. From the outset, we chose to devise a domain-agnostic approach, which abstracts from domain-specific aspects, such as the semantics of the execution of activities. While this has its merits in terms of generalizability and applicability of the proposed approach, it also limits the scope of the analysis. Extending the approach considering the semantics of activities, or at least a more fine-grained characterization of the rules, would afford the opportunity to tackle scenarios adhering more closely to the nuances of process execution and system updates in the real world. This, however, is technically not trivial and may only be achieved by considering specific real-world case studies.

Relatedly, the PAIS meta-model introduced in Section 3 deliberately captures a minimum set of entities. As discussed therein, aspects such as the organizational model (roles, access rights), fine-grained data dependencies, cross-process interactions, and the presentation layer are not covered. This limits the completeness of the impact analysis: updates affecting these aspects cannot be modeled or assessed with the current approach. Extending the meta-model to cover additional PAIS aspects is a direction for future work.

Moreover, the proposed approach assumes that a PAIS update is applied to all process instances, including those currently running. This is reflected in the operational impact analysis, which identifies all in-flight object instances

that are potentially affected by the update. This assumption differs from the versioning mechanisms found in activity-centric PAIS tools such as Activiti, Flowable, or Camunda, where process versioning ensures that structural changes apply only to new instances, while running instances continue under the previous version. Our approach is designed to assess the worst-case scenario in which the update affects all instances, thus providing a comprehensive view of the operational impact. Extending the approach to account for versioning strategies, where the impact is limited to new instances only, is a possible direction for future work.

More in detail, the proposed structural impact analyses rely on the partial order of the transitions of object-centric Petri nets without considering the semantics of the nets. Moreover, the proposed structural analyses cannot distinguish the impacted transitions that are in long-term dependencies and non-free choices since all the connected transitions are considered to be equally impacted.

The presented prior and posterior operational impact analyses build upon the suggested structural impact analysis, inheriting the limitation of the structural impact analysis. For instance, the prior operational impact analysis analyzes all the objects that potentially will perform the updated transitions. However, some objects may bypass the transition, e.g., a patient expected to perform surgery may die before it. Moreover, the posterior operational impact analysis analyzes all the objects that potentially performed the changed transitions, although some objects might have skipped the transition in reality.

Moreover, the accuracy of the operational impact analysis depends on

the alignment between the digital twin interface model and the event log. When the event log contains process variants or exceptions that are not captured in the OCPN model, e.g., if a material is issued without a prior goods receipt, or if activities are executed in an unexpected order, the token-based replay produces misaligned tokens. This may lead to over- or under-estimation of the number of impacted objects, as the replay cannot correctly place tokens for behavior not represented in the model. While we apply the state-of-the-art technique to address such misalignment, depending on the input event log and the degree of its deviation from the process model, the derived operational states may not accurately represent the actual behavior. In consequence, this introduces noise into the operational impact analysis, potentially leading to less reliable results. Improving the robustness of the operational impact analysis to unseen process behavior is a direction for future work.

Next, the suggested function-wise performance impact analysis simplifies the calculation of the difference in function-wise performance measures by comparing the entire change period with an arbitrary comparison period. However, an action may only have performance-wise impacts on the process after a certain time, e.g., adding more resources will only make a difference after some resources are effectively at work. In such cases, the effective change period should be considered for the comparison. Moreover, the suggested object-wise performance analysis does not isolate the objects subject to action instances to evaluate the object-wise performance impact. Instead, it focuses on analyzing all the existing objects of the impacted type in the process, indirectly evaluating changes in object-wise performance measures.

Finally, the current approach is that the performance impact analysis requires the change to be applied before its impacts can be measured. While this provides accurate data regarding the performance outcomes, it limits the proactive value of the analysis. The key advantage of using digital twins lies in their ability to predict the impact of changes before they are implemented, allowing decision-makers to anticipate potential issues or benefits. In its current form, our performance impact analysis offers limited foresight since it relies on post-implementation data, which may reduce its utility for analysts looking to mitigate risks associated with system updates.

7.3. Future Work

A first line of future work derives from the limited focus of the proposed approach on function updates only. Future work should investigate the interplay between such function updates and changes of the process-level schema, which could be handled using existing state of the art methods, possibly adapted to the object-centric scenario.

We will explore the integration of simulation techniques within the digital twin framework. By simulating the performance of system updates prior to their implementation, we aim to provide an estimation of their impact, thus enhancing the decision-making process. This simulation-based approach will allow analysts to predict how key performance indicators may shift following an update, without needing to rely solely on historical data.

Another direction of future work is to improve the performance impact analysis such that it completely isolates the instances affected by changes to provide more realistic performance impact measures. Finally, we also plan to evaluate the proposed approach's ease of use and usefulness with business

analysts in real-world situations.

8. Conclusions

In this paper, we proposed an approach to impact analysis of PAIS updates based on a DT-IM. PAIS updates are modeled as changes in the configuration of a DT-IM. Then, the PAIS entities impacted by PAIS updates are identified, which allows for measuring the structural/operational/performance impacts of PAIS updates. We have implemented a Web-based prototype of the approach and discussed two case studies, one in which the prototype is coupled to a simulated PAIS resembling a standard Procure-to-Pay process in SAP ERP, and one in which the PAIS is simulated using real-life event data from a loan application process.

References

- [1] M. Dumas, W. M. P. van der Aalst, A. H. T. Hofstede, Process-aware information systems: bridging people and software through process technology, John Wiley & Sons, 2005.
- [2] J. Mendling, B. T. Pentland, J. Recker, Building a complementary agenda for business process management and digital innovation, 2020.
- [3] P. Dadam, M. Reichert, The adept project: a decade of research and development for robust and flexible process support: Challenges and achievements, Computer Science-Research and Development 23 (2009) 81–97.

- [4] P. Soffer, B. Golany, D. Dori, Erp modeling: a comprehensive approach, *Information systems* 28 (2003) 673–690.
- [5] M. Parhizkar, M. Comuzzi, Impact analysis of erp post-implementation modifications: Design, tool support and evaluation, *Computers in Industry* 84 (2017) 25–38.
- [6] M. Comuzzi, J. Vonk, P. Grefen, Measures and mechanisms for process monitoring in evolving business networks, *Data & knowledge engineering* 71 (2012) 1–28.
- [7] G. Park, M. Comuzzi, W. M. P. van der Aalst, Analyzing process-aware information system updates using digital twins of organizations, in: R. S. S. Guizzardi, J. Ralyté, X. Franch (Eds.), *Research Challenges in Information Science - 16th International Conference, RCIS 2022, Barcelona, Spain, May 17-20, 2022, Proceedings*, volume 446 of *Lecture Notes in Business Information Processing*, Springer, 2022, pp. 159–176.
- [8] L. Reinkemeyer, *Business View: Towards a Digital Enabled Organization*, Springer International Publishing, Cham, 2020, pp. 197–206.
- [9] W. M. P. van der Aalst, A. Berti, Discovering object-centric Petri nets, *Fundam. Informaticae* 175 (2020) 1–40.
- [10] G. Park, W. M. P. van der Aalst, Realizing a digital twin of an organization using action-oriented process mining, in: *ICPM 2021*, 2021, pp. 104–111.

- [11] H. Österle, J. Becker, U. Frank, T. Hess, D. Karagiannis, H. Krcmar, P. Loos, P. Mertens, A. Oberweis, E. J. Sinz, Memorandum on design-oriented information systems research, *European journal of information systems* 20 (2011) 7–10.
- [12] M. Weske, *Business process management: Concepts, Languages, Architectures*, Springer, 2007.
- [13] B. Weber, M. Reichert, S. Rinderle-Ma, Change patterns and change support features—enhancing flexibility in process-aware information systems, *Data & knowledge engineering* 66 (2008) 438–466.
- [14] M. Reichert, B. Weber, *Enabling flexibility in process-aware information systems: challenges, methods, technologies*, Springer, 2012.
- [15] M. Reichert, P. Dadam, The adept project: A decade of research and development for robust and flexible process support – challenges and achievements, *Computer Science – Research and Development* (2009).
- [16] M. Reichert, S. Rinderle-Ma, P. Dadam, Flexibility in process-aware information systems, *Transactions on Petri Nets and Other Models of Concurrency II: Special Issue on Concurrency in Process-Aware Information Systems* (2009) 115–135.
- [17] M. Reichert, S. Rinderle-Ma, P. Dadam, Enabling adaptive process-aware information systems with ADEPT2, *Enterprise Information Systems* (2010).
- [18] N. Hasan, S. J. Miah, Y. Bao, M. R. Hoque, Factors affecting post-implementation success of enterprise resource planning systems: a per-

- spective of business process performance, *Enterprise Information Systems* 13 (2019) 1217–1244.
- [19] G. Feldman, H. Shah, C. Chapman, A. Amini, Technological, organizational, and environmental drivers for enterprise systems upgrade, *Industrial Management & Data Systems* 116 (2016) 1636–1655.
 - [20] T. Oseni, S. V. Foster, R. Mahbubur, S. P. Smith, et al., A framework for erp post-implementation amendments: A literature analysis, *Australasian Journal of Information Systems* 21 (2017).
 - [21] F. Casati, S. Ceri, B. Pernici, G. Pozzi, Workflow evolution, *Data & Knowledge Engineering* 24 (1998) 211–238.
 - [22] M. Döhring, H. A. Reijers, S. Smirnov, Configuration vs. adaptation for business process variant maintenance: an empirical study, *Information Systems* 39 (2014) 108–133.
 - [23] N. Assy, N. N. Chan, W. Gaaloul, An automated approach for assisting the design of configurable process models, *IEEE Transactions on Services Computing* 8 (2015) 874–888.
 - [24] M. La Rosa, M. Dumas, A. H. Ter Hofstede, J. Mendling, Configurable multi-perspective business process models, *Information Systems* 36 (2011) 313–340.
 - [25] M. Adams, A. H. Ter Hofstede, D. Edmond, W. M. Van Der Aalst, Worklets: A service-oriented implementation of dynamic flexibility in workflows, in: *On the Move to Meaningful Internet Systems 2006: CoopIS, DOA, GADA, and ODBASE.*, Springer, 2006, pp. 291–308.

- [26] N. R. van Beest, E. Kaldeli, P. Bulanov, J. C. Wortmann, A. Lazovik, Automated runtime repair of business processes, *Information Systems* 39 (2014) 45–79.
- [27] P. Soffer, B. Golany, D. Dori, Aligning an erp system with enterprise requirements: An object-process based approach, *Computers in industry* 56 (2005) 639–662.
- [28] Y.-Y. Lin, Y. Nagai, T.-H. Chiang, H.-K. Chiang, Succerp: The design science based integration of ecs and erp in post-implementation stage, *International Journal of Engineering Business Management* 13 (2021) 18479790211008812.
- [29] M. Comuzzi, M. Parhizkar, A methodology for enterprise systems post-implementation change management, *Industrial Management & Data Systems* (2017).
- [30] C. Pahl, Pattern-based software architecture for service-oriented software systems, *e-Informatica Software Engineering Journal* (2010).
- [31] A. Ahmad, C. Pahl, Customisable transformation-driven evolution for service architectures, in: 2011 15th European Conference on Software Maintenance and Reengineering, IEEE, 2011, pp. 373–376.
- [32] J. O. Kephart, D. M. Chess, The vision of autonomic computing, *IEEE Computer* 36 (2003) 41–50.
- [33] F. Author, S. Author, Adaptiflow: An extensible framework for event-driven autonomy in cloud microservices, in: *Proceedings of the 2025 International Conference on Autonomic Computing*, IEEE, 2025.

- [34] M. Laguna, J. Marklund, Business process modeling, simulation and design, Chapman and Hall/CRC, 2018.
- [35] B. Wurm, M. Becker, B. T. Pentland, K. Lyytinen, B. Weber, T. Grisold, J. Mendling, W. Kremser, Digital twins of organizations: A socio-technical view on challenges and opportunities for future research, *Communications of the Association for Information Systems* 52 (2023) 552–565.
- [36] M. Dumas, Constructing digital twins for accurate and reliable what-if business process analysis., in: *Business Process Management Workshops*, volume 2938, 2021, pp. 23–27.
- [37] F. Fornari, I. Compagnucci, M. C. D. Donato, Y. Bertrand, Digital twins of business processes: A research manifesto, *Internet of Things* (2024).
- [38] I. Compagnucci, B. Re, E. S. Asensio, M. Snoeck, A digital process twin conceptual architecture for what-if process analysis, in: *International Conference on Enterprise Design, Operations, and Computing*, Springer, 2024, pp. 373–388.
- [39] P. Royal, What is a business rule?, in: *Building Modern Business Applications: Reactive Cloud Architecture for Java, Spring, and PostgreSQL*, Springer, 2022, pp. 39–46.
- [40] IBM, What are business rules, <https://www.ibm.com/topics/business-rules>, 2024. Accessed: 2024-09-19.

- [41] D. Hey, K. A. Healy, Defining business rules: what are they, really?, https://www.businessrulesgroup.org/first_paper/BRG-whatBR_3ed.pdf, 2000. Accessed: 2024-09-19.
- [42] B. Hamraz, N. H. Caldwell, P. J. Clarkson, A holistic categorization framework for literature on engineering change management, *Systems Engineering* 16 (2013) 473–505.
- [43] G. Park, J. N. Adams, W. M. P. van der Aalst, Opera: Object-centric performance analysis, in: J. Ralyté, S. Chakravarthy, M. Mohania, M. A. Jeusfeld, K. Karlapalem (Eds.), *Conceptual Modeling - 41st International Conference, ER 2022, Hyderabad, India, October 17-20, 2022, Proceedings*, volume 13607 of *Lecture Notes in Computer Science*, Springer, 2022, pp. 281–292.
- [44] A. Berti, J. Herforth, M. S. Qafari, W. M. van der Aalst, Graph-based feature extraction on object-centric event logs, *International Journal of Data Science and Analytics* 18 (2024) 139–155.
- [45] B. van Dongen, Bpi challenge 2017, 2017. URL: https://data.4tu.nl/articles/_/12696884/1. doi:10.4121/UUID:5F3067DF-F10B-45DA-B98B-86AE4C7A310B.