The International Journal of Advanced Manufacturing Technology (2021) 116:835–845
https://doi.org/10.1007/s00170-021-07490-9

ORIGINAL ARTICLE

Real-time event-based platform for the development of digital twin applications

Carlos Eduardo Belman López[1]

Received: 12 March 2021 / Accepted: 9 June 2021 / Published online: 24 June 2021

Abstract
Digital twin has become essential to modern industrial developments and production paradigms. Digital twin provides support to users and processes in decision-making creating high-fidelity virtual models from physical objects in order to simulate their behaviors, predict their states, provide feedbacks, and if possible be optimized by themselves. The literature indicates an urgent need to develop digital twin applications. These applications require a digital platform that complies with DT requirements and allows all physical objects, virtual models, and industrial systems to communicate and integrate with each other. The contribution of this paper is to provide an analysis about digital twin (meaning and modeling), and to present a platform that works as: (1) a modern distributed system that runs as a cluster and can elastically scale to handle and integrate all the business applications, systems, and production data even the most massive data volumes; (2) a storage system that keeps data as long as necessary and provides real guarantees in delivery, persistence, performance (real time), reliability, and processing; (3) a real-time event-based platform that supports the requirements of digital twin applications including the management and support of different digital twin versions.

Keywords Event-based architecture · Digital twin · Real-time information · Smart production systems

1 Introduction

Digital twin (DT) is fundamental in the development of modern production systems. DT consists of the virtual representation of a machine, work cell, production line, or factory, where data analysis and simulations can be executed to extract knowledge about their status, predict maintenance needs, anticipate operational problems, among other utilities [39]. DT provides a unique way to reflect in the virtual world the properties of physical entities such as shape, position, state, movement, among others, using data acquisition techniques (such as smart sensors and ubiquitous technologies) and data transfer protocols. DT can at the same time be used for monitoring, diagnostics, forecasting, and optimization through artificial intelligence, data analysis, and machine learning algorithms [36].

✉ Carlos Eduardo Belman López
carlosbelman@gmail.com

1 Department in Engineering Sciences, TecNM, Celaya, Mexico

But the lack of a solid knowledge about DT (term, modeling, and development tools) has stopped the development of DT applications. Lu et al. [31] presented a reference DT model, some application scenarios, and current challenges. Jones et al. [21] presented a systematic literature review to find the key DT concepts in order to conceptualize a framework for DT development. Tao et al. [44] have highlighted the differences and correlation between DT and cyber-physical systems (CPS), although they are similar concepts, are not identical from many perspectives including origin, development, engineering practices, cyber-physical mapping, and core elements, being the DT a key part in the design and development of cyber-physical systems. Chen [11] conducted an expanded literature review on applications of cyber-physical systems highlighting areas such as smart manufacturing, transportation systems, energy management, process control, smart cities, and smart homes. Chen [12] performed a review about theoretical foundations for cyber-physical systems standing out research categories such as integration, modeling, system design, and architecture. Tao et al. [45] proposed a 5-dimensional DT model. These dimensions are physical entities, virtual models, services, data, and connections. From the perspective of this 5-dimensional DT model, Qi et al. [36]

investigated and summarized the key technologies and tools for DT implementations providing references in technologies, tools, and frameworks for future applications. Lee et al. [27] and Lee et al. [28] proposed a systematic architecture named 5C for developing and deploying cyber-physical systems into manufacturing applications. The 5C architecture includes smart connection level, data-to-information conversion level, cyber level, cognition level, and configuration level. Xu and Duan [52] showed the relationship between big data, DT, and cyber-physical systems in the context of Industry 4.0, emphasizing that in general, there are two main functional components to handle this relationship: system infrastructure and data analytics. System infrastructure oversees connectivity to ensure real-time communication between facilities and devices in order to capture, store, and process data, while data analytics focus on improving product personalization and resource efficiency in Industry 4.0 using descriptive, predictive, and prescriptive analysis. Shelden [41] pointed out the potential of cyber-physical environments where DT is an analytical model that synchronizes what is happening in the physical world, performing analysis, making predictions, and responding to events. Stark et al. [42] analyzed the development and operation of DTs for technical systems and services proposing the DT 8-dimension model.

Additionally, the literature indicates an urgent need to develop DT applications existing in few cases where DT is correctly modeled and applied. Even a lot of studies on CPS do not consider the development of DT even though it is a core element of a successful CPS design. Liu et al. [29] presented the development of a cyber-physical system mainly supported by data exchange using OPC UA and MTConnect, and Tamas and Murar [43] showed a vertical integration scenario between a user story and a cobot as a CPS. Both cases did not specify any DT model, properties, and characteristics. Theorin et al. [47] developed an architecture for event-based manufacturing known as LISA. It allows the integration of devices and services, but without considering the DT development or modeling for any device, production line, or manufacturing system. Angulo et al. [3] described a SOA-based infrastructure with the ability to integrate production information from various data sources to support performance monitoring into manufacturing companies. This proposal makes extensive use of RESTful web services and point-to-point and synchronized communication, which today is insufficient for the large number of events on the manufacturing workshops.

Also, current approaches for DT implementations lack conceptual basis, which complicates the applicability of the DT vision in design and production tasks [40]. Furthermore, the development of DT applications is a complex and iterative process characterized by high modularity, integration [15, 43], real-time data exchange, service orientation, robustness, security [49], data-driven decision making [52], and distributed architecture [6].

All these features and innovations require data as the angular building block. Data is essential to direct and monitor industrial processes and operations. Therefore, data must be collected, filtered, stored, and finally converted into information to make decisions within enterprise applications and systems [3]. At the same time, smart sensors and IoT technology have exponentially increased data availability and visibility in order to improve the control in production and manufacturing systems, where the degree of intelligence in the companies will depend on data-based innovations that allow to have the information about resources and processes when, where, and in the way it is needed [54]. Nowadays, intelligent production goes beyond automating workshops and depends on platforms that allow companies to achieve high autonomy and optimization levels. Furthermore, efficiency in production systems depends on the interconnection between many devices with different communication, processing, storage, and power supply characteristic [55]. Although today there are messaging systems, ETL pipelines, IoT frameworks among other tools, a platform, which allows to publish, subscribe, store, and process in real time the huge amount of data and events generated on production floors [33], is necessary. The literature and the practice also report limitations for DT implementations such as insufficient synchronization between the physical and digital world to establish closed loops, the missing of high-fidelity DT models for simulation and virtual testing, difficulties in the prediction of complex systems, and challenges for gathering and processing large data sets based on a solid framework or platform and a comprehensive reference model for DT [40]. Therefore, it is fundamental to develop a digital platform with a software infrastructure that complies with DT requirements and allows all physical objects, virtual models, and business applications to communicate and integrate with each other. For these reasons, the contribution of this research is to provide an analysis about DT (meaning and modeling), and to present a platform that works as follows: (1) a modern distributed system that runs as a cluster and can elastically scale to handle and integrate all the business applications, systems, and production data even the most massive data volumes; (2) a storage system that keeps data as long as necessary and provides real guarantees in delivery, persistence, performance (real time), reliability, and processing; (3) a real-time event-based platform that supports the requirements of DT applications including the management and support of different DT versions.

The remainder of this paper is organized as follows: Section 2 provides an in-depth discussion about DT (meaning and modeling). Section 3 presents the real-time event-based platform for DT applications. Section 4 shows a case study highlighting the implementation details. Section 5 concludes the research work and outlines future work.

2 Digital twin (meaning and modeling)

Currently, product design cannot be limited only to the design of a physical device. Rather, the design should encompass the device, the services enabled by the device, and the infrastructure that supports those services [39]. In addition, the lack of a deep understanding and solid knowledge about DT (term, modeling, and development tools) has stopped the vision and development of DT applications for intelligent production scenarios [31]. For this reason, this section discusses in more detail about DT (meaning and modeling).

DT is a term attributed to Michael Grieves and his work in conjunction with John Vickers, where Grieves presented the concept at a conference about product lifecycle management in 2003. For Grieves and Vickers, DT is a set of virtual information that fully describes a potential or real physical product, manufactured from the micro atomic level to the macro geometric level. At its optimum, any information that can be obtained from the physical inspection of a product can also be obtained from DT [18]. DT acts as a mirror of the real world, providing a way to simulate, predict, and optimize production systems and processes [31].

The vision for DT means an exhaustive physical and functional description of a component, product, or system, which includes useful information for all the current and following lifecycle stages. The idea of designing and developing a DT also refers to producing a copy of a part, product, or process and using it for reasoning about other instances of the same class, establishing a relation between multiple copies [40].

Shelden [41] considers a DT a virtual asset or simulation, running simultaneously in real time with its corresponding physical twin, where the physical and digital twins are reciprocally connected by sensors and actuators. Hence, the potential in DT is obtained using analytical models continuously synchronized between what is happening in the physical world, analyzing events, predicting conditions and states in the future, and acting through interventions (actuators).

DT provides decision-making support to users and processes creating high-fidelity virtual models from physical objects in order to simulate their behaviors, predict their states, and provide feedbacks. DT is also able to exchange all this production information with different human-machine interfaces (HMI), enterprise systems or with the cloud [29]. But the literature indicates an urgent need to develop DT applications for devices and production resources that allow the realization of cyber physical production systems and smart factories, but the lack of a solid knowledge about DT (not only the term but also the modeling) has also delayed these developments.

Grieves initially described the DT in three components: a physical product, a virtual representation of this product, and the bidirectional connections that feed data from the physical to the virtual representation, and information and processes coming from the virtual towards the physical representation. Grieves described this flow as a cycle (mirroring or twinning) that sends data from the physical to the virtual world and responds information and processes from the virtual to the physical world [18]. Schleich et al. [40] proposed a reference model, which serves as a DT of physical products in design and production tasks. In this approach, important properties such as scalability, interoperability, expansibility, and fidelity, as well as different operations like composition, decomposition, conversion, and evaluation are addressed. Scalability refers to the capability to provide a vision and information at different scales of the product (from small details to large systems). Interoperability refers to the ability to convert, combine, and establish equivalence between different model representations. Expansibility refers to the capability to integrate, add, or replace models. Finally, fidelity refers to the ability to describe the closeness to the physical product.

One of the most widely used architectures for modeling the DT is provided by Tao et al. [45], which consists of modeling the DT using five dimensions. These dimensions are physical entities, virtual models, services, data, and connections. A physical entity is the core of DT and could consist of an object, product, system, process, and even a factory. They perform tasks according to the physical laws and uncertainty within the environment. Virtual models are the replica of physical entities and reproduce the physical geometry, properties, behavior, and rules. The virtual model also reflects physical phenomena such as deformations, fractures, breakdowns, and failures. Behavior model explains the state, performance, deterioration, coordination, movements, and responding mechanisms against sudden changes in the environment. The rules provide the DT with logical skills such as autonomous judgment, reasoning, evaluation, and decision-making [36]. Services are essential components in the DT, providing users with applications for prediction, simulation, verification, virtual experimentation, and optimization, including intelligent diagnostics for health management in physical assets and processes [28]. Services transform the physical processes in robust and reliable, reduce costs, and improve performance while decreasing defects [8].

Stark et al. [42] analyzed the development and operation of DTs for technical systems and services proposing the DT 8-dimension model. These dimensions are integration breadth, connectivity modes, update frequency, CPS intelligence, simulation capabilities, digital model richness, human interaction, and product lifecycle. Although the five-dimension DT model proposed by Tao et al. [45] and the 8-dimension model proposed by Stark et al. [42] described a good part of the main DT elements, there are certain characteristics that remain adrift. Jones et al. [21] characterized the DT in 13 fundamental features: physical entity, virtual entity, physical environment, virtual environment, state and parameters, realization, metrology, twinning, twinning rate, physical-to-virtual connection, virtual-to-physical connection, physical processes, and virtual processes.

A physical entity represents a physical element in the real world such as an autonomous vehicle, a device, a product, a system, or a factory. A virtual entity or a virtual model is the replica of the physical entity in the virtual world [36]. The physical environment refers to the "measurable and quantifiable" space in the real world where the physical entity is located. Parameters refer to data types (geometry, functionality, location, activities, times, performance, among others), information, and processes transmitted between the physical and virtual entity. The state represents the current DT conditions and measured parameters, while the fidelity represents the accuracy and degree of replication with respect to the physical entity, describing the number of parameters transferred between both worlds. Twinning consists of synchronizing the physical and virtual state and parameters, for example, measuring the current parameters into the physical entity and bringing those same values or state in the virtual model. Twinning rate is the frequency in which synchronization occurs, either from the physical to virtual world or vice versa, generally expressed in real time. This implies that any change in a twin (physical or virtual) must be instantly reflected in the other.

The physical-to-virtual connection measures the physical parameters (metrology), transfers the values, and realizes them in the virtual environment and model. The virtual-to-physical connection quantifies (metrology), transfers, and realizes the information from the virtual model to the physical entity. This connection implies the functionality to perform the change in the physical state. Therefore, virtual processes (or services) determine the optimal measurements for the parameters, while virtual-to-physical connection (in conjunction with metrology and realization) ensures that those optimal values are effectively applied and carried out in the physical entity.

Finally, virtual processes (or services) refer to activities performed using the virtual model within the virtual environment. Virtual processes allow to carry out prediction, simulation, experimentation, virtual optimization, and verification activities supported by cloud technologies [10], machine learning algorithms [51], deep learning [50], and Big Data techniques [37]. It should be highlighted that DT modeling used in this research is the proposed by Jones et al. [21].

3 Platform architecture

DT consists of a dynamic and bidirectional mapping between a physical object and its corresponding virtual model through a middleware architecture that abstracts these physical and virtual counterparts. Therefore, DT applications require a platform that obtains and keeps synchronized the physical and virtual parts so that enterprise and operational systems can make decisions at the right place and time [31]. In addition, the low-level data generated on the production workshops must be refined to real-time information that supports decision-making [47]. Also, intelligent platforms with the ability to react to real-time events are necessary. These platforms combine key concepts, such as multilayer architectures, service orientation, information modeling, integration of legacy systems, data capture (acquired through multiple sensors and data sources), and data persistence [2]. The development of these platforms must be able to integrate devices and services at all levels. Additionally, events in a factory happen all the time and the ability to react to them as they occur makes it much easier to build services that drive business operation and provide feedback to engineers [33]. For these reasons, the contribution of this research is to present a platform that works as follows: (1) a modern distributed system that runs as a cluster and can elastically scale to handle and integrate all the business applications, systems, and production data even the most massive data volumes; (2) a storage system that keeps data as long as necessary and provides real guarantees in delivery, persistence, performance (real time), reliability, and processing; (3) a real-time event-based platform that supports the requirements of DT applications including the management and support of different DT versions. The following sections present in detail the architectural style and the different layers that make up the proposed platform.

3.1 Architectural style (event-oriented architecture)

The traditional industrial integration approach is to connect applications based on point-to-point (PtP) connections through a client/server architecture, but when the number of connections increases, the system becomes very rigid and complex to maintain. In addition, it is common to use proprietary protocols to communicate objects with each other. This is the normal scenario when PLCs from different vendors are connected. Another challenge is the communication between the different levels of ISA95 [20]. A common solution in the industry is to use OPC to standardize the way to access network devices [34]. The main problem with only using OPC is that it tends to become a PtP solution, for example, when PLC variables must be known in many devices and the companies have to include too much logic in the devices to aggregate, transform, and access data via OPC. The industry has been slow to migrate to new approaches, primarily due to the cost of replacing these legacy systems. However, migration has been recently accelerated with the advent of event-driven architectures [47].

Event-driven architecture is a distributed and asynchronous architecture pattern used to produce highly scalable and adaptable applications that can be used for small, large, and complex applications [38]. This architecture is comprised of highly decoupled, single-purpose event processing components that receive and process events asynchronously [47]. The event-driven architecture consists of two main topologies, the mediator, and the broker. The mediator topology is commonly used when you need to orchestrate multiple steps within an event through a central mediator, while the broker topology is used when you want to chain events without the use of a central mediator. The simplest and most common implementation for the mediator topology is through open-source frameworks such as Spring Integration, Apache Camel, or Mule ESB. For more sophisticated mediation and orchestration, BPEL can also be used. BPEL is similar to XML language and describes the data and steps required to process events. For very large applications that require much more sophisticated orchestration (including steps involving human interactions), the mediator can be implemented using a business process manager (BPM) such as jBPM.

The broker topology differs from mediator topology in that there is no central mediator and messages are distributed through the event-handling components. This topology is useful when there are relatively simple event flows that do not require centralized orchestration. There are two main types of architecture components within the broker topology: the broker and the event handler component. The broker can be centralized and contains all the channels used within the event flow. The channels contained in the broker can be message queues, message topics, or a combination of both [38]. The broker is the topology used in this investigation. Finally, event-driven architectures enable service choreography, where each service reacts to published events. The entity that creates an event only needs to know that the event occurred and does not need to know who is interested in the event or how it will be processed. With event-driven architectures, applications go from a synchronized communication that often has blocking problems to an asynchronous, distributed, scalable, and non-blocking communication [47].

Publish and Subscribe (Pub/Sub) is another pattern used in the proposed platform. Pub/Sub pattern is typically part of event-driven architectures. Most event-driven applications start with a simple message queue, for example, an application that directly receives information about a process to later display certain metrics on a dashboard. This solution works well when a few processes are monitored. But over time, more processes are being analyzed, and new applications and services are generated in such a way that they need data coming from all the processes involved such as in Fig. 1. Consequently, engineers could decide to solve this problem, adding a Pub/Sub platform that receives the metrics of all the processes involved. This solution could work initially and even reduce the complexity of a PtP or SOA architecture. But later, this scenario increases in complexity when the business requires analyzing not only the metrics and parameters but also the behavior of users and operators, data coming from the environment, external systems, and even logs of certain machines and legacy systems, in order to provide all this information to developers of machine learning and deep learning systems. Therefore, developing Pub/Sub platforms is much better than using PtP or SOA architectures, but there are still major drawbacks such as duplication, the need to maintain and support various systems, and the uncertainty that new applications, services, and use cases could arrive anytime as soon as the business grows [33].

In addition to the broker topology and Pub/Sub pattern, the platform uses a multilayer architecture (see Fig. 2) composed of the following parts: communication and integration layer,

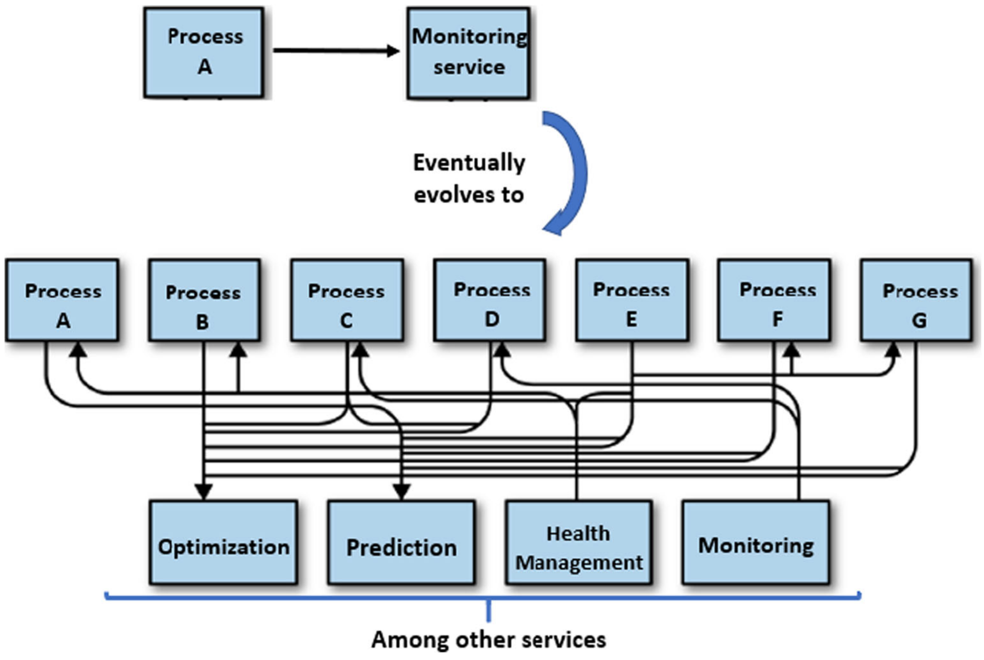


Fig. 1 Problems of PtP architecture

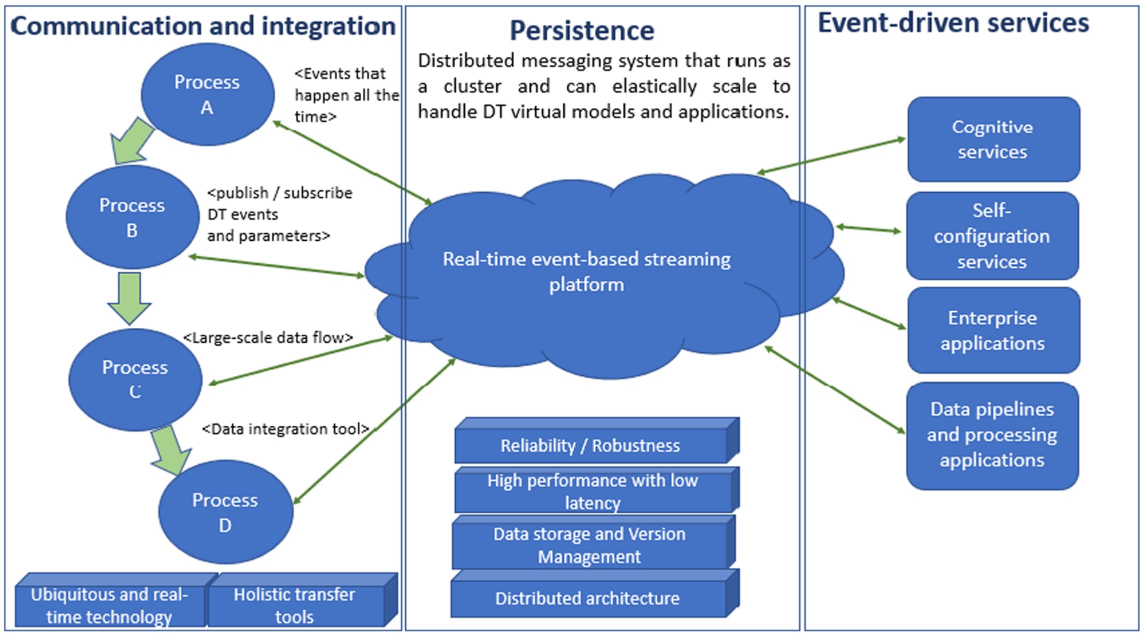

Communication and integration
Process A
<Events that happen all the time>
Process B
<publish / subscribe DT events and parameters>
Process C
<Large-scale data flow>
<Data integration tool>
Process D
Ubiquitous and real-time technology
Holistic transfer tools
Persistence
Distributed messaging system that runs as a cluster and can elastically scale to handle DT virtual models and applications.
Real-time event-based streaming platform
Reliability / Robustness
High performance with low latency
Data storage and Version Management
Distributed architecture
Event-driven services
Cognitive services
Self-configuration services
Enterprise applications
Data pipelines and processing applications

Fig. 2 Platform architecture

persistence layer, and event-driven services layer. The following subsections explain each layer in detail.

3.2 Communication and integration layer (twinning tools)

The integration and communication layer provides a secure and reliable channel of communication between operation and production processes. This layer captures and communicates the DT state (conditions and measured parameters) through ubiquitous computing and twinning tools, respectively.

In smart production, it is essential to make the right decisions, provide flexible responses to interruptions, and optimize production in short periods of time [22]. Today, data is essential to speed up decision making, where the increase in volume, variety, and velocity in data has been triggered by the widespread use of ubiquitous sensors and RFID tags along the production lines [53]. The communication and integration layer enables connectivity and guarantees real-time communication between facilities and devices, including components related to sense, capture, transfer, and access data in a distributed environment [52].

Ubiquitous computing enables physical entities to sense, capture, and access data without time and place restrictions, providing the most effective way for perceiving information from the environment [48].

Figure 2 displays the components in the integration layer composed mainly by two elements: ubiquitous technology and holistic transfer tools. These ubiquitous elements are built by ubiquitous and intelligent sensors. An ordinary sensor can be turned into a ubiquitous sensor by adding a network module to it. Ubiquitous technology also includes images, RFID tags, Auto ID, Kinect, GPS, GSM, among others. For example, RFID tags and Auto ID technology can be used to automatically track objects in real time, increasing visibility, business opportunities, and helping to automate operations [13].

The holistic transfer tools (twinning tools) are responsible for data transmission, allowing the interconnection of all the different objects in the factory using IoT protocols. Widely used data transfer protocols include Bluetooth, ZigBee, Zwave, Wi-Fi, and NFC. Bluetooth is commonly used for vehicle networks and detection applications. ZigBee is the most popular WSN network protocol and has low power consumption, very suitable for ubiquitous detection [28, 30]. Z-wave has very low power consumption suitable for smart home and healthcare applications. Wi-Fi is a wireless network protocol based on the IEEE 802.11 standards, while NFC is commonly seen in mobile phone payment systems. Meanwhile, long-range network protocols include SigFox, Neul, and LoRaWAN, in addition to GSM and mobile communication technologies. These long-range protocols are commonly used within smart cities and environmental applications to transmit data in ranges from 2 to 200 km [54].

3.3 Persistence layer

When a failure occurs, the platform must be able to recover properly and behave as if it had not failed [47]. Therefore, the platform must offer guarantees in delivery, persistence, performance (real time), reliability, and processing. The persistence layer should also offer low latency in data storage and query execution, with the ability to handle high transaction volumes [37], structured and unstructured data, possess real-time performance, scalability, high availability, data integrity and consistency [17], where cloud computing and platforms as a service provide effective solutions to such challenges [32]. This translates into a storage with real guarantees in performance and delivery, where the data is replicated, reliable, and persistent and can be kept as long as necessary. In addition, the data can be stored in cloud servers that provide support to decision-making systems and develop a decentralized and distributed architecture [53].

But in the industries, events happen all the time; this implies publish and subscribe messages that contain the current DT parameters and status (according to the DT model). These messages are produced and consumed on the platform and must have a structure or scheme in such a way that they can be easily understood. A widely used standard for message structure is JSON and XML. However, these standards lack certain characteristics such as robust handling of data types and, even more importantly, compatibility between versions of the same schema or model. A DT model can evolve over time as a result of changes in the physical part, changes in modeling interests during the life cycle, among others. In these cases, different DT model versions must be captured, stored, and integrated in a compatible way according to their evolution over time [31]. For these reasons, a platform that supports DT applications requires version management principles; it means an efficient version management of DT model. Although messaging systems add a layer that decouples the development of services, a tight coupling between different services can still be introduced through these systems. Messages that are serialized in objects using JSON as a transport protocol may cause problems with sudden changes in the message model structure, and even more so when the services involved do not correctly handle different versions of this structure. JSON does not support versioning natively. However, Apache Avro [46] enables version control to address the need for message version management and thus efficiently achieve version management of DT model [9].

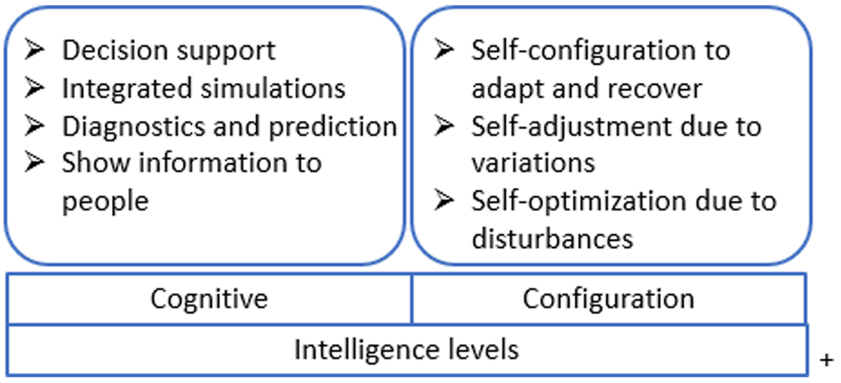
3.4 Event-driven services layer

The services layer provides data-based and event-driven services that allow prediction, simulation, verification, virtual experimentation, optimization, and monitoring of physical entities and processes, including smart maintenance services for health management [27]. The services transform the physical entities and processes into robust and reliable, reduce costs, and improve performance while decreasing defects [8]. The services are classified into two types (see Fig. 3), cognitive services and self-configuration services [27, 28].

Cognitive services generate deep knowledge about the monitored system, provide support in decision-making, generate diagnoses and predictions using machine learning and deep learning algorithms, and show this knowledge to users. Since the status and information about machines and process is available, people can make better decisions, for example, to decide on the priority of tasks or to optimize processes [5].

Cognitive services are implemented through data analysis and obtain information from data stored in the persistence layer. These can be classified into three types: descriptive analysis based on statistical functions, predictive analysis, and prescriptive analysis [52]. Predictive analytics have become the best source to extract knowledge in the production and manufacturing area [26]. Predictive methods can be categorized into 5 types: regression, decision trees, artificial neural networks, support vector machines, and Bayesian analysis. Finally, prescriptive analyses are important because they seek the optimal plan with the lowest total cost; there are two main types: mathematical programming and heuristic searches. While mathematical programming is designed to find the global optimal solution, the heuristic search is designed to find local optimal solutions in short times [52].

The self-configuration services represent the feedback from the DT virtual model to the physical entity and act as a control and supervision so that the machines are set up by themselves, applying to the monitored system the corrective and preventive decisions taken at the cognitive level [27]. Self-configuration services allow to production resources


➢ Decision support
➢ Integrated simulations
➢ Diagnostics and prediction
➢ Show information to people
➢ Self-configuration to adapt and recover
➢ Self-adjustment due to variations
➢ Self-optimization due to disturbances
Cognitive
Configuration
Intelligence levels
+
Fig. 3 Types of services

(machinery, robots, storage systems, transportation, and production facilities) to gain autonomy and the ability to self-control and self-configure in response to different situations, according to the knowledge acquired [20]. Through cognitive and configuration services, intelligence is decentralized throughout processes, providing greater stability and flexibility in operations [25].

But with Big Data, new data mining and analysis techniques are necessary, since traditional methods can fail trying to extract useful information from complex structures [7]. As data grows larger and larger, deep learning will play a key role in providing analytical solutions applied to production and manufacturing [14].

4 Case study and implementation details

The selected case study is a real industrial flotation process (mining plant). Hardware sensors like temperature, pH, flow, density, and all continuous process variables were collected every 20 s with no transformations. Quality variables like silica content (%) and iron ore content (%) are quality measurements obtained by laboratory analysis. A sample of the iron ore pulp is collected in the field/shop floor every 15 min. Those samples are sent to lab for analysis. So, on every 2 h, the laboratory gives a feedback of quality analysis; in other words, only every 2 h engineers have a quality measurement of the product stream (iron ore concentrate), which gives you a sense of the quality of the product (iron ore pulp concentrate). The objective in this case study is to predict silica concentrate (%) every minute and help engineers to act in preventive and optimized way, mitigating the iron (%) that could have gone to tailings. The complete data set can be downloaded from Kaggle [23]. The data is sent to the platform with the help of a Raspberry Pi and RESTful interfaces programmed in Python [35] that simulate and generate events every 20 s. DT model is built by the following parameters: Iron Feed (%), Silica Feed (%), Starch Flow, Amina Flow, Ore Pulp Flow, Ore Pulp pH, Ore Pulp Density, Iron Concentrate (%), Flotation Column Air Flow (1 to 7), Flotation Column Level (1 to 7), and Silica Concentrate (%).

High performance in data processing is the key to close the gap between data streams and information modeling using the DT. Additionally, industrial data analytics require low latency in data processing to enable critical applications to run [31]. For this reason, Kafka is the selected framework for implementing the broker topology in the proposed platform.

Kafka [4] is a lightweight, high-performance platform that allows you to asynchronously send streams of messages from an application to one or more applications and services. Written in Java, Kafka has become the standard messaging bus for many cloud-based applications because its functionality and architecture is highly scalable, distributed, fault-tolerant, secure, and reliable, and with excellent performance [9]. Kafka combines three key capabilities for implementing the proposed platform: (1) Kafka publishes and subscribes streams of events, including tools for importing and exporting data from other systems. (2) Streams of events are permanently and reliably stored as long as necessary. (3) Streams of messages are processed as they occur, also at a large scale. Figure 4 shows the implementation details and the mapping between the platform architecture and the DT elements

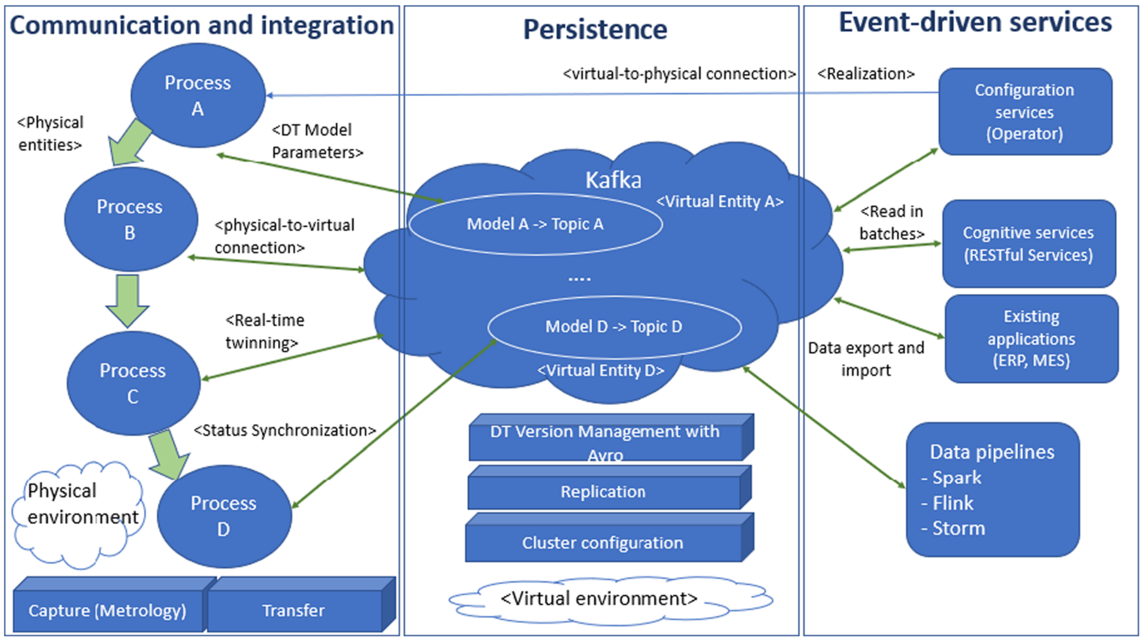


Fig. 4 Implementation details and mapping between the platform architecture and the DT elements

proposed by Jones et al. [21]. Twinning is carried out as follows: event producers (physical entities) can write to the platform in real time according to how the events happen (using the capture and transfer components). Event consumers (applications and services) read and process events (stored in the persistence layer) in batches every minute, thus reducing network overhead and producing responses very close to real time.

Messages represent the DT state and contain the real-time value for each parameter according to the DT model. Messages exchanged on the platform are published in topics representing the virtual entity, to later be consumed by applications and services subscribed to the respective topics. These applications and services can be machine learning systems, dashboards and reports for the users, or data pipelines that feed other business systems.

Message version management (to handle different DT model versions) is achieved through Avro. One of the most interesting features of Avro, and what makes it suitable for use on this platform, is that when the DT model changes or evolves to obtain greater fidelity, the platform allows these new messages to be written without any problem. In this way, the applications and services that are consuming the data can continue to process messages without requiring any changes or updates, solving one of the greatest challenges when implementing DT applications, such as the management of different DT versions as indicated by Lu et al. [31].

Additionally, the platform (implemented using Kafka) has the ability to elastically scale across multiple servers and data is replicated (with a replication factor set to 3) to protect against data loss due to any server failure. The creation of replicas also allows maintenance work on the servers, maintaining high availability for clients (applications and services) using a cluster configuration with 3 brokers. All these configurations make up the virtual environment. In addition, the reliability and continuous delivery capabilities through Kafka allow its integration with processing systems, such as Apache Storm, Apache Spark, Apache Flink, Apache Samza, among other processing systems that use Kafka as a reliable data source.

The cognitive services were developed using Artificial neural networks (ANNs) and Python [35] in conjunction with TensorFlow [1] and Keras [16]. The training was carried out using a computer equipped with an Intel Core i5-6200U processor (2.40 GHz), 16 GB of RAM, 222 GB of hard disk and Windows 10. ANN architecture consisted of an input layer; three hidden layers with 32, 64, and 128 neurons; and an output layer. ANNs were trained using Adam optimizer [24] to minimize the mean square error. In order to avoid overfitting, dropout was set to 0.5 in the hidden layers. Finally, ANNs were exposed as a RESTful service using Flask. Flask [19] is an ideal microframework for building RESTful services due to its lightweight nature.

The self-configuration services represent the feedback produced in the DT virtual model for the physical entity and allow the machines to be configured according to the corrective and preventive decisions made at the cognitive level. These services, together with the virtual-to-physical connection, allow to close the loop between the hypotheses and decisions generated in the virtual environment and the consequences carried out in the physical environment. An aspect that is frequently omitted in the literature is the role of humans and operators in this DT loop. If someone, for example, uses a DT to predict the health of a machinery using cognitive services implemented by predictive models and deep learning, to finally send mechanical engineers to replace the necessary components in the machinery according to cognitive services, the mechanical engineer in this scenario performs the process of virtual-to-physical connection, realization, and configuration, thus closing the DT cycle correctly. This is the approach used in this implementation, where the decisions made by the cognitive services are carried out by the corresponding operators, closing the DT cycle properly and complying with the definitions in the DT modeling proposed by Jones et al. [21].

5 Conclusions and future work

The proposed platform has shown that it is applicable for developing DT applications in various areas such as manufacturing, production, or mining, and it is easily applicable to many other areas, where processes are executed asynchronously, and the flow of products is non-linear and asynchronous. The proposed platform obeys design principles such as data orientation, scalability, robustness (high fault tolerance, it means the ability to function correctly even when failures occur), reliable (in data delivery and persistence), heterogeneity for managing diversity of data sources and end devices, real-time data, and processing at large scale, in addition to the flexibility provided by the integration of data, applications, and services. Thus, the platform transforms the physical entities of the company into a “connected factory” and provides a unique environment of trust that stimulates the creation and dissemination of knowledge based on data. Furthermore, the platform is modular and flexible (allowing the reconfiguration of production processes and systems) and meets data delivery requirements at the right time and place. Finally, the DT-based platform architecture enables the virtualization of processes and operations. This means the identification and abstraction of the logic behind the processes and operations for their translation to the virtual world in order to improve efficiency, productivity, and flexibility; predict behaviors; and reduce costs. For future work stands out the implementation of self-configuration services that allow executing and closing the DT cycle in a total autonomous way. Additionally, within a highly connected smart factory, the cybersecurity will be a key issue to be solved in the near future. Regarding the challenges in the development of DT applications, the inclusion of people is a priority. It means DT applications that connect workers with the production and manufacturing workshops. The representation of people, including personal data, health data, activities, and emotional states, can help to establish models to understand personal wellbeing and improve the working conditions on the production floors. By understanding the conditions of workers and their status, it is possible to design human-machine collaboration interfaces that improve people's physical and psychological health, increasing performance and productivity at the same time.

Funding The author would like to thank the Mexican Council of Science and Technology (CONACYT – Consejo Nacional de Ciencia y Tecnología) for financing this research by awarding a scholarship for postgraduate studies (under CVU 773443).

Data availability Not applicable

Code availability Not applicable

Declarations

Conflict of interest The author declares no competing interests.

References

1. Abadi M, Agarwal A, Barham P, Brevdo E (2015) TensorFlow: large-scale machine learning on heterogeneous systems. Retrieved from TensorFlow: https://www.tensorflow.org/
2. Alexopoulos K, Sipsas K, Xanthakis E, Makris S, Mourtzis D (2018) An industrial Internet of things based platform for context-aware information services in manufacturing. Int J Comput Integr Manuf 31:1–14. https://doi.org/10.1080/0951192X.2018.1500716
3. Angulo P, Guzmán C, Jiménez G, Romero D (2016) A service-oriented architecture and its ICT infrastructure to support eco-efficiency performance monitoring in manufacturing enterprises. Int J Comput Integr Manuf:202–214. https://doi.org/10.1080/0951192X.2016.1145810
4. Apache Software Foundation (2017) Apache Kafka. Retrieved from Apache Kafka. A distributed streaming platform.: https://kafka.apache.org/
5. Bagheri B, Yang S, Kao H-A, Lee J (2015) Cyber-physical systems architecture for self-aware machines in Industry 4.0 environment. IFAC-PapersOnLine:1622–1627. https://doi.org/10.1016/j.ifacol.2015.06.318
6. Belman-Lopez C, Jiménez-García J, Hernández-González S (2020) Análisis exhaustivo de los principios de diseño en el contexto de Industria 4.0. RIAI Rev Iberoam Autom Inform Ind 17:432–447. https://doi.org/10.4995/riai.2020.12579
7. Beysolow T II (2017) Introduction to Deep Learning Using R. Apress, San Francisco
8. Caggiano A (2018) Cloud-based manufacturing process monitoring for smart diagnosis services. Int J Comput Integr Manuf 31(7):612–623. https://doi.org/10.1080/0951192X.2018.1425552
9. Carnell J (2017) Spring Microservices in Action. Manning Publications Co., New York
10. Charro A, Schaefer D (2018) Cloud manufacturing as a new type of product-service system. Int J Comput Integr Manuf, pp 1018–1033. https://doi.org/10.1080/0951192X.2018.1493228
11. Chen H (2017) Applications of cyber-physical system: a literature review. J Ind Integr Manag 02:1–28. https://doi.org/10.1142/S2424862217500129
12. Chen H (2017) Theoretical foundations for cyber-physical systems: a literature review. J Ind Integr Manag 02:1–27. https://doi.org/10.1142/S2424862217500130
13. Chen T, Tsai H-R (2016) Ubiquitous manufacturing: current practices, challenges, and opportunities. Robot Comput Integr Manuf 45:126–132. https://doi.org/10.1016/j.rcim.2016.01.001
14. Chen X-W, Lin X (2014) Big Data deep learning: challenges and perspectives. IEEE Xplore (2)514–525. https://doi.org/10.1109/ACCESS.2014.2325029
15. Chen Y (2017) Integrated and intelligent manufacturing: perspectives and enablers. Engineering 3:588–595. https://doi.org/10.1016/J.ENG.2017.04.009
16. Chollet F (2015) Keras. Retrieved from Keras: https://keras.io
17. Gorton I, Klein J (2015) Distribution, data, deployment, software architecture convergence in Big Data systems. IEEE Comput Soc 32(3):78–85. https://doi.org/10.1109/MS.2014.51
18. Grieves M, Vickers J (2017) Digital twin: Mitigating unpredictable, undesirable emergent behavior in complex systems. Springer:1–30. https://doi.org/10.1007/978-3-319-38756-7_4
19. Grinberg M (2014) Flask web development. Developing web applications with Python. O'Reilly Media, Inc., Sebastopol
20. International Society of Automation (2021) ISA95, Enterprise-Control System Integration. Retrieved from International Society of Automation: https://www.isa.org/standards-and-publications/isa-standards/isa-standards-committees/isa95
21. Jones D, Snider C, Nassehi A, Yon J, Hicks B (2020) Characterising the digital twin: a systematic literature review. CIRP J Manuf Sci Technol 29:36–52. https://doi.org/10.1016/j.cirpj.2020.02.002
22. Kagermann H, Wahlster W, Helbig J (2013) Recommendations for implementing the strategic initiative INDUSTRIE 4.0. Final report of the Industrie 4.0 Working Group. Natl Acad Sci Eng (acatech): 1–82
23. Kaggle (2021) Quality prediction in a mining process. Retrieved from Kaggle: https://www.kaggle.com/edumagalhaes/quality-prediction-in-a-mining-process
24. Kingma D, Ba J (2015) Adam: a method for stochastic optimization. In: Proceedings of the 2015 International Conference on Learning Representations, San Diego. https://arxiv.org/abs/1412.6980
25. Klingenberg C (2017) Industry 4.0: what makes it a revolution? EurOMA. 1–11. ResearchGate.
26. Kusiak A (2017) Smart manufacturing. Int J Prod Res 56:508–517. https://doi.org/10.1080/00207543.2017.1351644
27. Lee J, Ardakani H, Yang S, Bagheri B (2015) Industrial big data analytics and cyber-physical systems for future maintenance & service innovation. Procedia CIRP 38:3–7
28. Lee J, Bagheri B, Kao H-A (2014) A cyber-physical systems architecture for Industry 4.0-based manufacturing systems. Soc Manuf Eng (SME) 3:18–23. https://doi.org/10.1016/j.mfglet.2014.12.001
29. Liu C, Vengayil H, Lu Y, Xu X (2019) A cyber-physical machine tools platform using OPC UA and MTConnect. J Manuf Syst 51:1–14. https://doi.org/10.1016/j.jmsy.2019.04.006
30. Liu Y, Peng Y, Wang B, Yao S, Liu Z (2017) Review on cyber-physical systems. IEEE/CAA Journal of Automatica Sinica 4:27–40. https://doi.org/10.1109/JAS.2017.7510349
31. Lu Y, Liu C, Wang K-K, Huang H, Xu X (2019) Digital twin-driven smart manufacturing: connotation, reference model,

applications and research issues. Robot Comput Integr Manuf 61: 101837. https://doi.org/10.1016/j.rcim.2019.101837

32. Moghaddam F, Ahmadi M, Eslami M (2015) Cloud computing challenges and opportunities: a survey. In: International Conference on Telematics and Future Generation Networks (TAFGEN). IEEE, Kuala Lumpur, pp 34–38. https://doi.org/10.1109/TAFGEN.2015.7289571
33. Narkhede N, Shapira G, Palino T (2017) Kafka: the definitive guide. Real-Time Data and Stream Processing at Scale. O'Reilly Media, Inc., Sebastopol
34. OPC Foundation (2021) OPC Foundation. Retrieved from OPC Foundation: https://opcfoundation.org/
35. Python Software Foundation (2020) Python. Retrieved from Python: https://www.python.org/
36. Qi Q, Tao F, Hu T, Anwer N, Liu A, Wei Y, Wang L, Nee A (2019) Enabling technologies and tools for digital twin. J Manuf Syst 58: 1–19. https://doi.org/10.1016/j.jmsy.2019.10.001
37. RS, RS (2017) Data mining with Big Data. In: Intelligent Systems and Control (ISCO). IEEE, Coimbatore, pp 246–250. https://doi.org/10.1109/ISCO.2017.7855990
38. Richards M (2015) Software Arquitecture Patterns. O'Reilly Media, Inc., Sebastopol
39. Rosen D (2019) Thoughts on design for intelligent manufacturing. Engineering 5:1–6. https://doi.org/10.1016/j.eng.2019.07.011
40. Schleich B, Anwer N, Mathieu L, Wartzack S (2016) Shaping the digital twin for design and production engineering. CIRP Ann Manuf Technol 66:1–4. https://doi.org/10.1016/j.cirp.2017.04.040
41. Shelden D (2018) Cyber-physical systems and the built environment. Technology|Architecture + Design 2:137–139. https://doi.org/10.1080/24751448.2018.1497358
42. Stark R, Fresemann C, Lindow K (2019) Development and operation of digital twins for technical systems and services. CIRP Ann Manuf Technol 68:129–132. https://doi.org/10.1016/j.cirp.2019.04.024
43. Tamas L, Murar M (2018) Smart CPS: vertical integration overview and user story with a cobotx. Int J Comput Integr Manuf 32: 504–521. https://doi.org/10.1080/0951192X.2018.1535196
44. Tao F, Qi Q, Wang L, Nee A (2019) Digital twins and cyber–physical systems toward smart manufacturing and Industry 4.0: correlation and comparison. Engineering:653–661. https://doi.org/10.1016/j.eng.2019.01.014
45. Tao F, Zhang M, Liu Y, Nee A (2018) Digital twin driven prognostics and health management for complex equipment. CIRP Ann Manuf Technol 67:1–4. https://doi.org/10.1016/j.cirp.2018.04.055
46. The Apache Software Foundation (2020) Apache Avro. Retrieved from Apache Avro: https://avro.apache.org/
47. Theorin A, Bengtsson K, Provost J, Lieder M, Johnsson C, Lundholm T, Lennartson B (2016) An event-driven manufacturing information system architecture for Industry 4.0. Int J Prod Res 55: 1297–1311. https://doi.org/10.1080/00207543.2016.1201604
48. Tian W, Zhao Y (2015) Optimized cloud resource management and scheduling. Morgan Kaufmann. https://doi.org/10.1016/C2013-0-13415-0
49. Tuptuk N, Hailes S (2018) Security of smart manufacturing systems. J Manuf Syst 47:93–106. https://doi.org/10.1016/j.jmsy.2018.04.007
50. Wang J, Ma Y, Zhang L, Gao R, Wu D (2018) Deep learning for smart manufacturing: Methods and applications. J Manuf Syst 48: 1–13. https://doi.org/10.1016/j.jmsy.2018.01.003
51. Wuest T, Weimer D, Irgens C, Thoben K-D (2016) Machine learning in manufacturing: advantages, challenges, and applications. Prod Manuf Res 4:23–45. https://doi.org/10.1080/21693277.2016.1192517
52. Xu LD, Duan L (2018) Big data for cyber physical systems in industry 4.0: a survey. Enterp Inf Syst 13:148–169. https://doi.org/10.1080/17517575.2018.1442934
53. Xu L, Xu E, Li L (2018) Industry 4.0: state of the art and future trends. Int J Prod Res *56*:2941–2962. https://doi.org/10.1080/00207543.2018.1444806
54. Yang H, Kumara S, Bukkapatnam S, Tsung F (2019) The Internet of Things for smart manufacturing: a review. IISE Transactions 51: 1–36. https://doi.org/10.1080/24725854.2018.1555383
55. Zhong R, Xu X, Wang L (2017) IoT-enabled smart factory visibility and traceability using laser-scanners. In: SME North American Manufacturing Research Conference. Procedia Manufacturing, pp 1–14