Computer Standards & Interfaces 97 (2026) 104108

Contents lists available at ScienceDirect

Computer Standards & Interfaces

journal homepage: www.elsevier.com/locate/csi

A requirement-driven method for process mining based on model-driven engineering

Selsabil Ines Bouhidel [a,*,1], Mohammed Mounir Bouhamed [b,1], Gregorio Diaz [c,*,1], Nabil Belala [a,1]

[a] MISC Laboratory, University of Abdelhamid Mehri Constantine 2, Constantine 25016, Algeria
[b] LIAP Laboratory, University of El Oued, P.O. Box 789, El Oued, 39000, Algeria
[c] Instituto de Investigación en Informática, Escuela Superior de Ingeniería Informática, Universidad de Castilla-La Mancha, 02071 Albacete, Spain

ARTICLE INFO

Dataset link: 10.17632/s7rjw8nnvm.2

Keywords:
Process mining
Data science
Meta-modeling
Business process management

ABSTRACT

Process mining analyzes business processes using event logs. Existing tools generate models to facilitate this task and improve the original business process, but the results are often unsatisfactory due to the complexity of the obtained models. Among the challenges faced in this context, we identify the misalignment with specific business requirements, preventing managers from accessing key data and making effective decisions. In this paper, we propose a requirement-driven approach centered on meta-modeling, which can help the development of process mining tools specially tailored to organizational needs. Thus, we introduce a requirement-driven method to address the critical challenge of model misalignment with required information. The method employs Model-Driven Engineering to simplify how process mining results are formulated, analyzed, and interpreted. The proposed method is iterative and involves several steps. First, a service manager defines a specific business question. Second, service managers and developers collaboratively establish a meta-model representing the target data. Third, developers extract relevant data using appropriate analysis techniques and visualize it. Finally, service managers and developers jointly interpret these visualizations to inform strategic decisions. This requirement-driven methodology empowers developers to concentrate on relevant information. Unlike general-purpose frameworks (e.g., ProM, Disco), this method emphasizes specificity, iterative refinement, and close stakeholder collaboration. By reducing cognitive overload through focused modeling and filtering of irrelevant data, organizations adopting this approach can achieve faster response times to business questions and develop specialized in-house analytical tools. This requirement-driven methodology, therefore, improves decision-making capabilities within process mining and across related analytical domains. We illustrate our methodology through a real business process taken from the literature owned by the VOLVO group. We use several examples of process mining to illustrate the benefits of the proposed methodology compared to existing tools which are unable to provide the required information.

1. Introduction

Process mining has emerged as a pivotal discipline within data science and business process management, enabling organizations to discover, monitor, and enhance operational processes by analyzing event logs [1]. In this context, an event log is a record of the execution of activities within a Business Process (BP). The importance of process mining is underscored by two key points: the size of the business market and the necessity to adapt BPs periodically to meet evolving business requirements [2]. Projected market growth further highlights the global relevance of effective business processes. For example, the Business Process Management (BPM) market was valued at USD 16.73 billion in 2025 and is expected to reach USD 29.26 billion by 2030. This growth, at an 11.83% Compound Annual Growth Rate, is driven by advances in artificial intelligence integration, cloud computing, and robotic process automation [3]. Similarly, the Business Process Outsourcing (BPO) market reached USD 302.62 billion in 2024 and is expected to attain USD 525.23 billion by 2030, representing a CAGR of 9.8% over the same period [4].

Process models are central to achieving these business improvements. Often based on Petri nets [5] or Business Process Model and

* Corresponding authors.
E-mail addresses: selsabil.bouhidel@univ-constantine2.dz (S.I. Bouhidel), bouhamed-mohammedmounir@univ-eloued.dz (M.M. Bouhamed), gregorio.diaz@uclm.es (G. Diaz), nabil.belala@univ-constantine2.dz (N. Belala).
[1] Researcher.

https://doi.org/10.1016/j.csi.2025.104108
Received 22 July 2025; Received in revised form 17 October 2025; Accepted 28 November 2025
Available online 4 December 2025
0920-5489/

Notation (BPMN) [6], such models offer critical insight for performance optimization, compliance verification, and strategic planing. As a result, organizations across various sectors are increasingly adopting process mining techniques to improve operational efficiency, ensure regulatory compliance, and drive data-informed decisions.

Despite growing adoption and development of sophisticated techniques, process mining still faces notable challenges. Chief among them, as regarded by the literature, is the absence of standardized evaluation criteria for process mining algorithms [7]. In addition, current tools often produce complex and unwieldy process models [8], complicating the selection of methods suitable for specific organizational settings.

While powerful, existing process mining tools frequently generate overly detailed models that are difficult to interpret [9]. This issue becomes particularly acute when addressing diverse and evolving business needs. Recent developments—such as precision-guided discovery [10] and improved algorithms for handling noisy or complex logs [11] have made progress. Yet, the issue of overcomplexity persists.

Furthermore, many widely used frameworks (e.g., ProM [12], Disco [13]), are too broad or inflexible to meet the needs of organizations with specific or evolving analytical requirements [1]. These general-purpose frameworks, designed for a wide array of scenarios, may fall short in terms of adaptability and usability to specific operational needs, leading to inefficiencies in both analysis tool development.

This paper presents a novel requirement-driven process mining method to address two critical limitations in current practice of process mining: the generation of excessively complex models and the inflexibility of general-purpose tools for specific organizational requirements. Our approach builds on the principles of Model-Driven Engineering (MDE) [14,15] and shifts the paradigm from adapting pre-built tools towards the construction of lightweight, customized analytical solutions. In this framework, specific business questions and organizational requirements directly guide the development of these solutions. The methodology detailed herein utilizes meta-models to precisely define the necessary data and analytical scope, facilitating the development of focused, in-house tools able to deliver actionable insights and support more effective decision-making.

The main contributions of this paper are:

1. A requirement-driven meta-modeling strategy that systematically integrates organizational objectives and stakeholder expectations into the process mining pipeline. This integration aims to ensure that discovered process models are not only accurate but also contextually meaningful and interpretable.
2. An iterative methodology for the cost-effective development of customized in-house process mining tools. This methodology empowers organizations to build specialized analytical assets, enhancing agility and enabling more precise data extraction and analysis.
3. Several process mining examples, based on a real business process from the VOLVO Group, are provided to illustrate our approach. The dataset and accompanying analysis notebook are available at the following public repository[2]

The presented approach aims to improve the overall utility and effectiveness of process mining applications by emphasizing specificity and iterative refinement. These improvements ultimately foster more informed and impactful strategic decisions within organizations.

Finally, the proposed method deliberately involves developers within each iteration cycle. This design choice enables a transparent cost–benefit evaluation: the development effort represents a measurable investment comparable to commercial tool licensing. As such, organizations can balance flexibility and sustainability when selecting between generic and requirement-specific analytical solutions.

The paper is organized in seven additional sections, where Section 2 provides background information on process mining and meta-modeling. Section 3 reviews related work in the field. Section 4 introduces the running example used throughout this paper. Section 5 details the proposed methodology. Section 6 presents practical examples illustrating the application of the developed method. Section 7 discusses the implications and limitations of the presented approach. Finally, Section 8 concludes the paper with key findings and directions for future research.

2. Background

This section introduces the background to the concepts used in this work. Specifically, it focuses on process mining, the event log structure utilized in process mining, and meta-modeling.

2.1. Process mining

Process Mining (PM) is a discipline that extracts knowledge from event logs generated by information systems to discover, monitor, and improve real-world processes. As established by [1], Process Mining combines methodologies from business process management, business analysis, and data mining. The field comprises three main phases: discovery, conformance checking, and model enhancement [1]. Fig. 1 illustrates these core phases.

Event logs are essential for Process Mining. These logs record sequences of process steps, grouping related events into cases (process instances). Logs typically include timestamps, resource information, and other contextual data (See Fig. 2).

Using event logs, Process Mining techniques model process flow, capture process variants and deviations, and quantify performance metrics such as cycle time, throughput, and quality [1]. Process Mining commonly uses models such as Petri nets, BPMN diagrams, and Directly-Follow Graphs (DFG) [1].

The key operations of these three techniques are described as follows [1]:

- **Process Discovery (PD)**: extracts models from event logs by analyzing event sequences, timestamps, and instances. These models are derived directly from data, without any predefined descriptions. They reconstruct actual process behavior and reveal deviations from prescribed designs.
- **Conformance Checking (CC)**: is a core technique in process mining. It compares event logs against a process model to detect deviations, such as missing, additional, or incorrectly ordered activities.
- **Model enhancement**: is the process of enhancing as business process based on what is discovered in process discovery and conformance checking. It repair incorrect models and address performance bottlenecks.

The discovery of business process models using existing process mining tools typically produces large and complex artifacts. While these models contain the data to answer a service manager's operational questions, they often fail to make this information accessible in practice. This fundamental disconnect reveals that standard tools are not inherently requirement-oriented.

2.2. Meta-modeling in model-driven engineering

MDE uses models to design and develop complex systems. MDE provides specialized tools and languages to automate many development steps, replacing traditional manual methods. This automation and abstraction enable developers to work at a higher level, significantly improving productivity, consistency, and maintainability [16].

[2] Bouhidel, Selsabil Ines; Bouhamed, Mohammed Mounir; Diaz, Gregorio; Belala, Nabil (2025), *"A requirement driven process mining Dataset"*, **Mendeley Data**, V2, doi: 10.17632/s7rjw8nnvm.2.

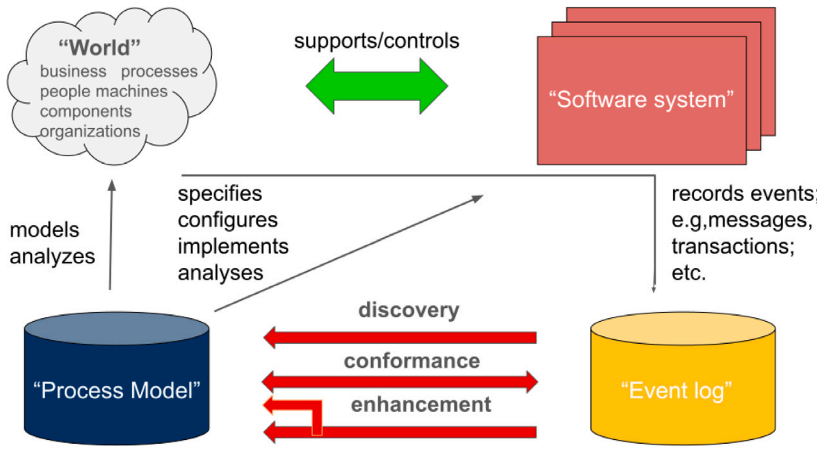

Fig. 1. The relation between the types of PM [1].

Case id	Event id	Properties				
		Timestamp	Activity	Resource	Cost	...
6	35654871	06-01-2011:15.02	register request	Mike	50	...
	35654873	06-01-2011:16.06	examine casually	Ellen	400	...
	35654874	07-01-2011:16.22	check ticket	Mike	100	...
	35654875	07-01-2011:16.52	decide	Sara	200	...
	35654877	16-01-2011:11.47	pay compensation	Mike	200	...

Fig. 2. Representative Structure of an Event Log [1].

Models are central to MDE. They simplify a system by capturing essential aspects for analysis and prediction. Crucially, models serve as the primary means of communication among stakeholders [14]. Models abstract systems by removing irrelevant details, managing complexity through decomposition, and predicting system behavior.

Meta-models structure models, defining their concepts, relationships, and constraints. Every model must adhere to its meta-model. While a model represents a system, its meta-model represents the language or framework used to build it. This hierarchy can extend to meta-meta-models, which define rules for meta-models. Meta-modeling ensures consistency and provides a formal basis for model construction [14].

MDE leverages models as central artifacts, meta-models to provide structure and formal basis, and precisely defined modeling languages to enable automation. MDE supports an iterative, platform-independent development process, enhancing productivity and simplifying complex system management.

Meta-modeling could guide developers in creating solutions that can effectively extract required information on demand. All changes are added to the background section.

3. Related works

Process mining significantly advances process discovery, conformance checking, and enhancement. Researchers have explored various approaches to identify process models from event logs. However, models are often large, complex, and difficult to interpret. They typically reflect a broad process view, which results in a lack of accuracy, specificity, and clarity.

This section systematically evaluates key studies to identify their respective contributions and limitations, which are summarized in Table 1. The evaluation adopts a requirement-oriented perspective, applying several systematic criteria: the extracted information, the tool used, is the approach consider new question based on business needs, model comprehensibility, scalability, and decision-making support. This framework facilitates a structured comparison that positions the proposed method against existing literature, moving beyond a purely descriptive review. While a full numerical benchmark is beyond the scope of this study, this qualitative analysis serves to highlight the specific novelty of the proposed approach.

Process discovery and conformance checking extensively use Petri nets. The ProM framework [12] demonstrates Petri nets' effectiveness in capturing concurrency and synchronization. Van der Aalst et al. [5,24] have shown that Petri nets offer formal advantages for workflow verification. Petri nets serve as a design language for complex workflows and provide powerful analysis techniques to verify workflow correctness. However, applying Petri nets to large-scale or dynamic event logs can generate overly complex models that are difficult to interpret and manage. Event-centric approaches such as the Event Relationship Graph proposed by [22] also address this issue by automating digital-twin generation, although applicability beyond specific manufacturing scenarios remains limited.

Several recent methods partially mitigate this complexity issue. For example, the enhanced eST-Miner [10]. The Alpha+++ algorithm [11] focus on avoiding implicit places, improving precision, and supporting the structured discovery of concurrent processes. The Alpha+++ algorithm extends the original Alpha algorithm by addressing noise, loops, and skips, and improving the discovery of control-flow structures. Consequently, the resulting models are often less complex; however, they may still overlook important aspects related to requirements. Another approach, [23] leverages statistical techniques, CUSUM and EWMA charts, to perform distribution-free anomaly detection, yet their practical implementation primarily relies on controlled simulation environments.

Similarly, hierarchical approaches use abstraction to discover multi-level models from activity trees, such as FlexHMiner [17]. These approaches effectively manage subprocess interleaving and enhance scalability. Nevertheless, aligning these models with detailed stakeholder requirements remains challenging, despite improving interpretability.

Table 1
PM approaches.

Reference	Modeling tool	Extracted model information	Tool used	Considering new questions	Comprehensibility	Scalability	Decision-making support
Augusto et al. [8]	BPMN	Control-flow patterns with balanced fitness and precision	Split Miner	no	Moderately comprehensible	Scalable for moderate data volumes, but limited for large logs	Basic operational insights only
Leno et al. [9]	Declarative Models	Correlated data conditions and refined rules	Custom Framework	no	Lower, declarative models are harder to interpret	Limited scalability depending on complexity	Weak, rule-based decision support
Lu et al. [17]	Activity Trees	Hierarchical process modeling for multi-level subprocesses	ProM	no	Moderate, hierarchy adds complexity	Scales with hierarchy depth	Provides structural insights only
Augusto et al. [18]	BPMN	Concurrency and inclusive decision patterns	Split Miner	no	Good, BPMN is readable	Scalable for moderate data volumes, but limited for large logs	Surface-level decision support
Tiftik et al. [19]	BPMN	Control-flow, data, and resource perspectives	ProM	no	Good comprehensibility via holistic view	Scalable with tool constraints	Decent support, cross-perspective
Aalst [20]	BPMN, Petri Nets, Process Trees	Process improvement and compliance	ProM	no	Moderate, learning curve with multiple formalisms	Scalable due to flexible tools	Moderate, compliance aids decisions
Groß et al. [10]	Petri Nets	Implicit places and structural accuracy aspects	eST-Miner	no	Lower; Petri Nets are less intuitive	Scalable in static settings only	Weak decision support
Kusters and van der Aalst [11]	Petri Nets	Concurrency, loops, noise filtering, and structured control-flow modeling for real-life process discovery	ProM	no	Lower, handles complexity	High scalability for noisy real data	Some support via noise filtering
Saraeian and Shirazi [21]	Uncertain BPMS	Event-based anomaly detector (pre-processor, conformance checker, optimizer with IPSO, Firefly, AdaBoost)	A BPMS extension	no	Low, uncertainty hard to interpret	Moderate scalability, tool-dependent	Moderate anomaly-driven decisions
Castiglione [22]	Event Relationship Graph	Automated generation of manufacturing digital-twin models via 3EM discovery	An event-centric miner	no	High for domain experts	Highly scalable event data processing	Strong automated decision paths
Mukherjee et al. [23]	CUSUM/EWMA control charts	Distribution-free anomaly detection via WRS and HFR statistics	Custom simulation experiments	no	Moderate	Scalable for process control scenarios	Good in anomaly contexts

Kalenkova et al. [6] showed that BPMN is recognized for its readability and alignment with business practices. BPMN supports creating conventional, understandable process models. In addition to its flat control-flow perspective, BPMN integrates subprocesses, data flows, and resources. This integration facilitates stakeholder communication within a single diagram. These characteristics make BPMN attractive for process miners and business users. Recent advancements enhance BPMN discovery by identifying true concurrency and inclusive choices, resulting in accurate, flexible models, such as Split Miner 2.0 [18]. However, BPMN models often struggle with adapting to evolving requirements, tending to be rigid and potentially not fully capturing dynamic business needs.

Data-driven approaches constitute another category. For instance, the Alpha Miner algorithm analyzes event log patterns to reconstruct process models [24]. The Alpha Miner predicts the amount of data needed for mining based on observed substructures and algorithm behavior. This prediction allows efficient data utilization and quantifies confidence [7]. Additionally, Leno et al. [9] introduced methods to enhance declarative discovery by incorporating correlated data conditions. These methods utilize clustering, rule mining, and re-description mining to improve rule precision. Further, an event-based anomaly detection method [21]integrate a extension of BPMS and utilize advanced optimization techniques (IPSO, Firefly) and ensemble learning (AdaBoost) but might add complexity to process mining systems.

Behavior-driven approaches discover deviations from usual behaviors [25]. For instance, the ILP Miner [26] uses Integer Linear Programming to enhance precision. Similarly, Split Miner [8] balances fitness, accuracy, and simplicity, producing behaviorally accurate, straightforward models while avoiding spaghetti-like structures.

Multi-perspective process mining integrates control-flow, data, and resource perspectives into a unified BPMN model. This integration enhances complex process representation, as exemplified by frameworks such as the one proposed in [19]. Additionally, the Process Mining Handbook [20] provides an overview of techniques including discovery, conformance checking, and enhancement. These techniques are essential for understanding and improving operational processes.

While existing techniques advance process discovery, they often produce models lacking contextual significance. Furthermore, aligning these models with stakeholder requirements and organizational goals demands additional mechanisms. Data-driven and behavior-driven systems, in particular, prioritize data patterns over explicit requirements or constraints.

The requirement-based meta-modeling approach addresses these limitations and effectively tackles the complexity challenge. By explicitly integrating stakeholder requirements, this approach ensures models remain actionable, intelligible, and well-aligned with real-world processes. This approach also reduces model size and yields representations that are more adaptable, precise, and easier to maintain.

The literature provides established criteria for quantitatively evaluating the complexity and quality of discovered process models. These indicators quantify the trade-off between model simplicity and behavioral accuracy. For instance, **structural metrics** assess properties such as the number of places, transitions, and arcs in a Petri net, alongside formal properties like soundness, to verify workflow correctness [11, 27]. Other structural approaches, such as cyclomatic complexity, quantify the control-flow to identify potential modeling challenges and error risks [21]. In parallel, **behavioral metrics** rely on conformance checking techniques to compare a model against an event log [24]. These techniques compute fitness, which measures how well a model reproduces observed behavior, and precision, which measures the extent to which it permits unobserved behavior. Recent work combines these structural and behavioral indicators to guide model simplification while preserving interpretability and alignment with stakeholder requirements.

Despite the availability of these metrics, scalability remains a critical challenge. Larger, more intricate process models inherently yield higher complexity scores, which hinder their practical usability without effective abstraction or hierarchical decomposition techniques. This reality establishes a dual purpose for these metrics in process mining: they not only serve as tools for evaluating final model quality but also act as essential guides for the iterative improvement and simplification of process models.

ProM [12] and Disco [13] are widely used tools for implementing process mining techniques. ProM is a versatile open-source platform supporting various plugins. However, ProM requires improved handling of large event logs and enhanced scalability [27]. Disco, presented by Fluxicon, offers an intuitive interface and powerful visualization. Despite these strengths, both tools often require customization to address specific challenges.

Custom tools attempt to address scalability but may still sacrifice usability, such as those introduced by Marlon Dumas et al. [28].

In summary, while approaches such as Petri Nets, BPMN, data-driven, and behavior-driven methods contribute to process mining, they face challenges. These challenges include complexity, rigidity, and issues related to contextual relevance and alignment with business requirements.

4. Running example

This section uses the VOLVO[3] Incident maNagement SysTem (VINST) as a running example to illustrate the process mining approach developed in this study. The VINST dataset, documented by [29], highlights the complexities inherent in real-world incident management systems. It features a rich event structure and diverse incident types, making it an ideal case for evaluating the proposed method. VOLVO IT service, aka VOLVO IT, utilizes VINST [30] to manage incidents affecting VOLVO product lines, including equipment breakdowns, quality control issues, supply chain disruptions, accidents, and regulatory changes. These incidents significantly impact production efficiency and operations. For instance, breakdowns halt production, quality issues require rework, and supply disruptions cause delays. The VINST dataset includes event logs recorded from March 2010 to May 2012 [29].

VOLVO IT established an incident management system [29] to address these issues. This system defines roles and responsibilities for resolving incidents promptly and efficiently. As described by [29], the process begins by determining whether the reported issue qualifies as an incident (see Fig. 3), which can be reported either manually or automatically. If it is not considered an incident, it is handled through other service desk routines. If confirmed as an incident, it is registered and classified by the first-line support, comprising various help desks (e.g., service desk, front desk, offline desk, desk-side support) as well as expert help desks. An attempt is then made to match the incident with an existing solution. If a solution is identified, the incident is resolved and closed. If not, it is further investigated at the same level. If the first line cannot resolve the incident, it is escalated to the second line of support, which conducts its own investigation and resolution attempt. The second-line support typically involves teams within Organization Line C or A2 [29]. If resolution is still not achieved, the incident is escalated to third-line support for deeper analysis. The third-line support usually consists of product experts, who are often also part of Organization Line C or A2, and handle incidents unresolved by the second-line support [29]. At each support level (1st, 2nd, or 3rd line), if the incident cannot be resolved internally and requires external expertise, it is forwarded to the supplier's support. Supplier support investigates the issue and may return it for internal resolution once a solution is identified. A Support Team (ST) is assigned at each step of the process to handle activities such as classification, investigation, matching, resolution, or escalation.

Once a resolution is found at any level, the case owner verifies the outcome before closing the incident. This structured approach aims to ensure efficient incident handling, minimize operational impact, and enhance client satisfaction [29].

[3] The VOLVO Group is a Swedish multinational manufacturing corporation specialized in producing trucks, buses, and construction equipment.

4.1. Original data

The VINST event log dataset, as described by [29], contains 15 distinct attributes. Table 2 presents the first 10 event logs from this dataset. Each incident is identified by a unique serial (SR) number. Multiple workers or owners may address an incident within a support team, and numerous log records track status and ownership changes until resolution [29].

According to [29], the dataset records include details such as serial number (identifying the service request), timestamp, case status, and sub-status, business impact, product, support country, case owner, support team, and organizational line (Table 2). The "owner" attribute identifies the task owner; two successive logs with the same SR but different owners indicate a work transfer. The full dataset comprises 7554 incidents and 65,553 events, averaging 8.7 events per incident [29].

4.2. Processed data

We preprocessed the raw VINST data for analysis. First, we calculated activity durations for each event (e.g., 7 days, 14:16:03) to enable temporal analysis. Next, we included only completed activities to focus on full process instances. We then extracted group names (e.g., "V13") and support lines (1st, 2nd, 3rd) from the "group line" column, providing key attributes for organizational analysis. Finally, we removed unnecessary columns to streamline the dataset. These steps ensured the data was suitable for our subsequent process mining analysis.

5. Proposed solution

This section presents a new requirement-driven method based on a model-driven engineering approach designed to address two key challenges in process mining: selecting appropriate algorithms and managing the variability of resulting models. First, we summarize the proposal, and we then present each step of our approach.

5.1. Overview

This subsection provides an overview of the proposed five-step iterative method for process mining (see Fig. 4). The method guides service managers and developers in mining and analyzing Business Processes (BPs) and their running instances. It also supports them in developing in-house process mining tools. Currently, no standard method exists for assessing and comparing process mining algorithms. Furthermore, existing tools often produce large, highly variable models that complicate their use in statistical analysis for managerial decision-making.

This approach supplements existing tools by establishing a company-specific library. This library helps provide tailored answers to specific business questions.

The proposed method involves five iterative steps:

- The service manager asks a question regarding the BP.
- The developer describes the targeted data as a meta-model.
- The developer extracts the data using data analysis tools.
- The service manager visualizes the results using a visualization library, with the developer's help.
- The service manager interprets the results to answer the question and make decisions.

The method is iterative: if the service manager is unsatisfied with the results, they initiate a new iteration by asking a new question.

Service managers initiate process mining when BP analysis is needed. They first consider existing frameworks, such as ProM and Disco, and any tools already developed in-house. If these tools satisfy their requirements, they use them. Otherwise, they follow the proposed method to develop a new in-house tool, thus expanding their collection.

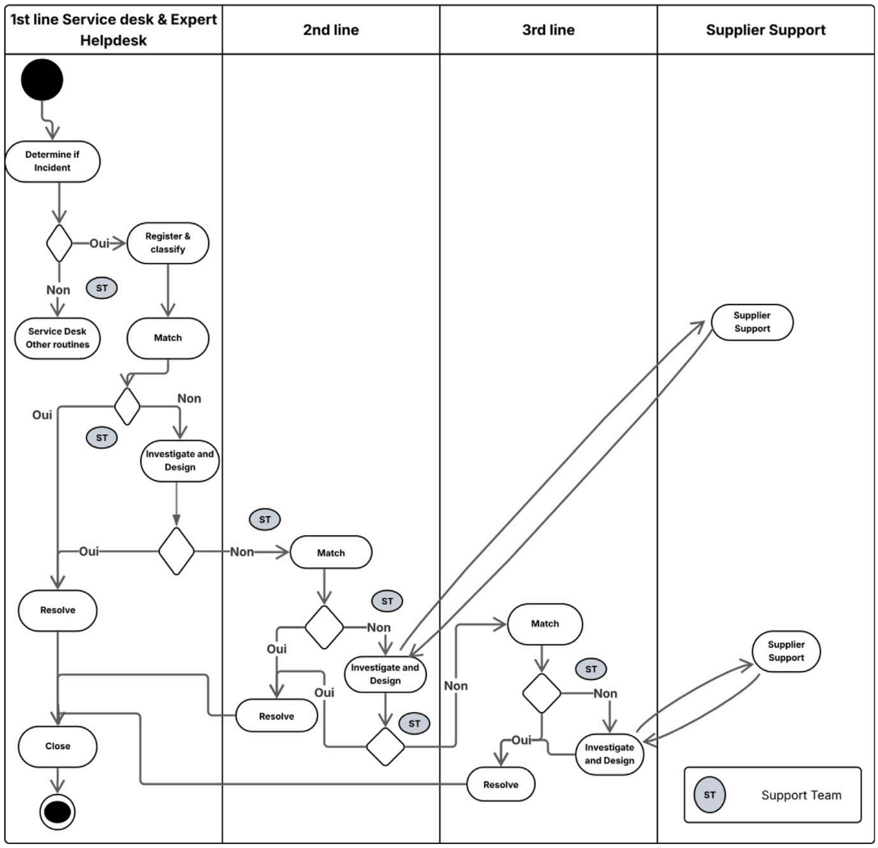

Fig. 3. Support Organization Structure.
Source: Adapted from [29].

Table 2
Table of the first 10 records from the event log.

Case ID	Activity	Resource	Complete timestamp	Variant	Variant index	Concept: name	Impact	Lifecycle: transition	Org: group	Org: role	Org country	Org involve	Product	Resource country
Case 1	Accepted-In Progress	Value 1	2006/01/11 23:49:42	Variant 77	77	Queued	High	Awaiting Assignmenr	Org line A2	A2_2	se	J11 2nd	PROD191	INDIA
Case 1	Accepted-Assigned	Value 1	2012/03/15 19:53:52	Variant 77	77	Accepted	High	In Progress	Org line A2	A2_2	se	J11 2nd	PROD191	INDIA
Case 1	Accepted-Assigned	Value 1	2012/03/15 19:56:17	Variant 77	77	Accepted	High	Assigned	Org line A2	A2_2	se	J11 2nd	PROD191	INDIA
Case 1	Accepted-In Progress	Value 1	2012/03/15 20:09:05	Variant 77	77	Accepted	High	In Progress	Org line A2	A2_2	se	J11 2nd	PROD191	INDIA
Case 1	Completed-Closed	Value 1	2012/03/15 20:11:33	Variant 77	77	Accepted	High	Closed	Org line A2	A2_2	se	J11 2nd	PROD191	INDIA
Case 2	Accepted-In Progress	Value 2	2006/11/07 18:00:36	Variant 78	78	Accepted	Medium	IN Progress	Org line A2	A2_2	cn	M1 2nd	PROD753	Sweden
Case 2	Accepted-In Progress	Value 2	2006/11/07 21:05:44	Variant 78	78	Accepted	Medium	In Progress	Org line A2	A2_2	cn	M1 2nd	PROD753	Sweden
Case 2	Accepted-Wait	Value 2	2009/12/02 22:24:32	Variant 78	78	Accepted	Medium	Wait	Org line A2	A2_2	cn	M1 2nd	PROD753	Sweden
Case 2	Accepted-In Progress	Value 2	2011/09/03 14:09:09	Variant 78	78	Accepted	Medium	In Progress	Org line A2	A2_2	cn	M1 2nd	PROD753	Sweden
Case 2	Accepted-In Progress	Value 3	2012/01/20 18:23:24	Variant 78	78	Accepted	Medium	In Progress	Org line A2	A2_2	cn	M1 2nd	PROD753	China

This need arises because the limited availability of specialized process mining tools often necessitates developing custom solutions. Therefore, the proposed method provides essential guidance for service managers and software developers throughout the process mining lifecycle.

The method's primary benefit is that it enables service managers to answer specific BP questions when no existing tool can help. It also facilitates building an internal library of tools for future analysis of the same or similar questions. This approach encourages developing small, focused tools, which helps reduce development costs compared to adding features to large, general frameworks like ProM and Disco, which target a wide range of processes and questions.

Developers use meta-modeling to specify the data needed for a specific question. This helps them focus on extracting only the required data, avoiding the time and resources needed to gather unnecessary information. It also helps the service manager focus on the relevant data, reducing distractions from irrelevant details.

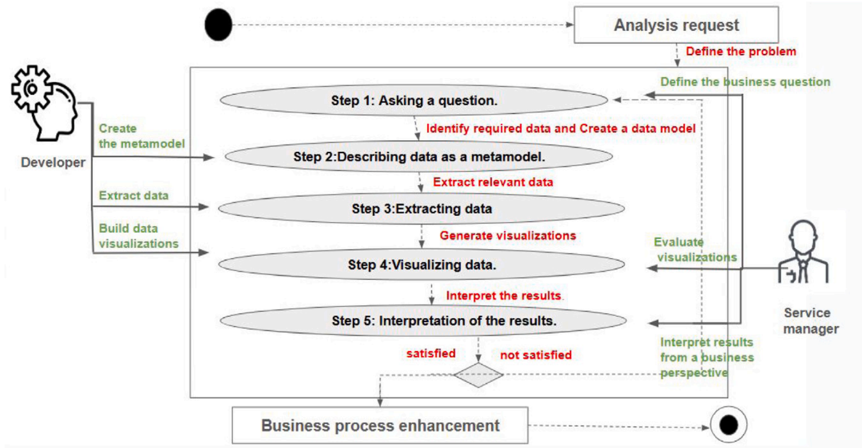

Fig. 4. Process mining steps.

5.2. Step 1: Define the question

In the first step, the service manager investigates the company's BP to identify areas for improvement. They use natural language questioning techniques to gather information and prepare specific questions about the process, facilitating clear communication with all stakeholders. The service manager then analyzes this information to pinpoint specific issues.

Inquiry questions typically include both quantitative and qualitative types. Examples include:

- What is the average, minimum, and maximum time to complete cases?
- What is the average service time for each task?
- How much time is spent between tasks in the process?
- How many people are involved in each case?
- What is the communication structure and interdependencies among people?
- How many transfers occur from one role to another?
- Who are the most important people in the communication flow?
- Who works on the same tasks?

5.3. Step 2: Specify the meta-model

As the second step, the developer creates a meta-model to describe the structure and key features of the target data. This involves selecting data sources, such as event logs, and defining the data structure. A meta-model simplifies complex data into a high-level overview, highlighting its attributes and relationships. This simplification is crucial for understanding the data structure and identifying gaps or inconsistencies, which is essential for accurate data analysis and visualization in the following steps.

5.4. Step 3: Extract the data

After the developer describes the target data as a meta-model, they extract the necessary data from the data source or event log. This involves using data analysis tools to explore and manipulate the data. The choice of tools depends on the data's complexity and the required analysis type, guided by the service manager's questions about the business process. Our implementation uses Python programming and NumPy to handle the data. This extracted data is then used to answer the service manager's questions, as discussed in later sections.

5.5. Step 4: Visualize the results

After extracting the data, the developer uses data visualization tools to create clear visuals. These visuals help the service manager quickly understand the analyzed data. The developer then presents these visualizations to the service manager for analysis and interpretation in Step 5.

5.6. Step 5: Interpret results and make decisions

In this final step, the service manager interprets the visualizations provided by the developer. The service manager then presents these results to the client, highlighting key patterns and trends. Using these visuals, the service manager answers the client's questions and explains the findings clearly, enabling the client to make informed decisions.

If the client is dissatisfied with the results, the service manager notes their feedback and generates a more detailed inquiry. The service manager forwards this inquiry to the developer for additional processing, restarting the method from Step 1. Notably, the developer can often refine the data they are already using rather than requiring new data extraction, making the process quicker and more flexible.

Once the client is content with the results, the developer incorporates the analysis and its resolution into the in-house tool.

6. Examples

This section presents real-world case studies to illustrate our proposed requirement-driven method, which is based on model-driven engineering. These case studies demonstrate the method application for addressing specific business questions and supporting tool development. We chose this descriptive approach over a complete empirical benchmark for three reasons. First, our primary goal is to introduce and demonstrate the method. Second, a full empirical benchmark would require datasets, tools, and resources beyond the scope of this study. Third, because our method is requirement-specific, quantitative comparisons with general-purpose tools would be misleading. We therefore provide transparent and reproducible descriptive analyses and explicitly report their limitations.

6.1. Example 1: Products are escalated to the third line

Step 1: Define the question. The service manager aims to improve the process by minimizing incidents not intercepted at the initial and secondary lines. They hypothesize that certain products are less likely to be intercepted at the first two lines compared to others. Therefore, they ask how often a product is escalated to the third line.

Step 2: Specify the meta-model. The developer specifies a meta-model to structure the data required to address the question. The data should reflect the incidents escalated to the third line. The objective is to identify the products escalated to the third line and their frequency.

Fig. 5 illustrates the specified meta-model. The specified meta-model contains one class and a table for analyzing incidents escalated to the third line. Each entry in this class assesses an individual product and includes the following details:

- Product: A unique identifier, for example, PROD582.
- Third_line_escalated_instances: The number of instances escalated in the third line.
- Total_instances: The overall number of the product instances.
- Third_line_escalated_rate (%): The incident rate for the product escalated to the third line.

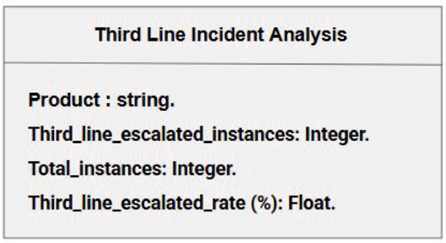

Fig. 5. Meta Model for Question One.

Step 3: Extract the data. The developer creates Python code to collect the data described in the meta-model shown in Fig. 5. The data extracted for the meta-model, Appendix Table 8, illustrates the products in the third line for which the escalate rate to the third line is greater than 10% of the total occurrences (“total instances”).

Table 3
Data extracted for Question One (Selected Products). The complete dataset is provided in Appendix Table 8.

Product	Third line escalated instances count	Total instances	Third line escalated rate (%)
PROD607	69	142	48.59
PROD604	31	71	43.66
PROD617	13	59	22.03
PROD611	14	43	32.56
PROD609	13	33	39.39
PROD194	6	28	21.43
PROD568	12	26	46.15
PROD655	5	20	25.00
PROD370	7	17	41.18
PROD5	7	16	43.75
PROD171	6	16	37.50
PROD582	5	15	33.33
PROD662	5	14	35.71
PROD80	5	10	50.00
PROD89	1	2	50.00
PROD610	1	2	50.00
PROD658	1	2	50.00

Step 4: Visualize the results. The developer used the extracted data from Appendix Table 8 so that the service manager could visualize the results. The visualization, Appendix Fig. 21, illustrates essential trends and patterns related to the meta-model’s first question. The x-axis lists the product names, while the y-axis measures the escalation rate. This representation highlights potential correlations or significant anomalies in the data. This visualization is crucial for improving performance.

Step 5: Interpret results and make decisions. The service manager interprets the results to answer the question (Appendix Fig. 21) and makes decisions.

Based on the analysis of results in Appendix Table 8, which is summarized in Table 3 and Fig. 6, the analysis identifies high escalation rates for some products. These include PROD607 (48.59%), PROD604 (43.66%), PROD617 (22.03%), and PROD611 (32.56%). These products also show the highest occurrence counts (142, 71, 59, and 43, respectively) and notable instances escalated to the third line (69, 31, 13, and 14, respectively). This potentially indicates complexities or challenges in their initial processing.

The data also includes products such as PROD609, PROD194, PROD568, PROD655, PROD370, PROD5, PROD171, PROD582, PROD662, and PROD80. These exhibit high rates (39.39%, 21.43%, 46.15%, 25%, 41.18%, 43.75%, 37.5%, 33.33%, 35.71%, and 50%, respectively). However, their total instances are low (between 10 and 33), and instances escalated to the third line are even lower (between 2 and 13).

Finally, the data presents products such as PROD89, PROD610, and PROD658. These products exhibit high rates of 50% each, but these rates derive from only two instances. Furthermore, only one instance was escalated to the third line.

Products with high rates but low total instances (e.g., fewer than 10) are considered less relevant for decision-making because the small sample size makes the rate unreliable. For example, a product with only two instances showing a 50% rate (one escalated to the third line) does not reliably predict its occurrence rate if it had 10 instances in a subsequent analysis.

Following this analysis, the service manager questioned the exclusion of products with fewer than 10 instances. The reasoning is that the total number of instances for these products is insignificant compared to others. This consequently impacts the reliability of decision-making for these products. For example, a product with only two instances showing a 50% rate does not reliably indicate its occurrence rate if it had 10 instances in a subsequent analysis using a new dataset. This is because it appeared only once on the third line in the original data.

By adhering to the service manager’s recommendations and filtering the data, the developer provided the updated visualization in Fig. 7 for the service manager’s review. The visualization in Fig. 7 illustrates how products are escalated to the third line after filtering. The X-axis lists product names, and the Y-axis represents the escalate rate. The analysis indicates that the products with fewer than 10 instances are smaller than the initial set.

The summary highlights products with high percentages and instance counts. This comprehensive data analysis offers a precise understanding of production challenges. It enables the service manager to make informed decisions and recommendations to their department. For instance, causal analysis of products with high escalate rates could pinpoint specific quality or process issues.

After the service manager validates the results and confirms their correctness and practicality, the developer can archive the analysis in the in-house library. If processing difficulties persist, the service manager should consider optimizing the initial processing lines to manage these complexities effectively. In this case, the service manager should consider initiating inquiries and restarting the method from the beginning, as the second example illustrates.

6.2. Example 2: Team dynamics

Step 1: Define the question. The service manager wants to know whether the organizational group has a role in escalating products to the third line. To address this, they ask: Do the skills or practices of certain organization group lead to more frequent escalations to the third line?

Step 2: Specify the meta-model. The developer specifies a meta-model to structure the data required to address the question. Specifically, the data should reflect:

- The number of instances where organizational groups transfer products to the third line;
- The total number of instances handled by each organizational group;

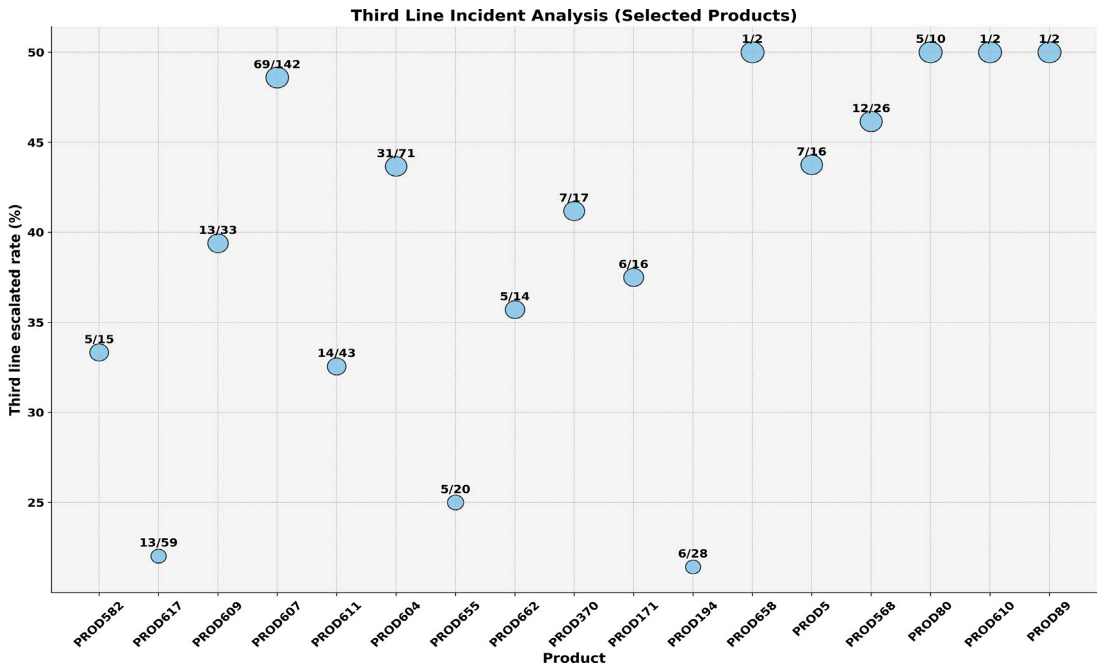

Fig. 6. Visualization of data Extracted for Question One (Selected Products). The complete Visualization is provided in Appendix Fig. 21

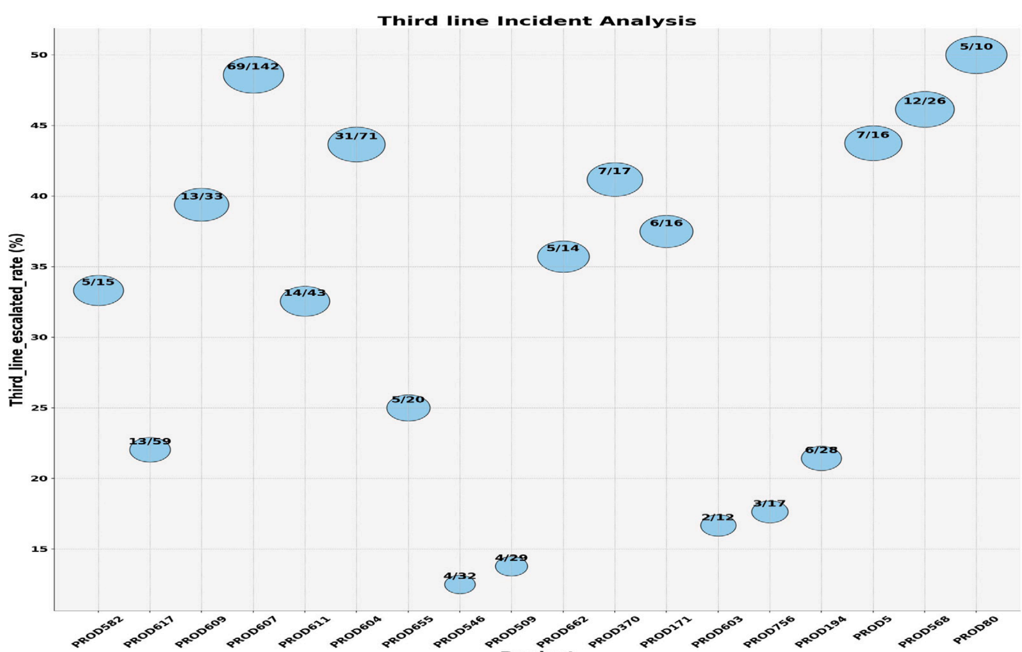

Fig. 7. Visualization of data Extracted with Filtering for Question One.

- The percentage of cases in which products were escalated to the third line for each group.

This analysis helps identify organizational groups that transfer products to the third line and the names of these transferred products. The specified meta-model, illustrated in Fig. 8, consists of one class and one table. These elements analyze the organization groups that escalate products to the third line and their escalation rates relative to the total instances. The meta-model includes the following attributes:

- Organization_group: the organizational group responsible for escalating the product to the third line (e.g., V5 and V13).
- Total_instances: the total number of instances handled by the group.
- Third_line_Organization_group_escalated_instances_cont: the number of times the group escalated a product to the third line;
- Rate (%): the percentage of escalated cases relative to the group's total workload.
- Product: the identifier of the escalated product (e.g., PROD765).

Table 4
Data extracted for meta model: Question Two.

Organization group	Total Instances	Third line Organization group escalated instances cont	Rate (%)	Product
G207	4	1	25.0	PROD765
V13	29	4	13.79	PROD582
N6	12	1	8.33	PROD370
G27	25	2	8.0	PROD611, PROD607
G276	32	1	3.12	PROD422
G51	241	1	0.41	PROD818
G96	1105	1	0.09	PROD611
Organizations with 0% escalate Rate	11721	0	0.0	None

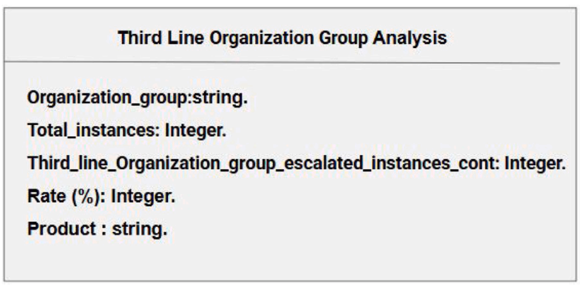

Fig. 8. Meta Model for Question two.

Table 5
Combined results of products and organization groups.

Product	Organization	Instances escalated to the 3rd Line
PROD582	V13	4
PROD607	G27	1
PROD765	G207	1
PROD818	G51	1
PROD611	G96	1
PROD611	G27	1
PROD422	G276	1
PROD370	N6	1
64 Products	Unknown	284

Step 3: Extract the data. The developer creates Python code to retrieve data outlined in the meta-model depicted in Fig. 8 and extracts relevant results (Table 4), which list each organizational group, total instances managed by the group, number of instances escalated to the third-line, its third-line escalate rate, and the associated product names.

Step 4: Visualize the results. The developer used the extracted data from Table 4 so that the service manager could visualize the results (Fig. 9). The visualization shows the results for Question 2 regarding the organization groups. The X-axis shows the organization groups' names, and the Y-axis shows the escalation rate. This visualization highlights the role of these organization groups in escalating products to the third line, which can help improve the escalation process.

Step 5: Interpret results and make decisions. The service manager interprets the results (Fig. 9) to answer the question and make decisions.

According to the analysis of these results (Table 4 and Fig. 9), the service manager observes that a relatively small number of organizational groups (averaging only seven) escalate products to the third line. In contrast, the remaining groups are classified as not escalating the product (see Table 5).

A closer look reveals that organizational groups with a 0% escalation rate to the third line exhibit 11,721 instances. Group G96 succeeded in escalating the product only once (PROD611), despite having 1105 recorded instances. Furthermore, six other organizational groups also have a significant number of instances but a low escalation rate. Specifically, groups G51, G276, N6, and G207 escalated the product once each (PROD818, PROD422, PROD370, and PROD765, respectively) despite their high total instance counts. Additionally, organizational group G27 successfully escalated two products (PROD611 and PROD607), and organizational group V13 escalated product PROD582 four times.

This leads the service manager to hypothesize that these teams may be capable of resolving most issues internally without escalation. While not conclusive, the evidence suggests that some groups may perform more effectively in early-stage resolution.

However, this result seems inconsistent with the findings from the Example 1, which indicated a high number of third-line escalations. This discrepancy prompts the service manager to consider two possible explanations: either the escalations are product-specific (i.e., due to complexity), or the organizational source data is incomplete.

The service manager attempts to obtain responses by generating a novel visualization based on the data from Examples 1 and 2.

The first hypothesis — that product complexity drives escalation — cannot be confirmed, as each organization appears to have escalated different products. The only shared case is PROD611, which was escalated by both G96 and G27, with each group contributing one instance. However, as shown in Appendix Table 8, PROD611 has a total of 14 escalations, meaning these two groups account for only 28% of its cases. Therefore, the organizational data alone does not explain the full pattern.

Further analysis of Fig. 5 and Appendix Table 8 reveals that only 11 third-line escalations can be linked to the seven named organizational groups. In contrast, the remaining 284 escalated instances, or 96.3%, lack a known originating organization (see Fig. 10). As a result, no reliable conclusions can be drawn about whether product complexity or team inefficiency is the driving factor.

Consequently, the service manager must ask the developer to propose a solution to improve the event log. The current system's event log lacks sufficient data for proper analysis due to missing information. Addressing this issue could involve either implementing a new system to capture more complete escalation data in the future or enhancing the existing system to improve the completeness and accuracy of the event logs.

6.3. Example 3: Products with the longest resolution times

Step 1: Define the question. The service manager asks: Which products have the longest resolution times?

Step 2: Specify the meta-model. The developer specifies a meta-model to structure the data required to address the question. The data reflect the average resolution times for all products. This analysis helps determine which products take the most time to resolve.

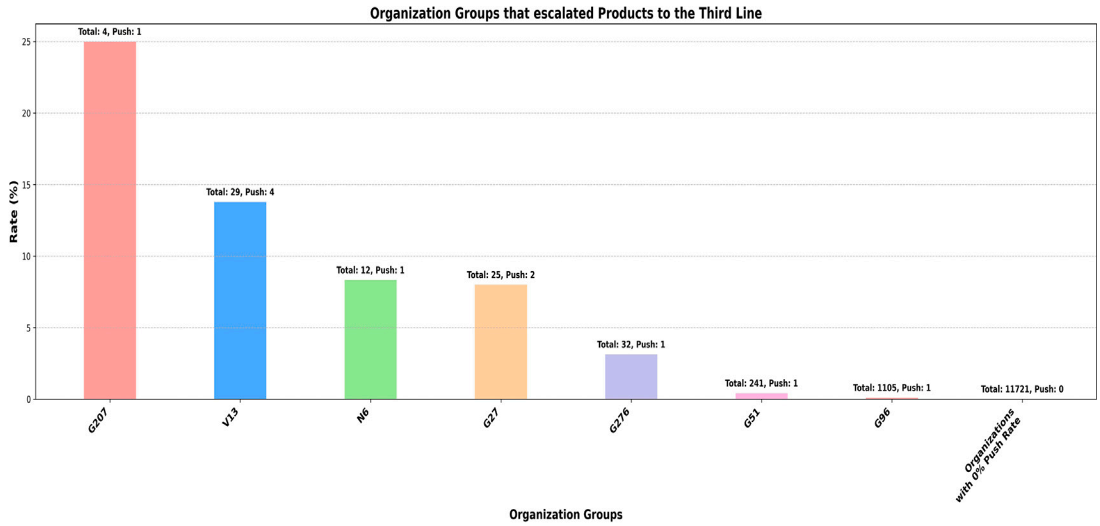

Fig. 9. Visualization of data Extracted for Question Two.

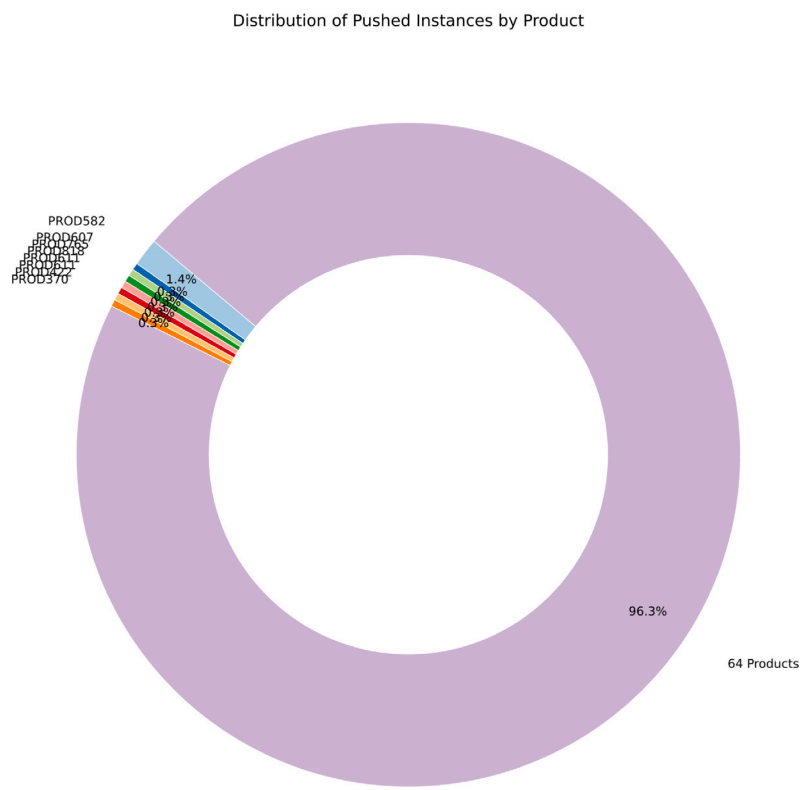

Fig. 10. A doughnut chart showing the distribution of escalated instances across different products.

The specified meta-model, Fig. 11, consists of a single class and a corresponding table. These are used to analyze the average resolution durations and highlight the slowest-resolving products. The meta-model includes the following attributes:

- **Product:** is a unique identifier for each product.
- **total_resolution_time:** the cumulative time spent resolving all incidents related to the product, including days, hours, and minutes (e.g., “7 days 00:00:11” or “59 days 02:48:23”).
- **incident_count:** the total number of resolved incidents per product.
- **avg_resolution_time:** the average time required to resolve a single incident.

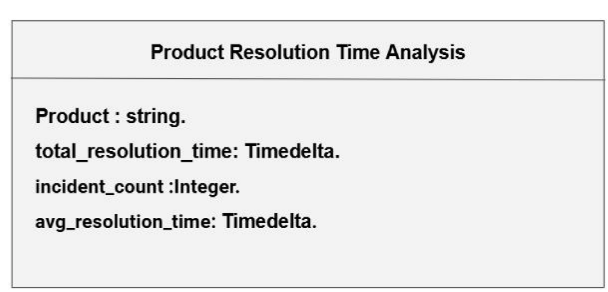

Fig. 11. Meta Model for Question three.

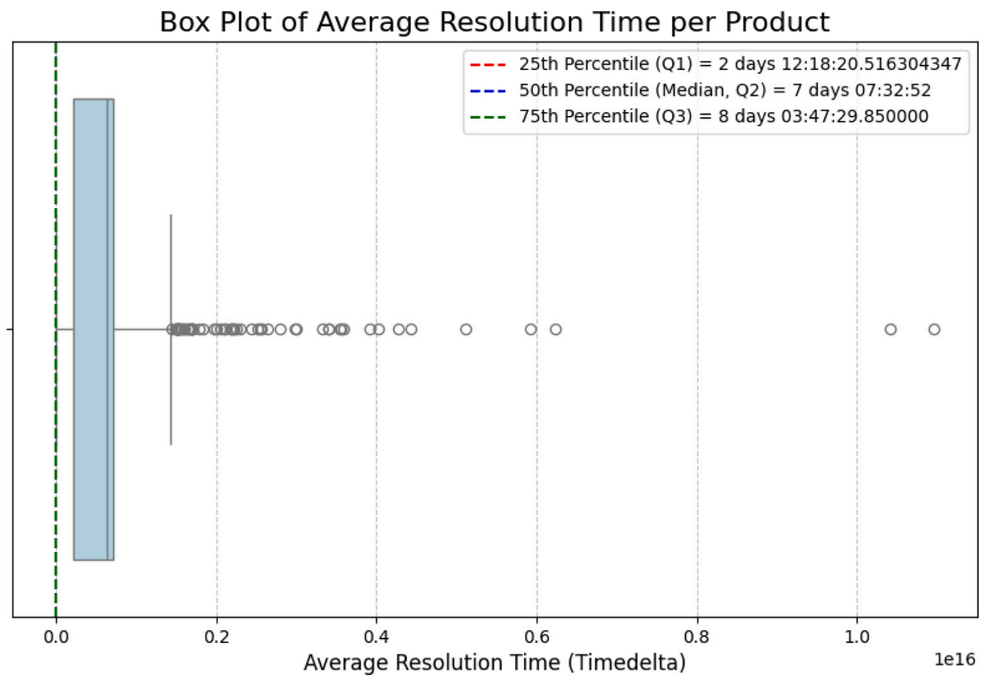

Fig. 12. Box plot for average resolution time for all products. The distribution is negatively skewed, with most products resolved near the lower quartile (Q1: 2 days, 12 h) and a median (Q2: 7 days, 8 h). Products exceeding the upper quartile (Q3: 8 days, 4 h) form a distinct outlier group, highlighting cases requiring management attention to improve resolution efficiency.

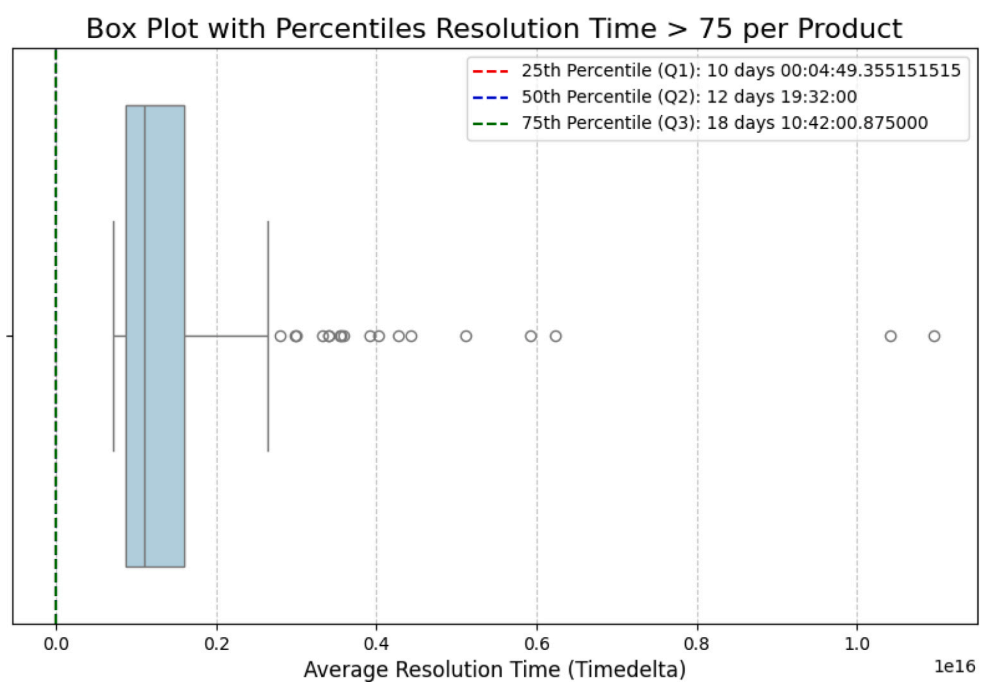

Fig. 13. Box plot: longest average resolution time for products > Q3. The distribution is positively skewed, with Q1 (approximately 10 days) and the median (Q2: approximately 12 days) indicating prolonged resolution, even for the fastest cases. Outliers above Q3 (18 days, 10 h) indicate products that face substantial process bottlenecks or inefficiencies requiring attention.

Step 3: Extract the data. The developer creates Python code to retrieve the data specified in the meta-model shown in Fig. 11. The developer then extracts the data (Appendix Table 9).

Given the large volume of data, the developer first takes an overall view using a box plot. This visualization helps subdivide the data into well-defined classes: Q1 (25%), Q2 (50%), and Q3 (75%).

The first box plot, Fig. 12, visualizes a negatively skewed distribution, where most resolution times cluster near the lower end of Q1, while fewer products exhibit longer resolution times. The Q1 value (2 days 12:18:21) and median Q2 value (7 days 7:32:52) suggest that resolution times for many products are typically around 5 days. However, products exceeding the Q3 quartile form a clear outlier group. These typically require more than 8 days, 3 h, 47 min, and 30 s for resolution. This longer resolution time aligns with the service manager's needs. The upper quartile is particularly important because it identifies a subset of products that require attention to enhance resolution efficiency.

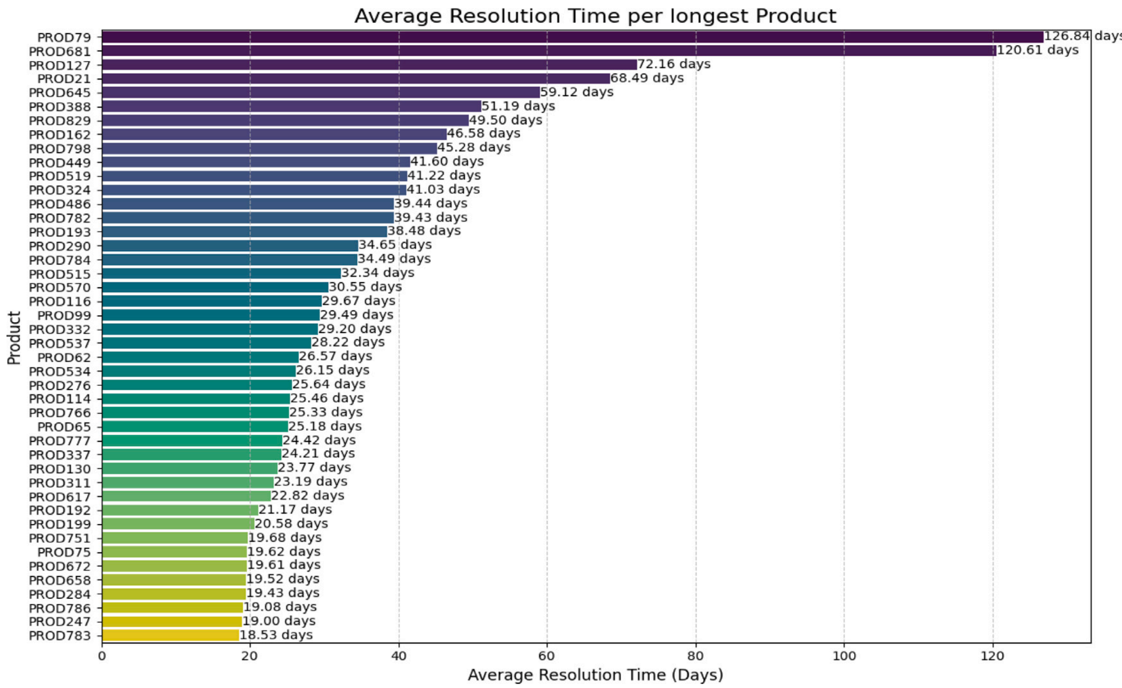

Fig. 14. Visualization of data extracted for Question Three.

Table 6
Analysis of incident impact levels and their resolution times.

Impact	Total incidents	Total resolution time	Average resolution time
High	531	2396 days, 17:41:31	4 days, 12:19:37
Low	5786	17348 days, 16:22:24	2 days, 23:57:40
Major	6	23 days, 11:35:52	3 days, 21:55:58
Medium	7544	25365 days, 04:04:18	3 days, 08:41:42

To focus the analysis, the developer filters the dataset to include only products with resolution times above Q3. Despite narrowing the scope, the filtered group still contains many products with long durations. As shown in Fig. 13, the developer creates a second box plot for this subset.

The second box plot (Fig. 13) focuses exclusively on products with resolution times exceeding the Q3 threshold identified earlier (see Fig. 12). This refined view reveals a positively skewed distribution, with most values concentrated towards the higher end of Q3. The Q1 value (approximately 10 days) suggests that even the fastest-resolving products in this group require significant time. The median Q2 (around 12 days) indicates that half of the products in this group require more than one week for resolution. The Q3 value (18 days, 10 h, and 42 min) marks the upper range of what could be considered typical. Above that threshold lie several extreme outliers. These outliers are especially important to examine, as they may indicate process bottlenecks or product-specific inefficiencies.

The data (Appendix Table 9) list products with the longest average resolution times. For each product, the table provides the total resolution time, number of incidents, and the computed average.

The developer uses the data from Appendix Table 9 to create a visualization 14 that ranks the products based on their average resolution time. This allows the service manager to quickly identify the slowest-resolving products within the long-duration subset.

Step 5:Interpret results and make decisions. The service manager interprets the results (Fig. 14) to answer the question and make decisions.

Based on the analysis of the results (Appendix Table 9 and Fig. 14), the service manager conducted a comparative analysis of the resolution times for the different products. The shortest average resolution time shown is 18 days 12:41:37.

In the first series of products (between PROD783 and PROD570), the average resolution time increases by approximately one day for each subsequent product. In contrast, the average resolution time in the second series (between PROD515 and PROD162) ranges from one to two days.

For other products, the analysis observed more notable variations. The resolution time for PROD829 is three days longer than its predecessor. The difference between PROD388 and PROD645 is eight days, and between PROD645 and PROD21 is seven days. Similarly, the difference between PROD21 and PROD127 is four days. A substantial increase of 48 days in resolution time was observed between PROD127 and PROD681, followed by a six-day difference between PROD681 and PROD79.

In conclusion, products with resolution times exceeding 46 days are classified as having extremely long resolution periods. These products include PROD829, PROD388, PROD645, PROD21, PROD127, PROD 681, and PROD79. For future analysis, the service manager should engage the development team to document this information in an internal knowledge library to ensure accessibility.

This analysis validates the service manager's initial question by highlighting potential complications that require performance adjustments.

Comparing the data from Example 1 and Example 3 reveals that seven products—PROD645, PROD537, PROD766, PROD617, PROD75, PROD658, and PROD247—appear in both the "longest average resolution time" and "products escalated to the third line" categories. These products exhibit both high average resolution times and frequent instances escalated to the third line, suggesting a need to improve their incident management processes. This finding enables the service manager to pose additional questions and analyze resources to implement performance enhancements.

6.4. Example 4: Impact levels of incidents and their resolution time

Step 1: Define the question. The service manager wants to know whether incidents with higher impact levels are resolved more quickly. To address this, they ask: Is there a correlation between incident impact level and resolution time?

Step 2: Specify the meta-model. The developer specifies a meta-model for structuring the data needed to answer the question. The data reflect incident resolution times categorized by impact level and the average resolution time for each impact level.

The specified meta-model, illustrated in Fig. 15, consists of one class and one table that analyzes the resolution times for incidents grouped by impact level. The meta-model includes the following attributes:

- **Impact Level:** a categorical variable representing the incident's severity (e.g., High, Medium, Low, Major);
- **incident_count:** the total number of incidents for each impact level;
- **total_resolution_time:** the cumulative time to resolve all incidents per impact level, in days, hours, and minutes (e.g., "5 days 12:19:30");
- **avg_resolution_time:** the average time to resolve a single incident for each impact level.

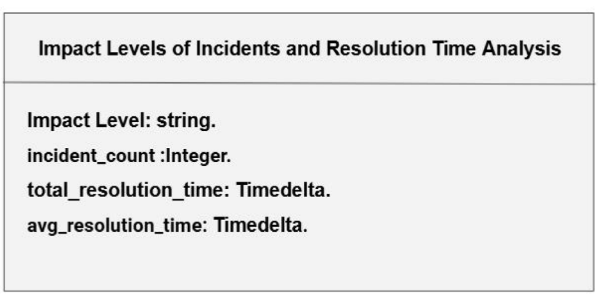

Fig. 15. Meta-Model for Question four.

Step 3: Extract the data. The developer creates Python code to retrieve the data outlined in the meta-model 15 and extracts the values shown in Table 6.

Table 6 summarizes, for each impact level, the total number of incidents, the combined resolution time, and the average time per incident.

Step 4: Visualize the results. The developer used Weighted Regression Analysis on the extracted data from Table 6 so that the service manager could visualize the results in Fig. 16. The developer also provided the summary table of this analysis, Table 7, for the service manager to interpret and draw conclusions.

The visualization, Fig. 16, illustrates the relationship between impact level (X-axis) and average resolution time in seconds (Y-axis). Each point represents a category of impact level. Bubble size reflects data magnitude for each point. The red regression line indicates a trend and highlights the correlation between higher impact levels and increased resolution times.

Step 5: Interpret results and make decisions. The service manager analyzed the statistical results presented in Fig. 16, Table 7, and Table 6 to determine whether a correlation exists between incident impact levels and their resolution times.

As shown in the data presented in Table 6, the average incident resolution times, ordered by impact level, are as follows:

1. Major: 3 days, 21 h, 55 min, 58 s (338,158 s);
2. Low: 2 days, 23 h, 57 min, 40 s (259,060 s);
3. Medium: 3 days, 8 h, 41 min, 42 s (290,502 s);
4. High: 4 days, 12 h, 19 min, 37 s (389,977 s).

Thus, these data demonstrate a clear positive correlation between incident impact and resolution time, with higher-impact incidents requiring substantially longer to resolve.

Fig. 16 illustrates a positive trend between incident impact and resolution time. A Weighted Least Squares (WLS) regression analysis confirms this descriptive relationship, with the model explaining a substantial portion of the variance in resolution time ($R^2 = 0.807$, adjusted $R^2 = 0.711$; see Table 7). Despite this strong model fit, however, the overall relationship is not statistically significant (F-statistic = 8.387, $p = 0.101$). Similarly, the coefficient for the impact level, while indicating an increase in resolution time of approximately 11.85 h (42,680 s) per impact level, does not reach the significance threshold of $\alpha = 0.05$.

The lack of statistical significance is likely attributable to the low statistical power of the analysis, a direct consequence of the limited degrees of freedom available from only four impact categories. Consequently, while the data suggest a positive correlation, they are insufficient to formally conclude that a relationship between incident impact and resolution time exists. Validating this potential trend and achieving sufficient statistical power would require a larger dataset with more observations. This limitation underscores the preliminary nature of the current finding.

7. Empirical evaluation

An exploratory survey, titled *Survey on Requirement-Driven Process Mining Approaches*, was conducted to empirically evaluate the proposed requirement-driven process mining approach. Thirty participants completed the seven-question survey.

7.1. Participant demographics

The participants reported diverse levels of expertise in data analysis and process mining. For data analysis, 50% identified as experts, 40% reported basic knowledge, and 10% were unfamiliar with the subject. Regarding process mining, 16.7% of participants were experts, 63.3% had basic knowledge, and 20% were unfamiliar. This diverse distribution of expertise ensures a balanced evaluation of the proposed approach from multiple viewpoints.

7.2. Key findings

The survey yielded four key findings regarding the perception and potential application of the proposed approach:

- **Comparative Effectiveness:** A significant majority of participants (93.3%; 28 out of 30) perceived the proposed requirement-driven approach as delivering superior results compared to traditional process mining tools. The remaining two participants (6.7%) were unsure (see Fig. 17).
- **Integration with Existing Tools:** When asked about combining the proposed method with existing tools, 56.7% of participants (17) stated they would definitely do so, and 36.7% (11) responded with "maybe, in some cases". Only two participants (6.7%) preferred to use a single tool exclusively. This result indicates that the proposed approach is viewed not as a replacement for existing tools, but as a valuable complement (see Fig. 18).
- **Cognitive Load Management:** When dealing with models that create a high cognitive load, 63.3% of participants (19) preferred the proposed approach, while 36.7% (11) favored a combination of methods. This highlights its utility in simplifying complex process analysis (see Fig. 19).

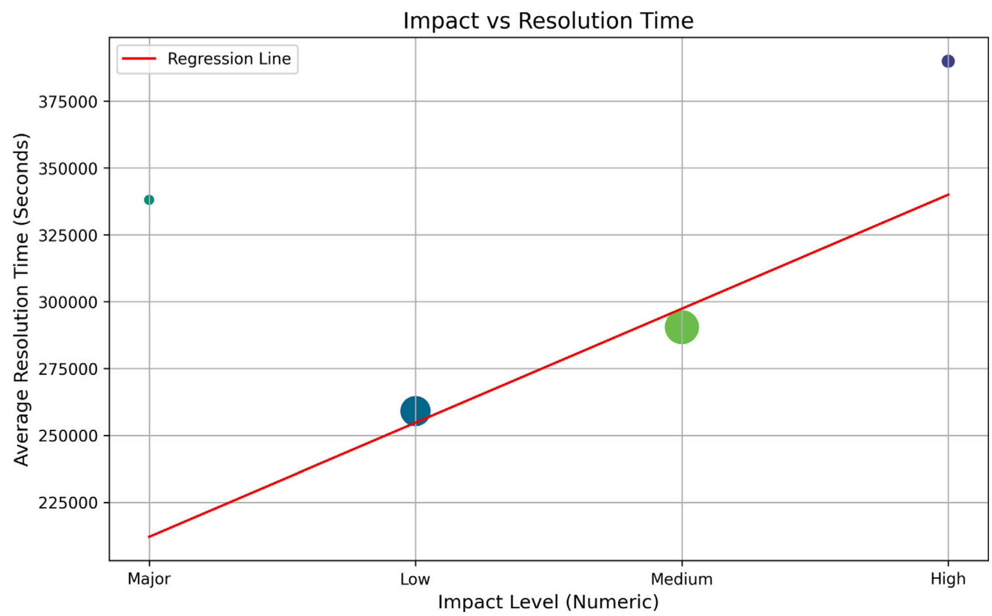

Fig. 16. Analysis of Impact Levels of Incidents and Their Resolution Times.

Table 7
WLS regression results.

Dep. Variable:	**Average_Resolution_Time_Seconds**			**R-squared:**	0.807
Model:	WLS			**Adj. R-squared:**	0.711
Method:	Least Squares			**F-statistic:**	8.387
Date:	Mon, 20 Jan 2025			**Prob (F-statistic):**	0.101
Time:	11:47:28			**Log-Likelihood:**	−46.604
No. Observations:	4			**AIC:**	97.21
Df Residuals:	2			**BIC:**	95.98
Df Model:	1				
Covariance Type:	nonrobust				

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
const	1.694e+05	3.95e+04	4.290	0.050	−502.738	3.39e+05
Impact_Numeric	4.268e+04	1.47e+04	2.896	0.101	−2.07e+04	1.06e+05

Omnibus:	nan	**Durbin-Watson:**	0.797
Prob(Omnibus):	nan	**Jarque–Bera (JB):**	0.174
Skew:	−0.108	**Prob(JB):**	0.917
Kurtosis:	2.002	**Cond. No.**	14.5

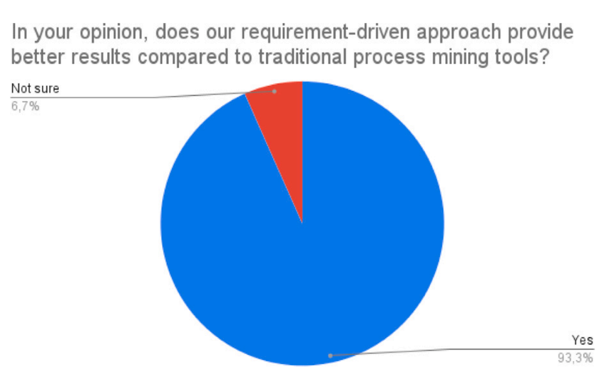

Fig. 17. Survey responses on the comparative effectiveness of the proposed approach versus traditional tools (N=30).

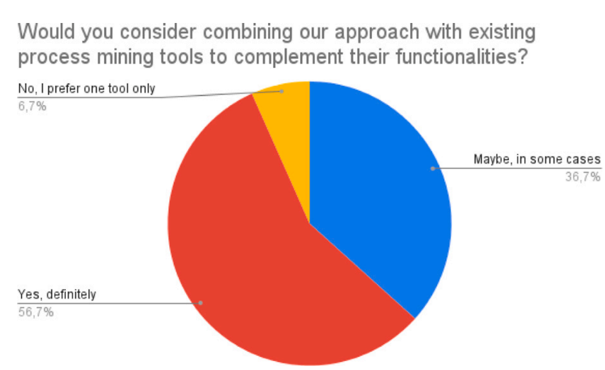

Fig. 18. Participant willingness to combine the proposed requirement-driven approach with existing process mining tools (N=30).

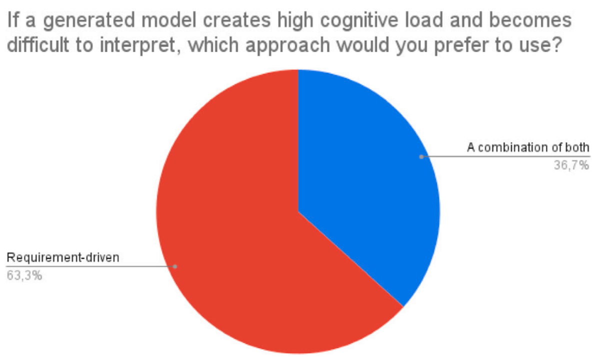

Fig. 19. Participant preference for an approach when faced with high cognitive load from complex models (N=30).

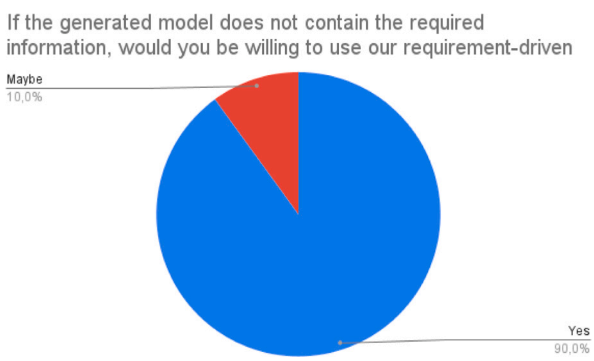

Fig. 20. Participant responses on using the proposed approach to refine traditional models that lack necessary information (N=30).

- **Refining Incomplete Models:** A vast majority of participants (90%; 27 out of 30) affirmed they would use the proposed approach to refine models generated by traditional tools that lack required information (see Fig. 20).

7.3. Qualitative insights

Open-ended survey responses highlighted several perceived strengths of the proposed approach, including:

- Enhanced focus on specific business requirements.
- Reduced complexity in the resulting process models.
- More targeted and interpretable analysis.

Participants also suggested potential improvements, such as greater automation and the need for validation across broader and more diverse datasets.

7.4. Limitations

While this survey provides preliminary validation of the proposed approach's practical relevance, its findings are exploratory. The primary limitation is the modest sample size (n=30), which restricts the generalizability of the results. Future work will focus on a larger-scale validation study, incorporating quantitative performance metrics to build upon these initial qualitative findings.

8. Discussion

This research introduces a requirement-driven method based on MDE that provides managers with process analysis outcomes tailored to their specific information needs. The proposed approach leverages core MDE mechanisms, such as the systematic mapping of requirements to executable results, to address documented shortcomings of traditional process mining. These shortcomings include cognitive overload from overly complex models and the misalignment of findings with business requirements. This method is not proposed as a replacement for established process mining tools like ProM and Disco. Instead, it functions as a complementary methodology for contexts where traditional tools produce results that are too general or cognitively demanding for managerial objectives. To supplement the descriptive evaluation, a preliminary exploratory survey was conducted comparing the proposed method with traditional process-mining tools. Participants consistently reported improved clarity and decision-making support when using the requirement-driven approach, particularly in contexts involving high cognitive load. While exploratory in scope, these results provide an empirical basis for future large-scale validation.

Traditional process mining methodologies typically favor exhaustive modeling, generating comprehensive models that include large amounts of irrelevant data. Although these representations may seem beneficial from a data completeness perspective, recent studies highlight significant downsides. Such models often cause information overload, increasing cognitive demand and negatively impacting managerial productivity and decision-making [31,32]. This finding aligns with Cognitive Load Theory (CLT), which emphasizes the limited processing capacity of users and suggests that analytical models should intentionally limit and prioritize outputs to enhance comprehension [31]. Within this framework, the proposed method offers a structured alternative that selectively simplifies process models to prioritize the interpretability and relevance of these models over mere data exhaustiveness. Moreover, once requirement-driven queries have been formalized and validated, they can be stored and reused as modular analytical components. This mechanism allows service managers to retrieve and adapt prior analyses without direct developer intervention, thereby reducing dependency and improving response times. Future integration with low-code or semi-automated platforms could further extend this self-service capability.

By systematically limiting outputs to elements explicitly requested by decision-makers, this MDE approach proactively manages cognitive load and generates artifacts with improved clarity, simplicity, and precision. The method uses explicit modeling techniques to create outputs tightly coupled to user requirements, thereby significantly reducing the cognitive complexity typical of traditional frameworks. This result is consistent with recent recommendations in computer-supported decision-making research advocating for methods that prioritize relevance and specificity over quantity and completeness [33,34]. Moreover, this targeted scope provides a valuable complement to automated tools by ensuring that insights remain directly aligned with stakeholder-defined needs while avoiding analytical noise.

Nevertheless, the proposed method has several operational and technical constraints. The primary limitation is the need for continuous technical involvement, as each new managerial request demands an explicit redefinition or adjustment of models, transformations, and executable artifacts. This dependency on software engineering support introduces significant operational overhead, which can affect responsiveness and resource allocation. In contrast, traditional tools (such as ProM or Disco) offer users greater exploratory autonomy, allowing for independent—though cognitively demanding—interaction with large datasets. To investigate this trade-off, an empirical evaluation involving thirty participants was conducted. The results confirmed that the requirement-driven approach significantly reduced cognitive effort and decision-making complexity compared to traditional tools, validating its effectiveness in managing the cognitive overload documented in information systems research [34,35].

A key implication of this research is the challenge of balancing targeted precision with user autonomy. Future research could explore integrating advanced techniques to automate or semi-automate the evolution of requirement-driven MDE systems. Techniques such as intelligent model inference, adaptive machine learning, and dynamic model adaptation could reduce the need for manual adjustments, lessen technical dependency, and enable sustainable cognitive load management, while preserving the decision-making advantages of a targeted approach. Additionally, future studies could incorporate controlled benchmarking to systematically assess the comparative performance of requirement-driven and traditional process mining tools across different organizational contexts.

In summary, this research demonstrates the practical feasibility and analytical advantages of integrating requirement-driven MDE techniques into process analysis. The study also highlights key directions for technical improvement, emphasizing automation and adaptability. Addressing these areas in future research will enhance the method's practical utility, operational scalability, and broader applicability in real-world managerial contexts.

9. Conclusion

Current process mining methods are often limited by model complexity, low adaptability, and poor alignment with business objectives. These limitations hinder the generation of actionable insights, particularly in dynamic operational environments.

This study introduces a requirement-driven, meta-modeling approach to address these limitations. The proposed approach directly incorporates user-defined requirements into the model discovery process, linking process models to organizational goals to improve their interpretability and adaptability. This linkage produces more precise, actionable, and context-aware business process representations. The study also developed a lightweight, scalable Python-based analytical tool. In contrast to generalist platforms, the developed tool offers solutions tailored to service managers, thereby enhancing usability and decision-making.

This work demonstrates the value of integrating technological innovation with practical application by aligning process mining more closely with organizational goals. Future research should focus on three key areas: (1) automating meta-model adaptation using techniques such as adaptive machine learning to reduce user dependency on technical experts; (2) systematically evaluating the performance and applicability of the approach across diverse industries and with varying data quality; and (3) enhancing the scalability of the analytical tool for extremely large event logs. To validate these advancements, subsequent work will involve systematically benchmarking the approach against multiple datasets, conducting controlled experiments to isolate the effects of specific features, and performing user studies to assess the correctness, robustness, and optimality of the generated solutions.

Beyond these computational contributions, this research provides practical guidance for implementing the requirement-driven approach in organizational contexts. The approach can be systematically embedded within enterprise and software development workflows, particularly Agile frameworks, to enhance its operational value. To facilitate this integration, the following guidelines map the requirement-driven approach onto established Agile ceremonies:

- **Backlog Refinement:** The service manager evaluates existing tools against requirements. If gaps exist, the approach systematically derives candidate solutions.
- **Sprint Planning:** The approach's outcomes (e.g., structured requirements and solution paths) inform sprint goals, ensuring alignment with validated needs.
- **Sprint Execution:** Teams implement features based on the derived requirements and solution paths, adapting to feedback.
- **Sprint Review and Retrospective:** Structured requirements provide objective criteria for assessing delivered increments against initial needs, facilitating the capture of reusable knowledge for future sprints.

Integrating the proposed approach within Agile cycles combines the adaptability of iterative development with the rigor of structured requirement analysis. This integration fosters consistency, reduces ad-hoc decision-making, and enhances clarity across development and managerial levels. The approach thus provides not only a methodological contribution but also a practical framework for deploying requirement-driven engineering in process mining.

Ultimately, this research advances process mining from a descriptive, analytical exercise to a prescriptive, strategic tool directly responsive to business needs. By embedding organizational requirements into the core of model discovery, this work provides a foundation for more effective and context-aware, data-driven decision-making.

CRediT authorship contribution statement

Selsabil Ines Bouhidel: Writing – original draft, Investigation, Conceptualization. **Mohammed Mounir Bouhamed:** Writing – review & editing, Writing – original draft, Project administration, Investigation, Conceptualization. **Gregorio Diaz:** Writing – review & editing, Validation, Supervision, Investigation, Funding acquisition. **Nabil Belala:** Writing – review & editing, Supervision, Conceptualization.

Declaration of Generative AI and AI-assisted technologies in the writing process

During the preparation of this work, the authors used ChatGPT (OpenAI) and Gemini (Google) to improve the language and readability of the manuscript. After using these tools, the authors reviewed and edited the content as needed and take full responsibility for the content of the publication.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Gregorio Diaz reports financial support and administrative support were provided by University of Castilla-La Mancha. Gregorio Diaz reports financial support was provided by Spain Ministry of Science and Innovation. Gregorio Diaz reports was provided by European Regional Development Fund. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research was funded by the Ministerio de Ciencia, Innovación y Universidades, Agencia Estatal de Investigación, MICIU/AEI/10.13039/501100011033 grant number PID2021-122215NB-C32, European Regional Development Found EU grant numbers PID2021-122215NB-C32 and 2022-GRIN-34113 and University of Castilla-La Mancha grant number 2022-GRIN-34113.

It is also supported by Direction Générale de la Recherche Scientifique et du Développement Technologique in Algeria under the PRFU project C00L07UN250220230008.

Appendix A. Extended tables

See Tables 8 and 9.

Appendix B. Extended figures

See Fig. 21.

Table 8
Data Extracted for Question One.

Product	Third line escalated instances count	Total instances	Third line escalated rate (%)
PROD582	5	15	33.33
PROD645	1	6	16.67
PROD617	13	59	22.03
PROD67	3	6	50.00
PROD803	1	4	25.00
PROD609	13	33	39.39
PROD607	69	142	48.59
PROD765	1	6	16.67
PROD493	1	6	16.67
PROD537	3	6	50.00
PROD611	14	43	32.56
PROD182	1	8	12.50
PROD604	31	71	43.66
PROD606	3	5	60.00
PROD655	5	20	25.00
PROD546	4	32	12.50
PROD75	1	2	50.00
PROD695	4	8	50.00
PROD509	4	29	13.79
PROD766	1	2	50.00
PROD422	1	4	25.00
PROD662	5	14	35.71
PROD73	4	8	50.00
PROD630	1	2	50.00
PROD70	2	5	40.00
PROD3	1	9	11.11
PROD370	7	17	41.18
PROD797	1	6	16.67
PROD171	6	16	37.50
PROD603	2	12	16.67
PROD756	3	17	17.65
PROD194	6	28	21.43
PROD222	1	6	16.67
PROD578	1	2	50.00
PROD750	3	6	50.00
PROD247	2	4	50.00
PROD353	1	5	20.00
PROD693	1	2	50.00
PROD87	1	6	16.67
PROD405	1	4	25.00
PROD598	2	6	33.33
PROD658	1	2	50.00
PROD622	1	7	14.29
PROD155	1	5	20.00
PROD74	2	4	50.00
PROD189	1	2	50.00
PROD589	1	6	16.67
PROD692	2	4	50.00
PROD614	1	2	50.00
PROD613	1	2	50.00
PROD419	1	8	12.50
PROD594	1	7	14.29
PROD771	1	7	14.29
PROD72	1	4	25.00
PROD5	7	16	43.75
PROD568	12	26	46.15
PROD761	1	2	50.00
PROD602	1	2	50.00
PROD71	1	2	50.00
PROD80	5	10	50.00
PROD559	1	8	12.50
PROD352	2	6	33.33
PROD102	2	4	50.00
PROD569	1	4	25.00
PROD610	1	2	50.00
PROD708	1	2	50.00
PROD57	1	2	50.00
PROD89	1	2	50.00
PROD301	1	2	50.00
PROD690	1	2	50.00

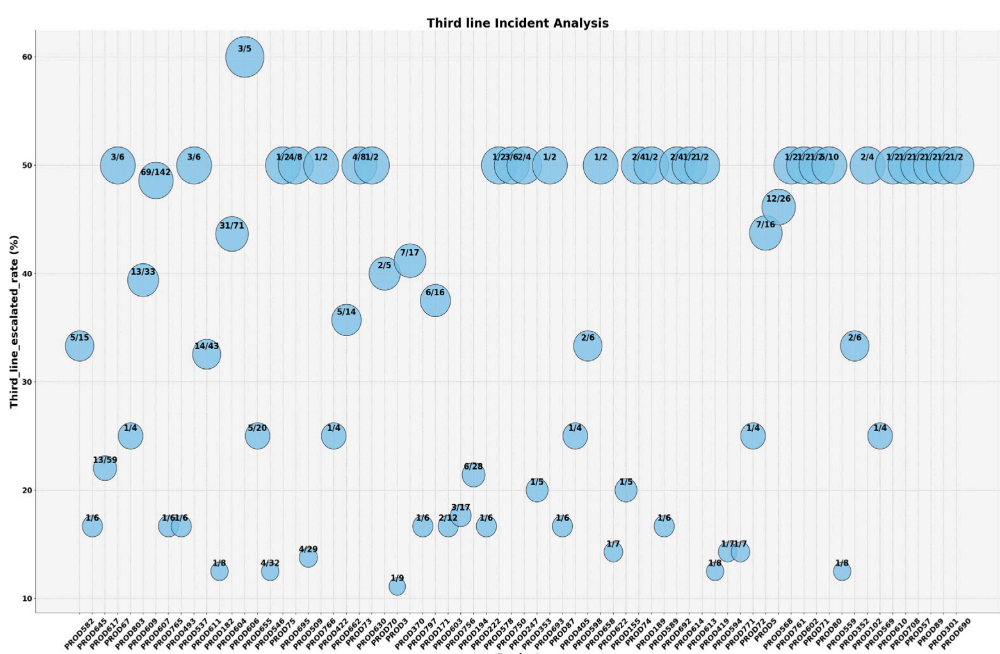

Fig. 21. Visualization of data Extracted for Question One.

Table 9
Average resolution time per product (Q75).

Product	Total resolution time	Incident count	Average resolution time
PROD783	37 days 01:23:13	2	18 days 12:41:36.500000
PROD247	37 days 23:49:05	2	18 days 23:54:32.500000
PROD786	95 days 09:40:38	5	19 days 01:56:07.600000
PROD284	38 days 20:48:21	2	19 days 10:24:10.500000
PROD658	19 days 12:22:47	1	19 days 12:22:47
PROD672	19 days 14:32:03	1	19 days 14:32:03
PROD75	19 days 14:56:36	1	19 days 14:56:36
PROD751	19 days 16:14:55	1	19 days 16:14:55
PROD199	41 days 03:38:57	2	20 days 13:49:28.500000
PROD192	42 days 08:17:45	2	21 days 04:08:52.500000
PROD617	684 days 12:20:52	30	22 days 19:36:41.733333333
PROD311	69 days 13:30:26	3	23 days 04:30:08.666666666
PROD130	47 days 13:01:44	2	23 days 18:30:52
PROD337	677 days 20:51:51	28	24 days 05:01:51.107142857
PROD777	24 days 10:10:37	1	24 days 10:10:37
PROD65	50 days 08:31:23	2	25 days 04:15:41.500000
PROD766	25 days 07:53:03	1	25 days 07:53:03
PROD114	178 days 04:36:45	7	25 days 10:56:40.714285714
PROD276	76 days 22:19:21	3	25 days 15:26:27
PROD534	52 days 07:07:38	2	26 days 03:33:49
PROD62	26 days 13:36:14	1	26 days 13:36:14
PROD537	84 days 15:45:00	3	28 days 05:15:00
PROD332	29 days 04:42:13	1	29 days 04:42:13
PROD99	58 days 23:31:30	2	29 days 11:45:45
PROD116	88 days 23:56:23	3	29 days 15:58:47.666666666
PROD570	30 days 13:17:41	1	30 days 13:17:41
PROD515	32 days 08:15:10	1	32 days 08:15:10
PROD784	34 days 11:44:38	1	34 days 11:44:38
PROD290	34 days 15:31:04	1	34 days 15:31:04
PROD193	38 days 11:32:43	1	38 days 11:32:43
PROD782	39 days 10:13:20	1	39 days 10:13:20
PROD486	315 days 11:44:50	8	39 days 10:28:06.250000
PROD324	41 days 00:45:06	1	41 days 00:45:06
PROD519	41 days 05:15:30	1	41 days 05:15:30
PROD449	41 days 14:29:19	1	41 days 14:29:19
PROD798	45 days 06:47:55	1	45 days 06:47:55
PROD162	46 days 13:55:45	1	46 days 13:55:45
PROD829	49 days 12:00:19	1	49 days 12:00:19
PROD388	358 days 07:14:31	7	51 days 04:27:47.285714285
PROD645	177 days 08:25:09	3	59 days 02:48:23
PROD21	68 days 11:51:21	1	68 days 11:51:21
PROD127	72 days 03:56:28	1	72 days 03:56:28
PROD681	120 days 14:45:15	1	120 days 14:45:15
PROD79	126 days 20:13:57	1	126 days 20:13:57

Data availability

Readers may find all the data obtained to reproduce the results presented in this paper and the corresponding source code in the following repository: Bouhidel, Selsabil Ines; Bouhamed, Mohammed Mounir; Diaz, Gregorio; Belala, Nabil (2025), *"A requirement driven process mining dataset*, **Mendeley Data**, V2, doi: 10.17632/s7rjw8nnvm.2.

References

[1] Wil M.P. van der Aalst, Process Mining: Data Science in Action, Springer, Heidelberg, ISBN: 978-3-662-49850-7, 2016, http://dx.doi.org/10.1007/978-3-662-49851-4.

[2] Mohammed Mounir Bouhamed, Allaoua Chaoui, Radouane Nouara, Gregorio Díaz, Abderraouf Dembri, Reducing the number of migrated instances during business process change: A graph rewriting approach, J. King Saud Univ.-Comput. Inf. Sci. 34 (9) (2022) 7720–7734.

[3] Mordor Intelligence, Business process management market — growth, trends, COVID-19 impact, and forecasts (2024–2029), 2024, https://www.mordorintelligence.com/industry-reports/business-process-management-market. (Accessed 28 May 2025).

[4] Grand View Research, Business process outsourcing market size, share & trends analysis report by service type (customer services, finance & accounting), by outsourcing type, by deployment, by end use, by region, and segment forecasts, 2025–2030, 2025, https://www.grandviewresearch.com/industry-analysis/business-process-outsourcing-bpo-market. (Accessed 9 December 2025).

[5] Wil Aalst, The application of Petri nets to workflow management, J. Circuits Syst. Comput. 8 (1998) 21–66, http://dx.doi.org/10.1142/S0218126698000043.

[6] Anna A. Kalenkova, Wil M.P. van der Aalst, Irina A. Lomazova, Vladimir A. Rubin, Process mining using BPMN: relating event logs and process models, in: Proceedings of the ACM/IEEE 19th International Conference on Model Driven Engineering Languages and Systems, 2016, pp. 123–123.

[7] Philip Weber, Behzad Bordbar, Peter Tino, A framework for the analysis of process mining algorithms, Syst. Man Cybern.: Syst. IEEE Trans. 43 (2013) 303–317, http://dx.doi.org/10.1109/TSMCA.2012.2195169.

[8] Adriano Augusto, R. Conforti, M. Dumas, M. Rosa, Artem Polyvyanyy, Split miner: automated discovery of accurate and simple business process models from event logs, Knowl. Inf. Syst. 59 (2019) 251–284, http://dx.doi.org/10.1007/s10115-018-1214-x.

[9] Volodymyr Leno, Marlon Dumas, Fabrizio Maria Maggi, Marcello La Rosa, Artem Polyvyanyy, Automated discovery of declarative process models with correlated data conditions, Inf. Syst. (ISSN: 0306-4379) 89 (2020) 101482, http://dx.doi.org/10.1016/j.is.2019.101482, URL: https://www.sciencedirect.com/science/article/pii/S0306437919305344.

[10] Felix Groß, Lisa Mannel, Wil Aalst, Enhancing the applicability of the eST-miner: Efficient precision-guided implicit place avoidance, 2023, pp. 121–128, http://dx.doi.org/10.1109/ICPM60904.2023.10271973.

[11] A. Kusters, Wil M.P. van der Aalst, Revisiting the alpha algorithm to enable real-life process discovery applications - extended report, 2023, http://dx.doi.org/10.48550/arXiv.2305.17767, arXiv, abs/2305.17767.

[12] Anon, Tool, 2020, URL: https://promtools.org/prom-documentation/. (Accessed 2020).

[13] Anon, Tool, 2020, URL: https://fluxicon.com/disco/. (Accessed 2020).

[14] Jean Bézivin, Olivier Gerbé, Towards a precise definition of the OMG/MDA framework, ISBN: 0-7695-1426-X, 2001, pp. 273–280, http://dx.doi.org/10.1109/ASE.2001.989813.

[15] Marco Brambilla, Jordi Cabot, Manuel Wimmer, Model-Driven Software Engineering in Practice, vol. 1, 2012, http://dx.doi.org/10.2200/S00441ED1V01Y201208SWE001.

[16] Skander Turki, Ingénierie système guidée par les modèles : Application du standard IEEE 15288, de l'architecture MDA et du langage SysML à la conception des systèmes mécatroniques, 2008.

[17] Xixi Lu, A. Gal, H. Reijers, Discovering hierarchical processes using flexible activity trees for event abstraction, in: 2020 2nd International Conference on Process Mining, ICPM, 2020, pp. 145–152, http://dx.doi.org/10.1109/ICPM49681.2020.00030.

[18] Adriano Augusto, Marlon Dumas, Marcello La Rosa, Automated discovery of process models with true concurrency and inclusive choices, ISBN: 978-3-030-72692-8, 2021, pp. 43–56, http://dx.doi.org/10.1007/978-3-030-72693-5_4.

[19] Merve Tiftik, Tugba Gurgen Erdogan, Ayça Kolukısa, A framework for multi-perspective process mining into a BPMN process model, Math. Biosci. Eng.: MBE 19 (2022) 11800–11820, http://dx.doi.org/10.3934/mbe.2022550.

[20] Wil Aalst, Process Mining: A 360 Degree Overview, ISBN: 978-3-031-08847-6, 2022, pp. 3–34, http://dx.doi.org/10.1007/978-3-031-08848-3_1.

[21] Shideh Saraeian, Babak Shirazi, Process mining-based anomaly detection of additive manufacturing process activities using a game theory modeling approach, Comput. Ind. Eng. 146 (2020) 106584, http://dx.doi.org/10.1016/j.cie.2020.106584.

[22] Claudio Castiglione, Automated generation of digital models for manufacturing systems: The event-centric process mining approach, Comput. Ind. Eng. (ISSN: 0360-8352) 197 (2024) 110596, http://dx.doi.org/10.1016/j.cie.2024.110596, URL: https://www.sciencedirect.com/science/article/pii/S0360835224007174.

[23] Amitava Mukherjee, Chong Zhi Lin, Michael Boon Chong, Comparisons of some distribution-free CUSUM and EWMA schemes and their applications in monitoring impurity in mining process flotation, Comput. Ind. Eng. (2019) http://dx.doi.org/10.1016/j.cie.2019.106059.

[24] W. van der Aalst, T. Weijters, L. Maruster, Workflow mining: discovering process models from event logs, IEEE Trans. Knowl. Data Eng. 16 (9) (2004) 1128–1142, http://dx.doi.org/10.1109/TKDE.2004.47.

[25] Massimiliano de Leoni, Wil Aalst, Marcus Dees, A general process mining framework for correlating, predicting and clustering dynamic behavior based on event logs, Inf. Syst. 56 (2015) http://dx.doi.org/10.1016/j.is.2015.07.003.

[26] Tpb Wiel, Process mining using integer linear programming, 2010, URL: https://api.semanticscholar.org/CorpusID:16736603.

[27] A. Rozinat, W.M.P. van der Aalst, Conformance checking of processes based on monitoring real behavior, Inf. Syst. (ISSN: 0306-4379) 33 (1) (2008) 64–95, http://dx.doi.org/10.1016/j.is.2007.07.001, URL: https://www.sciencedirect.com/science/article/pii/S030643790700049X.

[28] Marlon Dumas, Luciano García-Bañuelos, Process Mining Reloaded: Event Structures as a Unified Representation of Process Models and Event Logs, ISBN: 978-3-319-19487-5, 2015, pp. 33–48, http://dx.doi.org/10.1007/978-3-319-19488-2_2.

[29] Volvo IT, VINST data set, 2012, Document discussing the VINST dataset and its related processes such as incident and problem management.

[30] Volvo Group, Volvo IT contact information, 2025, https://www.volvogroup.com/en/contact-us/volvo-it.html. (Accessed 2024).

[31] Brianda Sígolo, Helen Casarin, Contributions of cognitive load theory to understanding information overload: a literature review, RDBCI Rev. Digit. Bibl. E Ciência Informação 22 (2024) p. e024027, http://dx.doi.org/10.20396/rdbci.v22i00.8677359/en.

[32] R.G. Letsholo, Marius Pretorius, Investigating managerial practices for data and information overload in decision making, J. Contemp. Manag. 13 (2016) 767–792.

[33] Ashley Braganza, Laurence Brooks, Daniel Nepelski, Maged Ali, Russ Moro, Resource management in big data initiatives: Processes and dynamic capabilities, J. Bus. Res. 70 (2016) http://dx.doi.org/10.1016/j.jbusres.2016.08.006.

[34] Anam Nusrat, Yong He, Adeel Luqman, Ankit Mehrotra, Amit Shankar, Unraveling the psychological and behavioral consequences of using enterprise social media (ESM) in mitigating the cyberslacking, Technol. Forecast. Soc. Change (ISSN: 0040-1625) 196 (2023) 122868, http://dx.doi.org/10.1016/j.techfore.2023.122868, URL: https://www.sciencedirect.com/science/article/pii/S004016252300553X.

[35] Martin Eppler, Jeanne Mengis, The concept of information overload: A review of literature from organization science, accounting, marketing, MIS, and related disciplines, Inf. Soc. 20 (2004) 325–344, http://dx.doi.org/10.1080/01972240490507974.