Digital Twin as a Proxy for Industrial Cyber-Physical Systems

Abdullah Aziz
Luleå University of Technology
Luleå, Sweden
abdullah.aziz@ltu.se

Olov Schelén
Luleå University of Technology
Luleå, Sweden
olov.schelen@ltu.se

Ulf Bodin
Luleå University of Technology
Luleå, Sweden
ulf.bodin@ltu.se

ABSTRACT

Digital twins (DTs) in industrial cyber-physical systems (ICPSs) are an essential concept for Industry 4.0. In recent years, DT architectures and frameworks for industry have been the focus of several research initiatives regarding analytics, predictive maintenance, virtual evaluation & verification, and fault detection & diagnosis. However, the resource constraint and battery-driven devices in an ICPS environment are facing challenges such as short life spans due to high energy consumption, lower availability of the services, and low security capabilities. To meet these challenges, this paper identifies some critical properties for DTs, including energy efficiency, high availability & state persistence, remote & contention control, and security. To provide services in ICPSs, we define the concept of a DT as a proxy (DTaaP) and a four-layer architectural model for the DTaaP that enables meeting the identified properties. We present a generic proof of concept (PoC) implementation by using Eclipse Ditto, which is an open-source digital twin framework. We evaluate our artifact against the listed properties through an experimental scenario. We show that the DTaaP improves energy efficiency, that the DT can serve as an anchor point for security, and that the DTaaP provides availability and persistence for duty cycling devices. We also identify some limitations of Eclipse Ditto for PoC DTaaP implementation, which can be a focus for future research.

KEYWORDS

Digital Twins, CPS, Industry 4.0, DTaaP.

ACM Reference Format:
Abdullah Aziz, Olov Schelén, and Ulf Bodin. 2023. Digital Twin as a Proxy for Industrial Cyber-Physical Systems. In *2023 10th International Conference on Wireless Communication and Sensor Networks (icWCSN) (icWCSN 2023), January 06–08, 2023, Chengdu, China.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3585967.3585982

1 INTRODUCTION

The Digital twins (DTs) refer to digital representations of physical assets, called physical twins (PTs), including systems, processes, and devices [21]. Michael Grieves developed the idea for the DT concept in 2003 and introduced it at the University of Michigan [10]. Since then, the DT paradigm has sparked much attention in academics and industry. Initially, the manufacturing environment adopted the concept of DTs and later adopted by the environment of the Industrial Internet of Things (IIoT) and cyber-physical systems (CPSs). CPSs [17] have been proposed as a fundamental concept of Industry 4.0 [15] architectures. CPSs incorporate sensing, computing, control, and networking into physical devices and infrastructure, allowing them to connect to cyberspace. A DT can be seen as a logical hub that emphasizes the Industry 4.0 characteristics of a product, device, or system. A DT may monitor and communicate with a physical entity [27]. A DT is linked to a CPS and appears to be conceptually comparable [31]. DTs can be implemented for monitoring, diagnostics, prediction, and control in industrial cyber-physical systems (ICPSs) [21]. A significant aspect of ICPSs is the ability to integrate information technologies, operational technologies such as embedded systems and control systems, and physical systems to generate new or enhanced functions.

1.1 Challenges

Although CPSs provide an increasing level of automation and integration across industrial environments, there are some challenges. With CPSs, most of the embedded systems are resource constrained devices, such as battery-driven sensors unable to provide services around the clock. Hence, the availability of such devices is challenging. The remote control of devices deployed in risky industrial sites is also an important challenge. Moreover, the governance of the devices and services is also challenging, e.g., who can access devices under what rules and who can manage these devices and their services. Security & authorization are other key challenges to address when monitoring and controlling devices remotely. Different devices provide various services; hence, sensitive services should have specific access rights. Unfortunately, CPS design and implementation is often ad hoc, which results in complex message-based solutions.

1.2 Scope

The scope of this article is to address these challenges by identifying some critical properties for digital twins and proposing an architectural model of digital twins as a proxy (DTaaP) for resource constrained devices in ICPSs. A generic implementation is outlined and evaluated in a particular use case. With DTaaP, a digital representation can be separated from its physical counterpart and act as a service proxy for communicating with consumers.

1.3 Contributions

The contribution of this article includes the definition of some critical properties for DTaaP, the proposal of a DTaaP four-layer architectural model enabling those properties, and a proof-of-concept (PoC) implementation. The four layers of the architecture are the device layer, communication layer, proxy layer, and application layer. Our proof-of-concept (PoC) of DTaaP implementation is based on

icWCSN 2023, January 06–08, 2023, Chengdu, China
© 2023 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-9846-6/23/01.
https://doi.org/10.1145/3585967.3585982

open source technologies and tools. In the PoC, we use the Eclipse Ditto digital twin framework (DTF) for our proxy layer since it is well suited for our DTaaP properties. Moreover, we design and implement an experimental scenario to show DTaaP in action. We evaluated our artifact by implementing an experimental scenario against the listed properties. Finally, we discuss the limitations in the current PoC implementation to be compliant with the properties.

1.4 Outline

The remainder of this paper is structured as follows. Section 2 explains the related work, which includes the literature on DT for exposing services. Section 3 defines some critical properties for digital twins. Section 4 proposes an architectural model for DT as a proxy. Section 5 provides a proof-of-concept implementation, and Section 6 provides an experimental demonstration. Section 7 describes the evaluation of the experiment against the critical properties. Section 8 describes the limitations in the current implementation with future research areas. Finally, Section 9 concludes the paper.

2 RELATED WORK

In recent years, DT architectures and frameworks have been the focus of many research initiatives [1, 11, 14, 29, 30, 32]. A five-dimensional digital twin (5D-DT), where “physical entity", “virtual entity", “connection", “data" and “services" are defined as the five dimensions, is presented in [32]. The generic digital twin architecture (GDTA) using the concept of 5D-DT was proposed in [29]. The GDTA model is parallel with the six information technology (IT) layers of the Reference Architecture Model for Industry 4.0 (RAMI 4.0) [2].

DT can be seen in different life cycle phases in industrial applications [19]. Common DT applications are iterative optimization [22, 35], virtual evaluation & verification [12, 24], real-time monitoring [18, 28, 33, 39], production control [38], asset management [6, 16], predictive maintenance [3, 23, 26], fault detection & diagnosis [34, 37], state monitoring [20, 36], and performance prediction [5, 8, 13].

Those works primarily use DTs for evaluation, prediction, and monitoring. However, a DT can also act as a proxy or middleware service provider to cater to resource constraints and battery-driven devices and devices that require secure and controlled sharing. Therefore, our focus in this article is to introduce and propose a concept of DTaaP with a PoC implementation.

3 CRITICAL PROPERTIES FOR DIGITAL TWINS

We identify the following properties of the DTaaP model needed to address previously mentioned challenges:

- Energy efficiency: Many service requests can be processed by a proxy, which does not consume power from the physical device and allows the physical device to duty cycle while the digital representations ensure service continuity. The latter is common for sensors that only perform periodic readings and wake up to deliver the latest measurement while sleeping in between.
- High Availability & State Persistence: Caching persistently stores the latest reported state of a device when the device is not connected. The persisting information can continue providing data to consumers, which can request the latest state information of the device anytime. Availability in the context of an ICPS refers to the ability of a consumer to access information or consume service even when a CPS is duty cycling, nonfunctioning, or facing failure. The DT of a CPS can not only provide availability at a specific time of failure but can also provide high availability by one-to-many or many-to-many mapping between DTs and PTs. In this way, when failures occur, a DT can continue to serve consumers by moving the operations from the failed CPS to another.
- Remote & Contention Control: Devices deployed in risky sites can be controlled (actuated) remotely by providing service functions from their digital replicas. An operator can remotely control a device by sending control commands through its digital replica from anywhere over the internet. Atomic access for controlling a device is important to control contention such that if an operator is performing some action on the device, no other user should have access to interrupt the action at that time.
- Security: It is error-prone and costly to make physical devices addressable across the internet. Instead, a digital counterpart can be used to face the world. Strong, generic, and computationally expensive security can be provided by a digital counterpart while the physical part can support security with only one digital counterpart (or a few), which reduces vulnerabilities and saves resources (e.g., by using only symmetric keys) for encryption.

4 DIGITAL TWIN AS A PROXY (DTAAP) : AN ARCHITECTURAL MODEL

teaserfigure

In this section, we present our proposed DTaaP architectural model as depicted in Figure 1. This architecture allows us to create a DT of a physical device based on its attributes and features, connect DTs and PTs, and enable interested authorized parties to consume the services of a PT through its DT proxy. The architectural model is composed of four main layers. Figure 1 shows the sequence of layers, from left to right.

4.1 Device Layer

All physical industrial resources, such as products, workers, equipment, technology, processes, environments, and facilities, are included in this layer. This layer of the proposed architectural model is a main industrial environment that a company wants to control. This layer encompasses all observable devices that must be monitored and sensed, as well as those that can be actuated and controlled.

4.2 Communication Layer

This layer is responsible for connecting a physical device to its virtual replica and establishes communication for their synchronization. This represents a communication channel for the exchange

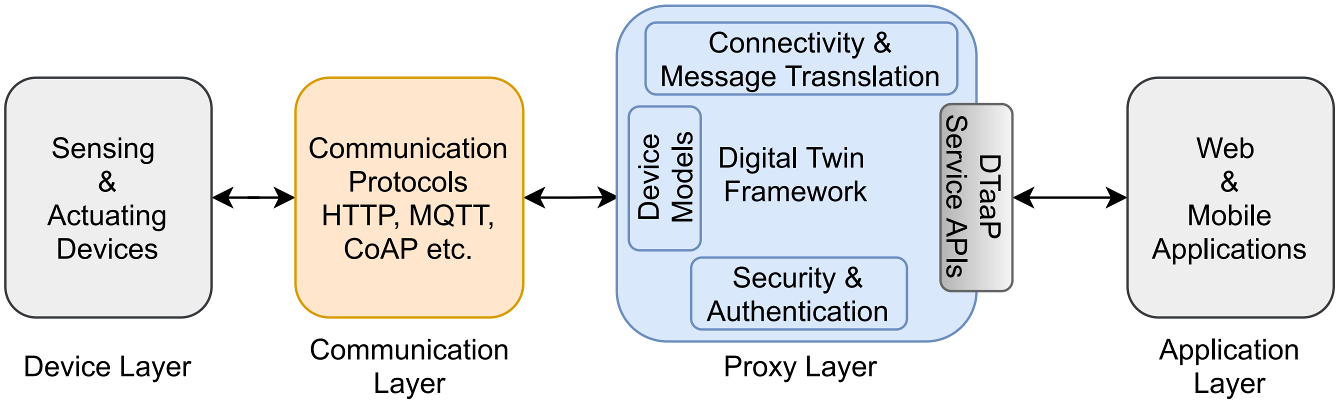
Sensing & Actuating Devices
Device Layer
Communication Protocols HTTP, MQTT, CoAP etc.
Communication Layer
Connectivity & Message Trasnslation
Device Models
Digital Twin Framework
DTaaP Service APIs
Security & Authentication
Proxy Layer
Web & Mobile Applications
Application Layer

Figure 1: DTaaP Architectural Model

of messages between PTs and DTs. The communication layer can utilize a suitable communication architecture such as an event-driven architecture (EDA) or a service-oriented architecture (SOA) according to the needs. This layer can provide one-way and two-way communication models for monitoring the sensed data and controlling the actuator with commands/responses, respectively.

4.3 Proxy Layer

The proxy layer is in charge of abstracting a real-world device, including all of its capabilities and features, as well as its digital representation. The digital twin framework (DTF) that stores digital twins according to device models, provides connectivity & message translation to communicate with digital twins, and ensures that digital twins can only be accessed by authorized parties, is deployed in this layer. Authorized access is achieved by policies that can manage the access rights for every single feature of a PT. The DTF is comprised of four main components:

4.3.1 Device Model. A device model is an object that describes the physical device and is primarily used as a hook for multiple features and attributes to this device. The device model contains the attributes that describe the device in more detail. The device model may contain an arbitrary number of features. A feature is used to manage all data and functionality of a device.

4.3.2 Connectivity & Message Translation. This DTF module is responsible for establishing connections with the communication layer and applications. An application from the application layer can consume services provided by a physical device through its digital twin. Similarly, a device in the device layer can synchronize its updated state or data with its digital twin through the communication layer. Moreover, message translation from a source to a destination can sometimes also be needed, which can also be handled by this DTF module.

4.3.3 Security & Authentication. This module is in charge of protecting the data and functionality of a digital twin. Each digital twin should have a policy that describes the authentication and authorization rights of interested parties. This module ensures that the requester has the right to see, use, or change the information he wants to access.

4.3.4 Service APIs. This module allows a digital twin of a device to act as a proxy by providing the service APIs. By using these DTF APIs, authorized users with valid access rights can manage a device and its features. APIs can also allow integration with the external world application for monitoring, controlling, analytics, and many other functions according to needs.

4.4 Application Layer

Different applications that can allow communication with a digital twin via the API services provided by the DTF can be deployed in the application layer. The applications in this layer cannot directly communicate with the devices in the device layer. Instead, the DT of a device is responsible for providing all the services to all authorized applications.

5 PROOF OF CONCEPT (POC): DTAAP IMPLEMENTATION

This section describes our generic proof of concept implementation of the DTaaP architecture using open source technologies. Section 5.1 provides a set of open source technologies for each layer of the architectural model, Section 5.2 describes the complete process of DT instantiation, and Section 5.3 describes major DTaaP service APIs.

5.1 Toolkit for DTaaP Implementation

In this subsection, we list the open source technologies and tools for each layer of the architecture model to implement the PoC.

5.1.1 Device Layer. For the device layer of the architectural model, various microcontrollers or development boards, such as “Raspberry Pi" and “Arduino", can be used. A device can have both sensing and actuating features. These features can be added by using various sensors and actuators.

5.1.2 Communication Layer. The communication layer has various protocols, such as the message queue telemetry transfer (MQTT) protocol , advanced message queuing protocol (AMQP), constrained application protocol (CoAP), hypertext transfer protocol (HTTP), and open platform communications united architecture (OPC-UA), that can be used based on the use case scenario and needs. For an event-driven communication protocol, the broker can be deployed in a gateway.

5.1.3 Proxy Layer. The proxy layer is responsible for creating and managing the DT of a PT and providing services to all authorized interested parties. Hence, the digital twin framework (DTF) is deployed in this layer that provides all the functionalities for connectivity, security, and services. There are various available digital twin frameworks provided by different vendors, such as Microsoft Azure Digital Twin [25]), the CPS Twinning framework [7], Eclipse Ditto [9], and Amazon Device Shadow [4]. We selected Eclipse Ditto as the DTF in the proxy layer to propose our DTaaP architectural model.

5.1.4 Application Layer. This layer offers various ways to develop applications to interact with the DTF by consuming the services provided by the DTF. Mobile, web or other control and monitoring applications can be developed in this layer. To interact with a DT through the application layer, a user must be authorized and have access rights to desired features of a DT.

5.2 DTaaP: Digital Twin Instantiation

This section illustrates the steps for DT instantiation in the Ditto DTF. The instantiation of a DT in Ditto requires the creation of policies for the access control, device model, and connection between a DT and PT.

5.2.1 Create Policy. A device can have various sensing and control features. The purpose of the policy is to define which users can access a DT and whether a requesting user has the access rights to perform read or write operations on the specific DT feature. Therefore, a DT should have a policy for each feature to control the access rights to that feature. Initially, we define the policy in a JSON object and store it in a file "*policy.json*". Later, to create a policy, an authorized admin user needs to make a *PUT* request to Ditto with a payload of "*policy.json*", as shown in Figure 2.

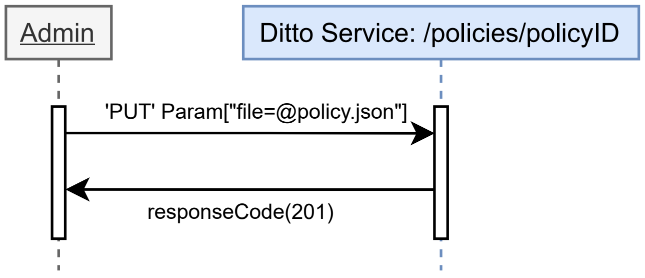
Admin
Ditto Service: /policies/policyID
'PUT' Param["file=@policy.json"]
responseCode(201)

Figure 2: Sequence diagram for creating a policy by consuming the Ditto service endpoint.

5.2.2 Create DT. The DT of a device can be created in the Ditto DTF by defining the device model in the JSON file "*deviceModel.json*". The device model contains information about the device, such as the device ID, attributes, and features. The device model also contains the policy ID of the created policy for access control of the attributes and features of this DT. To create a DT of a device, an authorized admin user needs to make a *PUT* request to Ditto with the payload of file "*deviceModel.json*", as shown in Figure 3.

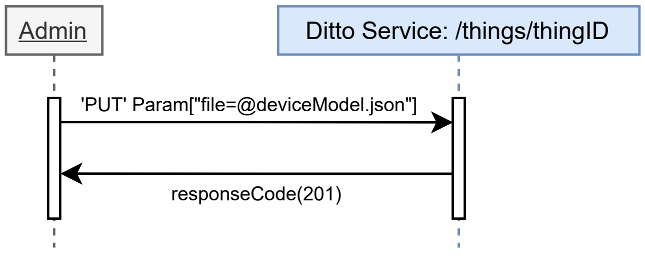
Admin
Ditto Service: /things/thingID
'PUT' Param["file=@deviceModel.json"]
responseCode(201)

Figure 3: Sequence diagram for creating a DT by consuming the Ditto service endpoint.

5.2.3 Create a Connection. Once a DT is created in the Ditto DTF, a connection needs to be created to establish communication between a DT and PT. A connection represents a communication channel for the one-way or two-way exchange of messages between a PT and DT. Ditto uses the Ditto protocol for communication between physical devices. This protocol is not a human-friendly protocol. To ensure proper communication between PT and a DT, sometimes it is necessary to map the incoming message from the device to the Ditto protocol. Based on the use case scenario and communication needs, a communication protocol and connection details need to be defined to establish a successful connection. The connection details should be defined in a JSON object that contains information about the *connectionType*, *connectionStatus*, *URI* of a gateway or broker, *source* and *target* information, and the *mapping context*. The connection details can be defined in the "*connectionDetails.json*" file; and to create the connection, an authorized admin user needs to make a *POST* request to Ditto with a payload of file "*connectionDetails.json*", as shown in Figure 4.

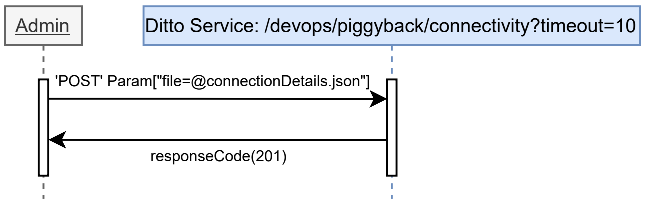
Admin
Ditto Service: /devops/piggyback/connectivity?timeout=10
'POST' Param["file=@connectionDetails.json"]
responseCode(201)

Figure 4: Sequence diagram for creating connections by consuming the Ditto service endpoint.

A DT will be successfully instantiated after establishing a communication channel between the DT and PT.

5.3 DTaaP: Service APIs

To allow applications to interact with a DT, DTF provides services such as REST-APIs. An application can consume these services to access a DT and its features. A user can monitor a specific feature of a device through its DT by consuming the GET REST-API of that feature. An application can also transmit control commands to a specific feature of a device through a DT by consuming services. The DTF also provides administrative services for managing policies, DTs, and their features. Only administrative services have the right to create, update, and delete the policies, DTs, and features of a DT.

6 EXPERIMENT: DTAAP IN ACTION

In this section, we design an experimental scenario to demonstrate how a DT can act as a proxy based on the proposed DTaaP architectural model and generic implementation described in Sections 4 and ??. For the purpose of prototyping and demonstrating the DTaaP model in action, we design an experiment with a one-to-one mapping between DTs and PTs. First, in Section 6, we describe the experimental scenario and setup. Later, in Section 6.3, we describe the DTaaP in action.

6.1 Experimental Scenario & Setup

In order to demonstrate the DTaaP in action, we design an experimental scenario of a device with both sensing and actuating abilities. The device has a temperature & humidity sensor as sensing features and a two-coloured LED for the control actuator. The idea here is that the device transmits the sensed data to its DT, and all the consumers can read the data from the DT. Moreover, for the LED actuator, an authorized user can transmit control messages to the device through its DT to allow secure remote control. The experimental setup with the tools and technologies for each layer is shown in Figure 5. For the device layer, we used a *Raspberry Pi 4 model B* with a *DHT11* temperature & humidity sensor to measure the temperature and humidity from the atmosphere. Similarly, for actuation, we used a two-coloured green and red LED. We developed this device using the *Raspbian OS* and coded both the sensing and actuating features in the *NodeJs* environment. We used and deployed an MQTT broker in the communication layer to establish an event-driven communication medium between a PT and DT. As our DTaaP PoC is based on the Eclipse Ditto DTF, we deployed Ditto in the proxy layer. Ditto allows abstracting a device in its DT and provides services both for consumer applications and for administration. For the application layer, we developed a web-based application that communicates with the Ditto DTF using AJAX calls. Different users can log into the DTF with the correct credentials through this web application. Users can monitor the state and latest information of a PT through its DT created in Ditto. Similarly, users can send control commands to actuators from the web application to its DT, which then transmits the command to the actual device.

6.2 Digital Twin Creation

Once all the layers of the architecture are developed and deployed successfully, as shown in Figure 5, we need to create a DT of the device in the Ditto DTF. We consume the administrative services of Ditto to complete the creation of a DT after defining the policy and then establishing a connection between the DT and its PT.

6.2.1 Policy Creation. First, we need to create a policy in the Ditto DTF. For our experimental scenario, we have defined a policy for two users: an “admin" and an “observer". We grant all the permissions and access rights to the “admin" user, and we only allow read access rights to the “observer" user so the “observer" can only obtain the latest state from the DT features. We created the policy in Ditto by making the request to Ditto, as shown in Figure 2.

6.2.2 Create DT. In this experimental device, we have two main features of our device: temperature and humidity. We used the same policy ID we created in the previous step to define the device model with temperature and humidity features. We created a DT in Ditto by making the request to Ditto, as shown in Figure 3.

6.2.3 Create Connection. Finally, to establish communication between a PT and DT, we created a two-way connection for monitoring the sensor state and for sending control commands to a two-coloured LED actuator based on MQTT topics. For the mapping function, we used JavaScript as a mapping method to translate the payload in the Ditto protocol. We created a connection between a DT in Ditto and its PT by making a *POST* request to Ditto with a payload of connection details, as shown in Figure 4.

6.3 DTaaP in Action

The DTaaP is active after implementing all the layers, creating a DT in the DTF, and establishing a connection between a PT and DT. Ditto exposes services as REST APIs for each DT to access using consumer applications and verifies access rights using the policy being used by a DT. As our PT is connected with its DT inside the DTF and all consumer applications only have access to the DT instead of being directly connected with the PT, the DT acts as a proxy, as shown in the architectural model.

The sensor module of our PT now only needs to send data to its DT instead of serving hundreds of consumers directly. The DT is responsible for maintaining the latest state and information of the PT. Even though the sensor module is in a sleep state, the DT can provide the latest sensor readings to all authorized interested parties. Whenever a sensor module wakes up, it reads the latest information and publishes it to its DT, which can then be read by authorized applications.

Similar to the sensor module of the PT, the actuator module only receives control commands from its DT. Whenever a user wants to turn the light to or from a specific colour, it sends the control command to the DT; the DT performs all the security and authorization checks; and if everything is fine, the DT transmits the command to the actual PT. In this way, a user can achieve remote control of the device.

7 EVALUATION: DTAAP PROPERTIES

In this section, we evaluated our DTaaP experimental scenario based on the properties defined in Section 1. We have listed four properties to achieve by implementing the DTaaP model, and the evaluation and demonstration of these properties are explained as follows:

7.1 Energy efficiency

We can evaluate the efficiency of this model by considering a scenario of our sensing module, which is a temperature & humidity sensor. Assume that a new battery allows this module to provide services 2000 times in its life. If 20 consumers consume this service 3 times per day (morning, afternoon, and evening), the module needs to provide service 20x3 = 60 times per day; hence, the battery will be dead in approximately 2000/60 = 33.34 days.

However, if the device only needs to write the sensed data to its DT three times per day, then all 20 consumers can obtain the sensed data from their DT without consuming the device energy. In this way, the device will provide service 3 times per day; hence, the battery will live for approximately 2000/3 = 666.67 days, which is

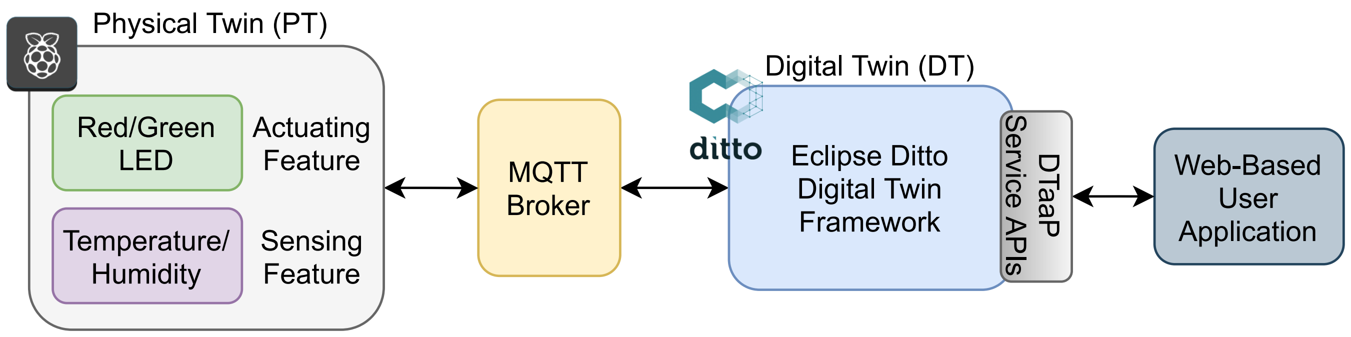

Figure 5: Experimental Scenario Setup

approximately twenty times more energy efficient than the previous scenario.

7.2 High Availability & Persistent State

The availability & persistent state is another important property of the DTaaP model. We achieve this property by establishing the temperature & humidity features in our experimental scenario. The sensor module wakes up and sends the sensor readings to the DT in intervals. All the users can obtain the desired feature readings from the DT without waking up the PT. The DT persists the last state of the specific feature and makes it available for all authorized consumers. As shown in Figure 6, a user is able to obtain the last monitored temperature from the DT, even when the sensor module is in a sleep state. Even if the sensor module is duty cycling or fails, the DT can continue proving the services based on the last persistent state. However, as in this example scenario, the mapping between a DT and a PT is one-to-one, and a DT cannot move the operations of a failed module to any other PT; hence, high availability cannot be demonstrated.

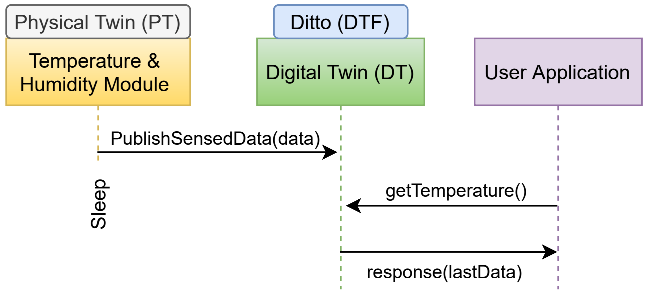

Figure 6: Sensed data are always available even when the device is in a sleep state.

7.3 Remote & Contention Control

As mentioned in Section 1, the remote control property of a device is considered very beneficial in risky industrial sites. In our DTaaP experimental setup, we have demonstrated LED actuator control via user application through a DT. Figure 7 shows the control command flow from the user application to a DT, which then transmits the control command to the PT LED actuator after verifying the authorization and access rights. The experimental example shows secure remote control in action; however, the contention control property that assures atomic control at a unit time is missing in the demonstration.

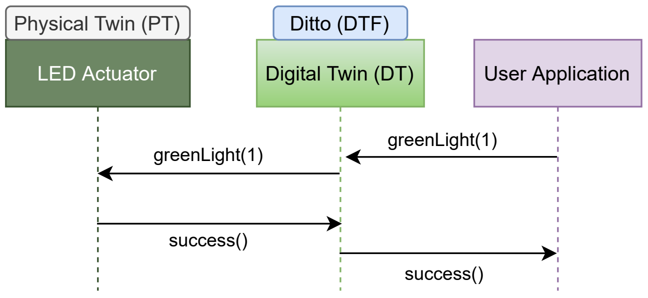

Figure 7: Remote control of an LED through a DT

7.4 Security

Resource constraint devices in an ICPS environment are not able to have rich security features against hundreds of users. In our experimental device setup, it is not possible to maintain records for hundreds of users in the device, whether in database records or shared keys. The device cannot even maintain an access control policy on each feature against all the users. Furthermore, the verification of security functions also needs extensive computation, which is unrealistic to achieve by the device. However, with the DTaaP model, the device only needs to maintain authorization and verification of one instance of its DT. The DTF in the DTaaP model is resource-rich and able to perform all the security functions on the DT before providing data to consumer applications and transmitting commands to a PT. In our experimental scenario, a user must log into the DTF with authorized credentials before consuming any service or transmitting control commands. Moreover, at the time of creating a DT, we specified a policy for access control of different features by various users. Hence, the security of the DTaaP model is very efficient.

8 LIMITATIONS & FUTURE WORK

The DTaaP model provides very beneficial properties in an ICPS environment, which are listed in Section 3. Although the PoC of the

DTaaP implementation is based on the best suitable open source tools and technologies, there are limitations to achieving some DTaaP properties, such as contention control and high availability, for these tools and technologies.

Consider a scenario where multiple users want to control the same actuator and send different control commands. For example, in our experimental scenario, assume user "A" wants to give a green signal by turning on the green light for ten seconds; and at the third second, user "B" wants to turn the signal to red, which will reverse the condition for user "A". Hence, contention control of an actuator to send control commands is sometimes needed, which cannot be achieved because the Ditto DTF currently does not support this feature.

Similarly, as the DTaaP is in an early stage, where we currently have a one-to-one mapping between a DT and PT, it cannot provide high availability that is useful to verify accurate information, to determine partial failures, or to replace a failed component with another one.

Another limitation of the DTaaP PoC implementation is that the Ditto DTF only stores the latest state of a PT in its DT and does not maintain the history. Some applications, such as analytical applications, can require historical state data of a PT from a DT.

These limitations of the DTaaP PoC implementation can be addressed in the future. High availability can be achieved by enriching the DTaaP model with one-to-many or many-to-many mappings between DTs and PTs. The contention control property can also be considered in future work by maintaining mutual exclusion if a user already holds control of the actuator and not allowing another user to control the same actuator at the same time. Maintaining historical data of a PT in its DT can also be the focus of future work. Multiple distributed DTs of a single PT for various operational needs could be another interesting research area to focus on in the future. Furthermore, we may consider more extensive evaluations, such as qualitative evaluations using a set of use cases against the listed properties and quantitative evaluations using more metrics, of the DTaaP model.

9 CONCLUSION

This paper proposed a DTaaP architectural model for a DT to act as a proxy for a PT. The DTaaP architectural model provides four main properties: energy efficiency, availability & state persistence, remote control, and security. These properties of the DTaaP architectural model address the challenges of resource constrained devices in an ICPS environment, such as long lifetime, availability of the device services around the clock, and rich security for authorization and control.

The proposed DTaaP architectural model has four layers: a device layer, a communication layer, a proxy layer, and an application layer. The PoC DTaaP implementation is based on the open source Eclipse Ditto DTF in the proxy layer. We designed and implemented an experimental scenario to evaluate our artifact against the listed properties.

The experimental evaluation shows how energy efficiency is improved and how the DT serves as an effective and efficient anchor point for security. Furthermore, the evaluation shows how availability and persistence are provided for duty cycling devices as the DT replicates their services.

ACKNOWLEDGMENTS

We would like to thank Dr. Mohammad Hanif at LTU for his valuable input and support in brainstorming and understanding the concept and limitations. This research work was funded by the Arrowhead Tools research project with grant agreement no. 826452.

REFERENCES

[1] Sailesh Abburu, Arne J Berre, Michael Jacoby, Dumitru Roman, Ljiljana Stojanovic, and Nenad Stojanovic. 2020. COGNITWIN–Hybrid and cognitive digital twins for the process industry. In *2020 IEEE International Conference on Engineering, Technology and Innovation (ICE/ITMC)*. IEEE, 1–8.

[2] Peter Adolphs, Heinz Bedenbender, Dagmar Dirzus, Martin Ehlich, Ulrich Epple, Martin Hankel, Roland Heidel, Michael Hoffmeister, Haimo Huhle, Bernd Kärcher, et al. 2015. Reference architecture model industrie 4.0 (rami4. 0). *ZVEI and VDI, Status report* (2015).

[3] P Aivaliotis, K Georgoulias, Z Arkouli, and S Makris. 2019. Methodology for enabling digital twin using advanced physics-based modelling in predictive maintenance. *Procedia Cirp* 81 (2019), 417–422.

[4] Amazon. 2021. *AWS IoT Device Shadow service - AWS IoT Core.* Retrieved May 15, 2021 from https://docs.aws.amazon.com/iot/latest/developerguide/iot-device-shadows.html

[5] Andrea Coraddu, Luca Oneto, Francesco Baldi, Francesca Cipollini, Mehmet Atlar, and Stefano Savio. 2019. Data-driven ship digital twin for estimating the speed loss caused by the marine fouling. *Ocean Engineering* 186 (2019), 106063.

[6] Zhuang Cunbo, Jianhua Liu, and Hui Xiong. 2018. Digital twin-based smart production management and control framework for the complex product assembly shop-floor. *The international journal of advanced manufacturing technology* 96, 1-4 (2018), 1149–1163.

[7] Matthias Eckhart and Andreas Ekelhart. 2018. *CPS Twinning framework.* Retrieved May 15, 2021 from https://github.com/sbaresearch/cps-twinning

[8] Matthias Eckhart and Andreas Ekelhart. 2018. Towards security-aware virtual environments for digital twins. In *Proceedings of the 4th ACM workshop on cyber-physical system security.* 61–72.

[9] Eclipse. 2021. *Eclipse Ditto.* Retrieved May 15, 2021 from https://github.com/eclipse/ditto

[10] Michael Grieves. 2005. *Product lifecycle management: driving the next generation of lean thinking: driving the next generation of lean thinking: driving the next generation of lean thinking.* McGraw Hill Professional.

[11] Michael Grieves. 2014. Digital twin: manufacturing excellence through virtual factory replication. *White paper* 1 (2014), 1–7.

[12] Jiapeng Guo, Ning Zhao, Lin Sun, and Saipeng Zhang. 2019. Modular based flexible digital twin for factory design. *Journal of Ambient Intelligence and Humanized Computing* 10, 3 (2019), 1189–1200.

[13] Hee Yong Jeon, Cedric Justin, and Dimitri N Mavris. 2019. Improving Prediction Capability of Quadcopter Through Digital Twin. In *AIAA Scitech 2019 Forum.* 1365.

[14] Klementina Josifovska, Enes Yigitbas, and Gregor Engels. 2019. Reference framework for digital twins within cyber-physical systems. In *2019 IEEE/ACM 5th International Workshop on Software Engineering for Smart Cyber-Physical Systems (SEsCPS).* IEEE, 25–31.

[15] Henning Kagermann, Wolfgang Wahlster, and Johannes Helbig. 2013. Securing the future of German manufacturing industry: Recommendations for implementing the strategic initiative INDUSTRIE 4.0. *Final report of the Industrie* 4, 0 (2013).

[16] Kalpana Kannan and N Arunachalam. 2019. A digital twin for grinding wheel: an information sharing platform for sustainable grinding process. *Journal of Manufacturing Science and Engineering* 141, 2 (2019).

[17] Edward A Lee. 2008. Cyber physical systems: Design challenges. In *2008 11th IEEE international symposium on object and component-oriented real-time distributed computing (ISORC).* IEEE, 363–369.

[18] Jiewu Leng, Hao Zhang, Douxi Yan, Qiang Liu, Xin Chen, and Ding Zhang. 2019. Digital twin-driven manufacturing cyber-physical system for parallel controlling of smart workshop. *Journal of ambient intelligence and humanized computing* 10, 3 (2019), 1155–1166.

[19] Mengnan Liu, Shuiliang Fang, Huiyue Dong, and Cunzhi Xu. 2021. Review of digital twin about concepts, technologies, and industrial applications. *Journal of Manufacturing Systems* 58 (2021), 346–361.

[20] Ruodan Lu and Ioannis Brilakis. 2019. Digital twinning of existing reinforced concrete bridges from labelled point clusters. *Automation in Construction* 105 (2019), 102837.

[21] Yuqian Lu, Chao Liu, I Kevin, Kai Wang, Huiyue Huang, and Xun Xu. 2020. Digital Twin-driven smart manufacturing: Connotation, reference model, applications and research issues. *Robotics and Computer-Integrated Manufacturing* 61 (2020), 101837.

[22] Eric Lutters. 2018. Pilot production environments driven by digital twins. *South African journal of industrial engineering* 29, 3 (2018), 40–53.

[23] Ryan Magargle, Lee Johnson, Padmesh Mandloi, Peyman Davoudabadi, Omkar Kesarkar, Sivasubramani Krishnaswamy, John Batteh, and Anand Pitchaikani. 2017. A simulation-based digital twin for model-driven health monitoring and predictive maintenance of an automotive braking system. In *Proceedings of the 12th International Modelica Conference, Prague, Czech Republic, May 15-17, 2017*. Linköping University Electronic Press, 35–46.

[24] Genevieve Martin, Christophe Marty, Robin Bornoff, Andras Poppe, Grigory Onushkin, Marta Rencz, and Joan Yu. 2019. Luminaire digital design flow with multi-domain digital twins of LEDs. *Energies* 12, 12 (2019), 2389.

[25] Microsoft. 2021. *Microsoft Azure Digital Twin.* Retrieved May 15, 2021 from https://docs.microsoft.com/en-gb/azure/digital-twins

[26] Daniel Olivotti, Sonja Dreyer, Benedikt Lebek, and Michael H Breitner. 2019. Creating the foundation for digital twins in the manufacturing industry: an integrated installed base management system. *Information Systems and e-Business Management* 17, 1 (2019), 89–116.

[27] Greyce Schroeder, Charles Steinmetz, Carlos Eduardo Pereira, Ivan Muller, Natanael Garcia, Danubia Espindola, and Ricardo Rodrigues. 2016. Visualising the digital twin using web services and augmented reality. In *2016 IEEE 14th international conference on industrial informatics (INDIN)*. IEEE, 522–527.

[28] Rafael M Soares, Maurício M Câmara, Thiago Feital, and José Carlos Pinto. 2019. Digital twin for monitoring of industrial multi-effect evaporation. *Processes* 7, 8 (2019), 537.

[29] Gernot Steindl, Martin Stagl, Lukas Kasper, Wolfgang Kastner, and Rene Hofmann. 2020. Generic digital twin architecture for industrial energy systems. *Applied Sciences* 10, 24 (2020), 8903.

[30] Behrang Ashtari Talkhestani, Tobias Jung, Benjamin Lindemann, Nada Sahlab, Nasser Jazdi, Wolfgang Schloegl, and Michael Weyrich. 2019. An architecture of an intelligent digital twin in a cyber-physical production system. *at-Automatisierungstechnik* 67, 9 (2019), 762–782.

[31] Fei Tao, Qinglin Qi, Lihui Wang, and AYC Nee. 2019. Digital twins and cyber–physical systems toward smart manufacturing and industry 4.0: Correlation and comparison. *Engineering* 5, 4 (2019), 653–661.

[32] Fei Tao, Meng Zhang, and A.Y.C. Nee. 2019. Chapter 3 - Five-Dimension Digital Twin Modeling and Its Key Technologies. In *Digital Twin Driven Smart Manufacturing*, Fei Tao, Meng Zhang, and A.Y.C. Nee (Eds.). Academic Press, 63–81. https://doi.org/10.1016/B978-0-12-817630-6.00003-5

[33] AMM Sharif Ullah. 2019. Modeling and simulation of complex manufacturing phenomena using sensor signals from the perspective of Industry 4.0. *Advanced Engineering Informatics* 39 (2019), 1–13.

[34] Jinjiang Wang, Lunkuan Ye, Robert X Gao, Chen Li, and Laibin Zhang. 2019. Digital Twin for rotating machinery fault diagnosis in smart manufacturing. *International Journal of Production Research* 57, 12 (2019), 3920–3934.

[35] Feng Xiang, Zhi Zhang, Ying Zuo, and Fei Tao. 2019. Digital twin driven green material optimal-selection towards sustainable manufacturing. *Procedia Cirp* 81 (2019), 1290–1294.

[36] Jiacheng Xie, Xuewen Wang, Zhaojian Yang, and Shangqing Hao. 2019. Virtual monitoring method for hydraulic supports based on digital twin theory. *Mining Technology* 128, 2 (2019), 77–87.

[37] Yan Xu, Yanming Sun, Xiaolong Liu, and Yonghua Zheng. 2019. A digital-twin-assisted fault diagnosis using deep transfer learning. *IEEE Access* 7 (2019), 19990–19999.

[38] Rongli Zhao, Douxi Yan, Qiang Liu, Jiewu Leng, Jiafu Wan, Xin Chen, and Xiafeng Zhang. 2019. Digital twin-driven cyber-physical system for autonomously controlling of micro punching system. *IEEE Access* 7 (2019), 9459–9469.

[39] Zexuan Zhu, Chao Liu, and Xun Xu. 2019. Visualisation of the digital twin data in manufacturing by using augmented reality. *Procedia Cirp* 81 (2019), 898–903.