Cyber Twins Supporting Industry 4.0 Application Development

Dinithi Bamunuarachchi, Abhik Banerjee, Prem Prakash Jayaraman, Dimitrios Georgakopoulos
(mbamunuarachchi,abanerjee,pjayaraman,dgeorgakopoulos)@swin.edu.au
Swinburne University of Technology
Melbourne, Australia

ABSTRACT

Industry 4.0 involves enhancing industrial processes with high-fidelity and high-value information from machines, workers, and products. Industry 4.0 applications improve production efficiency, product quality, etc., by using Internet of Things (IoT) and Artificial Intelligence (AI). Existing industry 4.0 application development approaches are centered on commercial IoT platforms that provide siloed development and runtime environments (leading to vendor lockdown) and only support individual sensors and actuators instead of entire machines. Therefore, Industry 4.0 applications need to construct representations of complex machines from such basic elements, which is a costly, error-prone, inefficient hindering portability across machines and plants. This paper proposes Cyber Twins, a comprehensive solution for efficient Industry 4.0 application development, testing, and portability. The Cyber Twins solution includes a model for machine representation and services that facilitate Industry 4.0 application development. Finally, a prototype Cyber Twin implementation is presented, with its functionality described using a sample Industry 4.0 application.

CCS CONCEPTS

• **Computing methodologies** → **Distributed computing methodologies**.

KEYWORDS

Industry 4.0 Applications, Cyber Twins, Digital Twins, IoT, Industrial IoT

ACM Reference Format:
Dinithi Bamunuarachchi, Abhik Banerjee, Prem Prakash Jayaraman, Dimitrios Georgakopoulos. 2020. Cyber Twins Supporting Industry 4.0 Application Development. In *The 18th International Conference on Advances in Mobile Computing and Multimedia (MoMM '20), November 30-December 2, 2020, Chiang Mai, Thailand*. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3428690.3429177

1 INTRODUCTION

Industry 4.0 is the latest trend in advancing manufacturing and it is underpinned by advancements in the Internet of Things (IoT) and Artificial Intelligence (AI) [11]. More specifically, IoT provides Industry 4.0 with the ability to capture high-fidelity and high-value data from manufacturing plants and related production processes. Industry 4.0 applications achieve this via the use of IoT platforms, such as Siemens' MindSphere [23], Microsoft's Azure IoT [18], and Software AG's Cumulocity IoT [6], which harvest data from machines (directly), people (via wearables and cameras), and raw materials/products (via consistency and quality measuring devices) in manufacturing plants. Complementary to IoT, advancements in AI, via the development of machine learning models that utilize such industrial IoT data, have allowed Industry 4.0 applications to improve manufacturing process efficiency, enable predictive machine maintenance, and improve product consistency/quality [9, 11]. While the benefits of such Industry 4.0 applications are well understood, the development of such applications is costly, error-prone, and inefficient. These challenges include 1) handling the complexity of exposing and integrating the data from machines, people, raw materials, and products in production processes and other Industry 4.0 applications, 2) developing reusable Industry 4.0 applications that can easily accommodate machine changes or deployment in another plant.

To explain these, consider an Industry 4.0 application for machine predictive maintenance. Predictive maintenance aims to reduce plant downtime due to a machine malfunction before its scheduled maintenance. Early maintenance prediction allows maintenance to be carried out on the machine when needed, eliminating unplanned stoppages and related loss of plant productivity. From the perspective of preventive maintenance, machines consist of actuators (e.g., motors), sensors (monitoring the actuators as well as the material being consumed by the machine during production, such as bottles, boxes, glue, etc.), hardware (e.g., onboard microprocessors and communications), and software (e.g., the machine's operating system that implements the control logic that takes machine settings as inputs, performs production-related actions via machine's actuators, translates machine sensor data to produce machine status and condition as outputs). The production process tasks the machines involved in the processes by applying process-related configurations, which we refer to as machine settings. An Industry 4.0 predictive maintenance application performs the following: 1) utilizes an IoT platform to collect sensor data related to machine health monitoring and supply levels from all the machines involved in each production process, 2) employs an AI model (e.g., predictive machine learning) to analyze such IoT data and determine the likelihood of an unplanned stoppage, 3) recommend maintenance for the machine(s) involved in these, and 4) recommends machine settings that can prevent future unplanned stoppages.

The development of such Industry 4.0 applications is currently inefficient and costly due to the limitations in IoT platforms for modeling and interacting with complex machines and lack of tools for comprehensive testing. More specifically, existing IoT platforms that are currently used in developing such Industry 4.0 can 1) only

MoMM '20, November 30-December 2, 2020, Chiang Mai, Thailand

ACM ISBN 978-1-4503-8924-2/20/11…$15.00
https://doi.org/10.1145/3428690.3429177

incorporate and interact with individual machine sensors and actuators instead of entire machines, and 2) they do not consider the control loop of complex industrial machines. Therefore, Industry 4.0 applications, such as our sample preventive maintenance application, must devise functional representations of such complex machines in the cyberspace from the basic sensor and actuator elements currently supported by existing IoT platforms. This approach hinders efficient application development and testing of Industry 4.0 applications and severely limits the portability of such applications across machines and plants. Furthermore, existing IoT platforms provide only proprietary development and runtime environments that lead to vendor lockdown.

To address the above challenges, this paper proposes Cyber Twins. This novel solution supports efficient Industry 4.0 application development and testing and enhances Industry 4.0 application portability across plants and machines. The specific contributions of the paper include:

- A Cyber Twin Model that allows semantically describing the composition of a machine. Industrial machines lack standardized and semantically rich descriptions that describe the composition of their sensors, actuators, and related IoT data that can be analyzed by IoT applications. Furthermore, the unavailability of a standardized semantic description reduces the Industry 4.0 applications' interoperability across machines and plants. The current standards in IoT that support describing and interacting with basic sensors and actuators fall short of representing and interacting with complex industrial machines that have a control loop. The proposed Cyber Twin Model allows providing cyber representations of complex industrial machines and interacting with them.
- Cyber Twin Services that allow operations to be carried out on Cyber Twins (the cyber representation of complex industrial machines) via the Cyber Twin service interface.
- A prototype implementation and evaluation of a Cyber Twin, using an Industry 4.0 use case, illustrating the ability of Cyber Twins to support the development of Industry 4.0 applications.

The rest of the paper is organized as follows: Section 2 provides sample Industry 4.0 applications to exemplify the Industry 4.0 application development challenges. Section 3 reviews the related work, and section 4 introduces the Cyber Twins. Next, section 5 presents a prototype implementation of a Cyber Twin and demonstrates the Cyber Twin-based Industry 4.0 application development using an Industry 4.0 use case. Finally, section 6 concludes the paper.

2 SAMPLE INDUSTRY 4.0 APPLICATIONS

In this section, in addition to the predictive maintenance application described earlier, we present two other examples to exemplify the challenges in developing Industry 4.0 applications.

2.1 Product Quality Improvement

Maintaining and improving product quality is key to success in manufacturing. These requirements relate directly to several Key Performance Indicators (KPIs) in a manufacturing plant, and the quality of the products needed to meet these targeted KPIs. Meeting the KPIs requires monitoring product quality, assessing and identifying any deviations from the KPIs, and making required adjustments to the production process to achieve the targeted KPIs. An Industry 4.0 application developed to support improving product quality requires the following:

- Data harvesting that requires access to sensor data and relevant machine settings.
- Data integration that requires integrating machine data with other data in a database within the factory environment.
- Conduct application testing by developing simulation-based models.

2.2 Productivity Improvement

Improving plant productivity requires improving the productivity and performance of workers in the manufacturing plant. Improving worker productivity and performance requires monitoring and assessing worker performance, identifying performance issues to enable timely training/re-training of the workers when required. An Industry 4.0 application developed to assess the performance of workers in a manufacturing plant requires:

- Data harvesting by collecting sensor data for monitoring work activities of factory workers (e.g., wearables worn on the wrist).
- Data integration that requires identifying individual workers and translate sensor data into the factory workers' work activities to compute the individual worker related KPIs.

As outlined in 2.1 and 2.2, the Industry 4.0 application development includes some repetitive tasks that are currently developed as a bespoke and siloed solution. An approach such as the proposed Cyber Twins can provide the required abstraction to the machines and people (via wearables), enabling the development of extensible Industry 4.0 applications. Such an Industry 4.0 application is also adaptable to changes in the plant environment (e.g., changes in machines) or redeployed easily in another plant with similar machines.

3 RELATED WORK

In this section, we review the current state-of-the-art in the literature relevant to the development of Industry 4.0 applications and existing Digital Twin solutions in the context of Industry 4.0.

3.1 Modeling Cyber Twins of Machines

In IoT, semantically rich ontology-based descriptions have been proposed and widely adopted for successfully coping with the heterogeneity of the sensors. The list of the proposed IoT related ontologies includes but is not limited to SSN[5], SOSA[13], IoT-O[22]. Semantic Sensor Network (SSN) ontology is a widely adopted standard ontology used in many research studies. To make SSN more lightweight and with the need to have the concepts such as actuation, SOSA has been introduced as a core ontology to SSN, and the SOSA/SSN ontology is published as a W3C recommendation and as an OGC standard for specifying the semantics of sensors, observations, and actuation. However, even though SOSA/SSN allows describing hardware elements in a machine such as sensors, actuators, and platform, it lacks the ability to describe software

elements, such as its operating system and the application element (e.g., machine settings).

Although semantic-based approaches are recommended to cope with various aspects of Industry 4.0, their propagation, and adaptation to the industrial context is hindered by the lack of standardized ontologies [25]. In the industry, information exchange formats such as MTConnect [12] support describing the machines and their data to enable data exchange between machines and industrial applications. Furthermore, the OPC UA standard allows vendor neutralism in information exchange in the industrial context [16]. OPC UA uses an object-oriented approach towards modeling and representing the machines and provides a generic object model representing the structure and state information. This generic model needs to be extended by vendors for describing their machines. Further, to make the model more meaningful to OPC UA clients, existing standards, such as MTConnect, and PackML, need to be incorporated into the model [10]. Some of the existing IoT platforms, such as Cumulocity IoT, provide a predefined set of sensor and control capabilities to support describing the machines and their data [6]. However, these are specific to the IoT platforms and lack standardization and meaning outside them.

The current industrial practices related to describing machines rely upon the standard-based descriptions. They lack the description power and querying capability provided by a semantically rich ontological representation of a machine. Also, the existing standardized IoT related ontologies such as SOSA/SSN don't describe the composition of a machine. As a step towards enhancing the existing support for Industry 4.0 application development, in this paper, we extend the widely adopted SOSA/SSN ontology to describe the composition of a machine.

3.2 Machine and Machine Data Integration

Integrating a machine with an Industry 4.0 application needs to enable communication between the two entities to send and receive data. In general, IoT platforms have been utilized to support integrating the machines with the Industry 4.0 applications.

The OpenIoT [24] is an open-source IoT platform that supports SSN [5] ontology-based integration of sensors and their data with IoT applications. The platform uses X-GSN [3] IoT middleware to retrieve data from sensors and annotate them with meta-data to support sensor data integration. X-GSN allows abstracting the heterogeneity of the physical sensors to the upper layers, such as the IoT platform and the IoT applications, via the concept of virtual sensors. The IoT data integrated with OpenIoT platform can be queried using SPARQL, and further, it can be exported to external data analysis and visualization services such as SensorDB [20]. However, OpenIoT neither provides semantic descriptions that can describe the machine's composition nor does it provide interfaces that support sending back actuation data (i.e., machine settings, actuation commands).

Industry 4.0 applications currently use IoT platforms such as Azure IoT [18], Cumulocity IoT [6], and MindSphere [23] for integrating industrial machines and the data they produce with related data analytics, such as machine learning models for predictive maintenance, and for improving production efficiency and product variation. However, existing IoT platforms typically provide IoT platform-specific software SDKs for integrating simple sensors, not complex machines, as the ones that are currently deployed in industrial plants [6, 18]. Therefore, developing Industry 4.0 applications via such existing SDKs requires industry 4.0 application to represent complex industrial machines as complex, interrelated collections of sensors and actuators. This is time consuming, costly, and error-prone. Furthermore, different IoT platforms provide proprietary, platform-specific domain-models for this and for building 4.0 applications [6, 18]. Hence, if the same machine is used by Industry 4.0 applications running on different IoT platforms, different IoT platform-specific machine representations and Industry 4,0 applications must be developed. In addition to the cost and effort required, this limits the reusability of the Industry 4.0 applications and leads to vendor-lockdown and bespoke solutions.

Another limitation of developing Industry 4.0 solutions using existing IoT platforms is that such platforms do not support testing of Industry 4.0 applications before they are deployed on actual machines. Cyber Twins include machine simulators or emulators that permit significant application testing before Cyber Twins switch Industry 4.0 application execution to the actual machines involved. Therefore, Cyber Twins reduce the time and cost required for Industry 4.0 application development and testing and also enable reuse of machines across Industry 4.0 applications.

3.3 Digital Twins

Digital Twins are virtual representations of physical objects. In the context of Industry 4.0 applications, Digital Twins (DTs) of machines and production lines have been mainly proposed for optimizing the production process and supporting the predictive maintenance.

Cai et al. have proposed a schematic for constructing the DT for a machine tool to support diagnosis and prognosis [2]. The DT fuses the machine's sensor data with process-related data and maintains a machine's characteristics profile to support the above aspects. However, it does not adopt standards for describing sensor data and cannot adequately support the reusability of the Industry 4.0 applications. Barthelmey et al. has proposed a dynamic DT for supporting predictive maintenance of a flexible production system [1]. The DT receives data in the AutomationML format, and the available analytical models are applied to the data for identifying predictive maintenance requirements. However, the solution is bespoke i.e., it only supports predictive maintenance and uses fixed semantics for describing the data to support only the predictive maintenance application. Schroeder et al. [21] have proposed an object-oriented data modeling mechanism for DTs of the industrial plant components using AutomationML. The DT data is exposed to third-party applications via a middleware platform, using JSON/REST interfaces. However, the lack of semantically rich data descriptions still limits the reusability of the applications that utilize the DT. Martin and Milan have proposed DTs for a production line to support the production processes' optimization using OPC data [26]. The production line data are used to create a simulation-based model and are used to identify and alert deviations from the optimal scenarios. Kamil et al. has proposed a DT for an experimental assembly system to support product quality control [28]. The assembly line data are collected using a PLC and standardized using an OPC server, and

delivered to the DT. However, OPC-based standardization lacks the full potential of rich semantic descriptions required for supporting the reusability and the adaptability of Industry 4.0 applications. Kychkin and Nikolaev have proposed a Mine Ventilation Control System architecture with DT for efficient and safe operation of a ventilation system [15]. In the proposed architecture, an add-on has been developed and integrated with the existing SCADA system to integrate the DT functionalities to the system. However, the lack of semantics reduces the reusability of the application in other contexts.

In summary, the existing state-of-the-art in DTs target specific applications (siloed) to demonstrate the benefits of a DT. They lack reusability and interoperability (i.e., they cannot support the new Industry 4.0 application that will use the same machine), hence fall short of addressing the Industry 4.0 application development challenges we introduced in Sections 1 and 2. To address the limitations of the existing DT research outcomes, we propose Cyber Twins to support the efficient development of reusable and adaptable Industry 4.0 applications.

4 THE CYBER TWINS

A machine consists of both hardware and software elements. The hardware elements of a machine include a hardware platform, sensors, and actuators. The hardware platform has (in some cases limited) computing and networking capability and hosts the machine's operating system. It can be a microcontroller like an Arduino or industrial controller hardware in an industrial machine. The machine's software elements include the Operating System (OS). The sensors (e.g., temperature sensors, photoresistors) and the actuators (e.g., relays, servo motors) are integrated with the machine's hardware platform, and the combination of both hardware and software makes up the machine. A machine can be utilized by an Industry 4.0 application to serve the purpose of the application by configuring it using different software programs or applying the required machine settings to it.

A machine may also have an internal control loop that determines its operations. For example, an evaporator machine accepts machine settings and the machine's operating system uses these to control its processing speed and evaporation temperature. The machine's operating system includes control logic that determines how each machine setting impacts the machine's operation. To determine the appropriate machine settings (e.g., the appropriate evaporation temperature required to eliminate variation in the product the machine produces), an IoT application must be tested to ensure that the machine setting it produces results in the desired evaporation temperature and even more importantly, the product consistency. Providing the ability to simulate such machine behavior can eliminate or significantly reduce the testing of Industry 4.0 application on the actual machine involved. The absence of a suitable abstraction for machines that support machine modeling, integration and testing leads to inefficiencies in Industry 4.0 application development.

A Cyber Twin (of a machine) provides an abstraction of a complex machine for Industry 4.0 applications that include everything that is needed to model, integrate and test the machine for the purpose of Industry 4.0 application development. In particular, a Cyber Twin provides a mechanism for semantically describing a complex machine that can potentially include control logic, and multiple sensors and actuators. A Cyber Twin provides a set of services (the Cyber Twin Services) that allows Industry 4.0 applications to integrate, simulate/emulate and use the machine. This includes using the actual machine or its simulator/emulator and the ability to switch from one to the other during application testing. Unlike existing IoT platforms that require integration of individual sensors to be handled by the Industry 4.0 application and provide no machine simulation/emulation, the proposed Cyber Twins-based approach has the following benefits:

- Reduction of time, cost, and effort required for developing an Industry 4.0 application by allowing an application to efficiently integrate and access all necessary sensor data it needs from a machine.
- Support for a variety of Industry 4.0 applications and eliminate repeated programming effort required for introducing new Industry 4.0 applications.
- Improves the reusability of Industry 4.0 applications by providing semantic abstractions of machines.

Figure 1 provides an overview of Cyber Twin-based Industry 4.0 applications. Each Industry 4.0 application may correspond to a use case where it requires interfacing with the Cyber Twins of one or more machines.

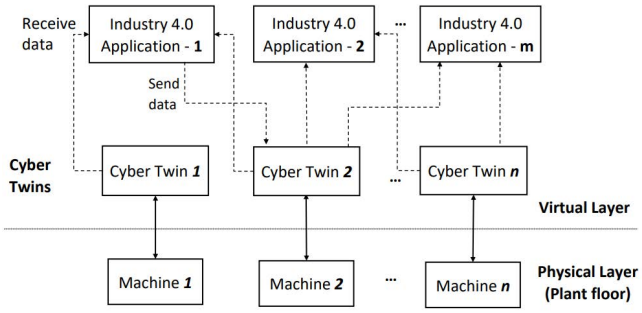


Figure 1: Cyber Twins-based Industry 4.0 Application Development

Next, we present a detailed description of the Cyber Twin Model used to describe machines and the Cyber Twin Services that provide interfaces to select, integrate, and use the machine or an emulator/simulator of the machine.

4.1 Cyber Twin Model

Modeling a machine first needs to describe the key hardware and software elements of the machine. The Cyber Twin Model uses an ontology-based approach for semantically describing a machine. The Cyber Twin ontology builds on the existing SOSA/SSN ontology. We propose specific concepts such as IoT device and its software elements, as illustrated in Figure 2 to describe a machine and to enable semantic query of its composition. The new concepts introduced in the Cyber Twins ontology are prefixed using 'ct' in Figure 2.

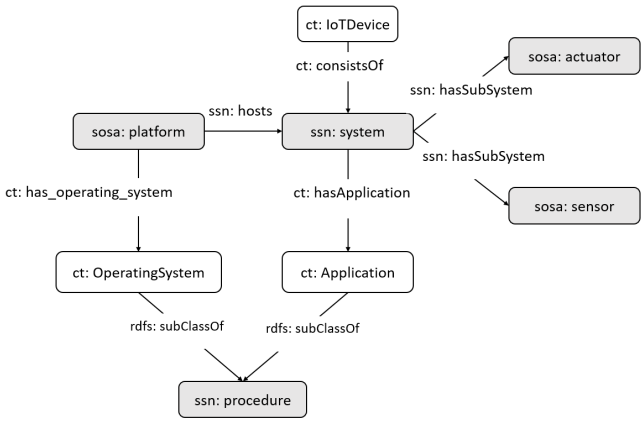


Figure 2: Cyber Twin ontology for modeling machines

- `IoT Device`: A machine that interact with the physical world and has (in some cases limited) computing and network capabilities that allow it to autonomously connect to the internet and send/receive data.
- `Application`: The application is deployed in the IoT device to utilize it for sensing and actuation. It can be the settings as well as a program. For example, for a temperature-sensing device, the application can be the settings (e.g., sensor polling frequency, data sending frequency), or a program (i.e., a program that collects periodic temperature measurements from the sensor and average the values) or both. Similarly, for a packaging machine, the application can be the machine settings that are applied to a machine for using it for producing a specific product type or in a production run.
- `Operating System`: The operating system software primarily controls the IoT device hardware. It consists of the components such as the drivers and the implementations of the communication protocols supported by the hardware. The operating system can also include the control program that implements the control logic of the machine.

4.2 Cyber Twin Services

The Cyber Twins provide a set of services to support Industry 4.0 application development, and they are categorized as:

- Device integration support services.
- Sensing and actuation support services.
- Testing support services.
- Querying support services.

The service categories are depicted in the service layer of Figure 3, and their functionalities are further described below.

- `Device integration support services`: Integrating an IoT device with an Industry 4.0 application requires the configuration of the IoT device to be used by the application. The configuration may require deploying a software program on the IoT device or changing the settings of the IoT device as needed for the application or both. Both kinds of operations are supported by the application deployment service (via the settings service and the program service) in the Cyber Twin service layer. It allows deploying the application on the IoT device or the emulator of the IoT device.
- `Sensing and actuation support services`: The Cyber Twin's actuation support service allows an Industry 4.0 application to send actuation commands to the actuator (A) of the IoT device via the actuation service (A) in the service layer. The actuation service (A) retrieves the status of the actuator (A) in the IoT device, semantically annotates such data, and sends the data to the Data query support service. The sensor data retrieving service (S) in the service layer retrieves the data from the IoT device sensor (S), semantically annotates such data, and sends the data to the data query support service. The data query support service stores all this in RDF format data and allows an Industry 4.0 application to query them semantically.
- `Testing support services`: To support Industry 4.0 application testing, a Cyber Twin allows an Industry 4.0 application to select, integrate, and use an emulator/simulator of the IoT device. In the device layer of Figure 3, a sensor simulator (Se) can be a software program that simulates the sensor (S) of the IoT device. It may also be a file that contains the historical data produced by the sensor (S), which are read and played back by the hardware emulator program to support the Industry 4.0 application testing.
- `Querying support services`: Allows an Industry 4.0 application to query the semantic description (ontological model) of the IoT device.

A Cyber Twin provides an API, as depicted in Figure 3, which allows an Industry 4.0 application to access and use the services described above.

5 A CYBER TWIN FRAMEWORK FOR INDUSTRY 4.0 APPLICATION DEVELOPMENT

Section 5.1 describes the Cyber Twins framework that implements the Cyber Twin Services described in Section 4.2. Section 5.2 provides an example of a simple Cyber Twin that has been developed using the framework introduced in Section 5.1. Section 5.3. presents an use case of Industry 4.0 application development using Cyber Twins and evaluates their performance.

5.1 Cyber Twin Framework Implementation

The Cyber Twins framework is designed and implemented as an actor-based system [27]. In the framework all the Cyber Twin Services are implemented as actors. The actor-based system exposes an HTTP-based web server, which allows an Industry 4.0 application to request the Cyber Twin Services. It supports the Modbus TCP/IP communication protocol, commonly used industrial protocols for communicating with machines. Table 1 depicts the tools and technologies used for the implementation. It currently has a Docker-based emulator that emulates a PLC server and the Modbus TCP/IP slaves connecting to the server.

We have introduced a Cyber Twin client-side library to support utilizing a Cyber Twin by an Industry 4.0 application. Figure 4 shows the sequence diagrams related to consuming different services provided by Cyber Twins (i.e., device integration support

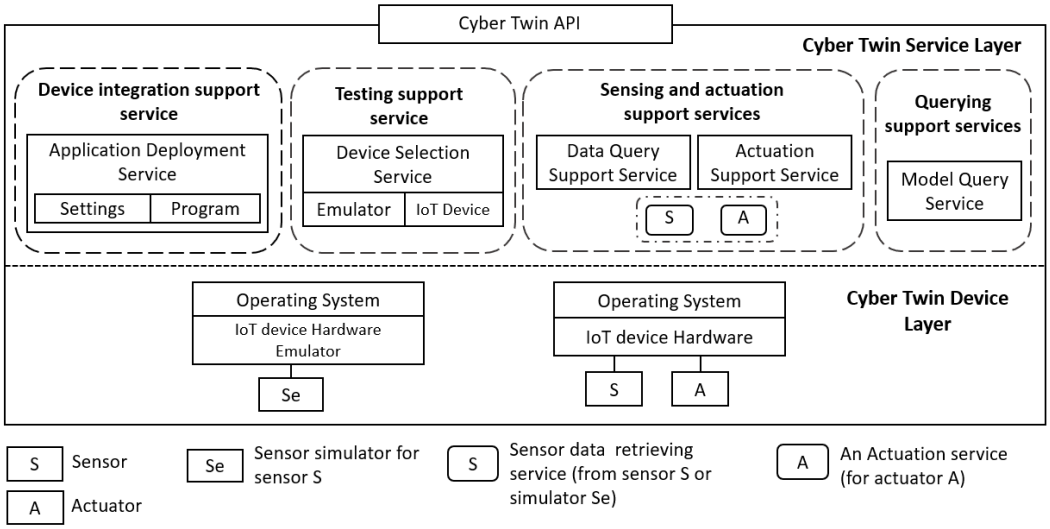


Figure 3: Cyber Twin Services to support Industry 4.0 application development

Table 1: Tools and technologies used for Cyber Twin framework implementation

Cyber Twin Components	Software Tools and Technologies
Cyber Twin actor-based system	Akka actor framework (2.6) [17], Java 1.8
Model query service	Eclipse rdf4j [7]
Data query support service	Eclipse rdf4j [7]
Modbus TCP/IP communication	EasyModbusTCP [8]
Emulator/simulator	Docker-based OpenPLC server [19]

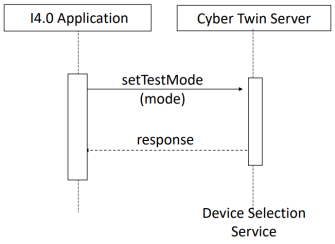
services, testing support services, sensing, and actuation support services, and querying support services). In the figures, the method calls initiated by the Industry 4.0 application are provided by the client library. Table 2 describes the client library's API for requesting the Cyber Twin Services, the service request parameters, and the service response parameters. It is also possible to send direct HTTP requests to the Cyber Twin to perform the same operations without using the implemented client library.

5.2 Use Case of an Industry 4.0 Application Using a Simple Cyber Twin

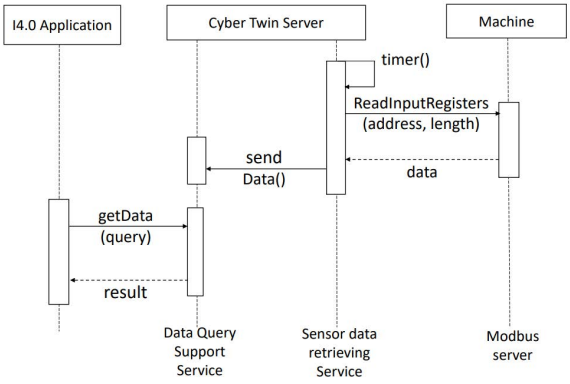
5.2.1 Use case of an Industry 4.0 Application. Consider a soft drink bottling plant for the Industry 4.0 application use case. In the plant, during the execution of its production process, the drink bottles are packaged into cardboard cartons. For gluing these cartons, the

Table 2: Cyber Twin API

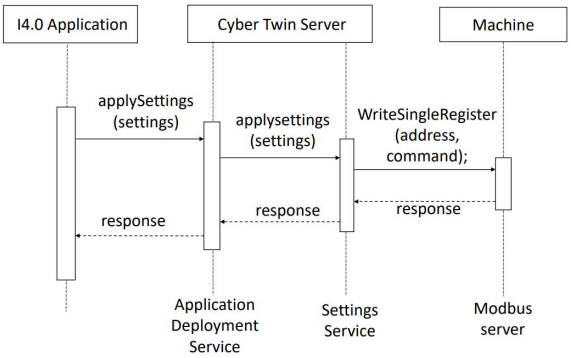
Cyber Twin Client API	Description
(1) CyberTwin (server,port)	Cyber Twin client object constructor parameter name: server, type: String, value: Cyber Twin server address. parameter name: port, type: Integer, value: Cyber Twin server port
(2) setTestMode (mode)	To select the emulator or the IoT device. parameter name: mode, type: Boolean, value: True/False. True indicates the selection of the emulator and the IoT device otherwise.
(3) getData(query)	To query the sensor and actuator data from the IoT device. parameter name: query, type: String, value: SPARQL-based query criterion. return type: QueryResult containing the data in the "result" parameter.
(4) applySettings (settings)	To send the settings to the Cyber Twin. parameter name: settings, type : String, value format: {'setting': 'x', 'value':'y'} 'x' indicates the setting identifier. 'y' indicate the value to be applied.
(5) queryModel (query)	To query the IoT device description. parameter name: query, type: String value: SPARQL-based query criterion. return type : QueryResult containing list of URIs in "result" parameter.



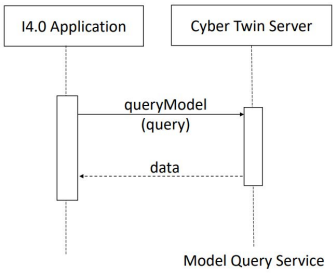
(a) Testing support services



(b) Sensing and actuation support services



(c) Device integration support services



(d) Querying support services

Figure 4: Industry 4.0 application requesting Cyber Twin Services via Cyber Twins client API.

plant employs a (hot-melt) adhesive melting machine. The machine melts the adhesive granulates (i.e., by increasing the temperature to a given set point) and makes the glue suitable for gluing the cartons. Depending on the packaging materials and the type of adhesive granulates that will be used in the process, the required glue temperature set point is configured during the setup of the production run. The machine has a temperature sensor for observing the glue temperature, and during the production run, it turns on a light to indicate when the desired glue temperature for the production is reached. Based on the indication, the factory workers start loading the packaging material. In this same plant, it is now required to introduce an Industry 4.0 application for predictive maintenance to support predicting unplanned machine downtimes of a packaging machine. Unplanned machine downtimes can be caused due to packaging issues, which in turn can be detected by monitoring the variations in the glue temperature. The Industry 4.0 application needs to monitor the glue temperature from the adhesive melting machine and analyze them to identify drops in the temperature which can lead to issues in the packaging and generate alerts to the factory operators.

5.2.2 Simple Cyber Twin of the Adhesive Melting Machine. The adhesive melting machine's operations are implemented using OpenPLC [19], installed on a PC that acts as a Modbus master, to which sensors and actuators are connected as slaves using the Modbus TCP/IP protocol. The temperature sensor is a thermistor (NTC B3960) connected to a (Wi-Fi-enabled) ESP8266. The actuator is a light-emitting diode connected to the ESP8266. It is turned on/off by the PLC if the temperature has reached a given setpoint (as explained in the use case). The Industry 4.0 application implementation setup is shown in Figure 5. Also, this machine is equipped with a web interface provided by the OpenPLC server [19] and for monitoring PLC variables, their names, register locations, data types, and values.

Figure 5: Industry 4.0 application implementation

Cyber Twin Model: Figure 6 depicts the Cyber Twin ontology-based representation of the machine's hardware and software elements. The ontology-extension captures the machine as an IoT device, describing the operating system (i.e., OpenPLC runtime) and its sensor (i.e., thermistor), actuator (i.e., light), and the platform

(i.e., Intel(R) i5-10210U). In the diagram, we have only depicted the machine's key elements to demonstrate the validity of the proposed extension. This description (or an extension of it) can be used to query and identify the machine's composition and integrate its sensor data with Industry 4.0 applications.

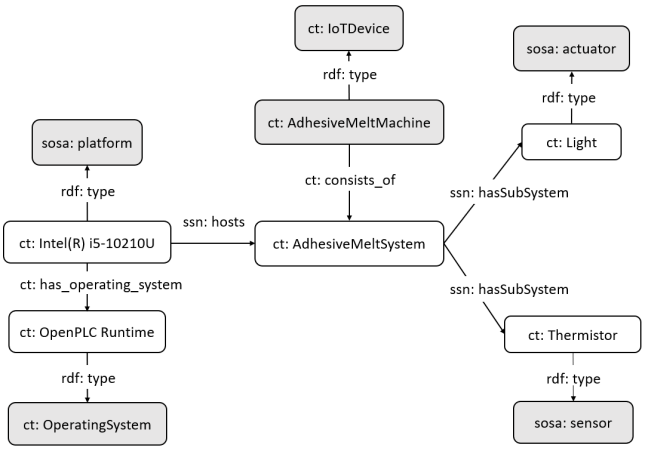


Figure 6: Cyber Twin ontology-based model

5.3 Cyber Twins-based Industry 4.0 Application Development and Evaluation

In the use case presented in Section 5.2.1, the application that implements the plant's production process (production process application) and the predictive maintenance application need to use the adhesive melting machine. In the respective application development processes, for integrating the machine and its data with the application and for sending settings to the machine, we have used the machine's Cyber Twin.

5.3.1 Demonstration of the Cyber Twin-based Industry 4.0 Application Development. To better demonstrate the production process interactions (i.e., setting the glue temperature setpoint) with the adhesive melting machine, we have simplified the process model of the production process and have included the "set the glue temperature set point" as a separate activity as illustrated in Figure 7. The process depicted in the figure is modeled using the Camunda modeling tool [4]. The process application and its activities are implemented using the relevant Camunda libraries, and the integration of the machine and its data are accomplished using the Cyber Twin Services as described next.

The Industry 4.0 application connects to the Cyber Twin Server (as shown in the line 7 of Figure 8) and sends the required setting to the machine (as shown in the lines 10 to 12). The "Application deployment service" provided by the Cyber Twin Services (i.e., Figure 4 (c)), is used here. In the implementation, the process application sets the machine setting, "TemperatureSetPoint", to the value "40".

For the demonstration, the process application was deployed and executed using the Camunda process execution engine [4]. The machine-setting delivery was verified by monitoring the setting

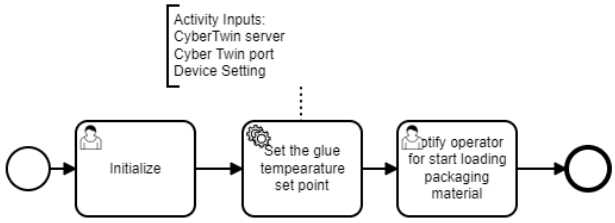


Figure 7: Activities in a simplified process model

```
6  //Create the Cyber Twin client object
7  CyberTwin twin = new CyberTwin("http://115.x.x.10", 9091);
8
9  //Send application settings
10 twin.applySettings(
11              "{\"setting\":\"TemperatureSetPoint\","
12            + " \"value\":\"40\"}");
```

Figure 8: Apply machine settings via the Cyber Twin

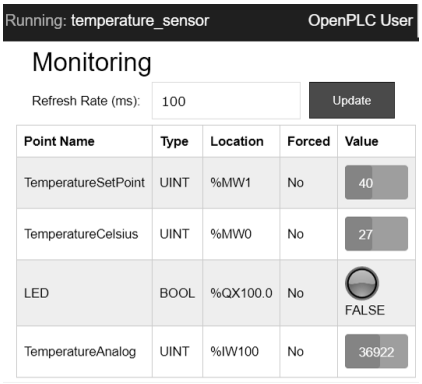


Figure 9: Temperature setpoint update

value using the PLC server's web interface. The web interface, as depicted in Figure 9, shows the table that lists the variables used in the PLC's control logic implementation. The table shows the details of the "TemepratureSetPoint" variable (i.e., which is set by the above process application) in the first row. It is a memory variable (%MW1) that can be read and write externally. As depicted, its value is updated to 40 and indicates the successful delivery of the setting update to the machine.

Next, the predictive maintenance application needs to integrate the machine and harvest the temperature sensor data. For implementing this, we have used the "Sensing and actuation support services" provided by the Cyber Twin Services (i.e., Figure 4 (b)), that allow data harvesting using SPARQL queries. The predictive maintenance application connects to the Cyber Twin server (as shown in line number 20 of Figure 10), send the required query to it, and retrieve the data matching the given query (line number 23). The SPARQL query exemplified in the figure, query the data that have unit Celsius. The sensor data retrieved is depicted in Figure 11.

```
19 //Create the Cyber Twin client object
20 CyberTwin twin = new CyberTwin("http://115.x.x.10", 9091);
21
22 //Query the Sensor data
23 QueryResult obj1 = twin.getData( query );
24
```

```
PREFIX sosa: <http://www.w3.org/ns/sosa/>
PREFIX ct: <http://www.example.org/cybertwin#>
PREFIX qudt-1-1: <http://qudt.org/1.1/schema/qudt#>
PREFIX qudt-unit-1-1: <http://qudt.org/1.1/vocab/unit#>

SELECT ?result
WHERE {
    ?value qudt-1-1:numericValue ?result .
    ?value qudt-1-1:unit qudt-unit-1-1:DegreeCelsius . }
```

SPARQL Query

Figure 10: Requesting the sensor data from the Cyber Twin

```
[[
result="28.0"^^
<http://www.w3.org/2001/XMLSchema#double>
]]
```

Figure 11: Industry 4.0 application response from Cyber Twin

As demonstrated, the Cyber Twin-based approach only needs a one-time effort to integrate a machine with its Cyber Twin. In the demonstration, introducing the predictive maintenance application required no additional effort to expose and integrate the machine's sensor data. The Cyber Twin provides required deep integration (sensor-level integration) with the machine's sensors to support any Industry 4.0 application to utilize the machine. This simplifies the Industry 4.0 application development process and improves its efficiency.

Further, the Cyber Twins enhances the semantic interoperability of Industry 4.0 applications. As shown in the predictive maintenance example, the Cyber Twins allow an Industry 4.0 application to semantically describe its desired sensor data inputs as a SPARQL query. It allows the same application to be used with another machine (e.g., an adhesive melting machine having a different model) as long as the Cyber Twin of the second machine produces the data matching the specified query criterion.

The Cyber Twin Services can encapsulate different programming interfaces provided by different machines and provide a generic programming interface to Industry 4.0 applications. The service interface improves the applications' reusability and allows them to be efficiently reused with different Cyber Twins that encapsulate different machines. The presented use case demonstrates the feasibility of the proposed approach for supporting Industry 4.0 application development and its capability to address the specific challenges we discussed. Next, we present a performance evaluation of the Cyber Twins.

5.3.2 Experimental Evaluation of the Cyber Twin Framework. For conducting a performance evaluation of the Cyber Twins server, we used the Cyber Twin of the adhesive melting machine that was deployed on a PC (processor: Intel Core I3-8145U | 2.11GHz, RAM: 8GB). The adhesive melting machine is emulated using the Cyber Twins OpenPLC emulator running on a Docker-based container. We have also used the software performance testing tool JMeter [14] to simulate the Industry 4.0 application interactions with the Cyber Twin server and support measuring the performances.

First, we evaluated the Cyber Twin's capability to support multiple applications executing in parallel. For the experiment, an Industry 4.0 application was represented by a single thread in JMeter. During the experiment, each application sent a fixed number (i.e., 1000) of requests to the Cyber Twin server sequentially, requesting temperature sensor data. The number of applications executing in parallel was increased from 1 up to 10, and the Cyber Twin response times were measured. The results are depicted in Figure 12. As per the results, the Cyber Twins server's performance is not impacted significantly by the growth in the number of clients.

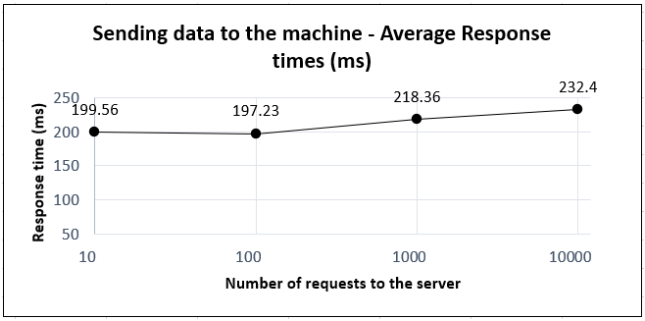


Figure 12: Cyber Twin response times for sending data to the machine

Next, we evaluated the Cyber Twins capability to support data delivery (e.g., actuation) to the machine with an increasing number of such requests. To avoid conflicts in machine control, we have assumed that only a single application (e.g., the production process application) delivers data to the machine. Hence, to measure the response time, we used a single Industry 4.0 application (i.e., a single thread in JMeter) and sent temperature setpoint value to the Cyber Twin server and thereby to the machine. During the experiment, we incremented the number of requests sent to the Cyber Twin server by the application, and the response times were recorded. The resulting graph is depicted in Figure 13, which shows that the Cyber Twin server response time is not impacted significantly by an increase in the number of requests.

6 CONCLUSION AND FUTURE WORK

In this paper we proposed Cyber Twins to support Industry 4.0 application development. Cyber Twins include a Cyber Twin ontology for semantically describing machines and the Cyber Twin Services that support Industry 4.0 applications in the tasks of modeling, integrating, simulating/emulating and using industrial machines. The feasibility of the proposed Cyber Twins was evaluated via the implementation and demonstration of two Industry 4.0 application

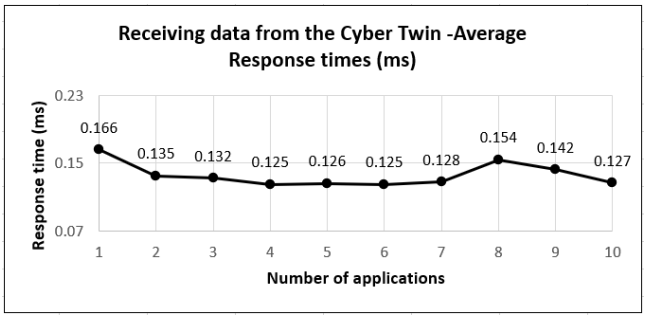


Figure 13: Cyber Twin response times when supporting multiple applications to retrieve data from the machine

use cases, namely predictive maintenance, and production process monitoring. Experimental evaluations validated the performance of the Cyber Twin and showed that it could be scaled up for use by multiple Industry 4.0 applications without a significant impact on performance.

As future work, we aim to compare and specifically measure the complexity and cost of involved in Cyber Twin-based Industry 4.0 application development with traditional IoT platform-based Industry 4.0 application development. In addition, we will utilize Cyber Twins in an ongoing industrial project. Finally, future work also includes extending the Cyber Twin Model to model and integrate processes, products, and people that are all important elements of Industry 4.0.

REFERENCES

[1] André Barthelmey, Eunseo Lee, Ramy Hana, and Jochen Deuse. 2019. Dynamic digital twin for predictive maintenance in flexible production systems. In *IECON 2019 - 45th Annual Conference of the IEEE Industrial Electronics Society*, Vol. 1. IEEE, Lisbon, Portugal, Portugal, 4209–4214.

[2] Yi Cai, Binil Starly, Paul Cohen, and Yuan-Shin Lee. 2017. Sensor Data and Information Fusion to Construct Digital-twins Virtual Machine Tools for Cyber-physical Manufacturing. *Procedia Manufacturing* 10 (2017), 1031–1042.

[3] Jean-Paul Calbimonte, Sofiane Sarni, Julien Eberle, and Karl Aberer. 2014. XGSN: An Open-source Semantic Sensing Middleware for the Web of Things. In *TC/SSN@ISWC*. CEUR, Garda, Trentino, Italy.

[4] Camunda. 2020. Process Automation reinvented for the Digital Enterprise. https://camunda.com/

[5] Michael Compton, Payam Barnaghi, Luis Bermudez, Raúl García-Castro, Oscar Corcho, Simon Cox, John Graybeal, Manfred Hauswirth, Cory Henson, Arthur Herzog, Vincent Huang, Krzysztof Janowicz, W. David Kelsey, Danh Le Phuoc, Laurent Lefort, Myriam Leggieri, Holger Neuhaus, Andriy Nikolov, Kevin Page, Alexandre Passant, Amit Sheth, and Kerry Taylor. 2012. The SSN ontology of the W3C semantic sensor network incubator group. *Journal of Web Semantics* 17 (2012), 25 – 32. https://doi.org/10.1016/j.websem.2012.05.003

[6] Cumulocity. 2020. Introduction to Cumulocity IoT. https://cumulocity.com/guides/concepts/introduction/

[7] Inc Eclipse Foundation. 2020. Eclipse rdf4j. https://rdf4j.org/

[8] Rossmann Engineering. 2020. THE STANDARD LIBRARY FOR MODBUS COMMUNICATION. http://easymodbustcp.net/en/

[9] Abdur Rahim Mohammad Forkan, Montori Federico, Dimitrios Georgakopoulos, Prem Prakash Jayaraman, Ali Yavari, and Ahsan Morshed. 2019. An Industrial IoT Solution for Evaluating Workers' Performance Via Activity Recognition. In *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, Dallas, TX, USA, USA, 1393–1403.

[10] OPC Foundation. 2020. Collaborations: PackML, PRODML, MDIS, AutomationM. https://opcconnect.opcfoundation.org/2016/09/collaborations-packml-prodml-mdis-automationml/

[11] Dimitrios Georgakopoulos, Prem Prakash Jayaraman, Maria Fazia, Massimo Villari, and Rajiv Ranjan. 2016. Internet of Things and Edge Cloud Computing Roadmap for Manufacturing. *IEEE Cloud Computing* 3, 4 (2016), 66–73.

[12] MTConnect Institute. 2020. MTConnect R Standard. https://www.mtconnect.org/standard-download20181

[13] Krzysztof Janowicz, Armin Haller, Simon J.D. Cox, Danh Le Phuoc, and Maxime Lefrançois. 2019. SOSA: A lightweight ontology for sensors, observations, samples, and actuators. *Journal of Web Semantics* 56 (2019), 1 – 10. https://doi.org/10.1016/j.websem.2018.06.003

[14] Apache JMeter. 2020. Apache Software Foundation. https://jmeter.apache.org/

[15] Aleksey Kychkin and Aleksadr Nikolaev. 2020. IoT-based Mine Ventilation Control System Architecture with Digital Twin. In *2020 International Conference on Industrial Engineering, Applications and Manufacturing (ICIEAM)*. IEEE, Sochi, Russia, Russia, 1–5.

[16] Stefan-Helmut Leitner and Wolfgang Mahnke. 2006. OPC UA - Service-oriented Architecture for Industrial Applications. *Softwaretechnik-Trends* 26, 6 (2006), 1–7.

[17] Inc Lightbend. 2020. Build powerful reactive, concurrent, and distributed applications more easily. https://akka.io/

[18] Microsoft. 2020. Understand and use device twins in IoT Hub. https://docs.microsoft.com/en-us/azure/iot-hub/iot-hub-devguide-device-twins

[19] Thiago Rodrigues Alves, Mario Buratto, Flavio Mauricio de Souza, and Thelma Virginia Rodrigues. 2014. OpenPLC: An open source alternative to automation. In *IEEE Global Humanitarian Technology Conference (GHTC 2014)*. IEEE, San Jose, CA, USA, 585–589.

[20] Ali Salehi, Jose Jimenez-Berni, David Deery, Doug Palmer, Edward Holland, Pablo Rozas-Larraondo, Scott Chapman, Dimitrios Georgakopoulos, and Robert Furbank. 2015. SensorDB: a virtual laboratory for the integration, visualization and analysis of varied biological sensor data. *Plant Methods* 11 (2015), 108–111. http://search.proquest.com/docview/1779613409/

[21] Greyce N Schroeder, Charles Steinmetz, Carlos E Pereira, and Danubia B Espindola. 2016. Digital Twin Data Modeling with AutomationML and a Communication Methodology for Data Exchange. *IFAC PapersOnLine* 49, 30 (2016), 12–17.

[22] Nicolas Seydoux, Khalil Drira, Nathalie Hernandez, and Thierry Monteil. 2016. IoT-O, a Core-Domain IoT Ontology to Represent Connected Devices Networks. In *Knowledge Engineering and Knowledge Management: 20th International Conference, EKAW 2016, Bologna, Italy, November 19-23, 2016, Proceedings 20*, Vol. 10024. Springer, Bologna, Italy, 561–576. https://hal.archives-ouvertes.fr/hal-01467853

[23] Siemens. 2020. MindSphere – the Industrial Internet of Things solution. https://new.siemens.com/au/en/products/software/discover-mindsphere.html

[24] John Soldatos, Nikos Kefalakis, Manfred Hauswirth, Martin Serrano, Jean-Paul Calbimonte, Mehdi Riahi, Karl Aberer, Prem Prakash Jayaraman, Arkady Zaslavsky, Ivana Podnar Žarko, Lea Skorin-Kapov, and Reinhard Herzog. 2015. OpenIoT: Open Source Internet-of-Things in the Cloud. In *Interoperability and Open-Source Solutions for the Internet of Things*, Ivana Podnar Žarko, Krešimir Pripužić, and Martin Serrano (Eds.). Springer International Publishing, Cham, 13–25.

[25] Matthias Thoma, Torsten Braun, Alexandru-Florian Antonescu, and Carsten Magerkurth. 2014. Managing Things and Services with Semantics: A Survey.

[26] Jan Vachalek, Lukas Bartalsky, Oliver Rovny, Dana Sismisova, Martin Morhac, and Milan Loksik. 2017. The digital twin of an industrial production line within the industry 4.0 concept. In *2017 21st International Conference on Process Control (PC)*. IEEE, Štrbské Pleso, Slovakia, 258–262.

[27] Wikipedia. 2020. Actor Model. https://en.wikipedia.org/wiki/Actor_model

[28] Kamil Židek, Ján Piteľ, Milan Adámek, Peter Lazorík, and Alexander Hošovský. 2020. Digital Twin of Experimental Smart Manufacturing Assembly System for Industry 4.0 Concept. *Sustainability* 12, 9 (2020), 3658. http://search.proquest.com/docview/2398421237/