Process Mining for Digital Twin Development of Industrial Cyber-Physical Systems

Francesco Vitale, Simone Guarino, Francesco Flammini, *Senior Member, IEEE*, Luca Faramondi, Nicola Mazzocca, and Roberto Setola, *Senior Member, IEEE*

***Abstract*—Digital twin development of industrial cyber-physical systems requires modeling, simulation, and monitoring to provide accurate digital replicas mimicking system dynamics. To this aim, modern solutions employ data-driven approaches to capture normal system dynamics using historical data, and anomaly detection using run-time data. However, these approaches are typically process-agnostic and have limited explainability, which negatively impacts trustworthiness. Hence, we propose a novel framework for the digital twin development of industrial cyber-physical systems based on process mining, which connects data-driven and process-based analyses. The framework implements an offline phase that systematically identifies the most suitable process model to capture normal dynamics and simulates faulty dynamics for what-if analysis. The subsequent online phase involves run-time monitoring for anomaly detection. We develop a proof-of-concept application of the framework addressing a water distribution case study that allows physical fault injection. Results highlight the importance of adopting high-quality process models to significantly improve anomaly detection and time performance.**

***Index Terms*—Conformance checking (CC), event logs, modeling, monitoring, process discovery (PD), simulation, water distribution testbed (WDT).**

Received 27 July 2024; accepted 14 September 2024. This work was supported by EU Horizon Europe R&I programme under Grant HORIZON-ER-JU-2022-ExplR-04. The work of Francesco Vitale and Nicola Mazzocca was supported in part by the Spoke 9 "Digital Society & Smart Cities' of ICSC—Centro Nazionale di Ricerca in High Performance-Computing, Big Data and Quantum Computing, and in part by European Union - NextGenerationEU under Grant PNRR-HPC, CUP: E63C22000980007. The work of Francesco Flammini was supported by the Swiss State Secretariat for Education, Research and Innovation (SERI) under Grant 23.00321 (Academics4Rail project). The work of Simone Guarino and Roberto Setola was supported by the Italian National Project INAIL BRIC 2023 ID 44 "Industrial Cyber Shield (ICS)" under Grant C83C22001460001. Paper no. TII-24-3789. *(Corresponding author: Francesco Vitale.)*

Francesco Vitale and Nicola Mazzocca are with the Department of Electrical Engineering and Information Technology, University of Naples Federico II, 80138 Naples, Italy (e-mail: francesco.vitale@unina.it; nicola.mazzocca@unina.it).

Simone Guarino, Luca Faramondi, and Roberto Setola are with the Departmental Faculty of Engineering, University of Campus Biomedico of Rome, 00128 Rome, Italy (e-mail: r.setola@unicampus.it; l.faramondi@unicampus.it; s.guarino@unicampus.it).

Francesco Flammini is with the Dalle Molle Institute for Artificial Intelligence, University of Applied Sciences and Arts of Southern Switzerland, 6962 Lugano, Switzerland, and also with the Department of Mathematics and Computer Science Ulisse Dini, University of Florence, I-50144 Florence, Italy (e-mail: francesco.flammini@supsi.ch, francesco.flammini@unifi.it).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TII.2024.3465600.

Digital Object Identifier 10.1109/TII.2024.3465600

I. Introduction

THE complexity of modern industrial cyber-physical systems (ICPSs) requires developers and business analysts to accurately model and acknowledge these systems' operation to assess their performance and dependability level. In this regard, the opportunity to model, simulate, and monitor ICPSs by digital twins enables the deployment of smart services, providing manufacturers with more accurate information to enhance production performance, facilitate quality-of-service modeling and evaluation, and ensure fault-tolerant management [1], [2]. In particular, the scientific literature has successfully demonstrated the effectiveness of digital twins for monitoring, simulation, and system efficiency in water distribution networks, where the distributed nature of the ICPS infrastructure poses significant challenges in terms of water management optimization, fault detection, and disaster risk reduction [3], [4].

Regardless of the specific application domain, the promising services provided by digital twins require appropriate ICPS infrastructure modeling to capture the same features and behavior of the physical process dynamics. These dynamics may involve the evolution of the state of an ICPS over time, e.g., the energy consumption of manufacturing systems [6], the water flow dynamics in water distribution [7], or the loss rate of network packets [8]. The analysis of the dynamics of an ICPS using digital twins may leverage model-based methods. Generally, such methods collect the dynamics of ICPSs with mathematical formalisms, such as systems of differential equations and the Petri net. However, since model-based methods require expertise and deep domain knowledge, cutting-edge digital twin development employs data-driven methods. These methods capture normal dynamics from historical data and can detect deviations from run-time data to enable recovery of the ICPS to an acceptable state [9]. Modern data-driven approaches rely on machine learning and/or deep learning, which is especially suited for discovering complex trends in data and long-term dependencies [10], [11], [12]. However, deep learning approaches have several drawbacks, such as large training time, high hardware resource usage, and black-box behavior that leads to hindered explainability [13], [14].

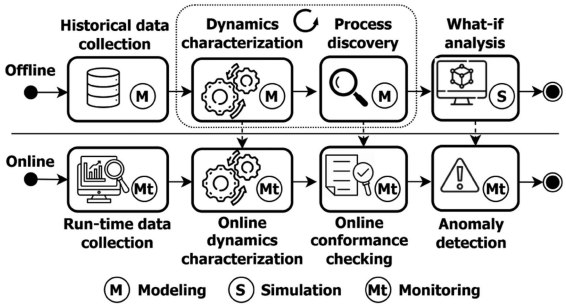

Fig. 1. High-level dynamic view of the offline and online phases of the proposed PM-based framework for digital twin development, with labeled circles indicating the modeling, simulation, and monitoring digital twin services.

Due to the difficulty in devising mathematical models and the limitations of existing data-driven approaches for digital twin development, we consider modeling the dynamics of ICPSs with process mining (PM). This research area allows connecting data-driven and process-based analyses with two sets of techniques: process discovery (PD) and conformance checking (CC) [15]. On the one hand, PD uses historical data to capture normal dynamics as process models using mathematical formalisms, such as the Petri net, and the process-oriented execution semantics of Petri nets improve the interpretability of the modeled dynamics and facilitate simulation. On the other hand, CC verifies whether the process model obtained with PD fits the data. Notably, the quality of the discovered models can be determined by evaluating four criteria, namely, *fitness*, *precision*, *generalization*, and *simplicity* [15]. However, the current scientific literature focuses on fitness and neglects the other metrics. This is a critical shortcoming because a process model that perfectly fits historical data does not necessarily mean it is high-quality. In fact, similarly to machine learning, PD algorithms may discover underfitting or overfitting models [15], leading to low values of precision and generalization, respectively. In addition to discovering and evaluating the quality of process models, PM allows monitoring for anomaly detection through CC in a wide range of domains, such as business processes and software applications [16], [17].

In light of the opportunities opened by PM as regards modeling, simulation, and monitoring, we propose a novel framework for digital twin development of ICPSs based on PM that follows the so-called monitor-analyze-plan-execute (MAPE) loop. This paradigm guides digital twin development through the MAPE steps, which help organize modeling, simulation, and monitoring as software services [18]. Fig. 1 shows a high-level dynamic view of the proposed framework, where the services involved in the MAPE loop are split into offline and online phases. On the one hand, the offline phase implements systematic modeling of the ICPS normal dynamics to identify the most suitable process model and simulation of faulty dynamics for what-if analysis. On the other hand, the online phase involves run-time monitoring for anomaly detection. The offline phase sets out: *historical data collection* from the ICPS plant, *dynamics characterization* using such data for preprocessing and event extraction, *PD* for modeling the normal dynamics of the ICPS, and *what-if analysis* using the model for simulating what could happen in the case of faults in the system. The systematic modeling part of the offline phase includes the iterative execution of dynamics characterization and PD to obtain the best ICPS process model by optimizing the fitness, precision, generalization, and simplicity quality metrics. The online phase implements monitoring by: *run-time data collection* during the ICPS plant operation, *online dynamics characterization* using such data for preprocessing and event extraction by reusing the same parameters as in the offline phase, *online CC* using the process model obtained in the offline phase, and *anomaly detection* in the diagnoses provided by CC, possibly leveraging supervised or unsupervised machine learning algorithms previously trained on the simulations of the offline phase.

We put forward a proof-of-concept application of the proposed framework using the water distribution testbed (WDT) [19], which is a real hardware-in-the-loop case study in a laboratory environment. The WDT is relevant to this work since it is an ICPS for water distribution that allows enforcing automatic control of fluid dynamics across its tanks and pipes and performing physical fault injection. In the offline phase, we demonstrate the systematic modeling approach, where different PD configurations are set to extract the most suited process models of the WDT. Each configuration involves a specific PD algorithm paired with a particular event extraction strategy applied to the collected WDT data. We consider a set of five PD algorithms: the inductive miner (IM), inductive miner infrequent (IMf) with 20% noise tolerance, α -miner, heuristic miner (HM), and integer linear programming-based miner (ILP). Next, the selected process models drive simulation and the subsequent anomaly detection in the online phase. In particular, the anomaly detection task involves adopting five different machine learning algorithms: one-class support vector machine (OC-SVM), isolation forest (IF), random forest (RF), extreme gradient boosting (XGBoost), and adaptive boosting (AdaBoost), trained using the so-called CC diagnoses [18]. The detection performance is then evaluated and compared with the baseline fitness thresholding (FT) approach [20]. The results show that the detection performance for each adopted anomaly detection technique is heavily impacted by the quality of the discovered models.

In summary, the main original contributions of this article are as follows.

1) Novel framework for digital twin development of ICPSs based on PM aimed at identifying the best ICPS process model by optimizing its fitness, precision, generalization, and simplicity.
2) Use of the process model to drive the simulation of faulty dynamics for what-if analysis and run-time monitoring for anomaly detection.
3) Application of the framework to a case study in a laboratory environment to evaluate the quality of process models obtained through different PD configurations and the performance of machine learning for anomaly detection.
4) Critical analysis of the results with a focus on high-quality process models from the best PD configurations to assess anomaly detection and time performance.

The rest of this article is organized as follows. Section II reviews existing approaches to digital twin development using PM. Section III provides the preliminaries to establish the definitions of the main PM inputs and outputs. Section IV presents a detailed view of the offline and online phases of the proposed framework for digital twin development based on PM. Section V describes the practical application of the framework to the WDT case study and showcases the results for both modeling and anomaly detection within the WDT. Finally, Section VI concludes this article.

II. Related Works

The scientific literature reports several methods to model ICPS dynamics. First, mathematical formulations can be devised to evaluate the state evolution of the physical plant. Examples of such dynamics are the modeling of the energy consumption of power systems and the traffic flow of vehicles [10], [26], [27], [28], [29]. Another approach involves formalisms to capture process models, such as the Petri net and the business process modeling notation (BPMN) [30], [31], [32]. These formalisms focus on modeling the relationships among state transitions of physical processes instead of the continuous state evolution of physical variables. Although mathematical and process-based formalisms are well-established modeling approaches, their development is made difficult by the high complexity of cyber-physical processes in ICPSs. Therefore, these models may be entirely or partially substituted by data-driven approaches leveraging machine learning and deep learning, where the normal behavior of the ICPS is modeled by monitoring the physical variables of the plants and learning their dynamics. For example, Faramondi et al. [33] proposed the application of neural networks and Bayesian networks to model the behavior of physical variables and network traffic of the WDT case study. Furthermore, Ding et al. [34] employed long short-term memory networks for modeling and detecting errors of ICPS sensors, computing hardware, and network traffic.

Despite the well-known advantages of machine learning and deep learning algorithms in terms of effective identification of trends and patterns and the handling of multimodal data, usually, these approaches are not suited for modeling and simulating ICPS dynamics. In fact, they focus primarily on tracking the behavior of local variables collected from individual ICPS devices and do not offer a comprehensive process-oriented view involving the state and state transitions of the ICPS. The difficulty of modeling the dynamics of ICPSs by mathematical formulations and process models, and the limitations of machine learning and deep learning techniques, including explainability and resource-constraint issues [13], [14], could be bridged by PM, which allows for extracting process models from event data by PD [15]. However, to extract a quality process model of ICPS dynamics, it is crucial to extract meaningful events from the ICPS monitored data. Furthermore, the PD algorithm could influence the quality of the process model as well.

As far as we are aware, the scientific literature lacks comprehensive work on systematic modeling approaches for digital twin development of ICPSs based on PM to investigate how different event data and different PD algorithms affect the quality of process models in terms of the fitness, precision, generalization, and simplicity metrics. In fact, the evaluation of all these metrics is crucial for identifying the best-suitable model to conduct simulation and anomaly detection [15]. Instead, the existing approaches usually capture a single process model related to the ICPS normative dynamics and use CC to check whether new dynamics align with it, thus detecting possible deviations. These approaches usually involve thresholding the fitness metric, which measures the degree of conformance of new behavior with the normative process model [20]. First, Accorsi and Stocker [21] applied PM for security analysis in a bank loan application scenario. The process is modeled with the BPMN through a set of interviews with the bank managers and transformed into a Petri net, and new traces are replayed against the Petri net via CC and those that did not fit perfectly were deemed anomalous and analyzed further. Next, Sarno et al. [23] inspected credit applications in a bank case study by integrating PM, fuzzy multiattribute decision-making and fuzzy association rule learning. In particular, CC checks recorded event logs and standard operating procedures, providing information about, e.g., skipped activities or wrong activity patterns, and this information is used to identify frauds. Moreover, Pecchia et al. [17] applied four different PD algorithms and CC to software logs. The analysis has been conducted over a dataset of 55 462 execution traces from three independent real-life applications. Their results demonstrate the failure detection capability of CC despite the presence of noisy application logs. Regarding ICPSs, Myers et al. [22] provided a comprehensive analysis of how to apply PM to industrial control systems, starting from device logs generation to analyzing such logs by CC. Although thresholding the fitness metric has shown satisfying performance in some cases, this approach could lead to poor detection performance in terms of many false positives and negatives in the case of noisy data [17], [20]. For this reason, researchers have been recently investigating the combination of PM and machine learning algorithms applied to the diagnostics provided by CC [18], [24], [25]. Nevertheless, such approaches only consider simulated traces, thus without evaluating realistic cyber-physical scenarios.

Table I relates the existing work to the key features that we outlined, namely, the model quality metrics, modeling process, simulation, anomaly detection, and the use of an ICPS case study, whereas all references neglect assessing the quality of multiple process models, our proposal involves a structured modeling approach where multiple Petri nets of the ICPS dynamics are extracted by varying the event log extraction strategy and the adopted PD algorithm. Moreover, our proposal investigates the impact of the quality of the process models on the simulation and anomaly detection tasks, where the performance is evaluated by exploring the combination of PM and machine learning in a real ICPS case study.

III. Preliminaries

In this section, we establish the definitions of the main PM inputs and outputs, commencing with the event log.

TABLE I
CHARACTERIZATION OF THE LITERATURE REGARDING PM-BASED MODELING AND ANOMALY DETECTION

Reference	Model quality metrics	Modeling process	Simulation	Anomaly detection	ICPS case study
[21]	Fitness metric	Single BPMN model evaluation without PD	✓	PM by fitness thresholding	✗
[22]	Fitness metric	Single Petri net evaluation using PD	✗	PM by fitness thresholding	✓
[17]	Fitness metric	Multiple Petri net evaluation using PD	✗	PM by fitness thresholding	✗
[23]	Fitness metric	Single BPMN model evaluation without PD	✗	PM + machine learning	✗
[24]	Fitness metric	Single BPMN model evaluation without PD	✓	PM + machine learning	✗
[18]	Fitness metric	Single BPMN model evaluation without PD	✓	PM + machine learning	✗
[25]	Fitness metric	Single BPMN model evaluation using PD	✓	PM + machine learning	✗
Our approach	Fitness, precision, generalization, and simplicity metrics	Multiple Petri net evaluation using PD	✓	PM + machine learning	✓

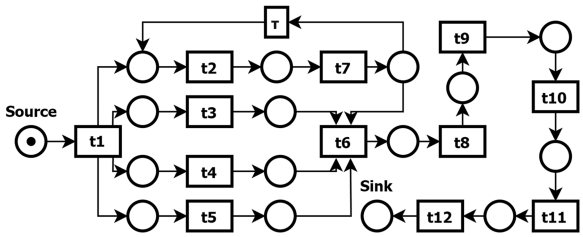

Fig. 2. Example sound workflow Petri net.

Definition 3.1 (Event log): Let us denote $\mathcal{A}$ as the universe of activities and $\mathcal{A}^*$ as the universe of possible sequences of activities (traces) in $\mathcal{A}$. An event log L is a multiset of k traces, where a multiset is a set such that there may be two equal yet distinct elements. Given $\mathcal{B}(\mathcal{A}^*)$ the set of multisets over $\mathcal{A}^*$, $L \in \mathcal{B}(\mathcal{A}^*)$.

In addition to event logs, PD and CC algorithms involve either using or producing a process model. The most widespread formalism in PM for process models is the Petri net.

Definition 3.2 (Petri net): Let P and Tr be two sets of nodes of a bipartite graph such that $P \cap Tr = \emptyset$, and $F \subseteq (P \times Tr) \cup (Tr \times P)$ a set of directed arcs. Furthermore, let $A \subseteq \mathcal{A}$ be a set of activities and $l : Tr \to A \cup \{\tau\}$ a function that associates to elements of Tr either an activity of A or the "silent" label τ, which represent unobservable activities that add legitimate behaviors not directly visible. A Petri net is the tuple (P, Tr, F, A, l), made of the places P, transitions Tr, arcs F, activities A, and labeling function l. Finally, $M \in \mathcal{B}(P)$ is the (current) *marking* of the Petri net, i.e., a multiset of tokens that determine which transition is allowed to *fire*. In the following, we denote $\mathcal{N}$ as the universe of Petri nets.

The Petri net can be extended to simulate a range of dynamics, from initial marking to final marking [35]. In addition, the Petri net to simulate should meet the soundness property. This property ensures that the final marking is always reached and that there are no dead parts. The most suited subclass of Petri nets that meet these requirements is the sound workflow Petri net [15]. Fig. 2 shows an example sound workflow Petri net. The round nodes are the set of places P, whereas the squared nodes are the set of transitions Tr, which are connected to a set of real activities A by a labeling function l. The `source` place has a token, which is the current—and initial—marking M of the Petri net. Furthermore, the Petri net has a τ transition.

IV. FRAMEWORK

In the following, we detail the offline and online phases of the proposed framework for digital twin development of ICPSs based on PM. Fig. 3 refines the high-level view in Fig. 1, showing the modules, depicted as rounded squares, involved in each offline and online digital twin service.

A. Offline Phase

1) Dynamics Characterization: First, *dynamics characterization* involves describing the evolution of the ICPS over time from historical data, which we consider a set of variables whose values are collected up to a given time. Such evolution is described in terms of the current state and state transitions.

Definition 4.1 (Current state and state transition): Let us denote $\mathcal{T} = \{1 \dots T\}$ the set of time steps up to $T \in \mathbb{N}$, $\mathcal{S}$ the universe of the possible states of a reference ICPS, and $\mathcal{V} = \{v_1, \dots, v_n : v_i(t) \in \mathbb{R}, t \in \mathcal{T}\}$ the set of n variables of a reference ICPS. Let $t \in \mathcal{T}$ be a time step, $v = (v_1, \dots, v_m) \in \mathcal{V}^m, m \leq n$ an m-tuple of variables, and $s \in \mathcal{S}$ a state of the ICPS. The *current state* of the reference ICPS is obtained by function

$$S : \mathcal{V}^m \times \mathcal{T} \to \mathcal{S}, \quad m \leq n \tag{1}$$

such that $S(v, t) = s$. Furthermore, let $t_1, t_2 \in \mathcal{T}$ be such that $t_2 = t_1 + 1 \wedge S(v, t_1) \neq S(v, t_2)$. The *state transition* of a reference ICPS is the triplet $st = (S(v, t_1), S(v, t_2), t_2)$. Hereafter, we consider the set of state transitions as the set of activities (see Definition 3.1).

To achieve meaningful process models, *preprocessing* may implement S by quantizing the numerical variables uniformly and assigning the current state based on the interval in which the value of the different variables falls. Subsequently, *event log extraction* involves building the event log from the identified state transitions.

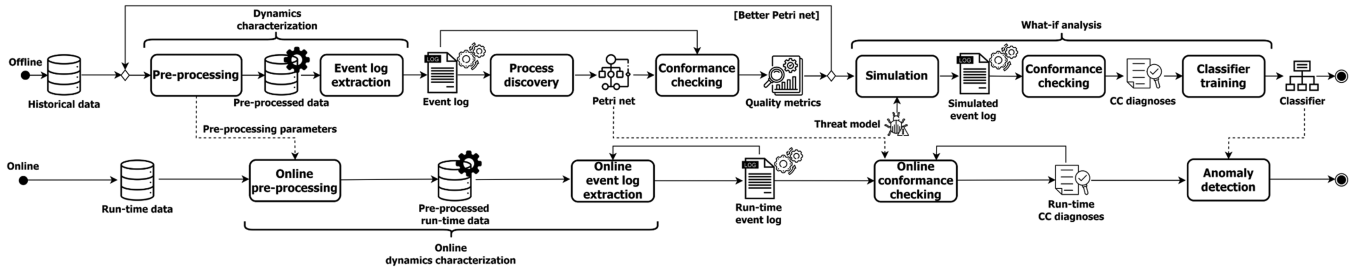

Fig. 3. Activity diagram that shows the steps that implement the digital twin services of the proposed PM-based framework.

2) Process Discovery: PD applies a PD algorithm to extract a set of control-flow relations between the state transitions of the event log. These relations can be encoded into a Petri net [15]. Hence, in general, a PD algorithm can be formulated as follows.

Definition 4.2 (PD algorithm): Let Λ be a set of control-flow relations, $L \in \mathcal{B}(\mathcal{A}^*)$ an event log, and $\bar{\mathcal{A}} = \mathcal{A} \cup \{\tau\}$ the set of state transitions joined with the silent transition τ. Let $\mathcal{P}(\bar{\mathcal{A}})$ be the power set of $\bar{\mathcal{A}}$, $\mathcal{P}(\bar{\mathcal{A}}) \times \mathcal{P}(\bar{\mathcal{A}}) \times \Lambda$ the set of triples containing elements of $\mathcal{P}(\bar{\mathcal{A}})$ and Λ, where each triple is made of two sets of state transitions and the related control-flow relation. Finally, let $\lambda : \mathcal{B}(\mathcal{A}^*) \to \mathcal{P}(\mathcal{P}(\bar{\mathcal{A}}) \times \mathcal{P}(\bar{\mathcal{A}}) \times \Lambda)$ be a function that associates an event log with a set of such triples and γ a function that encodes the results of λ in a Petri net $N \in \mathcal{N}$. A PD algorithm $\gamma \circ \lambda$ is the composite function

$$\gamma \circ \lambda : \mathcal{B}(\mathcal{A}^*) \to \mathcal{N} \tag{2}$$

such that $\gamma(\lambda(L)) = N$.

The set of control-flow relations Λ depends on the PD algorithm. For example, the IM [36] discovers the sequence $\rightarrow$, parallel $\wedge$, exclusive $\times$ and redo-loop $\circlearrowleft$ control-flow relations, and implements λ by recognizing these relations from splitting the event log into simpler parts. The Petri net in Fig. 2 was discovered using the IM, which identified, e.g., the triples $(\{\texttt{t2}\}, \{\texttt{t7}\}, \rightarrow)$, i.e., `t2` is causally followed by `t7`, and $(\{\texttt{t2}, \texttt{t7}\}, \{\tau\}, \circlearrowleft)$, i.e., the sequence of transitions `t2-t7` can loop via τ.

The types of control-flow relations handled and how these are identified in the event log by the PD algorithm limit the search space of the possible Petri nets. This limitation is called the *representational bias* and heavily affects the quality of the resulting Petri net [15]. For example, the α-miner struggles with capturing state transition loops because it only identifies those state transitions that either follow each other in a causal dependency or occur in parallel. In contrast, the IM can easily identify loops since it supports the redo-loop relation. In addition to the representational bias, PD algorithms differ based on their ability to deal with *noisy* event logs [15]. For example, the HM uses a "dependency threshold" that filters the control-flow relations in the event log by evaluating the frequency of occurrence of a given control-flow relation between a pair of state transitions: if the frequency is below the threshold, the control-flow relation is disregarded when building the Petri net.

As a final remark, note that the simplicity of the Petri net can be evaluated at this stage by analyzing its graph-based structure, e.g., the average number of incoming and outgoing arcs for each transition [37].

3) Conformance Checking: CC applies a CC algorithm to compare the traces of an event log with the Petri net, keeping track of event log elements that may highlight control-flow mismatches. This tracking results in CC diagnoses.

Definition 4.3 (CC diagnoses): Let $L \in \mathcal{B}(\mathcal{A}^*)$ be an event log of k traces and $N \in \mathcal{N}$ a Petri net. CC diagnoses $D \in \mathbb{R}^{k \times o}$ are collected by function

$$\mathcal{D} : \mathcal{B}(\mathcal{A}^*) \times \mathcal{N} \to \mathbb{R}^{k \times o} \tag{3}$$

such that $\mathcal{D}(L, N) = D$, where k is the number of traces and o is a number of event log elements.

The elements being tracked depend on the CC variant. For example, the token-based variant checks whether the execution of the traces in the event log causes missing and/or remaining tokens in the Petri net. A further example is the alignment-based variant, which attempts to align the execution of the traces in the event log with the best approximating path in the Petri net, tracking the activities that cause mismatches. These CC diagnoses can be used to evaluate a set of quality metrics, which are the fitness, precision, and generalization metrics [15].

Finally, note that the modeling part of the framework, comprising preprocessing, event log extraction, PD, and CC, is executed iteratively. This is to adjust the preprocessing and event extraction strategies and the PD algorithm until the best Petri net is obtained according to the quality metrics above.

4) What-If Analysis: What-if analysis carries out a set of *simulations* using the Petri net obtained during the modeling part to check what happens if faults are injected in the ICPS process. In particular, given a sound Petri net (see Section III), this can be extended to a stochastic Petri net to simulate a set of traces. Notably, this extension integrates the Petri net in Definition 3.2 with probability functions associated with the transitions to introduce stochastic delays and transition firing priorities in the presence of race conditions [35]. Subsequently, these traces are probabilistically injected with faults according to a specific threat model, which can be provided by domain experts or rely on generic fault injection techniques. The simulated event log undergoes *CC* so that the resulting faulty CC diagnoses can be used for *classifier training*. The resulting classifier could be the FT approach [20] or other supervised/unsupervised classifiers using the information in CC diagnoses [18].

B. Online Phase

Online dynamics characterization involves extracting the ICPS dynamics from run-time data monitored while the system is exercised. *Online preprocessing* applies to run-time data the same preprocessing parameters derived from historical data, such as the quantization step. This is critical because, e.g., run-time data could possibly exceed the lower and upper bounds for quantization adopted in the offline phase. Next, *online event log extraction* involves extracting state transitions from the collected data and updating the run-time event log. This can be achieved with two different strategies: incremental mode, where the event log is updated with each new state transition, and window mode, where the update occurs after a set of transitions has been collected [15]. When the run-time event log is updated, the framework checks whether it conforms to the Petri net obtained in the offline phase by *online CC* and updates the run-time CC diagnoses. Regardless of the specific event log update strategy, the run-time traces to check could be incomplete, hence requiring the application of slightly different CC solutions. In this article, we advocate the use of either online token-based CC [38] or alignment-based CC with prefix alignments [39] since these two solutions can provide the same types of CC diagnoses as in Definition 4.3. Finally, each time run-time CC diagnoses are updated, *anomaly detection* uses the classifier obtained in the offline phase to verify the presence of anomalies in the online execution. On the one hand, the classifier should accept normal behavior that exhibits small deviations from the norm. On the other hand, anomalous behavior should be detected and notified to avoid the propagation of errors throughout the ICPS, hence preventing failures.

V. Case Study and Experimental Results

In this section, we provide an overview of the WDT infrastructure and data collection, and show the proof-of-concept application of the framework to the WDT. Then, we split the experimental evaluation into two parts, where the first focuses on PM-based modeling of the WDT and the second assesses the quality of anomaly detection.

A. Water Distribution Testbed

The WDT is a scaled-down version of a water distribution infrastructure in a laboratory environment. The WDT consists of five polyurethane tanks ($T_1, \ldots, T_5$), 20 Evian©Series 263-model solenoid valves ($V_1, \ldots, V_{20}$), three mini-type pipe 151 410 pumps (P_1, P_2, P_3), one EK-DCP 2.2 pump (P_4), five WIKA©S-11 pressure sensors ($S_1, \ldots, S_5$), and eight manual valves to simulate water leaks from tanks or pipes. The volume of T_3 and T_4 is 15 l, whereas the volume of T_1, T_2, and T_5 is 30 l. Moreover, tanks are connected by cross-linked multilayer polyurethane pipes with an external diameter of 7/8". The water flow is controlled by a Modicon M340 programmable logic controller (PLC) equipped with BMX P342020 processors, DDM16025 discrete I/O, and AMM0600 mixed analog I/O modules. The physical plant is enriched with a virtual extension using `minicps`, which virtually adds three tanks (T_6, T_7, T_8), two pumps (P_5, P_6), two solenoid valves (V_{21}, V_{22}), three pressure sensors (S_6, S_7, S_8) under each tank, and three virtual PLCs for water-flow control. Specifically, the tanks are modeled with a volume of 30 l, the connecting pipes have an external diameter of 7/8", and the pumps have a flow rate of 4 l/min. The virtual extension is implemented using Python 3.9 and is run on an Ubuntu machine with the following characteristics: IntelXeonCPU E5-2620 v2 @ 2.10 GHz and 16 GB of RAM memory. WDT data from PLCs are collected by a Movicon 11.6 supervisory control and data acquisition workstation, which includes both a human machine interface for data visualization and a historian for data storage.

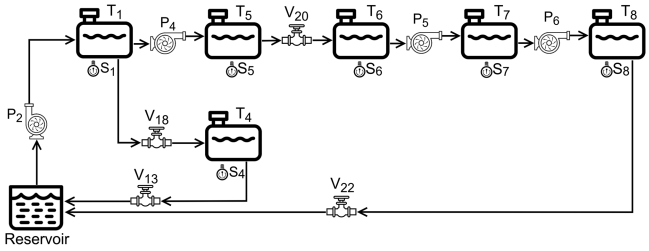

Fig. 4. Flow of water across tanks and the reservoir in the WDT.

TABLE II
Anomaly Scenarios for Physical Fault Injection in the WDT

Anomaly scenario	Affected resource	Cause	Effect
#1	V_{18}	Anomalous opening of V_{18}	Anomalous filling of T_4 from T_1
#2	V_{13}	Anomalous opening of V_{13}	Anomalous emptying of T_4
#3	P_5	Internal P_5 leak	Anomalous filling of T_7 from T_6
#4	V_{20}	V_{20} breakdown	Delay in filling T_6

The proposed case study involves data collection from both normal and anomalous dynamics of the WDT. Fig. 4 shows the normal water distribution dynamics enforced in the WDT. The water is pumped from the reservoir toward T_1, and then, the water follows two paths, of which the first involves T_4, and the second T_5, T_6, T_7, and T_8. Please notice that when valve V_{20} is activated, although the water is directed toward the reservoir, it is virtually redirected toward T_6 to initiate the simulated physical process in `minicps`. A *cycle* occurs when T_1 completes a filling-emptying cycle, and in our experimentation, a cycle is a *trace*. We collect a total of eight cycles as historical data to characterize normal dynamics. Anomalous dynamics are enforced by injecting physical faults on WDT pumps and valves. Table II lists the four anomaly scenarios in this case study. These scenarios are characterized by the affected system resources, the cause of the anomaly, and the effect on the physical process. The total number of collected cycles adopted as run-time data is 28, of which nine are anomalous and nineteen are normal.

B. Framework Application

1) Offline Phase: Dynamics characterization involves using the historical cycles collected from the WDT to obtain the system states and state transitions, which are then organized in traces to

build the event log. In more detail, *preprocessing* filters the historical data by selecting the sensors that indicate the water level in the six tanks involved in the WDT physical process depicted in Fig. 4 ($T_1, T_4, T_5, T_6, T_7, T_8$). Subsequently, each tank is assigned a set of states by quantizing the water level. For example, given a quantization (Q) level of 50%, the tank is in state $0\% - 50\%$ if the level stays within the interval that bounds the minimum and half the maximum capacity, and conversely, it is in state 50%–100%. Furthermore, state transitions are obtained for each tank by tracking when there are transitions from one state to another, i.e., from one water level to another. In this case study, we consider seven different Q levels, 50%, 33%, 25%, 20%, 16.6%, 14.2%, and 12.5%, leading to seven different sets of states and state transitions: by decreasing the Q level, we enable finer-grained analyses of the overall WDT dynamics. For each Q level, *event log extraction* draws complete traces by evaluating when T_1 completes a filling-emptying cycle and collects these traces into a single event log. Then, *PD* applies a PD algorithm to the event log. We consider: the α-miner, HM, IM, IMf with 20% noise tolerance, and ILP. The α-miner is the simplest and is included as the baseline. The HM slightly improves the α -miner by handling the directly-follows relation through causal nets and filtering noisy event logs. The IM applies the approach described in Section IV and always discovers sound Petri nets, but tends to generalize too much and discover complex Petri nets. The IMf extends the IM by filtering noisy event logs and reducing the complexity of the resulting Petri net. Finally, the ILP builds Petri nets by formulating a system of inequations based on causal control-flow relations, excluding the infrequent ones. The overall quality of the Petri net is assessed by *CC*. We use alignment-based CC to compute the fitness and precision, and token-based CC to compute generalization, and this choice is due to the Python library we use for the experiments. In total, we evaluate the quality of 35 different Petri nets, corresponding to each PD-Q configuration.

Subsequently, *what-if analysis* involves simulating these Petri nets to check what happens if faults are injected in the WDT dynamics. For each Petri net, a set of traces is simulated and injected with control-flow anomalies, which we set to occur uniformly with 15% probability. In this case study, we consider the following two control-flow anomalies: 1) unknown state transition, which may result from WDT water overflow, and 2) state transition order swap, which may result from WDT anomalous water leaks or pumping. In total, we generate one training event log per Petri net consisting of 3000 normal and 3000 anomalous traces. *CC* applies alignment-based CC to the simulated event logs to retrieve the CC diagnoses. These are then used during *classifier training* to train five different machine learning classifiers: RF, XGBoost, and AdaBoost for supervised learning, and IF and OC-SVM for unsupervised learning.

2) Online Phase: Online dynamics characterization involves extracting the WDT dynamics from the collected run-time data to build event logs. First, *online preprocessing* filters run-time data, applying the same seven Q levels adopted in the offline phase. Subsequently, *online event log extraction* extracts state transitions from the filtered data and builds the event log, which is updated in a window mode strategy after collecting a complete trace from data. Upon collecting a trace, *online CC* runs alignment-based CC with prefix alignment to build the run-time CC diagnoses. These are handled during *anomaly detection* by the trained classifiers to detect the four anomaly scenarios injected in the WDT (see Table II).

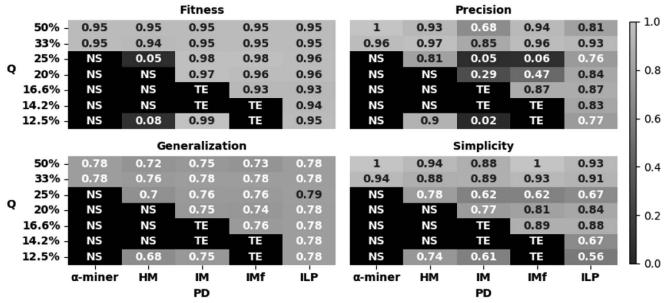

Fig. 5. Heatmaps related to the fitness, precision, generalization, and simplicity metrics per PD-Q pair. Heatmaps indicate those PD-Q pairs that either lead to NS Petri nets or TE when calculating the quality metrics.

C. Experimental Evaluation

The software for the experimentation is implemented using Python and several off-the-shelf libraries for data processing, machine learning, and PM, such as `scipy`, `sklearn`, and `pm4py`. The software was run on a desktop PC hosting an IntelCore i9-11900K CPU @ 3.50 GHz and 32 GB of RAM memory. The experimental setup and dataset for both normal and anomalous dynamics are available online.[1]

1) Modeling Results: Following the structured approach to modeling proposed in our framework, we aim to evaluate the fitness, precision, generalization, and simplicity of the PD algorithms in combination with the seven different Q levels above. Therefore, we have a two-factor experiment with factors PD and Q. The experiment involves adopting the normal cycles collected from the normal dynamics of the WDT. Specifically, out of the eight historical normal cycles, we randomly pick four of them to build a Petri nets by PD. The remaining four are used to evaluate the quality metrics. This process is repeated 20 times to account for statistical fluctuations. We set a timeout of 120 s for the application of alignment-based CC to evaluate the quality metrics.

Fig. 5 shows the heatmaps related to the fitness (top left), precision (top right), generalization (bottom left), and simplicity (bottom right) metrics per PD-Q pair. In general, the best Q level is 33% since this value allows all PD algorithms to achieve high fitness ($\geq$0.94), precision ($\geq$0.85), simplicity ($\geq$0.88), and generalization ($\geq$0.76). Moreover, the lower the Q level, the more likely the α-miner and HM could build nonsound (NS) Petri nets. In contrast, the IM, IMf, and ILP always build sound Petri nets, which is a pivotal property since the alignment-based CC implementation of `pm4py` cannot be applied to NS Petri nets. However, the IM and IMf tend to discover complex and low-precision Petri nets for several Q levels. Besides, the IM and IMf are more likely to show a timeout error (TE) for lower Q levels, e.g., 16.6% (IM), 14.2% (IM and IMf), and 12.5% (IMf).

[1][Online]. Available: https://github.com/francescovitale/pm_wdt_modeling_ad

TABLE III
TIME AND SPACE COMPLEXITY OF THE ILP COMPARED TO THE BASELINE α-MINER FOR EACH Q LEVEL

PD	Q	PD time [s]	CC time [s]	Places	Transitions	Arcs	Size
ILP (overall best)	**50%**	0.03 ± 0.00	<0.00 ± 0.00	11	12	24	5.11kB
	33%	0.12 ± 0.00	<0.00 ± 0.00	27	24	54	10.72kB
	25%	0.47 ± 0.03	0.01 ± 0.00	42	38	101	17.77kB
	20%	0.86 ± 0.04	0.02 ± 0.00	53	50	113	21.54kB
	16.6%	1.81 ± 0.15	0.08 ± 0.01	67	60	136	26.00kB
	14.2%	4.31 ± 0.27	0.28 ± 0.09	84	73	196	34.26kB
	12.5%	8.90 ± 0.31	0.57 ± 0.44	104	86	262	43.05kB
α-miner (baseline)	**50%**	<0.00 ± 0.00	<0.00 ± 0.00	13	12	24	7.80kB
	33%	<0.00 ± 0.00	<0.00 ± 0.00	28	24	54	16.68kB
	25%	<0.00 ± 0.00	N/A	42	36	85	25.51kB
	20%	<0.00 ± 0.00	N/A	52	48	105	32.00kB
	16.6%	<0.00 ± 0.00	N/A	68	60	135	41.11kB
	14.2%	<0.00 ± 0.00	N/A	85	72	173	51.55kB
	12.5%	<0.00 ± 0.00	N/A	106	84	229	66.88kB

N/A indicates that alignment-based CC is non-applicable due to the petri net being NS.

TABLE IV
SET OF HYPERPARAMETER VALUES EXPLORED DURING VALIDATION

		Hyperparameter values
Machine learning algorithm	**OC-SVM**	$\nu = \{0.1, 0.2, 0.3\}$ kernel = {'linear', 'poly', 'rbf', 'sigmoid'}
	RF	n_estimators = $\{100, 200, 500, 1000\}$
	XGBoost	$\eta = \{0.001, 0.01, 0.1, 0.3, 0.5, 1\}$ n_estimators = $\{100, 200, 500, 1000\}$ booster = {'gbtree', 'gblinear'}
	Adaboost	n_estimators= $\{100, 200, 500, 1000\}$ learning rate = $\{0.001, 0.01, 0.1, 0.3, 0.5, 1\}$
	IF	n_estimators= $\{100, 200, 500, 1000\}$ contamination = $\{0.1, 0.2, 0.3, 0.4\}$
Feature selection algorithm	**PCA with cumulative variance thresholding**	threshold = $\{0.95, 0.96, 0.97, 0.98, 0.99\}$

Notably, the ILP is the overall best since it achieves satisfying fitness, precision, generalization, and simplicity for all Q levels, builds sound Petri nets, and does not exceed the timeout for CC. We provide the detailed time and space complexity of ILP in Table III, comparing these to the baseline, i.e., the α-miner. Specifically, this table reports the time required to extract the Petri net (PD time) and for applying alignment-based CC for each normal trace (CC time), the number of places, transitions and arcs of the Petri net, and the size of the model. Although the α-miner is very fast at both discovering Petri nets and checking new traces, it leads to NS Petri nets for the majority of Q levels. Instead, the ILP is able to keep low PD and CC times while building sound Petri nets with lower memory usage for all Q levels.

2) Simulation and Anomaly Detection Results: In this part, we aim to evaluate the detection performance of several machine learning classifiers using the Petri nets obtained with the PD-Q configurations above. In addition, we evaluate the CC time for Petri nets discovered using such configurations. As mentioned, the classifiers are: RF, XGBoost, and AdaBoost for supervised learning, and IF and OC-SVM for unsupervised learning. Furthermore, we compare the results with the FT approach [20]. The classifiers were trained using the CC diagnoses obtained by simulation from the Petri net corresponding to the specific PD-Q configuration. In more detail, RF, XGBoost, AdaBoost, and IF were trained using both anomalous and normal simulated traces, whereas OC-SVM was trained exclusively using the normal data. Finally, FT was implemented by thresholding the fitness with the minimum value obtained from the normal simulated traces. In addition, the machine learning classifiers were validated through stratified five-fold cross-validation. Throughout the validation process, we standardized the CC diagnoses, employed feature selection with principal component analysis (PCA) and the cumulative variance thresholding criterion [40], and explored a set of hyperparameter values to maximize the performance metrics of each classifier (see Table IV).

First, we assess the impact of the Petri net quality on anomaly detection performance and CC time. To this aim, we consider the Petri nets obtained with 25% Q, for which the HM shows very low fitness (0.05), IM and IMf show very low precision (0.05 and 0.06, respectively), ILP results in the highest-quality PD algorithm, and the α-miner builds an NS Petri net. Table V reports the detection precision, recall, and F1 metrics of the

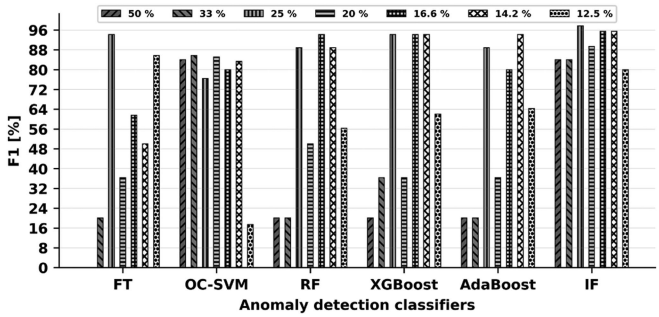

Fig. 6. Detection F1 score metric of FT, OC-SVM, IF, RF, Adaboost, and XGBoost classifiers related to the Petri nets obtained by ILP with 12.5%, 14.2%, 16.6%, 20%, 25%, 33%, and 50% Q.

aforementioned classifiers tested against the 28 (nine anomalous + 19 normal) run-time traces. It also includes the mean and standard deviation of the time required to retrieve the CC diagnoses from these traces (CC time). In general, the detection performance of all the classifiers heavily depends on the quality of the Petri net used to extract the CC diagnoses. In fact, regardless of the adopted classifier, the HM and IM result in the worst detection performance metrics. The highest detection performance is achieved by the ILP, which discovers the Petri net with the best quality metrics. Interestingly, the IMf, despite its low modeling precision, leads to high-detection performance, even surpassing the ILP when using RF ($P = 88.9, F1 = 88.9$). However, the IMf shows the worst CC time (~ 1.56s), whereas the ILP is the second fastest (~ 0.02 s). Overall, it can be noticed that IF outperforms the other classifiers in all cases, including the baseline FT approach, while XGBoost results in the best-performing supervised classifier.

Second, due to the ILP being the overall best for modeling the data at hand, we consider a detailed overview of the F1 score related to the ILP across the seven different Q levels, shown in Fig. 6. The best F1 score is achieved by the IF, which achieves the highest detection performance with $F1 = 97\%$ for 25% Q, whereas the worst performance is obtained by FT, with an F1 score of 0% for 50% Q. The OC-SVM generally shows high F1 values ($F1 > 75\%$), except for 12.5% Q, where F1 drops to 17%. At the same time, the supervised algorithms improve their performance as the Q level decreases, demonstrating that

TABLE V
DETECTION PRECISION (P), RECALL (R), AND F1 SCORE METRICS OF THE FT, OC-SVM, IF, RF, ADABOOST, AND XGBOOST CLASSIFIERS RELATED TO THE PETRI NETS OBTAINED BY HM, IM, IMF, AND ILP WITH 25% Q: ID INDICATES AN ILL-DEFINED VALUE

PD	CC time [s]	FT			OC-SVM			IF			RF			AdaBoost			XGBoost		
		P[%]	R[%]	F1[%]	P[%]	R[%]	F1[%]	P[%]	R[%]	F1[%]	P[%]	R[%]	F1[%]	P[%]	R[%]	F1[%]	P[%]	R[%]	F1[%]
HM	$\mathbf{0.01 \pm 0.007}$	ID	0.0	0.0	30.0	1.0	46.1	75.0	100.0	85.7	35.7	55.6	43.5	35.7	55.6	43.5	38.5	55.6	45.5
IM	1.55 ± 6.96	100.0	77.8	87.5	30.0	1.0	46.1	84.0	100.0	91.3	50.0	66.7	57.1	45.5	55.6	50.0	100.0	44.4	61.5
IMf	1.56 ± 7.04	**100.0**	**88.9**	**94.1**	90.9	47.6	62.5	**95.5**	**100.0**	**97.7**	**88.9**	**88.9**	**88.9**	88.9	**88.9**	88.9	88.9	**88.9**	88.9
ILP	0.02 ± 0.02	**100.0**	**88.9**	**94.1**	**92.9**	**61.9**	**74.3**	**95.5**	**100.0**	**97.7**	80.0	**88.9**	84.2	**100.0**	**88.9**	**94.1**	**100.0**	**88.9**	**94.1**

CC time column indicates the time required to retrieve the CC diagnoses from run-time traces.

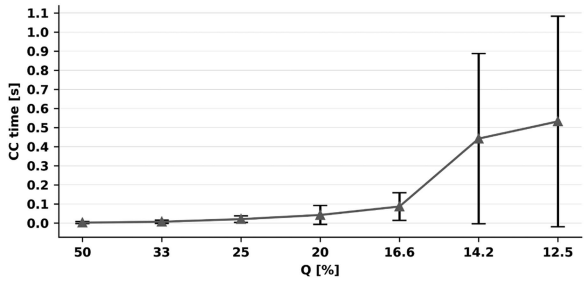

Fig. 7. Time required to retrieve the CC diagnoses from run-time traces for Petri nets obtained by ILP with 12.5%, 14.2%, 16.6%, 20%, 25%, 33%, and 50% Q.

the higher the level of modeling detail, the better the ability of supervised algorithms to recognize unknown test anomalies. However, similar to the unsupervised classifiers, there is a drop in performance at 12.5% Q for the supervised classifiers as well, with RF, XGBoost, and AdaBoost exhibiting F1 values of 56%, 62%, and 64%, respectively. Fig. 7 also reports the CC time for the run-time traces, which increases as the Q level decreases ($\sim$ 0.53 s at 12.5% Q), demonstrating that finer-grained analyses of the process dynamics require higher time to anomaly detection.

D. Threats to Validity

The results highlighted that both PD algorithms and the event log extraction strategy have an impact on the quality of the Petri nets and anomaly detection and time performance. In this section, we address a few additional internal and external threats to the validity of the approach.

1) Internal Threats to Validity: Since the experimental setup needs to collect the evolution of the fluid level across the tanks of the WDT, data quality can be affected by noise in fluid sensors. As a consequence, the evolution of fluid levels in normal cycles could differ. Furthermore, the injection of anomalies according to the scenarios in Table II requires disrupting the operation of the testbed by interfering with the flow of water. As such, operating on the testbed is subject to human error and may influence how the cycles are labeled. Finally, regarding the software, on the one hand, we have used the implementations provided by the `pm4py` packages, which could differ from those found in other frameworks, such as ProM. On the other hand, tuning the parameters of the machine learning algorithms can lead to slightly different results regarding detection performance.

2) External Threats to Validity: The proposed framework has been validated against a single ICPS case study, where the system dynamics have been collected by monitoring the evolution of fluid levels in the WDT tanks. However, the strategy used to characterize the states of the ICPS and to capture the state transitions in a Petri net can greatly differ based on the specific ICPS infrastructure. Therefore, further experiments with additional relevant ICPSs would be beneficial to fully validate the proposed framework.

VI. CONCLUSIONS AND FUTURE WORK

In this article, we presented a novel framework for the digital twin development of ICPSs based on PM, which combines data-driven and process-based analyses. The framework includes ICPS modeling, simulation, and monitoring through offline and online phases. The offline phase systematically extracts the most suitable Petri net model by using historical data collected as the ICPS operates in nominal conditions. The Petri net drives simulation and enables anomaly detection by building a classifier. Subsequently, the online phase involves run-time monitoring for anomaly detection, by using the Petri net and classifier obtained in the offline phase. We evaluated a proof-of-concept application to the WDT. The modeling results outlined that the ILP is the best PD algorithm since it achieves satisfying fitness ($\geq$0.93), precision ($\geq$0.76), generalization ($\geq$0.78), and simplicity ($\geq$0.56) for all the modeling configurations, and it also shows low time complexity for discovering Petri nets ($\leq$ 8.90 s) and checking traces ($\leq$ 0.57s), and always builds sound Petri nets. In addition, the simulation and anomaly detection results highlight the importance of adopting high-quality Petri nets to significantly improve anomaly detection and time performance. In fact, considering a specific modeling configuration, we showed that the ILP achieves satisfactory $F1$ ($\geq$74.3) and time complexity ($\sim$ 0.02 s) for all the classifiers, whereas the HM, which was the worst PD algorithm for that configuration, drops $F1$ performance as low as 0 with the baseline FT approach.

Based on the results of the experimentation, we believe the automatic building of digital twins through PD can play an essential role in the future evolution of ICPSs, by speeding up their development process, through the usage of artificial intelligence and machine learning. That also allows to manage their increasing complexity (i.e., size, distribution, and heterogeneity) and adaptation requirements to uncertainties, such as new threats and unknown operational scenarios due to rapidly evolving environments. In light of the above, and considering the threats to validity discussed in Section V-D, in the future, we plan to: develop more realistic simulations for better-informed what-if analyses, experiment with additional machine learning and deep learning algorithms to further generalize the anomaly detection results, and validate the framework against additional relevant ICPS case studies.

REFERENCES

[1] F. Tao, H. Zhang, A. Liu, and A. Y. Nee, "Digital twin in industry: State-of-the-art," *IEEE Trans. Ind. Informat.*, vol. 15, no. 4, pp. 2405–2415, Apr. 2019.

[2] F. Tao, B. Xiao, Q. Qi, J. Cheng, and P. Ji, "Digital twin modeling," *J. Manuf. Syst.*, vol. 64, pp. 372–389, 2022.

[3] H. M. Ramos et al., "Smart water grids and digital twin for the management of system efficiency in water distribution networks," *Water*, vol. 15, 2023, Art. no. 1129.

[4] H. J. Henriksen et al., "A new digital twin for climate change adaptation, water management, and disaster risk reduction (HIP digital twin)," *Water*, vol. 15, pp. 1–26, 2023, Art. no. 25.

[5] H. M. Ramos et al., "New challenges towards smart systems' efficiency by digital twin in water distribution networks," *Water*, vol. 14, 2022, Art. no. 1304.

[6] N. D. K. M. Eaty and P. Bagade, "Digital twin for electric vehicle battery management with incremental learning," *Expert Syst. Appl.*, vol. 229, 2023, Art. no. 120444.

[7] P. Conejos, F. Martínez Alzamora, M. Hervás, and J. Alonso Campos, "Development and use of a digital twin for the water supply and distribution network of valencia (Spain)," in *Proc. 17th Int. Conf CCWI*, 2019, pp. 1–4.

[8] L. Hui, M. Wang, L. Zhang, L. Lu, and Y. Cui, "Digital twin for networking: A data-driven performance modeling perspective," *IEEE Netw.*, vol. 37, no. 3, pp. 202–209, May/Jun. 2023.

[9] H. Huang, L. Yang, Y. Wang, X. Xu, and Y. Lu, "Digital twin-driven online anomaly detection for an automation system based on edge intelligence," *J. Manuf. Syst.*, vol. 59, pp. 138–150, 2021.

[10] P. T. Baboli, D. Babazadeh, and D. R. Kumara Bowatte, "Measurement-based modeling of smart grid dynamics: A digital twin approach," in *2020 10th Smart Grid Conf.*, 2020, pp. 1–6.

[11] M. Fahim, V. Sharma, T.-V. Cao, B. Canberk, and T. Q. Duong, "Machine learning-based digital twin for predictive modeling in wind turbines," *IEEE Access*, vol. 10, pp. 14184–14194, 2022.

[12] C. Quilodrán-Casas, V. L. Silva, R. Arcucci, C. E. Heaney, Y. Guo, and C. C. Pain, "Digital twins based on bidirectional LSTM and GAN for modelling the COVID-19 pandemic," *Neurocomputing*, vol. 470, pp. 11–28, 2022.

[13] S. Makridakis, E. Spiliotis, and V. Assimakopoulos, "Statistical and machine learning forecasting methods: Concerns and ways forward," *Plos One*, vol. 13, pp. 1–26, 2018.

[14] A. Rawal, J. McCoy, D. B. Rawat, B. M. Sadler, and R. S. Amant, "Recent advances in trustworthy explainable artificial intelligence: Status, challenges, and perspectives," *IEEE Trans. Artif. Intell.*, vol. 3, no. 6, pp. 852–866, Dec. 2022.

[15] W. M. P. van der Aalst and J. Carmona, *Process Mining Handbook*. Cham, Switzerland: Springer, 2022.

[16] J. Ko and M. Comuzzi, "A systematic review of anomaly detection for business process event logs," *Bus. Inf. Syst. Eng.*, vol. 65, pp. 441–462, 2023.

[17] A. Pecchia, I. Weber, M. Cinque, and Y. Ma, "Discovering process models for the analysis of application failures under uncertainty of event logs," *Knowl.-Based Syst.*, vol. 189, 2020, Art. no. 105054.

[18] A. De Benedictis, F. Flammini, N. Mazzocca, A. Somma, and F. Vitale, "Digital twins for anomaly detection in the industrial Internet of Things: Conceptual architecture and proof-of-concept," *IEEE Trans. Ind. Informat.*, vol. 19, no. 12, pp. 11553–11563, Dec. 2023.

[19] L. Faramondi, F. Flammini, S. Guarino, and R. Setola, "A hardware-in-the-loop water distribution testbed dataset for cyber-physical security testing," *IEEE Access*, vol. 9, pp. 122385–122396, 2021.

[20] F. de Lima Bezerra and J. Wainer, "Algorithms for anomaly detection of traces in logs of process aware information systems," *Inf. Syst.*, vol. 38, pp. 33–44, 2013.

[21] R. Accorsi and T. Stocker, "On the exploitation of process mining for security audits: The conformance checking case," in *Proc. 27th Annu. ACM Symp. Appl. Comput.*, 2012, pp. 1709–1716.

[22] D. Myers, S. Suriadi, K. Radke, and E. Foo, "Anomaly detection for industrial control systems using process mining," *Comput. Secur.*, vol. 78, pp. 103–125, 2018.

[23] R. Sarno, F. Sinaga, and K. R. Sungkono, "Anomaly detection in business processes using process mining and fuzzy association rule learning," *J. Big Data*, vol. 7, pp. 1–19, 2020.

[24] P. Singh et al., "Using log analytics and process mining to enable self-healing in the Internet of Things," *Environ. Syst. Decisions*, vol. 42, pp. 1–17, 2022.

[25] U. Singh, D. Gajjala, R. Khondoker, H. Gupta, A. Sinha, and O. Vyas, "Anomaly classification to enable self-healing in cyber physical systems using process mining," in *Proc. Int. Conf. Learn. Intell. Optim.*, 2023, pp. 1–15.

[26] B. Djordjević, E. Krmac, C.-Y. Lin, O. Fröidh, and B. Kordnejad, "An optimisation-based digital twin for automated operation of rail level crossings," *Expert Syst. Appl.*, vol. 239, 2024, Art. no. 122422.

[27] Y. Cui, F. Xiao, W. Wang, X. He, C. Zhang, and Y. Zhang, "Digital twin for power system steady-state modelling, simulation, and analysis," in *2020 IEEE 4th Conf. Energy Internet Energy Syst. Integration*, 2020, pp. 1233–1238.

[28] T. Zhang, X. Liu, Z. Luo, F. Dong, and Y. Jian, "Time series behavior modeling with digital twin for internet of vehicles," *EURASIP J. Wireless Commun. Netw.*, vol. 271, pp. 1–11, 2019.

[29] A. Jain, D. Nong, T. Nghiem, and R. Mangharam, "Digital twins for efficient modeling and control of buildings an integrated solution with scada systems," in *2018 Building Perform. Anal. Conf. SimBuild*, vol. 8, 2018, pp. 799–806.

[30] T. Brockhoff et al., "Process prediction with digital twins," in *2021 ACM/IEEE Int. Conf. Model Driven Eng. Lang. Syst. Companion*, 2021, pp. 182–187.

[31] A. Niati, C. Selma, D. Tamzalit, H. Bruneliere, N. Mebarki, and O. Cardin, "Towards a digital twin for cyber-physical production systems: A multi-paradigm modeling approach in the postal industry," in *Proc. 23rd ACM/IEEE Int. Conf. Model Driven Eng. Lang. Syst.: Companion*, 2020, pp. 1–7.

[32] I. Graja, S. Kallel, N. Guermouche, and A. H. Kacem, "BPMN4CPS: A BPMN extension for modeling cyber-physical systems," in *2016 IEEE 25th Int. Conf. Enabling Technol.: Infrastructure Collaborative Enterprises*, 2016, pp. 152–157.

[33] L. Faramondi, F. Flammini, S. Guarino, and R. Setola, "A hybrid behavior- and Bayesian network-based framework for cyber–physical anomaly detection," *Comput. Elect. Eng.*, vol. 112, 2023, Art. no. 108988.

[34] K. Ding, S. Ding, A. Morozov, T. Fabarisov, and K. Janschek, "On-line error detection and mitigation for time-series data of cyber-physical systems using deep learning based methods," in *2019 15th Eur. Dependable Comput. Conf.*, 2019, pp. 7–14.

[35] A. Rogge-Solti and M. Weske, "Prediction of remaining service execution time using stochastic petri nets with arbitrary firing delays," in *Proc. Serv.-Oriented Comput.: 11th Int. Conf.*, 2013, pp. 389–403.

[36] S. J. J. Leemans, D. Fahland, and W. M. P. van der Aalst, "Discovering block-structured process models from event logs - A constructive approach," in *Proc. Appl. Theory Petri Nets, Concurrency: 34th Int. Conf.*, 2013, pp. 311–329.

[37] L. Sánchez-González, F. García, J. Mendling, F. Ruiz, and M. Piattini, "Prediction of business process model quality based on structural metrics," in *Proc. Conceptual Model.–ER 2010*. Berlin, Heidelberg: Springer, 2010, pp. 458–463.

[38] I. Waspada, R. Sarno, E. S. Astuti, H. N. Prasetyo, and R. Budiraharjo, "Graph-based token replay for online conformance checking," *IEEE Access*, vol. 10, pp. 102737–102752, 2022.

[39] S. van Zelst, A. Bolt, M. Hassani, B. Dongen, and W. Aalst, "Online conformance checking: Relating event streams to process models using prefix-alignments," *Int. J. Data Sci. Analytics*, vol. 8, pp. 269–284, 2019.

[40] I. T. Jolliffe, *Principal Component Analysis for Special Types of Data*. NY, New York, USA: Springer, 2002, pp. 338–372.