An Orchestration System for Digital Twin Trustworthiness in Industrial Environments

N. Bicocchi, M. Fogli, C. Giannelli, E. Mingozzi, M. Picone and A. Virdis

IEEE Transactions on Industrial Informatics, 2026

Cite this as

N. Bicocchi, M. Fogli, C. Giannelli, E. Mingozzi, M. Picone and A. Virdis, "An Orchestration System for Digital Twin Trustworthiness in Industrial Environments," in *IEEE Transactions on Industrial Informatics*, doi: 10.1109/TII.2026.3692351.

An Orchestration System for Digital Twin Trustworthiness in Industrial Environments

Nicola Bicocchi, Mattia Fogli, *Member, IEEE*, Carlo Giannelli, *Senior Member, IEEE*, Enzo Mingozzi, *Member, IEEE*, Marco Picone, and Antonio Virdis, *Member, IEEE*

***Abstract*— Digital twins (DTs) are increasingly used in industrial environments to mediate interactions between applications and physical assets. However, existing orchestration approaches treat DTs as conventional software components, overlooking their cyber-physical nature and the resulting trustworthiness challenges. In such systems, violations of timing, resource, or data constraints can compromise the fidelity of the DT–physical twin (PT) relationship. In this paper, we propose a trustworthiness-driven orchestration system where trustworthiness is modeled as a multi-dimensional runtime invariant. The system implements a continuous control loop that monitors relevant metrics and enforces corrective actions, such as resource reallocation or migration across the cloud-to-edge continuum. We validate the approach through a Kubernetes-based prototype and large-scale simulations. Results show that the system effectively detects violations and restores DT–PT entanglement under dynamic conditions, demonstrating the feasibility of invariant-driven orchestration for industrial DTs.**

***Index Terms*— Digital Twins, Industry 4.0, Internet of Things, Orchestration, Trustworthiness**

I. Introduction

Early Internet of Things (IoT) implementations in industrial environments often relied on ad hoc approaches, where applications interacted directly with machines through their application programming interfaces (APIs) [1]. However, this approach introduced challenges due to the heterogeneity of machines and the potential for conflicting interactions from multiple applications [2], [3]. These limitations have driven the adoption of digital twins (DTs), which act as intermediaries between industrial machines, i.e., physical twins (PTs), and applications [4], [5]. Recent advances in microservices, service-mesh architectures, and quality of service (QoS) metrics have facilitated the adoption of DTs [6]. These approaches frequently treat DTs as conventional software components, overlooking their inherently cyber-physical nature [7], [8]. The cyber-physical nature of DTs implies that they not only process digital information but also influence physical processes. Neglecting this coupling can compromise *trustworthiness* in ways that standard software metrics cannot capture [6], [9].

Ensuring the trustworthiness of cyber-physical DTs in industrial environments poses significant challenges due to their real-time interaction with physical machines. For example, a DT controlling a robotic arm must be properly entangled with it to avoid production errors or collisions, a DT monitoring chemical reactions requires accurate and up-to-date sensor data to prevent the enforcement of unsafe commands, and a DT connected to multiple applications may be exposed to conflicting or malicious instructions that could compromise both digital and physical assets [2], [5]. Standard software and network metrics alone cannot capture these timing, safety, and security requirements. To address these challenges, we propose a DT orchestration system for trustworthiness in industrial environments. The scientific novelty of our work lies in introducing a trustworthiness-driven orchestration loop for DTs, which continuously monitors and coordinates multiple trustworthiness dimensions at runtime. Unlike conventional orchestration approaches that focus on software or network metrics, our system treats trustworthiness as a first-class, multi-dimensional runtime invariant: if a DT no longer satisfies the trustworthiness requirements, the system detects the violation and remediates it—for instance, by migrating the DT to a location that restores the invariant. This multi-dimensional orchestration loop represents a new principle for DT management in industrial environments, going beyond prior work that primarily considers static or software-centric orchestration [2], [8].

The contribution of this work is threefold. First, we extend the concept of trustworthiness to cyber-physical systems (CPSs), specifically targeting industrial environments [5], [7]. In doing so, we highlight how traditional notions of software trustworthiness—focusing on availability, reliability, and performance—are insufficient when systems directly interact with physical processes. Second, we propose and describe a dedicated orchestration system for DTs in industrial environments, in which trustworthiness is a core design criterion. The system enforces trustworthiness requirements across multiple dimensions. Third, we validate the proposed system through both experimental deployment in real industrial setups and

This work was supported in part by the Italian MUR in the framework of the CrossLab and the FoReLab projects (Departments of Excellence) and in part by the European Union – Next Generation EU, DATRUST PRIN 2022 PNRR Project under Grant P20225KTR4, CUP: I53D23006060001. *(Corresponding author: Antonio Virdis.)*

N. Bicocchi is with the Department of Engineering, University of Modena and Reggio Emilia, Modena 41125, Italy (e-mail: nicola.bicocchi@unimore.it).

M. Fogli and C. Giannelli are with the Department of Mathematics and Computer Science, University of Ferrara, Ferrara 44121, Italy (e-mail: {mattia.fogli, carlo.giannelli}@unife.it).

E. Mingozzi and A. Virdis are with the Department of Information Engineering, University of Pisa, Pisa 56122, Italy (e-mail: {enzo.mingozzi, antonio.virdis}@unipi.it).

M. Picone is with the Department of Sciences and Methods for Engineering, University of Modena and Reggio Emilia, Reggio Emilia 42122, Italy (e-mail: marco.picone@unimore.it).

large-scale simulations. The experimental evaluation demonstrates the system effectiveness in maintaining trustworthiness under diverse operational conditions, while the simulations allow us to assess scalability, performance, and robustness in scenarios that would be challenging to reproduce physically. Together, these evaluations provide empirical evidence of the system capability to ensure trustworthy operations of DTs in industrial environments.

The remainder of the paper is organized as follows. Section II provides background and motivation for DTs in industrial environments. Section III introduces the foundational concepts of DT trustworthiness. Section IV details the proposed orchestration system, including its design and implementation. Section V presents the evaluation, combining experimental validation and large-scale simulations. Finally, Section VI discusses conclusions and outlines directions for future research.

II. Background and Related Work

This section provides an overview of industrial DT deployments and reviews the related work most pertinent to our approach.

A. Background on Industrial DT Deployments

Modern industrial environments are typically organized into three layers: shop floor, plant, and enterprise (see Fig. 1a). The shop floor layer is further divided into three sub-layers and focuses on *field* industrial automation. This includes sensors and machines for measurement and actuation, programmable logic controllers (PLCs) that directly *control* machines, and Supervisory Control and Data Acquisition (SCADA) systems along with human-machine interfaces (HMIs) for *monitoring and supervision*. The plant layer is responsible for *managing* manufacturing processes and machine operations, typically using a manufacturing execution system (MES). The enterprise layer handles decision-making for *planning* operations, relying on an enterprise resource planning (ERP) system to gather data from various business assets, such as supply chains and production processes. As such, modern plants are characterized by various logical domains and networks, which may differ significantly in terms of both quantitative and qualitative requirements. These include computers directly connected to machines, edge nodes close to machines, or small-scale private data centers within control rooms. In this distributed environment, DTs are increasingly used as intermediaries between machines and applications. Specifically, DTs are deployed based on their intended purpose, the devices they interact with, and the required performance (see Fig. 1b).

For example, DTs that interact directly with machines and PLCs are traditionally deployed close to them, such as on-premises edge nodes, to achieve low latency and high data throughput. Instead, DTs interacting with MESs or ERPs are usually deployed within the control room, where more resources are available but at the cost of increased latency when interacting with machines. In such a scenario, DTs are deployed not only on-premises (on edge nodes or within the control room), but also on remote cloud nodes, thus trading abundant and scalable resources for potential network congestion and increased latency. Finally, to achieve a trade-off between performance and latency, it is also possible to run DTs on public nodes close to the plant, such as Multi-access Edge Computing (MEC) nodes managed by telecom operators.

Note that, on the one hand, DT orchestration across the cloud-to-edge continuum offers greater flexibility and can enable deployment solutions that would not be feasible otherwise. On the other hand, due to the cyber-physical nature of DTs, it is essential to adopt appropriate techniques for monitoring and assessing the suitability of nodes that could host and run DTs, as well as the underlying network interconnecting PTs, DTs, and applications.

Several metrics have been proposed to monitor the performance of networked devices. For example, the *Overall Equipment Effectiveness (OEE)* metric, originating from manufacturing, offers a comprehensive measure of production equipment efficiency by integrating availability, performance, and quality [10]. In contrast, *quality of experience (QoE)* focuses on the subjective experience of users interacting with digital services [11], *quality of information (QoI)* evaluates the intrinsic quality of data itself—critical for domains reliant on accurate and reliable information [12]—and *age of information (AoI)* captures the freshness or timeliness of data in real-time communication systems [13]. Similarly, but specifically designed for cyber-physical scenarios, the *Overall Digital Twin Entanglement (ODTE)* metric [14] has recently been proposed to measure in a concise yet expressive way the DT-PT entanglement (i.e., how well the DT represents the actual state of its PT, which in our case is an industrial machine). It is conceived as the product of three factors: *timeliness* (T), i.e., the distribution in time of status updates received from the PT, *reliability* (R), i.e., the ratio of the received state updates to the expected ones, and *availability* (A), i.e., the expected up-time of the PT from the perspective of the DT.

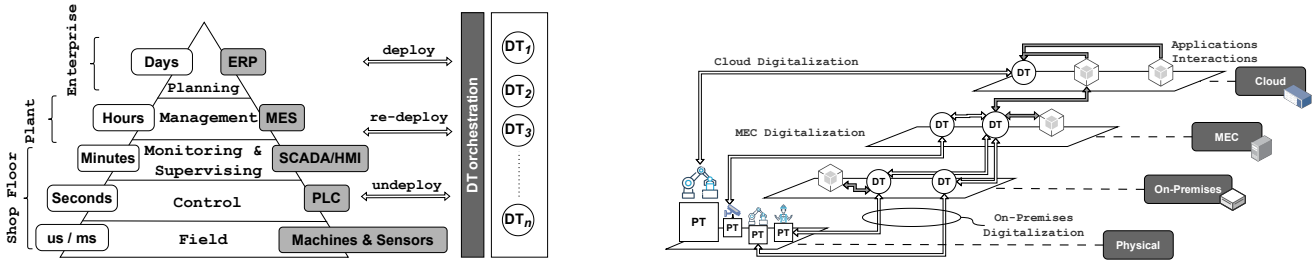
$$ODTE = T \times R \times A \tag{1}$$

It is worth noting that, while the ODTE metric captures important aspects of entanglement quality, it does not encompass the full range of potential challenges. For example, a DT might have access to sufficient computational resources and receive the required data in a timely manner, yet still provide an unreliable representation of the corresponding PT. This can arise due to various factors, such as human errors introducing inaccuracies, security breaches compromising data integrity or the DT itself, or insufficient quality of the upstream PT data, which may be incomplete or erroneous.

Given these limitations, we advocate for a holistic management approach based on the concept of trustworthiness in industrial settings. The aim is to orchestrate DTs by accounting for a combination of qualitative and quantitative factors, thereby ensuring more reliable and robust digital representations, as further elaborated in the following sections.

B. Background on Digital Twins

The role of DTs has recently gained renewed attention from both scientific and industrial communities. Research has focused on clarifying definitions and responsibilities while



(a) Hierarchical structure of industrial layers

(b) Distributed DT industrial environment

Fig. 1: Industrial environment organization: (a) hierarchical layering from field devices to enterprise planning, and (b) distributed orchestration framework spanning the cloud-to-edge continuum.

identifying challenges and opportunities, especially within IoT and Industrial Internet of Things (IIoT) applications [15]–[17]. DTs are increasingly regarded as integral components of CPS architectures, modeling physical assets and machines [18], encapsulating computational modules of physical components [19], and supporting patterns in the Reference Architecture Model Industrie 4.0 (RAMI 4.0) ecosystem, including manufacturing and administrative processes [20], [21]. Within this ecosystem, DTs are envisioned as flexible, adaptive software entities capable of enabling context-aware, autonomous, and adaptive applications across diverse domains [22].

Siqueira and Davis [23] reviewed the use of microservices for DTs, highlighting how this paradigm supports software infrastructures aligned with Industry 4.0. They note that while containerization is mature in cloud environments, embedded devices still face limited support, and full interoperability with orchestrators such as Kubernetes remains critical for coordinated DT deployment. Azarmipour et al. [24] explored dynamic management by modeling MESs and PLCs as compositions of microservices, while Damjanovic and Behrendt [25] emphasized leveraging containerized microservices for industrial DT deployment. Core components in these solutions often rely on technologies such as Apache Kafka, RabbitMQ, Elasticsearch, Grafana, and Hadoop. Despite these contributions, the formalization of software modeling and design patterns that capture the core properties and responsibilities of DTs remains an open research challenge [22], [26].

Trustworthiness in DTs has been addressed from multiple perspectives. Suhail et al. [27] combined data provenance with blockchain to guarantee trustworthy data in IIoTs settings, enabling reliable predictive maintenance. Sasikumar et al. [28] proposed blockchain-enabled DTs to support the emerging fifth industrial revolution. Vitale et al. [29] introduced a framework for industrial CPSs that integrates data-driven and process-based analyses, enhancing explainability and trustworthiness compared to purely data-driven methods. Bicocchi et al. [30] investigated the essential characteristics of DT trustworthiness, emphasizing mechanisms to enforce end-to-end trustworthiness while connecting applications with PTs.

While these works provide valuable insights on DT trustworthiness from the perspectives of data integrity, design, and general properties, they do not tackle the challenge of establishing and maintaining trustworthiness in complex industrial environments, where multiple heterogeneous DTs coexist to link applications with physical machinery.

III. DT Trustworthiness in Industry 4.0

The limitations affecting the ODTE metric (see Section II) underscore the importance of considering an extended set of aspects to ensure that a DT consistently mirrors its PT. As highlighted in [9], a comprehensive trustworthiness evaluation should account for several critical factors, including both quantitative and qualitative ones. To address the multifaceted nature of trustworthiness in industrial DT deployments, we identify six key requirements that drive the design of our orchestration system architecture. These requirements comprise quantitative dimensions—Entanglement Awareness (R1), Variable Load Resilience (R2), and Cloud-to-Edge Continuum Mobility (R3)—and qualitative dimensions—Declarative Description (R4), Observability (R5), and Security (R6). Together, they define the core architectural components and mechanisms required to ensure DTs are not only operationally efficient but *trustworthy* as a whole. Table I summarizes these requirements, highlighting their main objectives within industrial environments. The remainder of this section elaborates on each requirement, establishing the design principles for the orchestration system detailed in Section IV.

A. Entanglement Awareness

The first requirement, *Entanglement Awareness* (R1), concerns using a metric, e.g., the ODTE, to assess whether DTs accurately mirror their PTs. In this respect, industrial systems are typically organized into hierarchical zones, each with its own set of operational requirements and constraints. The interaction between DTs and PTs across different zones requires an architecture that understands these interdependencies and how they may influence entanglement monitoring and measurements. For instance, a DT overseeing an assembly line should be aware of other DTs operating in adjacent stations to ensure synchronized operations and prevent conflicts. A DT controlling a robotic arm in a car manufacturing plant would be impacted by the availability of parts managed by an upstream DT responsible for conveyor belt logistics. Effectively managing these dependencies helps minimize downtime and enhance coordination.

TABLE I: Trustworthiness requirements for industrial DT deployments

ID	Requirement	Key Objectives
R1	Entanglement Awareness	Monitor and assess DT–PT entanglement quality through metrics; manage inter-DT dependencies across hierarchical zones; ensure coordinated operations
R2	Variable Load Resilience	Dynamically allocate computational and memory resources based on operational demands; support augmentation functions; maintain real-time control loops under varying loads
R3	Cloud-to-Edge Continuum Mobility	Enable seamless DT migration across distributed nodes; optimize placement considering latency, resources, and operational constraints; adapt to evolving demands
R4	Declarative Description	Provide standardized, structured descriptions of PTs, DTs, and applications; distinguish deployment specifications from DT characteristics; enable automated deployment
R5	Observability	Correlate digital and physical performance indicators; integrate real-time monitoring with historical knowledge; support root-cause analysis and proactive decision-making
R6	Security	Embed security from the outset; implement role-based access control; ensure encrypted communication; systematically manage security policies across zones

B. Variable Load Resilience

Variable Load Resilience (R2) refers to the ability to manage computational and memory resources based on actual needs, adapting to changes in requirements and functionalities, as well as the incorporation of augmentation functions within DTs. In industrial environments, the amount of requests sent to DTs can fluctuate based on operational needs, with varying computational and network resource requirements (e.g., adding an intelligent functionality to detect anomalies in accelerometer data). Unlike traditional information technology (IT) systems, industrial environments are designed for operational technology (OT), which typically operates under fixed resource constraints. DTs should be capable of dynamically adjusting to these fluctuations without disrupting industrial operations. For example, if data processing demands rise due to predictive maintenance analytics running on multiple DTs, the system should efficiently reallocate computing resources. Furthermore, augmentation functions can be introduced to extend the capabilities of DTs, ensuring that real-time control loops are not affected. Load balancing mechanisms are critical to prevent overload on network segments that are essential for robotic controllers.

C. Cloud-to-Edge Continuum Mobility

Cloud-to-Edge Continuum Mobility (R3) enables DT instances to seamlessly migrate between distributed nodes (e.g., cloud, edge, and intermediate ones) based on operational needs. The system should be capable of understanding infrastructure layers, OT constraints, and asset monitoring to determine the optimal placement of DTs and applications. This mobility should consider varying governance models, stakeholder requirements, and infrastructure limitations. For instance, in a smart factory, a DT initially deployed in the cloud for artificial intelligence (AI) model training might need to migrate to an edge node for real-time inference to reduce latency. However, the edge node may not have sufficient computing resources to sustain the model (R2). Conversely, keeping the DT in the cloud preserves compute capacity but increases latency, which may break entanglement (R1). Therefore, placement decisions must account for multiple trustworthiness dimensions simultaneously. When orchestrating multiple DTs, placing all of them near their PTs may not be feasible due to limited resources, network constraints, and competing operational priorities. The orchestration system must therefore balance proximity to PTs with efficient resource utilization while adhering to industrial cybersecurity policies, operational constraints, and awareness of the target deployment characteristics. DTs should be flexible enough to move across nodes, ensuring they can adapt to evolving operational demands. Additionally, mobility should take into account the needs of different stakeholders, governance models, and the availability of cloud-native technology stacks to guarantee smooth and effective integration across diverse environments.

D. Declarative Descriptions

Declarative Descriptions (R4) provide a standardized, structured representation of DT requirements that extends beyond the configuration and characteristics of individual DTs to include associated applications and PTs. This approach ensures a consistent and comprehensive modeling of all relevant elements within the industrial environment. Additionally, a clear separation should be maintained between the description of a DT itself and the specifications concerning deployment, dependencies, and execution parameters of industrial applications. For example, a DT representing a machine should include not only metadata specifying permissible tool changes, power limits, and connectivity with ERP systems, but also a description of how associated applications interact with the DT and other components, such as OT modules or HMI devices. Such a uniform and structured representation is essential for enabling advanced functionalities, supporting automated deployment, ensuring operational consistency, and facilitating seamless integration across industrial ecosystems.

E. Observability

Observability (R5) provides deep insights into whether a DT is operating correctly and, if not, helps to identify the underlying causes. This can be achieved by integrating real-time monitoring with knowledge-based systems. Unlike conventional IT observability, which primarily focuses on software performance metrics, industrial observability must directly correlate digital metrics with physical performance indicators. Such a holistic approach ensures that both real-time operational data and historical knowledge contribute to

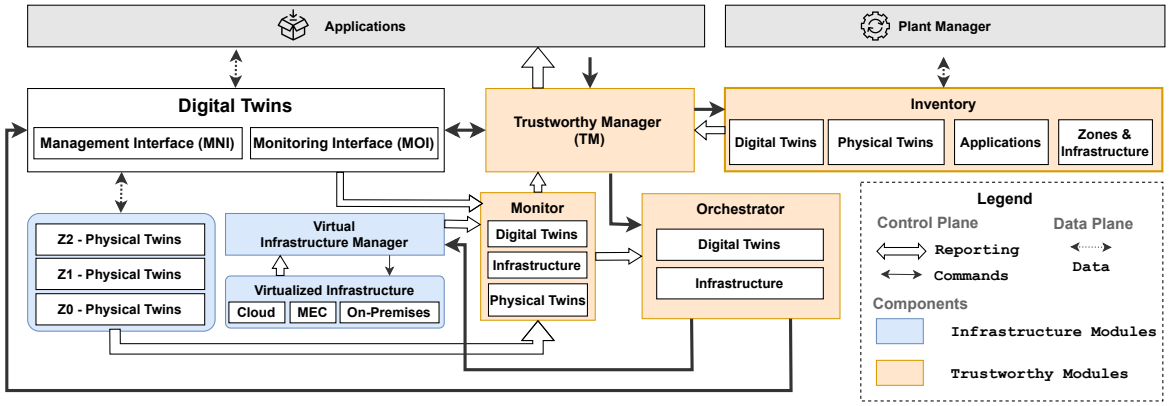


Fig. 2: Overview of the proposed industrial orchestration system for DT trustworthiness.

system awareness, diagnostics, and optimization. For example, a DT managing an industrial cooling system should combine real-time temperature measurements, fan speed telemetry, and predictive alerts with historical maintenance records and contextual knowledge. Full observability is especially critical in hierarchical and highly interconnected industrial ecosystems, where a single malfunction or performance degradation can propagate across multiple assets and zones. A comprehensive observability strategy is therefore essential to ensure reliability, efficiency, and safety, making it a core requirement for trustworthy DT deployments.

F. Security

Security (R6) must be integrated into industrial DT platforms from the outset, reflecting the critical nature of industrial operations. This involves ensuring that DTs do not compromise the underlying platform, and vice versa. Unlike conventional IT environments, industrial systems operate under strict safety and reliability requirements, making robust cybersecurity essential to prevent operational disruptions and potential threats. A structured, uniform approach should be adopted to systematically document and manage security policies, access control mechanisms, and vulnerabilities across different zones, considering their heterogeneous characteristics and operational constraints. For example, a DT interacting with a PLC should enforce role-based access control, maintain an up-to-date security inventory, and use encrypted communication channels to mitigate cyber threats such as unauthorized command injection or data tampering. By embedding security at every level—from DT models to their interactions with PTs—industrial environments can achieve reliable deployments that safeguard both digital and physical assets.

IV. Industrial Orchestration System for DT Trustworthiness

Based on the above considerations, we propose a trustworthiness-driven orchestration system for industrial environments that treats trustworthiness as a first-class, multi-dimensional runtime invariant, continuously monitoring and enforcing the requirements defined in Section III across the DT lifecycle (light-gray modules in Fig. 2). The system consists of functional components (dark-gray modules in Fig. 2) that receive DT-instantiation requests from applications, manage the lifecycle of DTs, and monitor their execution to ensure that trustworthiness requirements are met.

Applications can request services from a DT associated with a given PT (e.g., industrial machinery) and specify trustworthiness requirements accordingly. These requirements can range from performance targets (e.g., maximum response time) to deployment constraints (e.g., the DT must execute within zone Z2). These specifications are received by the Trustworthiness Manager (TM), which identifies a suitable DT from those available in the Digital Twin Applications Registry for the given PT. Each identified DT may include additional trustworthiness requirements to be handled by the TM, such as the requirement to run on graphics processing unit (GPU)-powered machines.

Once all requirements have been processed and a suitable DT candidate has been identified, the TM forwards an instantiation request to the Orchestrator, specifying the subset of computing nodes that satisfy the overall trustworthiness requirements. The Orchestrator gathers status information from these nodes via the Monitor module, which collects reports from each node and serves as a structured repository for metrics, logs, and events related to security threats, network performance, resource utilization, and DT execution. Based on the trustworthiness requirements provided by the TM and the real-time resource status obtained from the Monitor, the Orchestrator selects the most suitable node for DT deployment. The actual instantiation depends on the chosen node and can be carried out either through a *recipe*—such as a shell script or Ansible playbook executed directly on the target server—or by interacting with a Virtual Infrastructure Manager (VIM), for instance using a Helm chart in Kubernetes-orchestrated environments.

Once instantiated, DTs interact directly with their corresponding PTs as well as with applications, providing the requested services. During execution, the TM monitors the status of each DT through the Monitor module, which periodically collects performance data via the monitoring interface (MOI). By analyzing the information stored in the Monitor, the TM

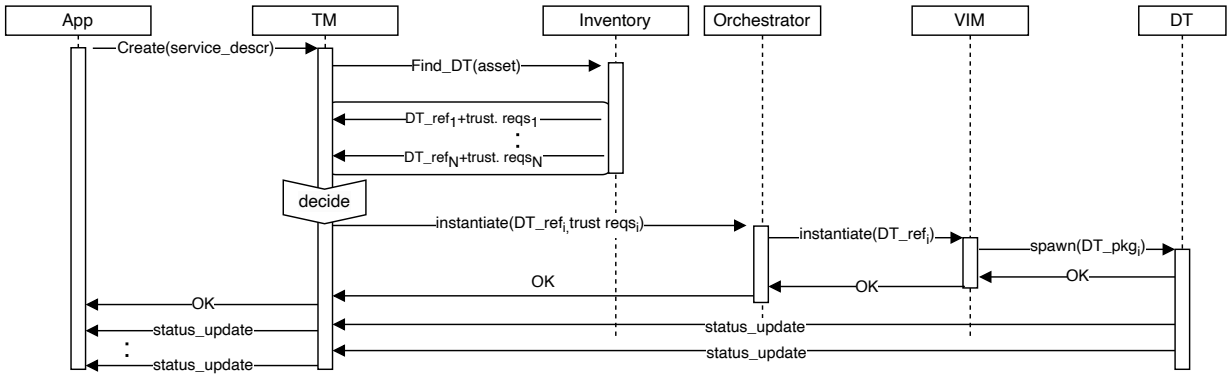


Fig. 3: Sequence diagram for the DT creation procedure.

can observe DT behavior, assess overall system performance, and detect anomalies over time, ensuring that the trustworthiness requirements specified by the application are continuously met. If any issues are detected, the TM may attempt to reconfigure the DT, for instance by adjusting resource allocation, modifying execution parameters via the management interface (MNI), or migrating the DT to another node. Should the problem persist, the TM may deallocate the DT and notify the application. This design relieves applications from the need to monitor DT execution or verify compliance with trustworthiness requirements. In this framework, any malfunctioning DT is considered untrusted and is prevented from interacting with applications. Additionally, the TM can leverage performance metrics from other computing or networking nodes to detect or even predict potential malfunctions in DT operations.

To clarify the interactions among the entities in the proposed platform, Fig. 3 illustrates the sequence diagram for the DT creation procedure. The process begins when an application issues a DT service request, specifying the service name and the target PT. The TM first processes this request by querying the inventory for DTs capable of handling the specified PT. The inventory responds with a list of references to suitable DTs, along with their associated trustworthiness requirements, which may include computing capacity, required capabilities (e.g., GPUs), maximum status-refresh intervals, and deployment constraints (e.g., restricted to zone Z1). Importantly, at this stage and in the subsequent steps, only references to DTs (such as their identifiers) are shared among nodes (see DT_ref_i in Fig. 3). Upon receiving the list, the TM selects the most appropriate DT candidate and issues an instantiation request to the Orchestrator. By doing so, the TM implicitly defines the subset of computing nodes where the DT may be deployed—for example, restricting execution to on-premises nodes or excluding cloud resources. This selection is guided by the specific requirements of the DT, particularly those related to updates of the state, which can only be satisfied by a subset of nodes due to their absolute or residual capacity.

The Orchestrator receives a reference to the selected DT along with its associated requirements and determines the actual node where the DT will be deployed. Here, the Orchestrator makes the final node selection—taking into account current resource utilization and availability—whereas the TM has previously identified only a subset of candidate nodes. This two-step decision-making process is necessary because the TM evaluates trustworthiness requirements across multiple DTs using a high-level view of the computing infrastructure, while the Orchestrator possesses detailed, up-to-date knowledge of the resource usage on each individual node.

The instantiation of the DT is carried out by the VIM, which uses the received DT reference to retrieve the corresponding software package (DT_pkg_i in Fig. 3) and deploys it on available nodes, depending on the deployment choice made by the Orchestrator. Once the DT is successfully spawned, all relevant entities, including the application involved in the creation process, are notified. Subsequently, the DT begins issuing periodic status updates to the TM.

In summary, the TM enforces a continuous control loop of observation, assessment, decision-making, and actuation to uphold trustworthiness requirements. It collects telemetry on *entanglement*, *load*, and *security* events, evaluates compliance, and triggers corrective actions as necessary: DTs may be migrated across the *cloud-to-edge continuum*, execution parameters adjusted via *declarative* interfaces with full *observability*, or, when integrity cannot be ensured, DTs decommissioned and prevented from further interaction with applications.

V. Evaluation

The evaluation of the proposed orchestration system is conducted in two stages. First, we perform a small-scale proof-of-concept prototype evaluation. Next, we carry out a system-level simulation at scale using an event-driven simulator.

The orchestration system treats trustworthiness as a first-class, multi-dimensional runtime invariant (see Table I). Our experiments focus on the three quantitative dimensions: Entanglement Awareness (R1), Variable Load Resilience (R2), and Cloud-to-Edge Continuum Mobility (R3). The ODTE metric monitors entanglement between each DT and its PT (R1): if entanglement is lost, the DT is no longer considered trustworthy. Violations may result from network latency, insufficient compute resources after a model upgrade (R2), or suboptimal placement. The system resolves such violations by migrating the DT to a location in the cloud-to-edge continuum (R3) that restores the trustworthiness invariant. The three

qualitative dimensions—Declarative Description (R4), Observability (R5), and Security (R6)—serve as structural enablers for this runtime control loop, addressed through design and implementation choices detailed in Section V-A.

Our evaluation emphasizes trustworthiness-driven orchestration under heterogeneous DT requirements, particularly the communication–compute trade-offs across on-premises, MEC, and cloud nodes that affect ODTE and timeliness. Active cyber-attacks and safety certification are beyond the scope of this work and are discussed as future directions.

A. Prototype

The prototype evaluation has two main objectives. First, we present an implementation of the industrial orchestration system for DT trustworthiness detailed in Section IV, ensuring alignment with the trustworthiness requirements outlined in Section III. Second, we demonstrate that the system can enforce corrective actions when the trustworthiness requirements are not met in a cloud-to-edge continuum scenario.

1) Implementation: We implemented the orchestration system described in Section IV using open-source, production-grade tools. The prototype leverages Kubernetes to provide VIM-like functionalities, ranging from declarative resource management to DT deployment. The orchestration system requires clear abstractions to describe a DT, including its properties, behavior, trustworthiness requirements, and deployment constraints (R4). We implemented this by extending Kubernetes with custom resource definitions (CRDs), defining custom data structures for DTs. Applications can request DTs declaratively via the Kubernetes APIs. The TM and Orchestrator are implemented as Kubernetes operators: the TM handles first-stage decision-making, selecting a DT that satisfies the application request, while the Orchestrator handles second-stage decision-making, choosing a suitable node for deployment. The runtime trustworthiness loop correlates digital and physical performance indicators (R5). This is achieved through Kubernetes, which collects infrastructure metrics, and Prometheus, Kiali, and Jaeger, which provide DT-level metrics, service-mesh visualization, and distributed tracing. The Orchestrator converts a DT resource into native Kubernetes objects for deployment, using node affinity to select an appropriate node along the cloud-to-edge continuum. Security requirements (R6), including access control and encrypted communication, are enforced using Kubernetes role-based access control (RBAC) and Istio. Once deployed, the orchestration system continuously monitors the DT to maintain the trustworthiness invariant at runtime. Importantly, this architecture is technology-agnostic; the prototype represents one possible instantiation of the proposed approach.

The implemented DTs are built using the White Label Digital Twin (WLDT) open-source framework [31], which enables the design, deployment, and management of modular and adaptable DTs. WLDT, a Java-based multithreaded stack, supports the definition of DTs, the implementation of shadowing and digitalization processes, and seamless interoperability with external applications. In our prototype, communication between physical and digital components is established via the Message Queuing Telemetry Transport (MQTT) protocol. To enhance trustworthiness, we extended the framework with a monitoring module that computes the ODTE metric, providing real-time assessment of entanglement between the physical and digital counterparts. Additionally, we integrated the MOI and MNI interfaces, enabling dynamic configurability and observability of the DT lifecycle in alignment with target operational requirements. For scalability and orchestration, each DT was containerized, ensuring lightweight, portable, and dynamic execution within the orchestration system.

TABLE II: Prototype evaluation: Experimental parameters

Epoch	**Latency**	T_d	**Model**	**CPU(s)**
E1 - Cloud	100 ms ± 20 ms	1 s	Simple	2000 mCPU
E2 - Cloud	100 ms ± 20 ms	100 ms	Simple	2000 mCPU
E3 - Edge	5 ms ± 1 ms	100 ms	Simple	250 mCPU
E4 - Edge	5 ms ± 1 ms	100 ms	Complex	250 mCPU
E5 - MEC	25 ms ± 2 ms	100 ms	Complex	1000 mCPU

The DTs digitalize a set of PTs emulated through a dedicated software simulator designed to replicate industrial machine behavior. The PT emulator, implemented in Python, offers high configurability in terms of data (via real dataset replay or generated values with specific random distributions), communication protocols (here via MQTT), serialization methods (JavaScript Object Notation (JSON) and YAML Ain't Markup Language (YAML)), message rate, and management of physical actions through received requests.

2) Setup: We provisioned the testbed in a private OpenStack environment using Terraform and configured it with Ansible. The setup comprised five Ubuntu 22.04 LTS instances: one (m1.small: 1 vCPU, 2 GiB memory) hosted the experiment scripts, while the remaining four formed the cluster. Within the cluster, one instance (m1.medium: 2 vCPU, 4 GiB memory) managed the control plane, and three instances (m1.large: 4 vCPU, 8 GiB memory) served as worker nodes.

The cluster was equipped with CRI-O as the container runtime, Kubernetes for container orchestration, Flannel for networking, and Istio as the service mesh. For monitoring and observability, Prometheus was deployed for automatic metric scraping, Kiali for service mesh visualization, and Jaeger for distributed tracing. Mosquitto facilitated MQTT communication between PTs and DTs, and Chaos Mesh enabled fault injection experiments. Finally, the DTs were deployed using Helm, the Kubernetes package manager.

3) Results: The evaluation of the prototype consisted of five epochs. Table II details the experimental parameters for each epoch. In particular, we used Chaos Mesh to inject the desired network latency, Kubernetes to vary CPU allocation, and the DT API to change the model complexity. The network latency was injected into the PT-DT communication link. The values were chosen to emulate the network conditions the DT would plausibly experience in a realistic cloud-to-edge continuum scenario [32], [33]. The PT published its state updates as MQTT messages—1 state update per second. We emulated the complexity of the model by making DTs find a given number of primes. Lastly, T_d represents the desired timeliness, i.e., how fresh digital states are required to be.

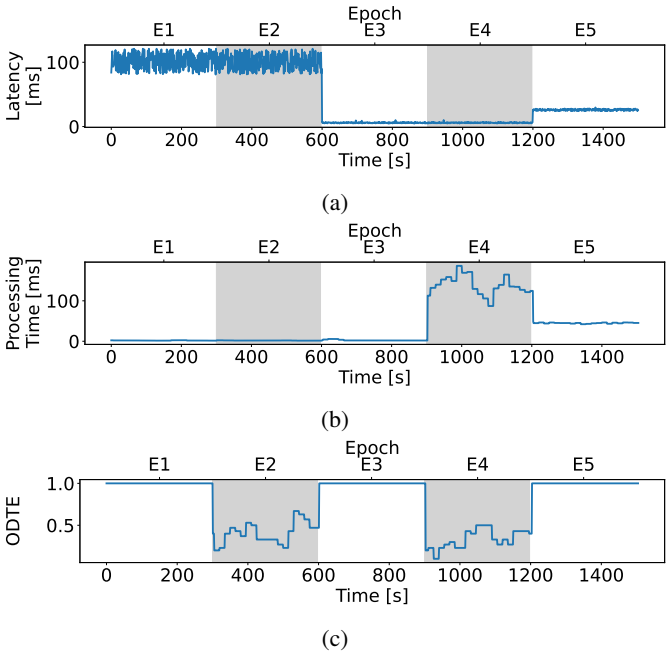


Fig. 4: Prototype evaluation: Network latency (a), processing time (b), and ODTE (c).

For example, $T_d = 100\,\text{ms}$ means that the DT is expected to produce its (digital) state after at most 100 ms upon reception of a (physical) state update sent by its counterpart.

Fig. 4 shows the network latency, processing time, and ODTE as experienced by the DT throughout the evaluation. As anticipated in Section II, the ODTE metric is a domain-specific measure for DTs quantifying how effectively the DT represents its PT in terms of expected performance. Processing time measures how long the DT takes to compute a digital state. This is influenced, on the one hand, by the number of available central processing units (CPUs), and, on the other hand, by the model complexity. Note that the higher the processing time, the longer it takes to produce a digital state. Processing time and network latency collectively impact T, a key factor in the ODTE metric (see Section II). The alternating background colors in Figs. 4a, 4b, and 4c visually indicate the transition from one epoch to the next. Each color shift marks the beginning of a new epoch.

During the first epoch (E1), the orchestration system prototype deployed the DT in the cloud, as this location satisfied the application requirements: a T_d of 1 s and a simple model (1K primes). In this phase, the ODTE metric remained at 1, indicating full entanglement between the DT and its PT. In the second epoch (E2), the required T_d was reduced to 100 ms. Given a cloud latency of 100 ms $\pm$ 20 ms, this target could not be met, causing the ODTE to drop—a trustworthiness violation (R1). The system identified network latency as the root cause and migrated the DT from the cloud to an on-premises edge node (R3), restoring entanglement. By the third epoch (E3), the ODTE returned to 1. Although edge nodes offer lower latency, they have constrained resources (from 2000 mCPU in the cloud to 250 mCPU at the edge). Nevertheless, this allocation was sufficient for the requested model. In the fourth epoch (E4), increasing the model complexity (from 1K to 10K primes) saturated the available compute resources, reducing entanglement. Here, the root cause was compute saturation at the edge (R2), not network latency. Unlike conventional orchestrators, which might scale up (move the DT to any node with sufficient resources) or scale out (spawn a replica), the proposed system preserves the cyber-physical relationship: relocating the DT to the cloud would resolve compute limitations but break entanglement. Instead, the system migrated the DT to the MEC (R3), providing enough compute while maintaining acceptable latency—a decision that jointly considers multiple trustworthiness dimensions. In the fifth epoch (E5), the corrective action successfully restored the trustworthiness invariant, with the ODTE consistently at 1.

TABLE III: Main simulation parameters

Parameter name	Value
#repetitions	5
sim duration	100 s
warm-up time	5 s
5G tx power (user)	17 dBm
5G tx power (base station)	23 dBm
5G bandwidth	5 MHz
avg network latency:	
on premises	5 ms $\pm$ 1 ms
edge	25 ms $\pm$ 2 ms
cloud	100 ms $\pm$ 20 ms
PT packet size	80 Bytes
Target timeliness (stringent)	0.6 s
Target timeliness (heavy)	1.6 s
#PTs	50

B. Simulation

To assess the performance of the orchestration system at scale, we used OMNeT++ as the event-based system-level simulator. The simulation framework includes comprehensive models of PTs and DT-based services, of the computing nodes hosting them, and of the underlying communication network. Key aspects of the simulation setup are detailed below.

We consider two DT archetypes inspired by realistic industrial workloads. The first represents latency-sensitive lightweight DTs (*stringent DTs*), typical of monitoring/control functions that require timely updates but limited computation. The second represents compute-intensive DTs with relaxed timeliness constraints (*heavy DTs*), typical of analytics or augmented DT functions that demand more processing while tolerating higher end-to-end delay. DTs and applications are hosted by computing nodes which can be located either on-premises, at MEC, or in the cloud, allowing the exploration of diverse use cases and deployment scenarios. Consistent with the prototype presented in Section V-A, each DT running on a computing node is allocated a fixed amount of computing power, expressed in mCPUs. Computing nodes across the continuum offer varying computational capacities, leading to different amounts of allocable per-DT CPU resources. Consequently, the actual CPU allocated to each DT depends both on the type of node (i.e., cloud, MEC, or on-premises) *and* on the number of DTs concurrently running on that node.

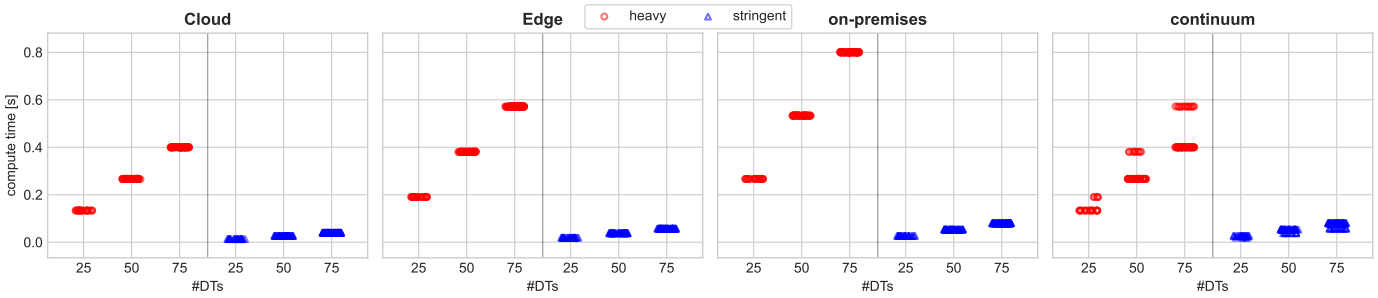


Fig. 5: Measured computation time for N DTs running heavy models and N running stringent ones, with N ranging from 25 to 75. Each dot represents the average performance of a DT.

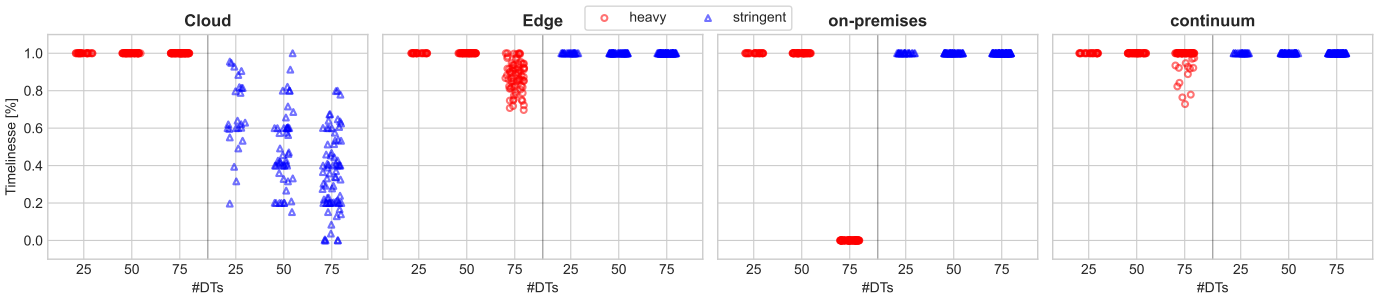


Fig. 6: Measured timeliness for N DTs running heavy models and N running stringent ones. N ranges from 25 to 75. Each dot represents the average performance of a DT.

Interconnections between PTs and computing nodes are based on either Ethernet for wired connections or private 5G networks for wireless connections. For this purpose, the simulator incorporates a detailed model of the Transmission Control Protocol (TCP)/Internet Protocol (IP) stack based on the INET library, and of 5G technologies including advanced scheduling and QoS support capabilities based on Simu5G [34]. Beyond being a general-purpose 5G user-plane simulator, the OMNeT++/INET+Simu5G toolchain has been recently used in multiple scientific works to evaluate industrial 5G scenarios, including smart-factory private-5G deployments with heterogeneous multi-flow QoS requirements as well as 5G–TSN/timing-sensitive settings for industrial automation [35]–[37]. To assess the robustness of the proposed orchestration strategy under realistic wireless load, we inject additional background traffic on the private 5G segment. In particular, we follow a 3GPP-compliant user-plane traffic model from 5G-ACIA guidelines [38], parameterized by message size, transfer interval, and (average/peak) data rate. We instantiate six HMI-related background traffic flows.

We compare four possible deployment scenarios depending on where DTs are deployed and on the communication technology they use:

- Cloud: DTs are hosted in the cloud, PTs use a private 5G network to communicate with DTs;
- MEC: DTs are hosted in the MEC, PTs use a private 5G network to communicate with DTs;
- On-premises: DTs are hosted on local servers, PTs use a wired network to communicate with DTs;
- Continuum: DTs are hosted on cloud, MEC, or on-premises depending on their communication requirements, PTs use accordingly either a wired or a private 5G network to communicate with DTs.

We simulate a number of PTs ranging from 50 to 150, deployed on a 100 m $\times$ 200 m floorplan. Half of the PTs are served by *stringent* DTs, and half are served by *heavy* DTs. To achieve statistical soundness, we run 5 independent replicas of each simulation scenario, with each run lasting 100 s. A summary of the simulation parameters is reported in Table III.

Fig. 6 reports the measured timeliness for both *stringent* and *heavy* DTs. The MEC and on-premises deployments achieve 100% timeliness for *stringent* DTs, but cannot accommodate *heavy* ones. Conversely, cloud deployments can handle *heavy* DTs but fail to meet the timeliness requirements of *stringent* DTs. The continuum deployment addresses both needs by placing most heavy DTs in the cloud, most stringent DTs on-premises, and a fraction of each type in the MEC, achieving 100% timeliness for both categories.

This is confirmed by the results in Fig. 5, which show the average time to compute DT status updates at computing nodes. On the one hand, the compute time experienced by *heavy* DTs in the continuum-deployment scenario is lower than the one in both the MEC and on-premises scenarios. As a result, the compute time of *heavy* DTs is comparable to the one obtained in the cloud-only deployment scenario. On the other hand, *stringent* DTs in the continuum-deployment scenario experience a compute time which is only slightly higher than that in the cloud-only deployment, as MEC and on-premises computing resources are used. This additional time is, however, low enough to meet the timeliness requirements.

VI. Conclusions

In this paper, we presented an orchestration system for DT trustworthiness, designed to ensure reliable interactions between industrial machinery and applications. The system integrates core trustworthiness requirements to dynamically manage DT instantiation and deployment while satisfying them. We validated the approach through both a working prototype and system-level simulations, demonstrating its capability to manage the DT lifecycle and preserve trustworthiness. The results confirm the feasibility and practical applicability of our approach in industrial scenarios. As future work, we will explore predictive, ML-assisted orchestration to anticipate ODTE and timeliness degradations under time-varying network and compute conditions, enabling proactive reconfiguration and migration before trustworthiness violations occur. Moreover, we aim to extend the orchestration layer with ISA/IEC 62443 to systematically capture and enforce industrial cybersecurity constraints across zones and assets.

Acknowledgement

AI tools were used solely for language editing and proofreading of the manuscript.

References

[1] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things (iot): A vision, architectural elements, and future directions," *Future generation computer systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[2] E. A. Lee, "Cyber-physical systems: A new frontier," *Embedded Systems Letters, IEEE*, vol. 3, no. 1, pp. 20–23, 2015.

[3] M. Wollschlaeger, M. Sauter, and J. Jasperneite, "The future of industrial communication: Automation networks in the era of the internet of things and industry 4.0," *IEEE Industrial Electronics Magazine*, vol. 11, no. 1, pp. 17–27, 2017.

[4] M. Grieves, "Digital twin: Manufacturing excellence through virtual factory replication," *White paper*, no. 1, 2017.

[5] F. Tao, M. Zhang, Y. Liu, and A. Y. C. Nee, "Digital twins and cyber-physical systems toward smart manufacturing and industry 4.0: Correlation and comparison," *Engineering*, vol. 5, no. 4, pp. 653–661, 2019.

[6] P. Bellavista, N. Bicocchi, M. Fogli, C. Giannelli, M. Mamei, and M. Picone, "Requirements and design patterns for adaptive, autonomous, and context-aware digital twins in industry 4.0 digital factories," *Computers in Industry*, vol. 149, p. 103918, 2023.

[7] E. A. Lee, "Cyber-physical systems: Design challenges," *11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC)*, pp. 363–369, 2008.

[8] R. Rajkumar, I. Lee, L. Sha, and J. Stankovic, "Cyber-physical systems: The next computing revolution," *Design automation conference (DAC)*, pp. 731–736, 2010.

[9] N. Bicocchi, M. Fogli, C. Giannelli, M. Picone, and A. Virdis, "Requirements and design architecture for digital twin end-to-end trustworthiness," *IEEE Internet Computing*, pp. 1–7, 2024.

[10] R. Iannone and M. E. Nenni, "Managing oee to optimize factory performance," *Operations Management*, pp. 31–50, 2013.

[11] A. A. Laghari, X. Zhang, Z. A. Shaikh, A. Khan, V. V. Estrela, and S. Izadi, "A review on quality of experience (qoe) in cloud computing," *Journal of Reliable Intelligent Environments*, vol. 10, no. 2, pp. 107–121, 2024.

[12] H. Baqa, N. B. Truong, N. Crespi, G. M. Lee, and F. Le Gall, "Quality of information as an indicator of trust in the internet of things," in *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*. IEEE, 2018, pp. 204–211.

[13] R. D. Yates, Y. Sun, D. R. Brown, S. K. Kaul, E. Modiano, and S. Ulukus, "Age of information: An introduction and survey," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 5, pp. 1183–1210, 2021.

[14] P. Bellavista, N. Bicocchi, M. Fogli, C. Giannelli, M. Mamei, and M. Picone, "Odte: A metric for digital twin entanglement," *IEEE Open Journal of the Communications Society*, 2024.

[15] F. Tao, H. Zhang, A. Liu, and A. Y. C. Nee, "Digital twin in industry: State-of-the-art," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 4, pp. 2405–2415, 2019.

[16] B. R. Barricelli, E. Casiraghi, and D. Fogli, "A survey on digital twin: Definitions, characteristics, applications, and design implications," *IEEE Access*, vol. 7, pp. 167 653–167 671, 2019.

[17] P. Bellavista, C. Giannelli, M. Mamei, M. Mendula, and M. Picone, "Digital twin oriented architecture for secure and qos aware intelligent communications in industrial environments," *Pervasive and Mobile Computing*, vol. 85, p. 101646, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1574119222000736

[18] K. Josifovska, E. Yigitbas, and G. Engels, "Reference framework for digital twins within cyber-physical systems," in *2019 IEEE/ACM 5th International Workshop on Software Engineering for Smart Cyber-Physical Systems (SEsCPS)*, 2019, pp. 25–31.

[19] K. M. Alam and A. El Saddik, "C2ps: A digital twin architecture reference model for the cloud-based cyber-physical systems," *IEEE Access*, vol. 5, pp. 2050–2062, 2017.

[20] R. Anderl, S. Haag, K. Schützer, and E. Zancul, "Digital twin technology – an approach for industrie 4.0 vertical and horizontal lifecycle integration," *it - Information Technology*, vol. 60, no. 3, pp. 125–132, 2018. [Online]. Available: https://doi.org/10.1515/itit-2017-0038

[21] E. Tantik and R. Anderl, "Potentials of the asset administration shell of industrie 4.0 for service-oriented business models," *Procedia CIRP*, vol. 64, pp. 363–368, 2017, 9th CIRP IPSS Conference: Circular Perspectives on PSS.

[22] K. Hribernik, G. Cabri, F. Mandreoli, and G. Mentzas, "Autonomous, context-aware, adaptive digital twins—state of the art and roadmap," *Computers in Industry*, vol. 133, p. 103508, 2021.

[23] F. Siqueira and J. Davis, "Service computing for industry 4.0: State of the art, challenges, and research opportunities," *ACM Computing Surveys*, vol. 54, 2022.

[24] M. Azarmipour, H. Elfaham, C. Gries, T. Kleinert, and U. Epple, "A service-based architecture for the interaction of control and mes systems in industry 4.0 environment," in *2020 IEEE 18th International Conference on Industrial Informatics (INDIN)*, vol. 1, 2020, pp. 217–222.

[25] V. Damjanovic-Behrendt and W. Behrendt, "An open source approach to the design and implementation of digital twins for smart manufacturing," *International Journal of Computer Integrated Manufacturing*, vol. 32, pp. 366–384, 2019.

[26] C. Koulamas and A. Kalogeras, "Cyber-physical systems and digital twins in the industrial internet of things [cyber-physical systems]," *Computer*, vol. 51, no. 11, pp. 95–98, nov 2018.

[27] S. Suhail, R. Hussain, R. Jurdak, and C. S. Hong, "Trustworthy digital twins in the industrial internet of things with blockchain," *IEEE Internet Computing*, vol. 26, no. 3, pp. 58–67, 2022.

[28] S. A., S. Vairavasundaram, K. Kotecha, I. V., L. Ravi, G. Selvachandran, and A. Abraham, "Blockchain-based trust mechanism for digital twin empowered industrial internet of things," *Future Generation Computer Systems*, vol. 141, pp. 16–27, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X22003636

[29] F. Vitale, S. Guarino, F. Flammini, L. Faramondi, N. Mazzocca, and R. Setola, "Process mining for digital twin development of industrial cyber-physical systems," *IEEE Transactions on Industrial Informatics*, vol. 21, no. 1, pp. 866–875, 2025.

[30] N. Bicocchi, M. Fogli, C. Giannelli, M. Picone, and A. Virdis, "Requirements and design architecture for digital twin end-to-end trustworthiness," *IEEE Internet Computing*, vol. 28, no. 4, pp. 31–39, 2024.

[31] M. Picone, M. Mamei, and F. Zambonelli, "Wldt: A general purpose library to build iot digital twins," *SoftwareX*, vol. 13, 2021.

[32] B. Galloway and G. P. Hancke, "Introduction to industrial control networks," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 2, pp. 860–880, 2013.

[33] X. Li, D. Li, J. Wan, A. V. Vasilakos, C.-F. Lai, and S. Wang, "A review of industrial wireless networks in the context of industry 4.0," *Wirel. Netw.*, vol. 23, no. 1, p. 23–41, Jan. 2017. [Online]. Available: https://doi.org/10.1007/s11276-015-1133-7

[34] G. Nardini, D. Sabella, G. Stea, P. Thakkar, and A. Virdis, "Simu5G–An OMNeT++ Library for End-to-End Performance Evaluation of 5G Networks," *IEEE Access*, vol. 8, pp. 181 176–181 191, 2020.

[35] M. Seliem, U. Roedig, C. Sreenan, and D. Pesch, "Qos-aware proportional fairness scheduling for multi-flow 5g ues: A smart factory perspective," 2025. [Online]. Available: https://arxiv.org/abs/2508.21783

[36] R. Debnath, M. S. Akinci, D. Ajith, and S. Steinhorst, "5gtq: Qos-aware 5g-tsn simulation framework," in *2023 IEEE 98th Vehicular Technology Conference (VTC2023-Fall)*, 2023, pp. 1–7.
[37] A. Bin Muslim, R. Tönjes, and T. Bauschert, "Synchronizing tsn devices via 802.1as over 5g networks," *Electronics*, vol. 13, no. 4, p. 768, 2024. [Online]. Available: https://www.mdpi.com/2079-9292/13/4/768
[38] 5G Alliance for Connected Industries and Automation (5G-ACIA), "A 5G Traffic Model for Industrial Use Cases," White Paper, Nov. 2019, accessed: Mar. 3, 2026. [Online]. Available: https://5g-acia.org/whitepapers/a-5g-traffic-model-for-industrial-use-cases/

Nicola Bicocchi is an Associate Professor at the University of Modena and Reggio Emilia whose research focuses on distributed systems, Internet of Things (IoT), cloud-edge computing, and Digital Twins. His work explores scalable software architectures, microservices, AI-enabled edge systems, and middleware for cyber-physical and industrial applications. He has published research in international journals and conferences on topics including Industrial IoT, pervasive computing, and adaptive Digital Twin systems.

Mattia Fogli (Member, IEEE) received his Ph.D. degree in computer engineering from the University of Ferrara, Italy, in 2024. He served as a Research Intern at the Florida Institute for Human & Machine Cognition, United States, from 2019 to 2020. He is currently an Assistant Professor at the University of Ferrara. His research interests include distributed systems, software engineering, and digital twins.

Carlo Giannelli (Senior Member, IEEE) received the Ph.D. degree in computer engineering from the University of Bologna, Italy, in 2008. He is currently an Associate Professor of Computer Science with the University of Ferrara, Italy. His primary research activities focus on industrial Internet of Things, digital twin management, cybersecurity in industry 4.0, software defined networking, and blockchain technologies. He serves on the editorial boards of ACM Computing Surveys, Computer Communications (Elsevier), and Journal on Wireless Communications and Networking (Springer).

Enzo Mingozzi (Member, IEEE) is a Full Professor with the University of Pisa, Pisa, Italy. His research focuses on networking systems and pervasive computing, and includes the design and performance evaluation of architectures, protocols, and algorithms for mobile edge/fog computing, IoT, and the quantum Internet. He has co-authored over 160 peer-reviewed papers, five book chapters, and holds ten patents.

Marco Picone is an Associate Professor at the University of Modena and Reggio Emilia working in the Distributed and Pervasive Intelligence (DIPI) Group at the Department of Sciences and Methods for Engineering (DISMI). He received the Ph.D. in Information Technology and the M.Sc. (cum Laude) in Computer Engineering from the University of Parma and he has also been a research visitor in the NetOS group at the Computer Laboratory, University of Cambridge (United Kingdom). He published several research papers and participated in many editorial activities for international journals, conferences and events. His research interests are currently mainly focused on Distributed Systems, Internet of Things, Edge/Fog Computing and Digital Twins.

Antonio Virdis (Member, IEEE) is Associate Professor at the University of Pisa, where he obtained his MSc degree in Computer System Engineering in 2011, and his PhD in Information Engineering in 2015. His research interests include Quality of Service, Edge Computing, network simulation and performance evaluation. He co-authored more than 80 peer-reviewed papers and 8 patents in the above fields. He led and has been involved in research projects supported by private industries and funded by the EU community and by Italian MUR. Among recent projects, he served as principal investigator for DATRUST project and as the University of Pisa lead for TWINKLE project, funded under the PRIN PNRR 2022 and PRIN PNRR programmes, respectively; both projects focus on the design and orchestration of digital twins. He is one of the authors of the SimuLTE and Simu5G opensource projects, for the system-level simulation of 4G and 5G communication networks.