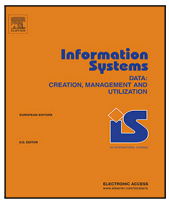
Information Systems 140 (2026) 102738

Contents lists available at ScienceDirect

Information Systems

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



Agentic Business Process Management: A research manifesto ☆,☆☆

Diego Calvanese [a], Angelo Casciani [g], Giuseppe De Giacomo [b], Marlon Dumas [c], Fabiana Fournier [d], Timotheus Kampik [e,f,*], Emanuele La Malfa [b], Lior Limonad [d], Andrea Marrella [g], Andreas Metzger [h], Marco Montali [a], Daniel Amyot [i], Peter Fettke [j,k], Artem Polyvyanyy [l], Stefanie Rinderle-Ma [m], Sebastian Sardiña [n], Niek Tax [o], Barbara Weber [p]

[a] *Free University of Bozen-Bolzano, Bozen-Bolzano, Italy*
[b] *University of Oxford, Oxford, United Kingdom*
[c] *University of Tartu, Tartu, Estonia*
[d] *IBM Research, Haifa, Israel*
[e] *Umeå University, Umeå, Sweden*
[f] *SAP, Berlin, Germany*
[g] *Sapienza Università di Roma, Rome, Italy*
[h] *paluno (Ruhr Institute for Software Technology), University of Duisburg Essen, Essen, Germany*
[i] *University of Ottawa, Ottawa, Canada*
[j] *German Research Center for Artificial Intelligence (DFKI), 66123 Saarbrücken, Germany*
[k] *Saarland University, 66123 Saarbrücken, Germany*
[l] *The University of Melbourne, Parkville, VIC 3010, Australia*
[m] *TUM School of Computation, Information, and Technology, TU Munich, Garching, Germany*
[n] *RMIT University, Melbourne, Australia*
[o] *Meta, London, United Kingdom*
[p] *University of St. Gallen, St. Gallen, Switzerland*

ARTICLE INFO

Keywords:
Business process management
Autonomous agents
Agentic AI
Framed autonomy
Explainability
Conversational actionability
Self-modification

ABSTRACT

This paper presents a manifesto that articulates the conceptual foundations of *Agentic Business Process Management* (APM), an extension of Business Process Management (BPM) for governing autonomous agents executing processes in organizations. From a management perspective, APM represents a paradigm shift from the traditional view on business processes. This shift is driven by the realization of process awareness by agent-oriented abstractions: software and human agents act as primary functional entities that perceive, reason, and act within explicit process frames. Thus, APM moves away from automation-oriented BPM towards systems in which autonomy is constrained, aligned, and made operational through *process aware* agents.

We introduce the core abstractions and architectural elements required to realize APM systems and elaborate on four key capabilities that agents in APM systems must support: *framed autonomy*, *explainability*, *conversational actionability*, and *self-modification*. These capabilities jointly ensure that agents' goals are aligned with organizational goals and that agents behave in a framed yet proactive manner in pursuing those goals. We discuss the extent to which the capabilities can be realized and identify research challenges whose resolution requires further advances in BPM, AI, and multi-agent systems. The manifesto thus serves as a roadmap for bridging these communities and for guiding the development of APM systems in practice.

☆ This article is part of a Special issue entitled: 'Autonomous Process Execution Systems' published in Information Systems.

☆☆ Given the role as Editor-in-Chief of this journal, Stefanie Rinderle-Ma had no involvement in the peer-review of this article and has no access to information regarding its peer-review. Full responsibility for the editorial process for this article was delegated to another journal editor.

* Corresponding author.
E-mail addresses: diego.calvanese@unibz.it (D. Calvanese), casciani@diag.uniroma1.it (A. Casciani), giuseppe.degiacomo@cs.ox.ac.uk (G. De Giacomo), marlon.dumas@ut.ee (M. Dumas), fabiana@il.ibm.com (F. Fournier), tkampik@cs.umu.se (T. Kampik), emanuele.lamalfa@cs.ox.ac.uk (E.L. Malfa), liorli@il.ibm.com (L. Limonad), marrella@diag.uniroma1.it (A. Marrella), andreas.metzger@paluno.uni-due.de (A. Metzger), marco.montali@unibz.it (M. Montali), damyot@uottawa.ca (D. Amyot), peter.fettke@dfki.de (P. Fettke), artem.polyvyanyy@unimelb.edu.au (A. Polyvyanyy), stefanie.rinderle-ma@tum.de (S. Rinderle-Ma), sebastian.sardina@rmit.edu.au (S. Sardiña), niek@meta.com (N. Tax), barbara.weber@unisg.ch (B. Weber).

https://doi.org/10.1016/j.is.2026.102738
Received 11 December 2025; Received in revised form 8 April 2026; Accepted 8 April 2026
Available online 17 April 2026

1. Introduction and motivation

With the advent of big data analytics, machine learning, and foundation models — notably, Large Language Models (LLMs) — organizations increasingly adopt data-driven artificial intelligence (AI) approaches to realize software systems that form the backbone of their operations. A long-term ambition leveraging these technologies to instill *autonomy* into software systems, meaning that they independently perceive, reason, and act to achieve their goals, emphasizing intentionality, goal-directed behavior, and constrained autonomy [1]. This lifts the burden of low-level decisions from knowledge workers and can ultimately reduce costs, alleviate workforce shortages, foster creativity and competitiveness, and improve the experiences of employees, customers, and other stakeholders involved [2]. Currently, efforts to achieve greater autonomy in software systems often attempt to make use of LLMs and are summarized under the umbrella term of *agentic AI*, thus promoting the notion of *agents* and *Multi-Agent Systems* (MAS) as the fundamental abstractions of software systems [3].

However, increasing the autonomy of LLM-based software agents also brings risks. On the business level, such agents may make decisions that are in violation of compliance rules, disregard social norms and expectations, or simply lead to directly adverse business outcomes [4]. On the technical level, reliance on LLMs entails inherent functional uncertainty that is difficult to test and debug, leading to system integration and maintenance challenges and, ultimately, to technical debt when replacing software agents requires making sense of entangled behaviors emerging from agent interactions.

To mitigate and control these risks, it is crucial to *govern* software agents in organizations. Agent governance has been a well-established line of research for several decades [5]. It originated primarily in the AI subfield of *normative MAS*, which focuses on regulating autonomous agent behavior through the specification of norms such as obligations, permissions, and prohibitions to ensure social order and compliance [6]. On the more practical side, LLM providers have issued recommendations for governing LLM-based agents [7]. Still, long-running research lines and pragmatic practice-based guidelines for governing agents lack a bridge to a practically well-established perspective of managing work in organizations.

Such a perspective is provided by Business Process Management (BPM) literature and practice. BPM is concerned with the management of (sets of) activities that are performed in coordination and jointly contribute to the accomplishment of organizational (business) goals [8,9]. With the advent of AI agents in modern organizations and the delegation of some activities to autonomous agents, it is imperative that BPM also encompasses the management of such agents acting within organizations. Indeed, BPM and MAS research have a strong joint tradition, dating back to the 1990s (cf. [10] for an overview). Still, most of the corresponding work focuses on allocating agents to execute tasks within a business process, rather than on managing autonomous agents as first-class citizens to ensure the achievement of business goals. While some exceptions exist — notably, work on process choreographies [11] and the emerging research directions of *agent system mining* [12,13], as well as of *agentic AI process observability* [14], within process mining — there is a lack of a holistic perspective on what is needed to apply and extend BPM to the governance of software agents in organizations. Moreover, BPM practitioners often lack a clear understanding of what constitutes an agent (and its associated benefits and risks), resulting in overlooked insights and missed opportunities for meaningful progress in the emerging domain of *Agentic (Business) Process Management* (APM) [10].

Consider an example of an APM system that facilitates the onboarding of new suppliers as part of a procurement process [15]. In such a system, there is a buyer agent and several supplier agents, some of which are human and some AI-based. Both types of agents (buyer and supplier) possess process awareness in the sense that they continuously align their individual goals and act in concert to meet the organizational objectives of the procurement process.

To address the above lack of a holistic perspective, this manifesto introduces the APM paradigm, with the notion of *framed agency* at its core. Specifically, we provide a joint position of academics focusing on both theoretical and applied aspects of the fields of BPM and autonomous agents, as well as that of industry experts. We identified and discussed the required capabilities of APM and associated research challenges during the Dagstuhl Seminar #25192 (AUTOBIZ[1]) and follow-up sessions. Specifically, after a series of talks by experts, participants split into working groups to further discuss individual topics of the research agenda. The results of these breakout groups were presented to all seminar participants, and their feedback was used to improve the findings. These were further discussed during the PMAI'25 workshop,[2] culminating in the vision of APM systems.

We first present the underlying concepts and core definitions of an APM system, followed by the main architectural elements, highlighting the key distinctions between the APM system level and the agent level (Section 2). Then, we discuss four key capabilities that are required to achieve responsible and effective management of AI agents in business processes: *framed autonomy*, *explainability*, *conversational actionability*, and *self-modification* (Section 3). These capabilities are purposefully ordered: Framing ensures that agents are process-aware and their actions are guard-railed. Explainability serves as a means to preserve the system's autonomy by having the agents articulate the rationale for their behavior, a prerequisite to deploying them. Conversational actionability then ensures effective execution, interaction, and governance. Finally, self-modification allows for continuous improvement to help move towards the long-term vision of self-improving software systems in the operational back-ends of organizations. The realization and delivery of these four key capabilities involves a series of research challenges, which are presented in Section 4. We conclude the paper with critically reflections and call for the establishment of APM formal foundations, as well as engineering and management practices, and for empirical research to confirm or challenge the assumptions made in this manifesto (Section 5).

2. Agentic business process management systems

This section first defines the fundamental concepts underlying an APM system and then elaborates on the main architectural elements that constitute such a system at both *macro* (system) and *micro* (agent) levels.

2.1. Fundamental concepts

In line with the AI and software engineering communities [19–21], we introduce the concept of an *agentic system* as a collection of (one or more) *individual* goal-driven agents that sense, reason, and act upon external stimuli to deliver (parts of) the functionality of a software system [19,20,22]. We use the term *agentic* to indicate that the APM system is agent-centric: agents constitute the primary functional entities responsible for executing business processes and serve as a major organizational backbone. This marks a paradigm shift from previously introduced AI-augmented BPM systems [23] and other *non-agent-centric* BPM approaches, where the term denotes that certain core entities (i.e., not agents) may exhibit some degree of agentic characteristic (e.g., autonomy), see [24].

[1] See https://www.dagstuhl.de/25192. We express our gratitude to the Scientific Directorate and staff of Schloss Dagstuhl for their invaluable support. We also thank our fellow participants for their engaging discussions.

[2] https://ceur-ws.org/Vol-4087/; intermediate proposals that this paper extends are provided in [15–18].

In an APM system, an *agent* is an autonomous entity situated in an environment [20,25]. Functionally, an agent operates through a continuous control loop [26], alternating between *sensing* its environment to update its internal state, *reasoning* to select actions that align with its goals, and *acting* to influence the environment. An agent is *proactive* and has its own thread of control, meaning that it makes (semi) autonomous decisions about which actions to perform and when [10,27]. In this sense, an agent processes information, generates content and knowledge, and proactively makes decisions and performs activities and tasks. This positions agents in contrast to classical components, objects, and services, which do not have goals and respond reactively to user prompts or inputs provided via an API [20,28,29]. For example, an agent may browse Web pages and make online purchases on behalf of a user; it may compare prices, select items, and complete checkouts [30].

The notion of an agent is sufficiently general to design environments in which heterogeneous entities perceive, reason, and act to achieve designated objectives. Within an agentic system, we distinguish the following categories of agents:

Human agents, such as process workers or process managers.

Software agents [20], which inhabit a software environment and whose tasks are executed by means of program code, *without deliberation*.

(Physically) Embodied agents [31] (a.k.a robots), whose decision-making and action directly affect the physical world.

AI agents, that accomplish tasks by means of deliberation via AI algorithms and models, for example, using generative AI techniques such as LLMs [32,33].

While an agentic system hosts a collection of inter-operating agents (i.e., it is a socio-technical system), it does not by itself realize any shared form of *process awareness*. Here, we define process awareness as the assurance that agents' inner workings conform to organizational processes and adhere to their operational constraints, regulations, and goals. Conceptually, we therefore regard an APM system as a system that explicitly realizes such process awareness in some concrete manner.

Accordingly, in the context of BPM, we define an APM system as follows:

Definition 2.1. An *Agentic Business Process Management* (APM) system is a socio-technical system jointly realized by a collection of agents, some of which are at least partially **process-aware**.

An APM system is considered socio-technical in the sense that it allows for a combination of social (human) and technical (technological) agents that interact and depend on each other to function effectively. This does not exclude fully autonomous systems with little to no intervention by any human agent. Collectively, agents execute business processes by virtue of each agent possessing a certain degree of process awareness. Thus, in an APM system, every agent is treated as *autonomous*, with its own goals, decision-making capabilities, and knowledge of (parts of) the business processes. Agents may employ *tools*, owned or shared, that reflect the objects, resources, functions, and services required to fulfill their goals.

We distinguish *autonomy* from *automation*. While automation denotes the execution of predefined tasks or workflows exactly as specified, APM systems facilitate autonomy, whereby agents can perceive, reason, and choose how to act within a process frame in order to achieve given goals. In other words, automation follows fixed rules, whereas autonomy in APM allows agents to make context-sensitive decisions while still respecting process awareness and constraints.

Revisiting our initial APM system example of the supplier onboarding process (Section 1), we can see there are two types of process-aware agents: buyers and suppliers, some human and some AI-based. The buyer agent is responsible for periodically disseminating Requests for Quotations (RFQs) to suppliers identified as suitable for meeting the manufacturing process targets and, in turn, evaluating the quotes they provide in order to decide which supplier to contract with. The supplier agents work to fulfill their winning bids within individual time frames determined by the contractual terms. Each individual agent may also pursue other, non–process-aware goals that reflect the agent's specific role. For example, a supplier agent may be restricted to a particular geographical region. An APM system may also include agents that are not process-aware, such as a designated legal agent responsible for ensuring the diligence of the contracts.

We remain agnostic in this example about the specific mechanism used to instrument the agents with process awareness (e.g., how their individual knowledge and goals are set and updated), as there may be different ways of framing it within an APM system. For the purposes of defining the essence of an APM system, however, it is important that APM agents incorporate some concrete realization of such a framing capability (see Section 3.1).

Considering the agent as a first-class citizen in an APM system, we define an agent as follows:

Definition 2.2. An *agent* in an APM system is a primary execution entity—an actor that perceives, reasons, and acts autonomously, with its autonomy framed to ensure process-aware behavior aimed at achieving process goals.

We may call an agent in an APM system an *APM agent*. It is important to note that once an agent becomes part of an APM system, its individual behavior and goals (i.e., micro-level work execution) are both framed and aligned. The former implies that the agent's behavior is directed at both complying with process constraints (the *frame*) and towards achieving the goals of the APM system processes (i.e., the macro level). The latter entails that its behavior is harmonized with that of the other agents in the system. While being process-aware is an inherent property of an APM agent, it is a qualitative property in the sense that different agents in an APM system may vary in their degree of process awareness and autonomy, and in the extent to which their behavior helps achieve the system's collective goals.

To realize this notion and equip an agent with the means to promote autonomy and process awareness in an APM system, we identify four essential capabilities the agent must possess: framing, explainability, conversational actionability, and self-modification. These capabilities, and the rationale for their inclusion, are further detailed in Section 3.

The actions of an agent are facilitated by the use of tools. We therefore define:

Definition 2.3. A *tool* in an APM system is a means accessible to an agent that augments its capacity to reason, and to perceive and act upon its environment.

Possible tools may include sensors, actuators, messaging, or other software functions and services. These tools may be accessed through communication protocols that define how information is exchanged.

In light of the presented concepts, next, we elaborate on the building blocks of an APM system and of an agent.

2.2. Conceptual architecture

This section introduces the conceptual architecture of an APM system. As visualized in Fig. 1, the architecture focuses on two distinct levels: the *APM system* representing the *macro level* management of work that establishes the process-awareness of its agents (Section 2.2.1), and the *APM agents*, i.e., the autonomous execution of work in the APM system on the *micro level* (Section 2.2.2).

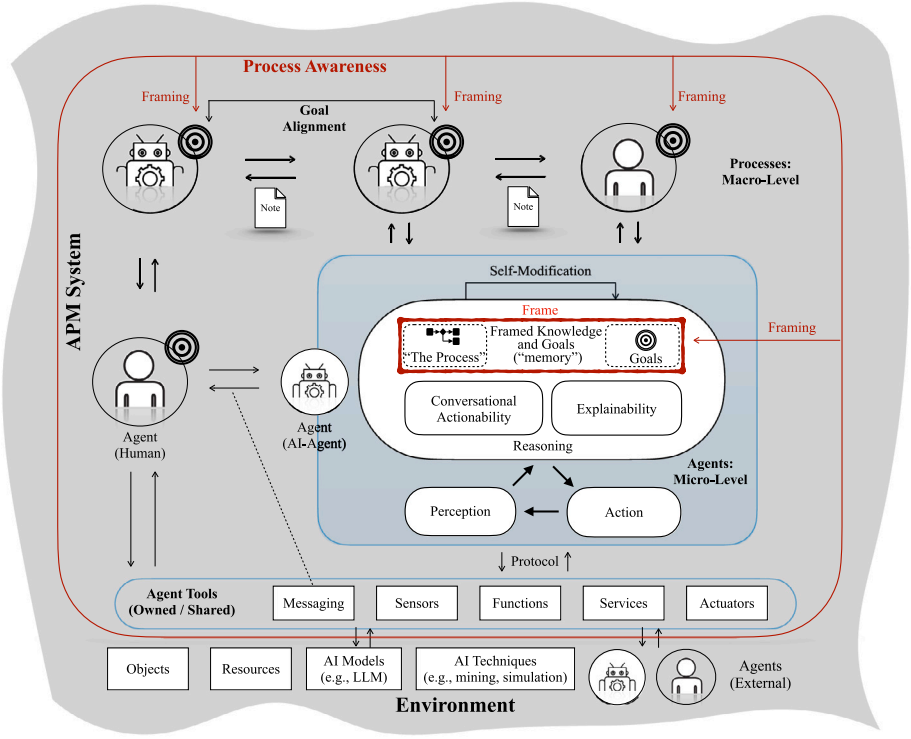


Fig. 1. Conceptual architecture of an APM system depicting its key actors, components (boxes), and interactions (arrows). *Process-awareness* (red-colored) is realized by a framing mechanism that constrains and aligns agents at the macro level, while at the micro level each process-aware agent runs a *Perceive–Reason–Act* loop over framed knowledge and goals (its internal view of "the process" and its goals), as agents interact with other process-aware human- and AI-agents in the system, and with the external environment (external agents, objects, resources, AI models/techniques), using tools (messaging, sensors, services, and actuators).

2.2.1. APM system

The APM system constitutes the "world" where agents live and act, and in which they can interact with each other and with the environment via tools (both individually owned and shared). These tools let the agents exchange data (e.g., *notes*, *messages*) with other agents in the APM system, employ a variety of AI models (e.g., LLMs) and techniques (e.g., simulation, mining), sense and manipulate external objects and resources, invoke remote functionalities via services, and interact with external agents that are not part of the APM system. To understand each other, agents rely on a shared language, ontology, or protocol [20]. Examples include Model Context Protocol (MCP), Agent Communication Protocol (ACP), Resource Description Framework (RDF), and Knowledge Query and Manipulation Language (KQML).

The *macro level* (process-aware management) represents the *processes* themselves, defined by collective, organization-level goals. The attainment of these goals is achieved by imposing process awareness and goal alignment on the agents through a *framing* mechanism.

Definition 2.4. Framing is a primary mechanism for ensuring process-awareness and goal alignment in an APM system, imposing restrictions on the autonomy of agents through their knowledge and goals.

Without framed autonomy, an agent could perform any action in pursuit of its own individual goals. It provides the normative and operational specifications that agents must adhere to, including the *lifecycle management* to create, suspend, or destroy agents as the process evolves. More concretely, the realization of the framing mechanism attends to two aspects:

Process-Awareness governs all agents' work towards common process objectives. Process awareness entails that an agent's autonomy is directed to ensure that its actions aim to fulfill the collective process goals.

Goal-Alignment sets the rules of engagement and structural relationships (e.g., hierarchies or coalitions), including the coordination of goals among agents that may work collaboratively. This also reflects the assignment of roles and segregation of duties throughout their lifecycle.

The functional manifestation of the framing mechanism is expressed through the capabilities of the agents (see next section), while its concrete realization is an engineering choice, since there are various ways in which it can be implemented—for example, by assigning collective responsibility to one or more agents, through orchestration, or via a shared memory (e.g., as part of the frame) that allows agents to progress along a mutual plan.

2.2.2. APM agent

The *micro level* (agent-oriented execution) consists of a collection of autonomous agents that jointly execute the process—and may participate in multiple processes.

Drawing on foundational ontologies like DOLCE [34] and GFO [35], agent-oriented modeling frameworks like *i** [36] and Tropos [37], and classic AI constructs such as BDI [38] and FIPA [39], as well as the mental model theory of reasoning [40], and in concert with the characteristics of an AI-Augmented BPM system [23], we consolidate here a unified specification schema for an APM agent.

As conceptualized in Fig. 1, an agent, from an architectural point of view, consists of three main conceptual modules [20]:

The **perception module** allows the agent to observe the state of its *environment*, *context* (situational constraints, location), and other agents. These agents operate within the agent's environment: some are part of the APM system (and thus process-aware), while others may be external to it. This module is responsible for managing the agent's *perception and sensing* attributes (e.g., sensors, percepts, belief update functions), ensuring the agent continuously updates its understanding of the world.

The **reasoning module** is the core component for knowledge representation and decision making. It processes sensory data and is continuously informed by the framing mechanism to maintain the

agent's *mental* (i.e., knowledge, perceptions, and beliefs) and *intentional* (i.e, desires, intentions, and goals) models. This also includes the agent's ability to self-modify, learn, and explain its reasoning. The agent's core knowledge includes its *identity* (i.e, agent ID, type), along with the agent's *behavioral dispositions and plans* (policies, triggers, reactions) to decide what to do next. This module also inherently provides a computational mechanism for decision-making, enabling the agent to continuously process its knowledge, revise and generate ongoing inferences, and translate its perceptions into actions. Recent advances in AI explore the use of LLMs for this purpose, although their true capability to fulfill this role remains under debate [41,42]. Several alternative approaches could support such a competence within an agent, including externalizing parts of the reasoning through specialized tools. Similar to the framing functionality, we intentionally leave the concrete realization of the decision-making component open.

The conceptual architecture introduces the following reasoning-related capabilities required for a generic agent to become *process-aware* within an APM system:

Framing (see 3.1). This module represents the internalization of the APM system's *framing*. Uniquely for an agent in an APM system, as a process-aware actor, the agent's knowledge includes two key internal elements: a mental model and an intentional model. The internal mental model serves to represent the process reality, viewed by the agent as *the process*, as perceived through the framing mechanism. It serves as the agent's "memory" regarding the process model, running instances, stakeholders, and all relevant process context. The internal intentional model maintains the agent's goals as they are shaped by its *social* and *normative* dispositions (i.e., obligations, roles, collective goals, and prohibitions). By explicitly consulting these frames, the reasoning component ensures that the agent's local constraints and objectives remain aligned with those of the overall process.

Explainability (see 3.2). Supported by the agent's *accountability* and *traceability* attributes (i.e., trace logs, auditability), *explainability* is the ability to articulate the rationale behind decisions. This ensures that process stakeholders can receive specific explanations of the workings of the socio-technical system.

Conversational Actionability (see 3.3). Building on the agent's *communication* and *interaction* attributes (i.e., protocols, roles, languages), this capability allows the agent to negotiate and coordinate with other agents and act on the environment. It also enables process stakeholders to trigger actions, either at design- or run-time, to change system behavior.

Self-Modification (see 3.4). Grounded in the agent's *perception* abilities, self-modification is an agent's capability to adapt and evolve over time. Agents self-modify to better reach individual goals and, when successfully *framed*, to achieve collective process goals.

Finally, the **action module** allows the agent to perform actions that change the state of the environment and send messages to other agents. It defines the agent's *capabilities* (i.e, skills, resources) and abilities to socially interact with other agents.

3. Envisioned capabilities of an APM system

This section gives an overview of the aforementioned key capabilities of an APM system: *framing*, *explainability*, *conversational actionability*, and *self-modification*, as initially proposed in [15–18].

3.1. Framing in an APM system

The guard-railing of an autonomous agent requires assuring it is operating within its current *frame*. Intuitively, a frame is a set of rules, restrictions, and regulations, which may evolve over time. Frames establish boundaries within which one or several agents in an APM system may operate with maximal flexibility, making autonomous decisions [43]. Frames may exist — at least — on *agent type*, *process*, and *organization* levels (as well as potentially across organizations).

More analytically, frames are *normative*: they specify deontic process requirements governing the behavior of the process. In contrast, classical process specification languages, such as BPMN and DECLARE, are primarily concerned with specifying behavior required to accomplish a business goal.[3] However, unlike informal definitions of business processes, e.g., as "sets of activities" performed to "jointly realize a business goal" [8, p. 5], goals remain largely implicit in these more formal and operationalizable process specification languages.

In previous works, operational specifications have been called *frames* as well [23]. Indeed, they can be considered a sort of *operational* frame. In APM systems "frames" focus, however, on the normative specification. When we need to distinguish, we call the two frames *normative frame* and *operational frame*, respectively. For example, an agent may be specified so that it must reject any student assignment submitted after the deadline. Such a requirement is considered a form of normative frame statement. An operational frame specification may complement this statement by instructing that, when rejecting an assignment, the agent shall: (1) retrieve the name and ID of the student who created the assignment, (2) record these details in the database, and (3) send the student an email informing them about the rejection. Alternatively, these three operational statements could be replaced by a single prohibition statement (i.e., a normative frame) not to delete the rejected assignment without informing the student who submitted it, leaving more room for autonomy in the behavior pursued by the agent, for example letting the agent decide on its own whether to first send an email to the student and then record the information in a database, or the other way around.

If there are no autonomous decision-makers, then the normative frame is just an additional condition over the operational frame; but if decision-making is possible, then the operational frame requires finding a strategy to satisfy the objective, whereas the normative frame requires choosing a strategy that remains within what is allowed (with respect to the frame).

Strategies for achieving goals under framed autonomy are associated with decision-makers, including software agents, giving rise to several problem setups, for centralized as well as distributed intelligence.

Centralized intelligence: We consider the "AI agents" as a single entity orchestrating the *process* that is executed in a mutually fully observable and coordinated manner. The *environment* may be stochastic and not fully observable. The frame is over the process. The single entity may have active or passive responsibility for the frame. If we have multiple agents, we may break down the problem into several of the above scenarios.

Distributed intelligence: We consider AI agents as distributed entities that enact the process as *resources*. This has wide-ranging implications: a resource may have only partial observability of what other resources are doing; coordination may be effortful, and resource-level

[3] BPMN is *imperative*, specifying — at least ostensibly — the exact control flow of activities to be executed, and is therefore well-suited for *operational* requirements. DECLARE , in contrast, is *declarative*, specifying constraints over permissible behavior. While this declarative nature aligns naturally with *normative* requirements, DECLARE is typically used to constrain execution rather than to explicitly capture deontic notions such as obligations, prohibitions, or permissions.

goals may be mutually inconsistent, or inconsistent with process-level goals. In such scenarios, we can frame individual resources, groups of resources, or the entire process. Accordingly, we need to assign responsibility to individual agents or groups thereof, and there may be strategic interactions affecting responsibility.

From these problem setups, we can derive three different blueprint scenarios for framed autonomy in business processes:

(i) we have a single decision-maker and place a frame on process behavior;
(ii) we have multiple decision-makers and place frames on individual decision-makers;
(iii) we have multiple decision-makers and place frame(s) on process behavior or parts thereof.

In practice, there may be additional variance to the scenarios. For example, normative frames may be partially represented within operational process specifications, restricting overall agent autonomy. An example is a purchasing process where purchase orders can only be created and paid through a central IT system that enforces normative rules, e.g., regarding four-eyes approval policies. Other parts of the global normative frame can potentially be projected to local agent-level norms. For example, overall spending limits may apply on the global level, but could be operationalized locally. Specifically, local operationalization of frames acts as a security and privacy boundary. Indeed, the frame limits an agent's perspective and capabilities to only its necessary process context, inherently restricting the potential impact if an agent is compromised or manipulated by a deliberately malicious actor.

3.2. Explainability in an APM system

As a prerequisite to effective execution and governance, an APM system should be explainable. Because an APM system is a composition of AI agents, this may be achieved by endowing agents with the intrinsic capability to explain their own behavior and actions, including those agents assigned collective process-awareness responsibilities. To this end, each agent's specification should include explicit instructions concerning the requirement to articulate the explanans (the explanation itself) in response to certain triggering situations or conditions (explanandum), as well as which explanation mechanisms — so-called *eXplainable AI* (XAI) techniques — may be employed.

Empowering agents in APM systems with explainability will help address important concerns, including the following:

Trust issues may arise among stakeholders — including process owners, business analysts, end users, and customers — who may hesitate to rely on agentic process recommendations or automated agent decisions when the underlying rationale is unclear.

Accountability for agent behavior requires that if an agent fails, it can explain the underlying rationale such that responsibility can be assigned and corrective actions can be implemented.

Biases may be perpetuated by AI and ML components underlying APM systems and their agents. Such biases may lead to discriminatory or unfair agent decisions; explainability is a prerequisite for detection and mitigation.

Compliance of agents' behavior with regulatory frameworks, such as the EU's GDPR and AI Act, needs to be demonstrable. This requires an increasing level of transparency, particularly in high-risk domains like finance, healthcare, and human resources, which are common areas for BPM applications.

Agents in APM systems make independent decisions, adapt to changing conditions, and learn from experience with minimal human intervention. Here, explainability offers a central mechanism through which agents can articulate the rationale behind their behavior. As such, explainability becomes a first-class citizen in the realization of APM systems, supporting agent autonomy from two perspectives: *(i)* enabling agents to independently resolve misalignment in other agents' behavior; *(ii)* reducing human intervention by making agent behavior understandable and transparent.

To account for the quality of explanations in APM systems (e.g., soundness, accuracy, usefulness, and interpretability), derived from situation-aware explainability [44–46] and causal processes [47,48], we postulate that the following are desirable properties about explainability in APM systems:

1. Ability to infer explanations that conform to the framing constraints and statements.
2. Ability to capture the richness of contextual information that affects agents' behavior and decisions.
3. Ability to reflect causal execution dependencies among actions in the agents' trajectories/behavior.
4. Ability to provide explanations that are interpretable to other agents (humans or digital).

3.3. Conversational actionability in an APM system

As the term suggests, conversational actionability refers to an agent's ability to combine interaction and enactment capabilities. This is essential in an APM system. On the one hand, agents need to integrate conversational capabilities to coordinate with one another and to receive instructions from, interact with, and report to the human agents involved in 'the process' in various roles (ranging from active participation to management). Knowing what the process is and who the agents assigned to relevant roles are may be considered part of an agent's mental model of the process. On the other hand, agents must be able to link such conversations to the corresponding enactment capabilities, making decisions and performing actions accordingly.

In [17], conversational actionability is positioned within AI-augmented BPM Systems as the ability of the system to handle these two requirements:

Process-aware conversation. The system can interact with users or external agents using a conversational interface to support, trigger, and guide actions related to the enactment of one or multiple processes.

Process-aware actionability. Conversations trigger business process executions, such as taking a decision or performing an action, as well as evolution actions, such as deciding on priorities or changes to the process.

When moving from a centralized execution system to an APM system, these capabilities must be collectively realized by the set of agents operating within the system. Instead of relying on centralized intelligence and a single point of truth, an APM system caters to federated perspectives. Each agent decides how it instantiates these two capabilities based on its own individually informed perspective about 'the process' (i.e., its frame). Hence, this may require the use of consensus resolution approaches or the ability of external agents to tolerate a possible variety among the results when interacting with different agents.

As for the conversation component, an agent may, in fact, interact with other agents using natural language and/or exploiting other unstructured and semi-structured information sources (such as diagrams and charts) or formal communication languages, such as multi-agent interaction protocols. Choosing which interaction modalities are embodied by the agent, manifested by the proper selection of tools in the

APM system, obviously also depends on whether an agent only interacts with other computational agents, and/or humans.

In terms of actionability, depending on the role played by the agent, it may need to cover one or more key functionalities (i.e., types of agent-enacted behavior; cf. [17]):

Query – to provide information on the process(es), either at the model level or regarding execution data pertaining to the current or past states of affairs;

Recommend – to provide insights and suggestions on the adaptation and future evolution of process instances;

Create – to elicit models from domain knowledge and process-relevant data;

Execute – triggering actions to move process instances to a new state.

In order to enact such behaviors, agents typically require interacting with tools providing different services or the same service with different functional and non-functional guarantees. For example, determining the likely time-to-completion for an order may be achieved using predictive monitoring techniques, employing simulation, or by a combination of the two. This creates the challenge of identifying the best mix of tools to realize an overall functionality in the "best way" possible (see Section 4.3).

3.4. Self-modification in an APM system

In order to adjust to ephemeral or permanent changes, APM systems must be able to self-modify. A fundamental distinction in self-modifying APM systems is between *adaptation* and *evolution* of both individual agents and agent coalitions. Adaptation and evolution fundamentally differ in scope, duration, permanence, and the type of knowledge they leverage.

Adaptation refers to short-term, instance-specific modifications [49,50] that address immediate, unforeseen issues during process execution. These modifications do not alter the underlying process model or schema, but rather constitute localized modifications performed by the agent within the constraints of the current process definition to handle exceptional circumstances or environmental changes that were not anticipated at design time. Adaptations are ephemeral—they affect only the current process instance and do not propagate to future executions unless explicitly learned and incorporated into the model (which then would be considered evolution). Such adaptations can be enacted individually by an agent, or they can be enacted collaboratively by engaging with other agents about how their shared perspective on the process may need to be adapted. Adaptations may be reactive (i.e., in response to perceived deviations or problems) or they may be proactive (i.e., performed based on predictions), see [18,51].

Evolution describes long-term modifications to the process logic, model, or policy that persist across multiple instances and executions. These longer-term modifications are typically shared among multiple agents in the APM system. Evolution may be informed by aggregated insights, patterns, and correlations discovered through analysis and learning from historical execution data, performance metrics, and repeated anomalies. Rather than responding to a single event, evolution synthesizes knowledge from multiple instances to identify systemic issues, improvement opportunities, or changing environmental conditions that warrant permanent changes to how the process is defined or executed. These modifications affect the process model itself and thereby influence its future instantiations, i.e., future processes [52]. Such evolution also entails the need to communicate and re-align how it is applied across all relevant agents in the system.

Ideally, the relationship between adaptation and evolution forms a feedback cycle in the APM system. Adaptations generate execution traces and performance data that feed into agents' learning mechanisms. When certain adaptive patterns prove consistently effective across multiple instances or contexts, they become candidates for evolutionary incorporation into the process model [53], and hence into the agent's frame. Conversely, evolution should reduce the frequency of certain types of adaptations by preemptively addressing known failure modes or inefficiencies. However, adaptations remain necessary for handling truly novel situations that have not been anticipated during design time [54] or not yet encountered frequently enough by any agent in the APM system to justify evolutionary modifications, or to address context-specific conditions that should not be generalized.

The distinction between adaptation and evolution has important implications for system design. Adaptation mechanisms must prioritize responsiveness, robustness, and security/safety under uncertainty, often operating with incomplete information and limited time for deliberation. Adaptation mechanisms require runtime monitoring, exception handling capabilities, and flexible execution engines that allow deviating from prescribed process execution paths. Evolution mechanisms, meanwhile, require sophisticated analytical capabilities: pattern mining across execution histories, causal inference to distinguish correlation from causation, statistical validation to ensure that observed patterns are not artifacts of noise or bias, and change management protocols to securely/safely deploy model modifications without disrupting ongoing operations.

4. Challenges

This section provides an overview of research challenges that require solving to further advance the relevance of APM. The challenges are categorized according to the four core capabilities of *framed autonomy*, *explainability*, *conversational actionability*, and *self-modification*. Analogously to the capabilities in the previous section, these challenges were identified during the Dagstuhl seminar and published as PMAI workshop papers at ECAI [15–18]. We augmented and refined these challenges based on the instrumental feedback received from the PMAI workshop participants.

In addition, we introduce a set of *cross-cutting* challenges that affect several of the aforementioned capabilities.

4.1. Challenges regarding framed autonomy

Achieving framed autonomy in APM systems comes with practical challenges, ranging from fundamental questions about the notion of an agent in business processes to specification and operationalization.

F1: What is a pragmatic notion of an agent in the context of business process execution? Before the broad adoption of LLMs, the notion of an agent did not play a major role in the engineering of business information systems and the processes that run them. Consequently, practitioners cannot be expected to be familiar with the depth and sophistication of agent-related abstractions and the presented framing mechanism for the realization of process awareness. To the contrary, a practitioner may consider a software tool that makes use of an LLM as an agent, without much thought about further properties. Defining a more precise and robust notion of an agent that is still intuitively understandable by business process management practitioners can thus be considered a key prerequisite for framing agents' autonomy.

F2: How to elicit and specify frames? The elicitation and specification of mental and intentional frames requires a *frame meta-model*, and one or several specification languages. To this end, existing specification languages can be reused; potentially, several languages and their underlying concepts can be combined. For example, declarative approaches to process specification — such as DECLARE [55] and in more practical contexts business rule and query languages with temporal reasoning capabilities [56] — can be augmented with deontic notions to promote normativity to a first-class abstraction. For elicitation, both symbolic

and subsymbolic approaches can be used and fused. LLMs can generate frames or parts thereof from natural language text, whereas rule mining approaches can be applied to infer normative constraints from the traces of well-behaved agents and multi-agent systems.

F3: How to operationalize frames on real-world symbolic data? Once specified, frames need to be integrated with APM systems to ensure the agents' frame-compliance during runtime. A short- to mid-term prerequisite is the operationalization of frames using technologies that do, in fact, run in large organizations. Here, explainability is a necessity, considering the practical intricacy of normative requirements, as well as the scale of real-world symbolic queries and data. To address this runtime compliance, formal logic-based approaches [57] can provide the imperative logic foundation needed to enforce the operational boundary of the agent. By leveraging strict transition semantics, such formalizations can naturally guardrail the agent's execution to legal actions and safely handle exceptions via guarded execution constructs.

F4: How to design incentive and reinforcement mechanisms for frame-compliant agents? A central challenge in APM is understanding how to design agents whose reasoning and behavior consistently align with their prescribed frame, meaning the norms, goals, constraints, and process logic meant to guide their actions. The question is not merely how to make such a frame available to agents, but how to ensure that agents actively use it, internalize it, and treat it as instrumentally valuable. This requires developing ways to define, measure, and strengthen an agent's degree of frame compliance as a continuous, learnable property. A prerequisite are incentive structures that cultivate an agent's "desire" to be process-aware. Humans can be motivated through monetary compensation, perks, recognition, or intrinsic cultural alignment. AI agents require a functional counterpart that reshapes their internal utility models. The open problem is how to design rewards inherently as part of the frame, that make adherence to the frame beneficial for the agent while still preserving autonomy. To support this, reinforcement mechanisms must be woven directly into the agentic lifecycle. Over time, agents must learn which components of the frame are genuinely useful, which elements constrain them productively, and which parts might be deprioritized without violating safety, governance, or organizational policies.

4.2. Challenges regarding explainability

Challenges with respect to the explainability in APM systems center around technically sound explanations that are easily understood and actionable, given their intended and actual explainees.

X1: How to specify or let agents learn other agents' preferences regarding explanations? A variety of elicitation mechanisms may be used in an APM system to effectively capture explanation preferences through various channels, whether explicitly declared upfront in the agentic specification, interactively elicited through dialogue, or implicitly inferred from user behavior.

Systems must accommodate both static preferences that remain consistent and those that dynamically adapt to changing contexts, while supporting the natural evolution of preferences as the explainee's, human or digital, understanding develops. Here, the challenge is to find the sweet spot between keeping explanations up to date and not confusing the explainee (i.e., limiting its autonomy). Finally, we have to think about when and how to provide full versus incremental explanation updates.

X2: How to leverage and extend existing XAI techniques by the agents to generate explanations? Employing state-of-the-art XAI techniques for APM systems has several limitations. This is because making sense of agent behavior goes beyond explaining the AI models used under the hood; it must also account for the broader context in which agents perform their actions, the knowledge they may have acquired over time, possible cause-and-effect relationships among their actions, and the constraints (i.e., framing) imposed on their behavior. We therefore need new and enhanced explainability techniques that help agents explain the outcomes of their behaviors and decisions more broadly.

X3: How may one articulate actionable explanations (e.g., to other agents) to preserve autonomy? Besides being context-sensitive, explanations should be actionable—indicating to the explainee which corrective or mitigating actions could be taken to alter the state of the explanandum, particularly without escalating the situation to any external agent. In this way, the explainee may be able to autonomously act upon the condition at hand. However, further work is needed to devise a systematic approach that enables the explainer to determine the most effective content for the explainee, to elicit such corrective action—taking into account both the explanandum and the behavioral intentions of the explainee.

X4: When to generate explanations (generation time) and how long to preserve them? With respect to APM system performance and memory efficiency, it is important to assess whether explanations should be generated proactively wherever feasible, or whether their generation should be deferred until necessary. A related question concerns when outdated explanations ought to be retired [58].

X5: How to generate causally sound explanations? Not every pair of action executions is causally dependent. To generate sound explanations, it is therefore important to distinguish spuriously correlated actions (e.g., those that are merely temporally sequential) from causally dependent ones (i.e., when one action directly triggers the execution of another) within an agent's behavior or across multiple agents.

4.3. Challenges regarding conversational actionability

Challenges in conversational actionability pertain to how agents can realize a virtual circle of action and interaction with human principals such as domain experts and other agents in order to enact the process faithfully and efficiently.

A1: How to engage in conversations with human principals and other agents? The first challenge pertains the *conversational* facet of an APM system. This contains two distinct aspects. The first concerns the need for agents to interact with human principals—typically using natural language or domain-specific languages (such as process diagrams, dashboards, and the like). As for natural language in general, there are only few studies focusing on the ability of foundation models to converse about processes, with many open challenges discussed in a dedicated vision paper [59], and some preliminary contributions [60,61]. This is even more challenging when natural language conversations are mixed with domain-specific knowledge sources.

The second aspect pertains to agent-to-agent interactions and data exchange, which can be carried out using a variety of different means, from specific interaction protocols and recently developed human-readable token encoding formats (e.g., TOON[4]), and formal languages to more flexible forms (e.g., based on higher-level, informal languages).

A2: How to exploit capabilities of process management tools and services? The second challenge pertains to the *actionability* facet of an APM system. Agents need to relate conversations to goals and, in turn, to actions. This imperative raises a twofold challenge.

On the one hand, the actions in question are not under the direct responsibility of the agents, but require them to interact with different *actuators*, such as: (1) ERP, CRM, or other systems of record to perform transactions; (2) collaboration tools, e.g., to trigger notifications to human actors; (3) physical actuators, e.g., IoT devices or robots; (4) process or task automation tools to update the state of a process or

[4] https://github.com/toon-format/spec

to trigger predefined automation scripts; (5) dedicated planners and schedulers to trigger sequences of steps to achieve a desired state.

On the other hand, in deciding which action should be selected, agents typically need to acquire relevant information about the organization and, in particular, the current state of affairs as part of their individual mental model about the process. This can only be done if the agents can effectively exploit tools and services encapsulating specific, well-understood, and highly trusted functionality related to data and process intelligence, covering all phases of the process lifecycle (from model-driven analysis and simulation to online data querying, process mining components, as well as predictive and prescriptive components). This calls for encapsulating such components as tools into semantically rich descriptions that can be used by agents, describing functional and non-functional requirements and enabling discovery (for example via MCP).

A3: How to compute and use key indicators on the overall behavior? A key challenge related to conversational actionability is how to compute and use key functional and non-functional process indicators obtained when gathering information and taking actions. Such indicators range from standard KPIs related to performance, time, and cost, to a larger set of indicators measuring/estimating trust, usability, uncertainty, flexibility, and alike (see [17]). A set of KPIs to assess completeness and correctness of process models when compared to corresponding textual descriptions have been proposed [62] and vice versa for generated text from process models in [63]. Moreover, for example, an agent may need to ponder the impact of taking an action and, in doing so, may invoke a simulator and/or a black-box predictor. Each of these components would provide different levels of trust, precision, and uncertainty, and depending on how they are used (e.g., selecting one of them or selecting both and then aggregating the obtained outputs), such levels would propagate up in completely different ways. This is essential to obtain end-to-end indicators regarding the overall process execution.

A4: How to balance autonomy, delegation, and control? The presence of multiple, interacting agents, operating using different tools and AI models with different levels of trust, performance, cost, etc, makes it even more challenging to determine how to balance autonomy and human oversight, and how to achieve the best trade-off between delegation (improving performance) and control (mitigating undesired outcomes and retaining human responsibility).

4.4. Challenges regarding self-modification

Below, we present the relevant challenges related to the capability of APM systems to adapt and evolve.

M1: How to govern self-modification? For the agents in an APM system to operate safely and effectively, governance and human oversight must be built into their design. A key question is when to refrain from autonomy and return control to a human agent, e.g., in scenarios where the autonomous agent has insufficient confidence and is at risk of making mistakes. Research in AI planning, runtime monitoring may help establish thresholds or confidence bounds beyond which a process must escalate to human decision-makers. Techniques from the ML literature on prediction with reject option [64] (sometimes called "learning to defer") can be explored as a solution direction.

Similarly, determining when and how a human agent should validate a proposed process or policy modification requires a framework for explainable adaptation, where the autonomous agents present justifications for their suggested modifications. Solution directions may include methods from the field of explainable AI (XAI) [65] (see 3.2), or justifications may be provided in alternative forms such as simulations of expected outcomes.

Another major issue is aligning the planning and goals of the AI agents in the system with those of its human agents (i.e., process stakeholders) [66]. Multi-objective optimization techniques [67] can assist in balancing trade-offs between performance, cost, compliance, and user satisfaction while maintaining logical constraints defined in the process model. However, optimizing across conflicting objectives remains a challenging problem, especially when human values or other non-quantifiable criteria are involved.

M2: How to evaluate the success or failure of modifications? Quality assurance in fully autonomous scenarios introduces its own challenges. Without humans routinely checking outcomes, the agents in the APM systems must develop internal mechanisms for evaluating whether their adaptations "worked". Here, ML-based anomaly detection, performance baselining, and causal reasoning can help detect regressions or misbehavior. Formal methods for verification and validation of modifying process logic or agent policies also present a relevant research direction here. Generative AI, particularly LLMs, could also be used for generating justifications or summaries of decisions for human audits or by providing labels for downstream evaluation—an emerging area sometimes called LLM-as-a-judge [68]. Solutions to the quality assurance problem could use human input purely for evaluation, even in scenarios where the execution is fully automated. However, human input is costly, and hence a challenge is to make this cost-efficient with maximal autonomy. Directions to be explored could include active testing [69] or online testing [70].

M3: How to align concurrent evolution and adaptation of linked process executions among process-aware agents? Business processes rarely exist in isolation. In practice, APM agents operate within ecosystems of interconnected process executions, shared resources, and distributed agent coalitions that may themselves be undergoing evolution and adaptation. For example, agents engaged in related processes may undergo similar adaptations and thus may be able to share "lessons learned" from these common adaptations, thereby making the overall modification more effective (e.g., see [71], where this is explored for self-adaptive software systems). A key challenge is detecting and managing these interdependencies in real-time. APM agents must be aware not only of their own mental and intentional states (i.e., processes and goals), but also of the adaptation state of related agents engaged in the same processes. This requires mechanisms for inter-process communication, coordination protocols, and potentially a meta-level orchestration agent that monitors system-wide adaptations. Techniques from distributed systems, such as consensus algorithms, distributed constraint optimization, and multi-agent coordination [72], may provide foundational approaches. However, these must be adapted to the dynamic, semi-autonomous nature of APM systems, where agents might not have complete visibility into other agents' internal states or future intentions.

M4: How to enable continuous learning and adaptation management? A defining feature of self-modifying agents in APM systems is their ability to learn from experience and improve process behavior over time. This may be achieved utilizing process mining techniques by the agents themselves as a self-reflection mechanism to analyze their own recorded behaviors (a.k.a., agent trajectories). To support this, the agents must continuously record the adaptations they make and assess their impact both locally (per agent) and globally (across agents). Capturing this meta-knowledge creates a feedback loop where successful modifications reinforce future decisions and ineffective ones are pruned. AI planning and reinforcement learning techniques (while balancing exploration and exploitation; e.g., see [50,73]) are promising approaches to structuring such learning loops, especially when context-aware and enriched by causal modeling of intervention outcomes.

However, scaling these techniques may pose challenges: Long-running processes or high-volume APM systems face constraints of bounded memory and context. The agents in the system cannot retain or process each event they have ever observed. Hence, the agents must learn to construct and maintain bounded knowledge representations: compressed summaries, predictive state abstractions, or fixed-size windows of relevant execution history. Techniques from stream reasoning,

process mining over sliding windows, and transformer-based sequence models may offer solutions. Designing APM agents that know which parts of their history to remember, forget, or query becomes a core challenge for sustainable, real-time, continuous learning.

M5: How to model and measure uncertainty in APM systems? Nondeterminism in agentic behavior may arise in APM systems due to *(i)* stochastic AI techniques and models (such as LLMs); *(ii)* stochastic and/or drifting contexts; *(iii)* the difficult-to-anticipate actions of human agents. The agents in APM systems thus must operate under both aleatoric (inherent randomness) and epistemic (lack of knowledge) uncertainty [74]. It is well-known from the literature on uncertainty quantification and active learning that accurate estimates of specifically *epistemic* uncertainty are a prerequisite to learn and adapt policies to new environments in a data-efficient way [74–76].

Techniques from probabilistic modeling, Bayesian inference, and fuzzy logic are key for representing and reasoning about agentic uncertainty in process executions. In some domains, thresholds or confidence levels may be set by human experts (qualitative), while in others, Bayesian models or statistical metrics (quantitative) provide actionable measures. As an example, reliability estimates for proactive adaptation may be derived from ensembles of prediction models [77]. APM agents must combine these forms of knowledge, learning from past executions when quantitative models are feasible, while falling back on qualitative heuristics when data is sparse or ambiguous. Hybrid methods that integrate fuzzy logic or ensemble learning could further improve uncertainty modeling in domains with imprecise information. Deciding when to defer the decision-making back to a human agent is a classic task where most solutions involve uncertainty quantification [64].

4.5. Cross-cutting and complementary challenges

C1: Toward automated provisioning and onboarding of legacy BPM into apm. A key challenge in this direction lies in provisioning and onboarding legacy process assets into an APM system. Organizations often operate long-standing BPM and workflow systems, implemented across diverse technologies and custom solutions. Many such systems require extensive manual effort for data handling, exception management, and cross-system orchestration, which in turn introduces operational inefficiencies and error risks. Moreover, critical business logic is frequently fragmented across workflow diagrams, scripts, rules, and tacit human practices, making structured extraction and reinterpretation difficult. Transitioning to APM therefore demands not only technical migration but also semantic and operational transformation, namely *agent-centric process mining*, including the reinterpretation of workflows into agent-oriented roles, goals, policies, and behaviors, while preserving institutional knowledge and ensuring continuity of governance and auditability. How this process can be aided, automated, deployed, and validated may keep the industry busy for years to come, as organizations seek reliable methodologies and tooling to safely and incrementally modernize their automation capabilities.

C2: Holistic security and privacy in agentic workflows. APM systems introduce a unique attack surface, requiring novel security models (mainly based on role-based access control). The capability of *Conversational Actionability* implies that agents may ingest malicious prompts (prompt injection) or interact with compromised external agents, potentially leading to data exfiltration or unauthorized actions. Simultaneously, *Self-Modification* capabilities introduce the risk of "poisoning" agents' memories [78] to ultimately cause them to adopt unsafe behavioral patterns. Furthermore, while *explainability* is required for trust, it presents a privacy paradox: detailed explanations of agent reasoning may inadvertently leak sensitive business data or personally identifiable information to unauthorized third parties. To prevent deliberately malicious actors from modifying workflows, APM systems must adopt advanced defense mechanisms. Technical and semantic guardrails (such as gateways and strict input/output sanitization) can intercept malicious instructions, and more advanced architectural design patterns such as "Action-Selector" or "Plan-Then-Execute" can constrain agents to predefined API calls, preventing them from executing unauthorized workflow modifications [79]. Moreover, process mining techniques can be repurposed as real-time security monitors to automatically detect potentially malicious deviations in process execution [80]. However, genAI agent security is an emerging and evolving area of research (cf. [81] for a taxonomic overview) and, accordingly, APM systems face — just like any other systems that deploy genAI agents — many security challenges that remain unsolved. An example of an open challenge is "indirect prompt injection" [82], where malicious instructions are hidden within external data or documents the agent is tasked to process. Thus, a major cross-cutting challenge is the development of security frameworks that can govern the trade-off between agent autonomy and information security. As overly rigid guard-railing limits adaptive capabilities, it is important to find a balance that ensures agents can negotiate and adapt without compromising the integrity and confidentiality of the process-aware environment.

C3: Benchmarking and evaluation frameworks for APM. While individual capabilities of AI models (e.g., LLM reasoning) are frequently benchmarked, there is a distinct lack of holistic evaluation frameworks for agentic systems in a process management context. In addition to well-established BPM characteristics, such as time, cost, and quality, we require novel metrics to assess how well the agents in APM systems deliver their capabilities (see Section 3). While this also calls for novel benchmark datasets to evaluate how well agents adhere to these qualities in stochastic environments, the benchmark datasets may be compromised by *data pollution* (aka. *data contamination* or *benchmark contamination*) [83,84]. This occurs when an AI model, used as a tool in the APM system, has been accidentally or intentionally exposed to the benchmark data during its training or fine-tuning phase. This has several negative consequences: *(a)* The AI model's performance will be artificially inflated, giving a false sense of its capabilities; *(b)* the purpose of a benchmark is to objectively measure performance on unseen data, yet contamination undermines this, making it difficult to reliably perform comparisons; *(c)* the AI agents employing such a tool that excels on a contaminated benchmark are likely to perform poorly in real-world applications where they encounters truly novel data. These are very important challenges to overcome to move APM from theoretical viability to industrial adoption, allowing organizations to quantify the return on investment (ROI) and risk profile of deploying autonomous workforce agents.

C4: Liability and accountability. The shift from human-centric to agentic process execution creates a "responsibility gap" that cuts across legal and organizational dimensions. When the agents in an APM system operate within *framed autonomy*, determining liability for adverse outcomes becomes complex: is the error attributable to the software developer, the organization enforcing the regulations (frame) onto the APM system, the specific AI model provider, or the emergent behavior of the multi-agent system? This challenge is exacerbated by *self-modification*, where an agent's behavior may drift significantly from its initial design. Research must align APM system architectures with emerging legal frameworks (such as the EU AI Act) to establish clear chains of accountability. This involves not only technical traceability but also the development of "social" contracts between human and digital agents, defining the boundaries of delegation and the precise conditions under which human intervention is legally and operationally mandatory.

C5: Methods for engineering APM systems. One complementary challenge is to build on the aforementioned challenges for establishing essential architectural and technical building blocks of agentic process management and deliver novel *analysis and design methods* to support

the engineering of APM systems.[5] Such engineering methods may build on agent-oriented analysis and design methods, such as AAII, Gaia, Tropos, or Prometheus [20].

5. Conclusions and outlook

Our work extends the BPM community's well-established tradition of writing manifestos. Notably, the *Process Mining Manifesto* [85] has stood the test of time as a rallying call for the community, facilitating the establishment of a data science tradition within BPM research and ultimately anticipating the emergence of process mining offerings as important components in large ERP vendors' software suites. In the realm of BPM and AI, the community has developed several conceptual visions. Pre-dating widespread LLM adoption and new-generation AI agents, the manifesto on *AI-augmented Business Process Management Systems* (ABPMS) focuses on the integration of AI capabilities into holistic process management and execution systems [23], introducing the vision of framing in BPM while leaving the agentic perspective *implicit*. As a complementary viewpoint, the *augmented process execution* proposal is intelligence/analytics-oriented and provides a four-level pyramid, ranging from basic *descriptive* analytics via *prediction* and *prescription* to augmentation [86]. Paralleling the emergence of LLM-augmented process management software, the *Large Process Models* vision sketches and discusses AI-based BPM in the age of LLMs, highlighting LLM potential to facilitate some tasks, while also emphasizing that achieving self-improving processes (essentially in the sense of an ABPMS) requires, even in human-in-the-loop scenarios, more than current-state LLMs [59].

Overall, the following four vision statements have anticipated at least some ideas that are largely accepted in practice:

1. Intelligent business process execution requires the collaboration of autonomous software agents and humans;
2. These agents require symbolic *frames*;
3. While LLM-based process modeling and mining capabilities play a role in mainstream BPM software offerings, LLMs alone are insufficient for truly intelligent BPM systems.
4. More broadly, LLM-based technologies are driving the *autonomous enterprise* era through novel agentic AI platforms, making it crucial to examine their technological implications for agentic BPM platforms as a central backbone.

Continuing this tradition, we hope that our manifesto will both anticipate developments in BPM theory and practice and help researchers and practitioners embrace recent technological developments while planning for long-term sustainable impact on the discipline and the organizations that practice it. In this context, we highlight the interdisciplinary nature of APM research as a bridge between BPM, autonomous agents, and machine learning. This space is being significantly reshaped by the recent trend of LLM-based agents, often portrayed in industry as a silver bullet, an innovation wave that the BPM community should actively engage with.

Parts of the APM vision can be practically realized by applying recent advances in research areas such as declarative process specification and conversational process modeling and mining. Still, to fully realize APM systems, this manifesto should primarily be interpreted as a call to action for further research. Notably, we intentionally leave open the realization and implementation details of key functionalities such as process-awareness and framing, an agent's framed knowledge and goals representation, as well as the aforementioned agent capabilities for explainability, conversational actionability, and self-modification. We invite diverse concrete implementations of the presented vision and conceptual architecture.

Future work could address challenges along (but not limited to) the following lines:

Establish formal abstractions for frames and goals. To better support framed autonomy in BPM, we suggest to *(i)* introduce first-class abstractions for mental models of processes, goals and normative frames; *(ii)* develop and evaluate algorithms for synthesizing provably frame-compliant and performant operational specifications from frames, goals, and environmental information; *(iii)* demonstrate the applicability of the abstractions and algorithms in the context of real-world business information systems.

Ensure explainability of *execution* and *management*. Explainability for APM systems requires capabilities of technically sound explanations on process, agent, and MAS levels that are dynamically adjusted such that they are interpretable for the potentially heterogeneous group of target agents. The explanations should be actionable on both instance (execution) and process (management) levels as a means to maximize agentic autonomy.

Ensure actionability of conversational approaches. Ensuring that agents behave faithfully and efficiently with respect to process goals requires research on how software agents can converse with human principals and other software agents in a process context, as well as with tools and traditional process-aware systems, how to assess agent behavior, and how to trade off autonomy and control.

Move self-adapting agents towards enterprise reality. The deployment of concurrently evolving, process-aware, and self-adaptive agents in real-world enterprises requires new research on the governance of self-adaptation and systematic frameworks for the comprehensive assessment of the consequences of self-adaptation.

In this context, we advocate for research that *(i)* integrates perspectives from both BPM and agents research; *(ii)* advances strong formal foundations that give principle-based guarantees, *(iii)* develops engineering results based on re-usable software artifacts and reproducible empirical evaluations grounded in systematic benchmarks; *(iv)* ultimately studies the value provided to businesses and society at large. Finally, we emphasize that a sustainable roadmap towards APM requires considering *agent* notions that encompass not only LLM-based "genAI agents" but also include human agents that remain crucial for organizational and societal success.

CRediT authorship contribution statement

Diego Calvanese: Writing – review & editing, Writing – original draft. **Angelo Casciani:** Writing – review & editing, Writing – original draft. **Giuseppe De Giacomo:** Writing – review & editing, Writing – original draft. **Marlon Dumas:** Writing – review & editing, Writing – original draft. **Fabiana Fournier:** Writing – review & editing, Writing – original draft. **Timotheus Kampik:** Writing – review & editing, Writing – original draft. **Emanuele La Malfa:** Writing – review & editing, Writing – original draft. **Lior Limonad:** Writing – review & editing, Writing – original draft. **Andrea Marrella:** Writing – review & editing, Writing – original draft. **Andreas Metzger:** Writing – review & editing, Writing – original draft. **Marco Montali:** Writing – review & editing, Writing – original draft. **Daniel Amyot:** Writing – review & editing. **Peter Fettke:** Writing – review & editing. **Artem Polyvyanyy:** Writing – review & editing. **Stefanie Rinderle-Ma:** Writing – review & editing. **Sebastian Sardiña:** Writing – review & editing. **Niek Tax:** Writing – review & editing. **Barbara Weber:** Writing – review & editing.

[5] e.g., see the CAiSE 2026 workshop EPAIS.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Timotheus Kampik reports financial support was provided by Wallenberg AI, Autonomous Systems and Software Program (WASP), Knut and Alice Wallenberg Foundation. 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

The article was derived from working group notes as published in the report of Dagstuhl Seminar 25192, "AUTOBIZ: Pushing the Boundaries of AI-Driven Process Execution and Adaptation"; these notes were further refined into several peer-reviewed workshop papers for the PMAI workshop at ECAI 2025 [15–18], which form the basis of Sections 3 and 4 of this paper.

AC conducted this research while enrolled in the Italian National Doctorate in Artificial Intelligence run by Sapienza University of Rome, Italy (CUP B53C23003500006, Mission 4 NRRP Generic Research Scholarship, Component 1). TK was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation, Sweden.

Data availability

No data was used for the research described in the article.

References

[1] M. Wooldridge, N.R. Jennings, Intelligent agents: Theory and practice, Knowl. Eng. Rev. 10 (2) (1995) 115–152, http://dx.doi.org/10.1017/S0269888900008122.

[2] Y. Shen, X. Zhang, The impact of artificial intelligence on employment: the role of virtual agglomeration, Humanit. Soc. Sci. Commun. 11 (1) (2024) 1–14, http://dx.doi.org/10.1057/s41599-024-02647-9.

[3] J. He, C. Treude, D. Lo, LLM-based multi-agent systems for software engineering: Literature review, vision, and the road ahead, ACM Trans. Softw. Eng. Methodol. 34 (5) (2025) 124, http://dx.doi.org/10.1145/3712003.

[4] T. Warren, Microsoft's plan to fix the web with AI has already hit an embarrassing security flaw, 2025, URL https://www.theverge.com/news/719617/microsoft-nlweb-security-flaw-agentic-web, (Accessed 15 November 2025), The Verge.

[5] T. Kampik, A. Mansour, O. Boissier, S. Kirrane, J. Padget, T.R. Payne, M.P. Singh, V. Tamma, A. Zimmermann, Governance of autonomous agents on the web: Challenges and opportunities, ACM Trans. Internet Technol. 22 (4) (2022) 104, http://dx.doi.org/10.1145/3507910.

[6] D. Gabbay, J. Horty, X. Parent, R. Van der Meyden, L. van der Torre, et al., Handbook of Deontic Logic and Normative Systems, Volume 2, College Publications, 2021, URL https://www.collegepublications.co.uk/handbooks/?00005.

[7] Y. Shavit, S. Agarwal, M. Brundage, S. Adler, C. O'Keefe, R. Campbell, T. Lee, P. Mishkin, T. Eloundou, A. Hickey, K. Slama, L. Ahmad, P. McMillan, A. Beutel, A. Passos, D.G. Robinson, Practices for governing agentic AI systems, 2023, URL https://cdn.openai.com/papers/practices-for-governing-agentic-ai-systems.pdf, (Accessed: 30 September 2025), OpenAI white paper.

[8] M. Weske, Business Process Management: Concepts, Languages, Architectures, Springer, 2024, http://dx.doi.org/10.1007/978-3-662-69518-0.

[9] M. Dumas, M. La Rosa, J. Mendling, H.A. Reijers, Fundamentals of Business Process Management, Springer, 2018, http://dx.doi.org/10.1007/978-3-662-56509-4.

[10] H. Vu, N. Klievtsova, H. Leopold, S. Rinderle-Ma, T. Kampik, Agentic business process management: Practitioner perspectives on agent governance in business processes, in: Business Process Management: Responsible BPM Forum, Process Technology Forum, Educators Forum. BPM 2025, in: LNBIP, vol. 565, Springer, 2026, pp. 29–43, http://dx.doi.org/10.1007/978-3-032-02936-2_3.

[11] G. Decker, Design and Analysis of Process Choreographies (Ph.D. thesis), Universität Potsdam, Germany, 2009, URL https://nbn-resolving.org/urn:nbn:de:kobv:517-opus-40761.

[12] A. Tour, A. Polyvyanyy, A.A. Kalenkova, A. Senderovich, Agent Miner: An algorithm for discovering agent systems from event data, in: Business Process Management, Springer, 2023, pp. 284–302, http://dx.doi.org/10.1007/978-3-031-41620-0_17.

[13] A. Tour, A. Polyvyanyy, A.A. Kalenkova, Agent system mining: Vision, benefits, and challenges, IEEE Access 9 (2021) 99480–99494, http://dx.doi.org/10.1109/ACCESS.2021.3095464.

[14] F. Fournier, L. Limonad, Y. David, Agentic process observability: Discovering behavioral variability, in: Proceedings of the Process Management in the AI Era, PMAI 2025, in: CEUR Workshop Proceedings, vol. 4087, CEUR-WS.org, 2025, p. paper3, URL https://ceur-ws.org/Vol-4087.

[15] P. Fettke, F. Fournier, L. Limonad, A. Metzger, S. Rinderle-Ma, B. Weber, XABPs: Towards explainable autonomous business processes, in: Proceedings of the Process Management in the AI Era (PMAI 2025), in: CEUR Workshop Proceedings, vol. 4087, CEUR-WS.org, 2025, p. paper2, URL https://ceur-ws.org/Vol-4087.

[16] D. Calvanese, G. De Giacomo, T. Kampik, Y. Lesperance, A. Marrella, A. Matta, et al., Autonomy in business process execution: Why we need first-class abstractions for goals and normative frames, in: Proceedings of the Process Management in the AI Era, PMAI 2025, in: CEUR Workshop Proceedings, vol. 4087, CEUR-WS.org, 2025, p. paper6, URL https://ceur-ws.org/Vol-4087.

[17] M. Montali, M. Comuzzi, I. Teinemaa, D. Amyot, M. Dumas, Towards conversational actionability in AI-augmented business process management systems, in: Proceedings of the Process Management in the AI Era, PMAI 2025, in: CEUR Workshop Proceedings, vol. 4087, CEUR-WS.org, 2025, p. paper8, URL https://ceur-ws.org/Vol-4087.

[18] A. Elyasaf, A. Metzger, S. Sardina, A. Senderovich, E.S. Asensio, N. Tax, Toward self-modifying autonomous business process systems, in: Proceedings of the Process Management in the AI Era, PMAI 2025, in: CEUR Workshop Proceedings, vol. 4087, CEUR-WS.org, 2025, p. paper1, URL https://ceur-ws.org/Vol-4087.

[19] R. Sapkota, K.I. Roumeliotis, M. Karkee, AI agents vs. Agentic AI: a conceptual taxonomy, applications and challenges, 2025, CoRR, arXiv:2505.10468.

[20] M. Wooldridge, An Introduction to Multiagent Systems, second ed., John wiley & sons, 2009.

[21] D. Amalfitano, A. Metzger, M. Autili, T. Fulcini, T. Hey, J. Keim, P. Pelliccione, V. Scotti, A. Koziolek, R. Mirandola, A. Vogelsang, A research roadmap for augmenting software engineering processes and software products with generative AI, 2025, arXiv:2510.26275.

[22] N.R. Jennings, On agent-based software engineering, Artificial Intelligence 117 (2) (2000) 277–296, http://dx.doi.org/10.1016/S0004-3702(99)00107-1.

[23] M. Dumas, F. Fournier, L. Limonad, A. Marrella, M. Montali, J.-R. Rehse, R. Accorsi, D. Calvanese, G. De Giacomo, D. Fahland, et al., AI-augmented business process management systems: a research manifesto, ACM Trans. Manag. Inf. Syst. 14 (1) (2023) 1–19, http://dx.doi.org/10.1145/3576047.

[24] P. Fettke, H.-G. Fill, J. Köpcke, LLM, LAM, LxM agent: From talking to acting machines, Enterp. Model. Inf. Syst. Archit. (EMISAJ) 20 (2025) http://dx.doi.org/10.18417/emisa.20.3.

[25] M. Wooldridge, N.R. Jennings, D. Kinny, A methodology for agent-oriented analysis and design, in: Proceedings of the Third Annual Conference on Autonomous Agents, 1999, pp. 69–76, http://dx.doi.org/10.1145/301136.301165.

[26] S. Russell, P. Norvig, Artificial Intelligence: A Modern Approach (4th Edition), Pearson, 2020, URL http://aima.cs.berkeley.edu/.

[27] H. Li, H. Zhang, A.E. Hassan, The rise of AI teammates in software engineering (SE) 3.0: How autonomous coding agents are reshaping software engineering, 2025, CoRR, arXiv:2507.15003.

[28] I. Weber, Large language models as software components: A taxonomy for LLM-integrated applications, 2024, arXiv:2406.10300.

[29] A.E. Hassan, D. Lin, G.K. Rajbahadur, K. Gallaba, F.R. Côgo, B. Chen, H. Zhang, K. Thangarajah, G.A. Oliva, J.J. Lin, W.M. Abdullah, Z.M.J. Jiang, Rethinking software engineering in the era of foundation models: A curated catalogue of challenges in the development of trustworthy FMware, in: M. d'Amorim (Ed.), Companion Proceedings of the 32nd ACM International Conference on the Foundations of Software Engineering, FSE 2024, ACM, 2024, pp. 294–305, http://dx.doi.org/10.1145/3663529.3663849.

[30] I. Gabriel, G. Keeling, A. Manzini, J. Evans, We need a new ethics for a world of AI agents, Nature 644 (2025) URL https://www.nature.com/articles/d41586-025-02454-5.

[31] J. Cassell, Embodied conversational agents: representation and intelligence in user interfaces, AI Mag. 22 (4) (2001) 67–83, http://dx.doi.org/10.1609/aimag.v22i4.1593.

[32] T. Guo, X. Chen, Y. Wang, R. Chang, S. Pei, N.V. Chawla, O. Wiest, X. Zhang, Large language model based multi-agents: A survey of progress and challenges, in: Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI 2024, Jeju, South Korea, August 3-9, 2024, ijcai.org, 2024, pp. 8048–8057, URL https://www.ijcai.org/proceedings/2024/890.

[33] K. Tran, D. Dao, M. Nguyen, Q. Pham, B. O'Sullivan, H.D. Nguyen, Multi-agent collaboration mechanisms: A survey of LLMs, 2025, CoRR, arXiv:2501.06322.

[34] S. Borgo, R. Ferrario, A. Gangemi, N. Guarino, C. Masolo, D. Porello, E.M. Sanfilippo, L. Vieu, DOLCE: a descriptive ontology for linguistic and cognitive engineering, Appl. Ontol. 17 (1) (2022) 45–69, http://dx.doi.org/10.3233/AO-210259.

[35] F. Loebe, P. Burek, H. Herre, GFO: the general formal ontology, Appl. Ontol. 17 (1) (2022) 71–106, http://dx.doi.org/10.3233/AO-220264.

[36] X. Franch, J.C. Sampaio Leite, G. Mussbacher, J. Mylopoulos, A. Perini (Eds.), Social Modeling Using the i* Framework, Springer International Publishing, Cham, Switzerland, 2024, http://dx.doi.org/10.1007/978-3-031-72107-6.

[37] F. Giunchiglia, J. Mylopoulos, A. Perini, The Tropos software development methodology: processes, models and diagrams, in: The First International Joint Conference on Autonomous Agents & Multiagent Systems, AAMAS 2002, ACM, 2002, pp. 35–36, http://dx.doi.org/10.1145/544741.544748.

[38] A.S. Rao, M.P. Georgeff, BDI agents: From theory to practice, in: V.R. Lesser, L. Gasser (Eds.), Proceedings of the First International Conference on Multiagent Systems, June 12-14, 1995, San Francisco, California, USA, The MIT Press, 1995, pp. 312–319, URL https://cdn.aaai.org/ICMAS/1995/ICMAS95-042.pdf.

[39] S. Poslad, P. Charlton, Standardizing agent interoperability: the FIPA approach, in: Mutli-Agents Systems and Applications, Springer-Verlag, 2001, pp. 98–117, http://dx.doi.org/10.1007/3-540-47745-4_5.

[40] P.N. Johnson-Laird, R.M.J. Byrne, Deduction, Psychology Press, 1991.

[41] Y. Saxena, S. Chopra, A.M. Tripathi, Evaluating consistency and reasoning capabilities of large language models, in: 2024 Second International Conference on Data Science and Information System, ICDSIS, 2024, pp. 1–5, http://dx.doi.org/10.1109/ICDSIS61070.2024.10594233.

[42] I. Khalid, A.M. Nourollah, S. Schockaert, Large language and reasoning models are shallow disjunctive reasoners, in: W. Che, J. Nabende, E. Shutova, M.T. Pilehvar (Eds.), Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Vienna, Austria, 2025, pp. 8843–8869, http://dx.doi.org/10.18653/v1/2025.acl-long.433.

[43] G. Acitelli, A. Alman, F.M. Maggi, A. Marrella, Achieving framed autonomy in AI-augmented business process management systems through automated planning, Inf. Syst. 133 (2025) 102573, http://dx.doi.org/10.1016/j.is.2025.102573.

[44] G. Amit, F. Fournier, S. Gur, L. Limonad, Model-informed LIME extension for business process explainability, in: PMAI@IJCAI'22, in: CEUR Workshop Proceedings, vol. 3310, CEUR-WS.org, 2022, p. paper1, URL https://ceur-ws.org/Vol-3310.

[45] G. Amit, F. Fournier, L. Limonad, I. Skarbovsky, Situation-aware explainability for business processes enabled by complex events, in: BPM-W 2022, in: LNBIP, vol. 460, Springer, 2022, pp. 45–57, http://dx.doi.org/10.1007/978-3-031-25383-6_5.

[46] D. Fahland, F. Fournier, L. Limonad, I. Skarbovsky, A.J.E. Swevels, How well can large language models explain business processes as perceived by users? Data Knowl. Eng. 157 (2025) 102416, http://dx.doi.org/10.1016/j.datak.2025.102416.

[47] F. Fournier, L. Limonad, I. Skarbovsky, Y. David, The WHY in business processes: Discovery of causal execution dependencies, Künstliche Intell. 39 (2025) 197–219, http://dx.doi.org/10.1007/s13218-024-00883-4.

[48] Y. David, F. Fournier, L. Limonad, I. Skarbovsky, The WHY in business processes: Unification of causal process models, in: Business Process Management Forum, Springer Nature Switzerland, 2026, pp. 40–57, http://dx.doi.org/10.1007/978-3-032-02929-4_3.

[49] D. Weyns, An introduction to self-adaptive systems: A contemporary software engineering perspective, John Wiley & Sons, 2020, http://dx.doi.org/10.1002/9781119574910.

[50] A. Palm, A. Metzger, K. Pohl, Online reinforcement learning for self-adaptive information systems, in: S. Dustdar, E. Yu, C. Salinesi, D. Rieu, V. Pant (Eds.), Advanced Information Systems Engineering - 32nd International Conference, CAiSE 2020, in: LNCS, 12127, Springer, 2020, pp. 169–184, http://dx.doi.org/10.1007/978-3-030-49435-3_11.

[51] A. Metzger, T. Kley, A. Rothweiler, K. Pohl, Automatically reconciling the trade-off between prediction accuracy and earliness in prescriptive business process monitoring, Inf. Syst. 118 (2023) 102254, http://dx.doi.org/10.1016/J.IS.2023.102254.

[52] S. Rinderle, M. Reichert, P. Dadam, Correctness criteria for dynamic changes in workflow systems - a survey, Data Knowl. Eng. 50 (1) (2004) 9–34, http://dx.doi.org/10.1016/J.DATAK.2004.01.002.

[53] B. Weber, M. Reichert, S. Rinderle-Ma, W. Wild, Providing integrated life cycle support in process-aware information systems, Int. J. Coop. Inf. Syst. 18 (1) (2009) 115–165, http://dx.doi.org/10.1142/S0218843009001999.

[54] A. Marron, L. Limonad, S. Pollack, D. Harel, Expecting the unexpected: developing autonomous-system design principles for reacting to unpredicted events and conditions, in: Proceedings of the IEEE/ACM 15th International Symposium on Software Engineering for Adaptive and Self-Managing Systems, SEAMS '20, ACM New York, USA, 2020, pp. 167–173, http://dx.doi.org/10.1145/3387939.3391607.

[55] C. Di Ciccio, M. Montali, Declarative process specifications: Reasoning, discovery, monitoring, in: W.M.P. van der Aalst, J. Carmona (Eds.), Process Mining Handbook, in: LNBIP, vol. 448, Springer, Cham, 2022, pp. 108–152, http://dx.doi.org/10.1007/978-3-031-08848-3_4.

[56] T. Kampik, C. Okulmus, Expressive power and complexity results for SIGNAL, an industry-scale process query language, in: BPM 2024 Forum, in: LNBIP, 526, Springer, 2024, pp. 3–19, http://dx.doi.org/10.1007/978-3-031-70418-5_1.

[57] A. Casciani, S. Agostinelli, Y. Lespérance, A. Marrella, S. Sardiña, Formal semantics for knowledge representation and automated reasoning in BPMN process models, Inf. Syst. 140 (2026) 102718, http://dx.doi.org/10.1016/j.is.2026.102718.

[58] N. Mehdiyev, P. Fettke, Explainable artificial intelligence for process mining: A general overview and application of a novel local explanation approach for predictive process monitoring, in: Interpretable AI: A Perspective of Granular Computing, Springer, 2021, pp. 1–18, http://dx.doi.org/10.1007/978-3-030-64949-4_1.

[59] T. Kampik, C. Warmuth, A. Rebmann, R. Agam, L. Egger, A. Gerber, J. Hoffart, J. Kolk, P. Herzig, G. Decker, H. van der Aa, A. Polyvyanyy, S. Rinderle-Ma, I. Weber, M. Weidlich, Large process models: A vision for business process management in the age of generative AI, KI - Künstliche Intell. 39 (2024) 81–95, http://dx.doi.org/10.1007/s13218-024-00863-8.

[60] L.F. Das Neves, C. Zerva, A. Gianola, A proposal for handling query ambiguity for process mining tasks, in: Proceedings of the 7th International Workshop on Artificial Intelligence and Formal Verification, Logic, Automata, and Synthesis (OVERLAY 2025), CEUR, 2025, pp. 107–115.

[61] M.L. Bernardi, A. Casciani, M. Cimitile, A. Marrella, Conversing with business process-aware large language models: the BPLLM framework, J. Intell. Inf. Syst. 62 (6) (2024) 1607–1629, http://dx.doi.org/10.1007/S10844-024-00898-1.

[62] N. Klievtsova, J. Benzin, T. Kampik, J. Mangler, S. Rinderle-Ma, Conversational process modelling: State of the art, applications, and implications in practice, in: C.D. Francescomarino, A. Burattin, C. Janiesch, S. Sadiq (Eds.), Business Process Management Forum - BPM 2023 Forum, Utrecht, the Netherlands, September 11-15, 2023, Proceedings, in: Lecture Notes in Business Information Processing, 490, Springer, 2023, pp. 319–336, http://dx.doi.org/10.1007/978-3-031-41623-1_19.

[63] N. Klievtsova, J. Mangler, T. Kampik, S. Rinderle-Ma, Utilizing process models in the requirements engineering process through Model2Text transformation, in: G. Liebel, I. Hadar, P. Spoletini (Eds.), 32nd IEEE International Requirements Engineering Conference, RE 2024, Reykjavik, Iceland, June 24-28, 2024, IEEE, 2024, pp. 205–217, http://dx.doi.org/10.1109/RE59067.2024.00028.

[64] K. Hendrickx, L. Perini, D. Van der Plas, W. Meert, J. Davis, Machine learning with a reject option: A survey, Mach. Learn. 113 (5) (2024) 3073–3110, http://dx.doi.org/10.1007/s10994-024-06534-x.

[65] R. Dwivedi, D. Dave, H. Naik, S. Singhal, R. Omer, P. Patel, B. Qian, Z. Wen, T. Shah, G. Morgan, et al., Explainable AI (XAI): Core ideas, techniques, and solutions, ACM Comput. Surv. 55 (9) (2023) 1–33, http://dx.doi.org/10.1145/3561048.

[66] S. Sreedharan, A. Kulkarni, S. Kambhampati, Explainable human-AI interaction, Synthesis lectures on artificial intelligence and machine learning, Springer International Publishing, Cham, Switzerland, 2022, http://dx.doi.org/10.1007/978-3-031-03767-2.

[67] R.T. Marler, J.S. Arora, Survey of multi-objective optimization methods for engineering, Struct. Multidiscip. Optim. 26 (2004) 369–395, http://dx.doi.org/10.1007/s00158-003-0368-6.

[68] L. Zheng, W.-L. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. Xing, et al., Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena, Adv. Neural Inf. Process. Syst. 36 (2023) 46595–46623, http://dx.doi.org/10.48550/arXiv.2306.05685.

[69] J. Kossen, S. Farquhar, Y. Gal, T. Rainforth, Active testing: Sample-efficient model evaluation, in: International Conference on Machine Learning, 139, PMLR, 2021, pp. 5753–5763, URL https://proceedings.mlr.press/v139/kossen21a.html.

[70] O. Sammodi, A. Metzger, X. Franch, M. Oriol, J. Marco, K. Pohl, Usage-based online testing for proactive adaptation of service-based applications, in: Proceedings of the 35th Annual IEEE International Computer Software and Applications Conference, COMPSAC 2011, Munich, Germany, 18-22 July 2011, IEEE Computer Society, 2011, pp. 582–587, http://dx.doi.org/10.1109/COMPSAC.2011.81.

[71] P.-A. Dragan, A. Metzger, K. Pohl, Coordinated online reinforcement learning for self-adaptive systems using factored Q-Learning, in: 6th IEEE International Conference on Autonomic Computing and Self-Organizing Systems - ACSOS 2025, IEEE CS, USA, 2025, pp. 76–87, http://dx.doi.org/10.1109/ACSOS66086.2025.00024.

[72] Y. Shoham, K. Leyton-Brown, Multiagent systems: Algorithmic, game-theoretic, and logical foundations, Cambridge University Press, Cambridge, UK, 2008, http://dx.doi.org/10.1017/CBO9780511811654.

[73] S. Satyal, I. Weber, H.-y. Paik, C. Di Ciccio, J. Mendling, Business process improvement with the AB-BPM methodology, Inf. Syst. 84 (2019) 283–298, http://dx.doi.org/10.1016/j.is.2018.06.007.

[74] E. Hüllermeier, W. Waegeman, Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods, Mach. Learn. 110 (3) (2021) 457–506, http://dx.doi.org/10.1007/s10994-021-05946-3.

[75] V.-L. Nguyen, M.H. Shaker, E. Hüllermeier, How to measure uncertainty in uncertainty sampling for active learning, Mach. Learn. 111 (1) (2022) 89–122, http://dx.doi.org/10.1007/s10994-021-06003-9.

[76] A. Tharwat, W. Schenck, A survey on active learning: State-of-the-art, practical challenges and research directions, Mathematics 11 (4) (2023) 820, http://dx.doi.org/10.3390/math11040820.

[77] A. Metzger, T. Kley, A. Palm, Triggering proactive business process adaptations via online reinforcement learning, in: D. Fahland, C. Ghidini, J. Becker, M. Dumas (Eds.), Business Process Management - 18th International Conference, BPM 2020, in: LNCS, 12168, Springer, 2020, pp. 273–290, http://dx.doi.org/10.1007/978-3-030-58666-9_16.

[78] V. Torra, M. Bras-Amorós, Memory poisoning and secure multi-agent systems, 2026, arXiv:2603.20357.

[79] L. Beurer-Kellner, B. Buesser, A. Cretu, E. Debenedetti, D. Dobos, D. Fabian, M. Fischer, D. Froelicher, K. Grosse, D. Naeff, E. Ozoani, A. Paverd, F. Tramèr, V. Volhejn, Design patterns for securing LLM agents against prompt injections, 2025, http://dx.doi.org/10.48550/ARXIV.2506.08837, CoRR, arXiv:2506.08837.

[80] J.L. Cobo-Ariza, J. Arregui, A.M.R. Quintero, Á.J. Varela-Vaca, M.T. Gómez-López, Explaining the compliance of security policies for GDPR in business processes, in: Advanced Information Systems Engineering Workshops - CAiSE 2025 Workshops, Vienna, Austria, June 16-20, 2025, Proceedings, in: Lecture Notes in Business Information Processing, Springer, 2025, pp. 355–367, http://dx.doi.org/10.1007/978-3-031-94931-9_29.

[81] A. Chhabra, S. Datta, S.K. Nahin, P. Mohapatra, Agentic AI security: Threats, defenses, evaluation, and open challenges, IEEE Access (2026) http://dx.doi.org/10.1109/ACCESS.2026.3675554, 1–1.

[82] Z. Wang, V. Siu, Z. Ye, T. Shi, Y. Nie, X. Zhao, C. Wang, W. Guo, D. Song, AGENTVIGIL: automatic black-box red-teaming for indirect prompt injection against LLM agents, in: Findings of the Association for Computational Linguistics: EMNLP 2025, Suzhou, China, November 4-9, 2025, Association for Computational Linguistics, 2025, pp. 23159–23172, URL https://aclanthology.org/2025.findings-emnlp.1258/.

[83] S. Baltes, F. Angermeir, C. Arora, M.M. Barón, C. Chen, L. Böhme, F. Calefato, N. Ernst, D. Falessi, B. Fitzgerald, D. Fucci, M. Kalinowski, S. Lambiase, D. Russo, M. Lungu, L. Prechelt, P. Ralph, C. Treude, S. Wagner, Guidelines for empirical studies in software engineering involving large language models, 2025, arXiv:2508.15503.

[84] K. Zhou, Y. Zhu, Z. Chen, W. Chen, W.X. Zhao, X. Chen, Y. Lin, J.-R. Wen, J. Han, Don't make your LLM an evaluation benchmark cheater, 2023, arXiv:2311.01964.

[85] W. van der Aalst, A. Adriansyah, A.K.A. de Medeiros, F. Arcieri, T. Baier, T. Blickle, J.C. Bose, P. van den Brand, R. Brandtjen, J. Buijs, A. Burattin, J. Carmona, M. Castellanos, J. Claes, J. Cook, N. Costantini, F. Curbera, E. Damiani, M. de Leoni, P. Delias, B.F. van Dongen, M. Dumas, S. Dustdar, D. Fahland, D.R. Ferreira, W. Gaaloul, F. van Geffen, S. Goel, C. Günther, A. Guzzo, P. Harmon, A. ter Hofstede, J. Hoogland, J.E. Ingvaldsen, K. Kato, R. Kuhn, A. Kumar, M. La Rosa, F. Maggi, D. Malerba, R.S. Mans, A. Manuel, M. McCreesh, P. Mello, J. Mendling, M. Montali, H.R. Motahari-Nezhad, M. zur Muehlen, J. Munoz-Gama, L. Pontieri, J. Ribeiro, A. Rozinat, H. Seguel Pérez, R. Seguel Pérez, M. Sepúlveda, J. Sinur, P. Soffer, M. Song, A. Sperduti, G. Stilo, C. Stoel, K. Swenson, M. Talamo, W. Tan, C. Turner, J. Vanthienen, G. Varvaressos, E. Verbeek, M. Verdonk, R. Vigo, J. Wang, B. Weber, M. Weidlich, T. Weijters, L. Wen, M. Westergaard, M. Wynn, Process mining manifesto, in: F. Daniel, K. Barkaoui, S. Dustdar (Eds.), Business Process Management Workshops, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, pp. 169–194, http://dx.doi.org/10.1007/978-3-642-28108-2_19.

[86] D. Chapela-Campa, M. Dumas, From process mining to augmented process execution, Softw. Syst. Model. 22 (6) (2023) 1977–1986, http://dx.doi.org/10.1007/S10270-023-01132-2.