C. Alcaraz, H. Guzman and J. Lopez, "Adaptive Digital Twin: Protection, deception, and testing", *Future Generation Computer Systems*, vol. 179, 2026.
http://doi.org/10.1016/j.future.2025.108357
NICS Lab. Publications: https://www.nics.uma.es/publications

Adaptive Digital Twin: Protection, Deception, and Testing

Cristina Alcaraz[*], Hector Guzman[†], and Javier Lopez[‡]

Computer Science Department, University of Malaga,
Campus de Teatinos s/n, 29071, Malaga, Spain

Abstract

A Digital Twin (DT) is a cutting-edge technology that has gained relevance in recent years, demonstrating huge potential for the simulation of processes and the provision of valuable insights to improve and optimise systems. Leveraging a high degree of fidelity in replicating real-world processes, DTs are being explored for advanced applications such as deception and proactive protection of critical infrastructures. However, this same advantage also raises concerns with respect to a system's exposure, as the detailed digital representation may introduce new cybersecurity risks. With the aim of assisting the growth of this technology, this paper presents an adaptive DT solution that facilitates the configuration of particular components of the digital system, tailoring different application scenarios specifically for protection, deception, and testing purposes. Finally, the proposed architecture is tested under a specific IoT-oriented use case to validate, experiment, and extract conclusions of the proposed solution.

Keywords: Digital Twin, Critical Infrastructures, Internet of Things, Protection, Deception, Testing

1 Introduction

A Digital Twin (DT) is a dynamic, digital counterpart of a physical system that is continuously synchronised with real-time data throughout the system lifecycle, as presented in El Saddik et al. (2021) and Eramo et al. (2022). In contrast to static models, DTs establish a bi-directional connection with their physical counterparts, enabling not only the reflection of system states but also the influence of real-world operations through valuable insights and feedback. They serve a wide range of functions including design, testing, simulation, diagnosis, and operational refinement. This

[*]alcaraz@uma.es
[†]hectorguzman422@uma.es
[‡]javierlopez@uma.es

real-time mirroring allows organisations to observe, manage, and optimise complex systems effectively. One of the key advantages of DTs is their ability to safely simulate hypothetical scenarios, which supports performance enhancement, risk mitigation, and strategic decision-making without disrupting current operations.

According to Barricelli et al. (2019), the origins of the DT concept can be traced back to the 1970s, when the National Aeronautics and Space Administration (NASA) developed methods to monitor and diagnose physical components during aerospace missions. However, the modern understanding of DT was introduced by Michael Grieves within the context of Product Lifecycle Management (PLM), where he highlighted the need for real-time interaction between digital and physical assets to optimize system's performance across their lifecycle. As pointed out in Alcaraz and Lopez (2022), a DT is generally conceived as a group of physical and/or virtual machines or computer-based models that simulate, emulate, or replicate the behaviour of a physical entity. It is also described as a system that couples physical entities with their digital counterparts, enabling organisations to enhance performance, efficiency, and decision-making through this interaction.

To conceptualise DT, Grieves proposed a framework composed of three interconnected spaces that work together to replicate and manage a physical system. Figure 1 illustrates a representation of this model, adapted to reflect its core structure and main functions.

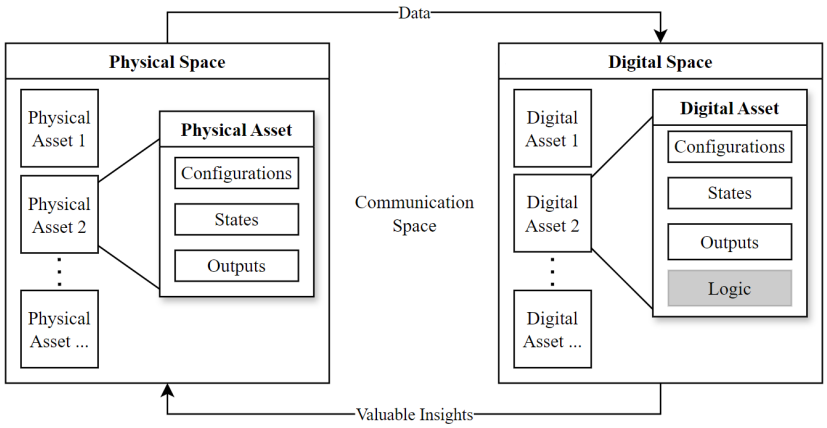

Figure 1: Digital Twin Spaces

- *Physical space*: This includes real-world components and operational technologies, such as sensors, actuators, and controllers (e.g. Programmable Logic Controllers (PLCs)). These elements are responsible for continuously collecting operational data and executing control commands to regulate system behaviour.

- *Digital space*: A virtual representation of physical assets, built using digital models that simulate states, conditions, and configurations of the system. This space allows for the collection and analysis of operational data to better understand behaviour, predict potential failures, and optimize the system performance.

- *Communication space*: This refers to the interface that enables the bidirectional data flow between the physical and digital environments. It allows the DT to receive real-time data from the physical system, process the data to derive insights, and send back adjustments to enhance system efficiency and security.

While the key advantage of the DT over conventional simulation lies in its bidirectional flow of information, where the digital model not only mirrors the physical system but also influences it by providing actionable insights, this same feature also introduces concerns regarding a system's exposure. The DT can be exploited by adversaries to obtain valuable information about system structure and operation, as its faithful replication means that the entire infrastructure is digitally represented. In addition, since the DT can directly impact the physical system, a compromised digital counterpart may result in decisions that negatively affect the real system, rather than protecting it. Recent research has already raised this cybersecurity issue, such as Mun et al. (2025) and Alcaraz and Lopez (2022).

To maintain the competitive advantages of DT while preventing exposure in certain scenarios, ADTwin is proposed as an Adaptive Digital Twin solution that combines DT qualities with supervised, protective, and deceptive capabilities for *Protection, Deception, and Testing* applications (hereinafter denoted as PDT). In this regard, the "adaptive" feature refers to the capability of the system to dynamically regulate the visibility and behaviour of its elements, selectively exposing real system information depending on whether the purpose is protection, deception, or testing (PDT). This enables ADTwin to switch between modes and tailor its responses according to the application scenario. Delving deeper into PDT applications, a protection application is assumed as the perfect implementation of a DT aiming to study system behaviour and provide the best insights to feed back into the system. In contraposition, a deception application refers to scenarios where a DT can be applied as a High-Interaction (HI) decoy that gathers adversarial knowledge for further improvement of system security. Last but not least, a testing application is suggested for *what-if* scenarios, which may compromise the integrity of the system, but whose knowledge is valuable for risk management as also indicated by the European Cyber Security Organisation in ECSO WG6 (2023). Each of these applications requires specific configurations that ADTwin envisions in order to dynamically manage in a next-level DT definition.

Therefore, the **main contribution of this article** is to provide cybersecurity capabilities driven by adaptive DTs in the three applications mentioned above, PDT, with a special focus on their usefulness for critical digitised scenarios. To this end, this article is structured as follows. Section 2 presents the current related work in the field of DT that supports this approach. Section 3 enumerates the necessary requirements for an adaptive PDT solution. Section 4 describes the proposed architecture and how its behaviour is adapted for each particular application. The logic behind the adaptive architecture is showcased in Section 5, and a specific Internet of Things (IoT) Use Case (UC) is addressed in Section 6. Section 7 focuses on experimentation with the proposed approach applied to the specific UC described in Section 7, under certain circumstances; whereas Section 8 presents the validation of the proposed system across several evaluation dimensions. Finally, Section 9 outlines the conclusions and future work.

2 Related work

At present, the use of DTs is widespread not only to demonstrate testing capabilities and represent specific anomalous scenarios, but also to demonstrate protection capabilities. In Homaei et al. (2024) and Alcaraz and Lopez (2022) the authors already provided an overview of the current state of the art to determine how the simulation paradigm can benefit risk management, monitoring, prediction, detection, response, and deception operations. However, given that this article focuses specifically on DT-guided protection, deception, and testing, the review of related work is limited to these particular topics. As pointed out by Iqbal et al. (2024) and Suhail et al. (2024), DTs are well-equipped systems that, unlike traditional High-Interaction Honeypot (HIH), have the ability to represent the observed physical counterpart with a high degree of fidelity and make autonomous decisions about it. By simulating real-world assets, DTs can provide a more immersive and engaging navigation environment, enabling dynamic adaptation of defensive strategies to intensify Cyber Threat Intelligence (CTI) and refine response tactics against advanced and potential attacks.

All these attractive deception features have encouraged the proposal of several approaches. Yigit et al. (2023) detail TwinPot, an advanced honeypot system based on DT technology with application in seaports. Its main objective is to observe adversarial behaviour and provide feedback to cybersecurity mechanisms by automatically classifying different types of attacks. To do so, TwinPot (i) controls adversarial navigation by diverting the threat away from the real system and protecting the real infrastructure, and (ii) collects information to identify behaviour patterns and vulnerabilities. In addition, Liatifis et al. (2024) propose SiHoneypot, an HIH that integrates DT technology to simulate the states of autonomous vehicle sensors, such as LiDAR (Light Detection and Ranging). The integrated DT acts as an intermediary component, managing communication between the LiDAR honeypot and the control panel. The honeypot data is transmitted to a centralised server, where a hash value is calculated and sent to the DT for verification. The DT compares this hash with pre-existing values to detect anomalies or potential cyber threats. In the event of a discrepancy, an alert is triggered, causing the honeypot to strengthen its monitoring and track suspicious activity. Similarly, Durão et al. (2018) introduce DECEPTWIN for application with the Internet of Vehicles (IoV) given the increased risk of security threats that interconnected vehicles bring and the limited interaction capabilities of traditional honeypot-based approaches. This approach also aims to combine the technology with blockchain to address integrity and traceability within this decentralised infrastructure.

Nintsiou et al. (2023) also propose a DT-backed deception approach called DiTwinI-Hon. This DT is capable of evolving deception strategies from previous adversarial activity. The idea is to leverage information gathered from attacks to polish and improve the honeypot's response mechanisms. In contrast, INCEPTION is proposed by Suhail et al. (2024). The approach includes a DT as a central element of a deception platform, which goes beyond what has been seen so far, where the DT is the component that deceives malicious actors and diverts them from the real environment. Thus, INCEPTION does not only seek to leverage the advantages of DT realism, but also to dynamically embed deception strategies into such realism. There is a clear trade-off between the level of realism and the risks of exposure, which is introduced as "fidelity". According

to Schleich et al. (2017), fidelity refers to the ability to conform to the physical model, while Durão et al. (2018) emphasise this condition of simulation as a prerequisite for providing mirror guarantees. Also Alcaraz and Lopez (2022) attribute fidelity to the DT's ability to show a reality equivalent to its counterpart, since deviations could lead to invalid conclusions and knowledge. Thus, the greater the fidelity of the simulation, the more realistic the scenario will be, increasing the risks of exposure and the complexity of the virtual counterpart as well.

Table 1: Comparison of DT-based deception approaches

Reference	**Technology**	**Fidelity**	**Purpose**	**Scope**	**Action**
Yigit et al. (2023)	TwinPot	High	Research	Multiple	Dynamic
Liatifis et al. (2024)	SiHoneypot	High	Both	Single	Static
Nintsiou et al. (2023)	DiTwinIHon	High	Research	Single	Dynamic
Suhail et al. (2024)	INCEPTION	Adaptive	Production	Multiple	Dynamic
Iqbal et al. (2024)	DECEPTWIN	High	Production	Multiple	Dynamic
Proposed approach	ADTwin	Adaptive	Both (PDT)	Multiple	Adaptive

Table 1 summarises the characteristics of each approach, highlighting the leverages of ADTwin, whose advantages stem from its adaptability across different dimensions. The classification presents key attributes of the solutions. The degree of fidelity is generally high, as DT-based honeypots stand out for closely mimicking real assets. However, some approaches introduce an adaptive fidelity, meaning that the level of detail or realism can be dynamically adjusted (i.e. high when realism is required, or reduced when it is necessary to protect sensitive information from exposure or deceive attackers). The purpose of deception may vary, ranging from research strategies that primarily gather insights about the attacks and adversaries behaviour, to production systems that actively engage attackers to divert them away from critical components. With respect to the scope of protection, it can focus on a specific type of attack or include multiple decoy tools to detect a wider spectrum of attacks. Ultimately, the level of action refers to how the DT can influence the real counterpart. While static models have a fixed configuration to always act the same way, dynamic models provide the real counterpart with feedback to change its behaviour. Adaptive approaches can decide when this feedback is necessary or when it could negatively affect to the real assets.

The comparison showcases the leverages of the proposed approach. In terms of fidelity, it presents a high degree of adaptation, to control the exposure and complexity within the digital part (later discussed in Section 4), and proactive actions in the real counterpart, in order to isolate critical assets from the real world and permit complex tests, failures of which may cause huge losses. Those attributes contribute to the PDT applications, allowing specific configurations for each scenario.

3 Adaptive requirements for PDT

To specify the ADTwin architecture, a structured approach based on a set of requirements that characterise its potential features is defined in this section. Specifically, the considered methodology follows three main sequential actions:

- Action 1: Identification of requirements that address protection, deception, and testing (or simply PDT) from a general perspective;
- Action 2: Association of these requirements with the three main ADTwin PDT applications; and
- Action 3: Mapping of each requirement to the corresponding DT capability levels, which are also presented in detail later in this section.

Action 1 comprises five primary requirements assuming an "adaptive" nature so as to allow the system to tailor to each PDT application context - emphasising the leverages identified in Table 1. They are as follows:

Table 2: Requirements contribution to capability levels compliance for each of the application scenarios

	Protection Requirements					**Deception Requirements**					**Testing Requirements**				
	Sync.	**Comm.**	**Monit.**	**Decis.**	**Imit.**	**Sync.**	**Comm.**	**Monit.**	**Decis.**	**Imit.**	**Sync.**	**Comm.**	**Monit.**	**Decis.**	**Imit.**
ITU CLs	**SR-P**	**CR-P**	**MR-P**	**DR-P**	**IR-P**	**SR-D**	**CR-D**	**MR-D**	**DR-D**	**IR-D**	**SR-T**	**CR-T**	**MR-T**	**DR-T**	**IR-T**
CL1-Representation	F	-	-	-	F	P	-	-	-	F	P	-	-	-	F
CL2-Communication	-	F	-	-	-	-	P	-	-	-	-	P	-	-	-
CL3-Analysis	-	-	F	-	-	-	-	F	-	-	-	-	F	-	-
CL4-Optimization	-	F	-	F	-	-	P	-	F	-	-	P	-	P	-
CL5-Symbiosis	F	F	F	F	F	P	P	F	P	F	P	P	P	X	F

Legend	
F	Fully contributes
P	Partially contributes
X	Does not permit
-	Not Applicable

- SR - *Synchronisation*: Refers to how the states of real-world assets are synchronised with the digital assets, directly connecting with the concept of system fidelity. While full-synchronisation perfectly mirrors the entire structure of the real counterpart, maintaining a consistent DT model, partial synchronisation allows for increased system discrepancy by simulating selected components. This approach can even enable complete simulation of the real system when needed.
- CR - *Communication*: Since the communication space can permit on-way or dual transmission of information between parts, CR includes from one-way communication, where the digital part shadows the real one, to dual dialogue in which digital insights feed back into the system. This bidirectionality concept is explored and referred to as the *level of integration* in Kritzinger et al. (2018), which

is also related to the level of action mentioned above. This also means that one-way communication constrains the DT to a static role, while dual dialogue enables dynamic models whose feedback can actively influence the real counterpart.

- MR - *Monitoring*: The monitoring process concerns how the functioning of the system is observed, analysed, and transformed into valuable insights.
- DR - *Decision-Making*: The decision-making requirement facilitates autonomous behaviour based on previous knowledge. In the framework of critical infrastructures, this process can be supervised to minimise potential risks.
- IR - *Imitation*: Related to the degree of fidelity in terms of responses, protocols, and structure between the real part and its digital equivalent. While a flawless imitation is generally preferred, it also brings complexity and cannot always be fully achieved.

As part of Action 2, PDT requirements, which are labelled as "-[P/D/T]", are correlated to those identified in Action 1 and are summarised in Table 3. More specifically, the protection (P) application demands full-synchronisation (SR-P) in order to achieve a high level of fidelity; this way, insights with direct connection to the real assets can be extracted for further system strengthening. This transfer of information requires a dual communication space (CR-P) and full monitoring (MR-P) of the system. Additionally, autonomous decision-making (DR-P) settles on the actions that can be automatically triggered to protect the infrastructure.

Table 3: Requirements attribution for specific application purpose

	Synchronisation	*Communication*	*Monitoring*	*Decision-Making*	*Imitation*
Scenario	**SR**	**CR**	**MR**	**DR**	**IR**
Protection	Full	Dual	Full	Autonomous	Flawless
Deception	Partial	Constrained	Full	Semi-supervised	Flawless
Testing	Partial	Constrained	Dedicated	Supervised	Flawless

Contrarily, the deception (D) application takes advantage of partial-synchronisation (SR-D) to deceive adversaries by simulating part of the environment, providing a lower level of fidelity to preserve knowledge about the real functioning that attackers must ignore. This configuration constrains the communication (CR-D), as decoys must not affect the actions taken in the real counterpart or adversarial attacks over synchronised assets. System monitoring should be intensified (MR-D) to collect as much information as possible about the attack. In terms of autonomous decision-making, employing a semi-supervised model (DR-D) enables the selection of critical actions that must be restricted, thereby preventing compromising security even if attackers manage to bypass deception mechanisms.

Lastly, the testing (T) application requires partial-synchronisation (SR-T) to test the system under special conditions that would not normally be reached, but which impact understanding is crucial to further improve the response of the system. Thus, fidelity is reduced in order to alter the system representation, enabling the reproduction

of otherwise inaccessible scenarios for analysis and testing. Due to the consequences that tests may have if exposed to the real system, communication must be constrained (CR-T), and decision-making fully supervised (DR-T), facilitating the system to make recommendations and alerts while safeguarding the current state in the real world. For a better analysis of consequences, dedicated monitoring (MR-T) allows focusing on target assets. Given the specific requirements for each application, it is worth mentioning that all of them count on a flawless degree of imitation (IR-P, IR-D, IR-T), which means that the DT must respond exactly as the real counterpart would, replicating all system functions, protocols, and even errors.

To measure the expanding functionality and autonomy of a DT, the International Telecommunication Union (ITU) defines the standard ITU (2023), also mentioned in Alcaraz and Lopez (2025), which establishes five Capability Levels (CL). CL1 holds the representation level, where the DT captures physical network status and behaviour to mirror the system. The interaction level, CL2, comprises the bilateral communication that empowers the influence of the DT on the physical space. CL3 incorporates data analysis and inference to detect anomalies and carry out predictions. The optimization level is described in CL4 and encompasses Artificial Intelligence (AI), expert knowledge, and big data, permitting real-time decision-making, suggestions, and management. Lastly, CL5 envisions the perfect symbiosis between physical and digital spaces, achieving autonomous reconfiguration of the network.

Action 3 matches each of the previous ADTwin requirements with the CL they contribute to meet, as shown in Table 2. The matching is considered *full* for the ones that directly permit the specific CL; *partial* if the requirement equips the CL with limited functionality; *null* for the requirements that are not congruent with the capability; and *not applicable* if it does not specifically affect the capability.

As for protection, all the requirements contribute to fulfil the totality of the CLs, achieving the ideal notion of a DT according to the standard. The application is perceived as the perfect symbiosis between the two worlds with a strong relevance of the communication space, which enables real-time decision through the monitoring, analysis, and comprehension of the functioning of the system. Regarding the deception application, the CLs are adapted to provide the best benefits from the previous application, while maintaining a certain degree of discrepancy and security by means of simulation, decoys, and restricted communications. Even with this configuration, the application can partially comply with the five CLs, serving as a perfect approach to understand adversarial behaviour and protect targeted assets in response. Last but not least, the testing application employs configurations similar to deception. However, the experimental nature of the approach restricts any impact on the real assets to human eye decision, hindering the fifth CL, and therefore acting closer to a digital shadow approach. Nevertheless, this degree of adaption leverages the extraction of knowledge out of extreme or unusual casuistry, facilitating the understanding of how the system would react and the elaboration of solutions in advance.

The layered structure of the CLs is crucial, as compliance at each CL inherently limits subsequent levels, at most, to the same degree. ADTwin offers a novel approach by permitting partial implementation of these capabilities, thereby enabling the adaptation of subsequent levels to the specific scenario.

4 Adaptive DT-based architecture

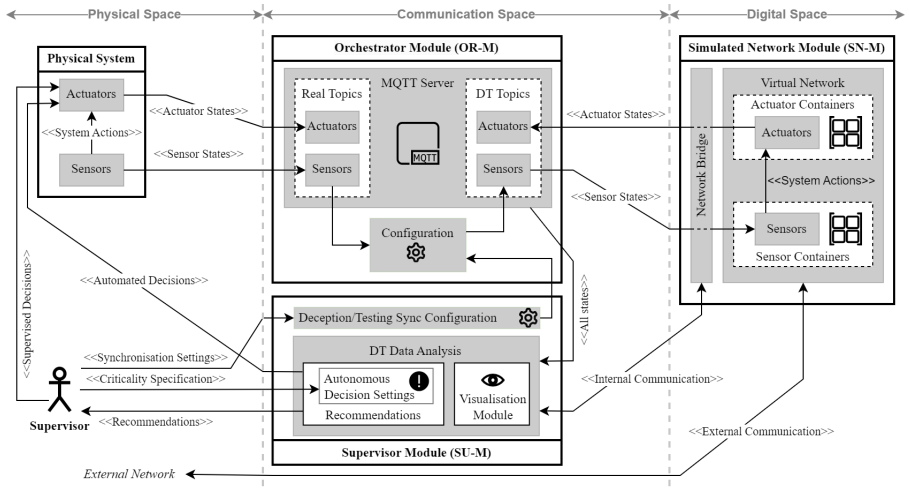

Figure 2: General architecture for ADTwin

The architecture designed for ADTwin in Figure 2 separates the responsibilities into different modules that cooperate to represent the real-world and offer adaption capabilities that further characterise the system, complying with the framework proposed by Grieves and conceptualised in Figure 1. Operating within the physical space, the physical system is considered the first component of the architecture, offering a genuine outlook of the real system behaviour and states that enable a crucial understanding of the environment. The selected physical infrastructure is composed of different sensors, human inputs and logic that generates deterministic outputs that determine the actions to carry out by the actuators. With the aim of eliminating central governance, all the devices work as stand-alone systems, representing an IoT network where devices receive and send messages with the configurations to be applied, which are structured in JavaScript Object Notation (JSON) for better human legibility, as shown in Listing 1 for a movement sensor.

```
{
    "name": "Movement Sensor",
    "status": "active",
    "information": { "press": "no" },
    "variables": {
        "start_time": 978927218.6141062,
        "hold": "no",
        "light_on": false
    }
}
```

Listing 1: Example of a sensor configuration

Moving into the communication space, the core of ADT, the *Orchestrator Module* (OR-M) is the most relevant module of the approach. It is responsible for adapting the

CL1 by dynamically adjusting the synchronisation level through a publisher-subscriber infrastructure that tracks the states of real-world sensors and delivers the states for the digital counterpart, either mirroring the real assets or simulating their values depending on the configuration. Going deeper into the logic of this module, a Message Queuing Telemetry Transport (MQTT) broker is set in the middle of the communication between real and digital worlds. The broker comprises identical pairs of topics for each input and output device in the infrastructure that hold the actual value of a real asset and its counterpart in the digital network. Regarding sensors, while real topics are directly published by the physical system, the supervisor configuration allows either linking real and digital sensor topics, genuinely mimicking the system state, or simulating digital assets providing a fixed value or a generation function. The corresponding value is subsequently delivered from OR-M to the digital counterpart for logic execution. Contrarily, actuator topics do not facilitate synchronisation configuration as they are always computed and published by the corresponding part, just serving as a comparison between the deterministic results from the execution of the logic in both parts, driven by the sensor states.

Moving to the next module in Figure 2, the *Simulated Network Module* (SN-M) holds the digital space in an assemblage of virtual containers that accurately mirrors the behaviour of the system, enhancing CL1 and allowing for the straightforward monitoring required by CL3. To identically imitate the IoT network, it is crucial to maintain the IP infrastructure unchanged. For that reason, this module deploys containers in an indistinguishable virtual network interface and assigns to devices the exact same IP directions that they use in the physical system. To deal with the problem of IP collisions, the module bridges virtual IPs to equivalent localhost IPs in the MQTT network, benefitting the two networks to be distinguishable for OR-M. The simulated network is composed of two main container groups, one that demands sensor states from the digital topics in OR-M, and another that simulates actuators and provides to OR-M the topics with the states for comparative purposes. Both container associations mimic the logic of the real system to provide the exact input-output workflow, also implementing the same communication protocols that the physical system uses.

In last instance, returning to the communication space, the *Supervisor Module* (SU-M) holds the supervisor configurations and decision-making in compliance with CL4. The module employs data analysis techniques to provide suggestions and automatic actions. The difference between these two outputs is that while automatic actions are directly applied to the physical system, suggestions require supervisor approval as they may influence critical aspects of the environment. The supervisor also interacts with this module to adjust OR-M synchronisation settings and specify the set of autonomous decisions that ADTwin can make in respect to the real-system, complying with CL2 and CL5. In addition, a *Visualisation Module* (VI-M) is also provided inside SU-M, representing the ADTwin status in a user-friendly interface, as shown in Figures 3 and 4. For instance, the visualisation allows for the comparison of actuator states in both the physical and digital systems, highlighting discrepancies in the behaviour when systems are being provided with the same inputs, or displaying affected components when the input is synthetically altered for testing purposes.

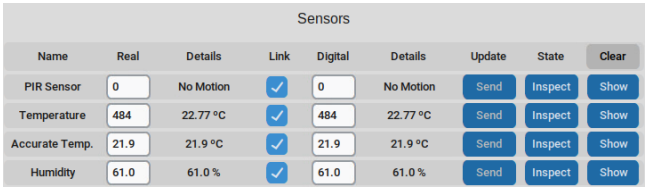

Figure 3: Sensors status view in the visualisation module

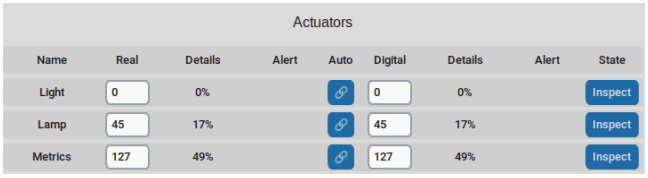

Figure 4: Actuators status view in the visualisation module

4.1 Twin view for protection

For the protection application, specific requirements are indispensable to offer a perfect symbiosis between the two counterparts, providing the so-called "twin view" of the system. In terms of architecture, OR-M is needed to directly transmit real asset MQTT topics to the digital topics according to SR-P, preventing the alteration of any of the inputs. SN-M subscribes to these topics to produce the actuator states in the individual devices, which must correspond to the real actuator conditions, as the logic is flawlessly imitated for IR-P compliance. The system is fully monitored (MR-P) and the selected communication technologies enable the tracking of sent configurations by delving into the JSON information. Therefore, network traffic, configurations and communication patterns between devices can be extracted and transferred to SU-M for further analysis.

SU-M is where autonomous decision-making under supervisor predefined rules (DR-P) is made, which settles on the actions that are triggered to protect the infrastructure. Additionally, to make these actions effective, the architecture enables the communication of automated decisions directly into the physical system (CR-P). For this approach, VI-M provides an overview of the assets states, pointing out discrepancies between actuators situation that may entail system vulnerabilities as illustrated in Figure 5.

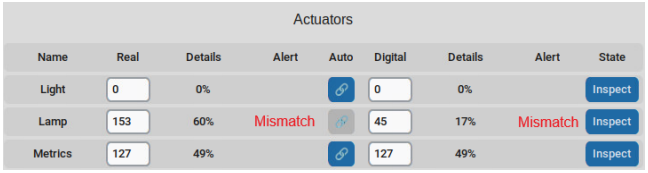

Figure 5: Actuator discrepancy alert in the visualisation module

4.2 Shadow view for deception and testing

In contrast to the view for protection, deception and testing applications adjust the functioning of the modules to adapt the whole ADTwin system to a "shadow view", with different purposes and infinite discrepancy degrees. In respect to the architecture, in order to offer a partial synchronisation, OR-M applies the configuration that decides which digital topics are linked to their real equal and which are fixed to a provided value or simulated with a generation function, complying with (SR-D, SR-T). Those offline values are directly published by OR-M to the MQTT server, which acts as a low-level physical interface for the input reads, making them indistinguishable from real values at the top-level logic.

Although synthetic states may cause outputs in SN-M that are different from the ones in the real-world network, the system logic imitation is still flawless (IR-D, IR-T) and will be consistent inside the virtual network. Regarding monitoring, while this module can observe the full system for deception application the same way it does for protection (MR-D), it can also focus on specific targets for testing purposes (MR-T). As for analysis and decision-making, SU-M collects all the monitoring insights and enables both automatic decisions, that are directly sent to the physical system, and suggestions, which the supervisor must review before implementing. This semi-supervised approach directly fits DR-D, while it is restricted to only suggestions for testing applications to preserve the physical system from the impact of experimental scenarios (DR-T). As the supervisor can control which decision are applied to the physical system, the communication space is constrained according to CR-D and CR-T.

5 Adaptive logic for ADTwin

The leverage of ADTwin lies in the adaption that each component of the architecture implements. The logic behind this capability, far from being complex, is focalised, effective, and comprehensible for humans, as it comprises an expert eye for the protection of critical infrastructures. This section aims to provide a formal perspective of the main implementation elements that form the adaptive logic of ADTwin for the aforementioned PDT applications.

5.1 Synchronisation adaption

Synchronisation adaption mainly occurs in OR-M. Nonetheless, it depends on all the modules, the most important being the adaption instrument in the approach. Synchronisation settings begin in SU-M, where the supervisor adjusts the discrepancy level of the DT according to the application by defining the sensor configurations. The example in Algorithm 1 shows the definition of a temperature sensor including information such as its IPs in real and digital counterparts, the MQTT topics that hold the states of this asset, the link option to synchronise or not the sensor, and the function that generates the virtual values when synchronisation is deactivated.

Once the configuration is updated, OR-M applies the rules and manages the MQTT server workflow, allowing direct connections to synchronised assets, but executing

Algorithm 1 Synchronisation settings for the temperature sensor

Require: real_ip is the IP address in the actual system
Require: virtual_ip is the IP address in the digital twin
Require: real_topic is the MQTT topic for the real sensor
Require: virtual_topic is the MQTT topic for the digital sensor
Require: link_to_real indicates whether to link to the real sensor value (TRUE/FALSE)
Require: value_generator is a function to generate values

```
temperature_sensor ← Sensor(real_ip, virtual_ip, real_topic, virtual_topic, link_to_real, value_generator)
sensors["temperature_sensor"] ← temperature_sensor
```

generation functions for the others as exposed in Algorithm 2. This configuration only applies for input assets like sensors, as outputs are the result of system logic with the given entries and must be computed in an equivalent digital stream to maintain a consistent state of the system.

Algorithm 2 MQTT orchestration

Require: sensors is the dictionary containing all sensors
Require: sensor_key is the key of a specific sensor

```
sensor ← sensors[sensor_key]
if sensor.link == TRUE then
    publish(sensor.topic_digital, get(sensor.topic_real))
else
    simulated_value ← sensor.function()
    publish(sensor.topic_digital, simulated_value)
end if
```

In last instance, the digital input topics published in the MQTT server are employed by SN-M to feed the request of sensor containers. Each container is responsible of executing the logic of one device, which requires interacting with the MQTT interface, which acts as the counterpart of the hardware in the virtual network to provide the input states. This interface subscribes to all digital MQTT topics and stores the last published value for each input asset. Thus, the newest state of the asset is available whenever the logic needs it. Regarding the containers acting as actuators, they also require the MQTT interface in order to register the variations of actuator states in their respective digital output topics. Algorithms 3, 4, and 5 illustrate, respectively, the initialization of the MQTT interface, the reading of sensor values, and the writing of actuator values.

The proposed implementation aims for a transparent execution of the inside-network logic, not being affected by the origin of the input, which is only managed by OR-M. Thus, the synchronisation level can be effectively adapted to comply with PDT requirements.

Algorithm 3 MQTT interface initialisation

Require: mqtt is an instance of the MQTT client
Require: values is a dictionary to store sensor values
Require: store_sensor is a function defined to (i) receive a topic, (ii) identify the corresponding sensor, and (iii) store the value in the sensor position in the dictionary

```
mqtt ← MQTTClient()
mqtt.subscribe("dt/sensors/")
values ← {}
mqtt.on_message = store_sensor
```

Algorithm 4 Read sensor value

Require: values is a dictionary to store sensor values
Require: sensor is the name of the sensor to read

```
if values.get(sensor) exists then
    return values.get(sensor)
else
    return None
end if
```

Algorithm 5 Write actuator value

Require: mqtt is an instance of the MQTT client
Require: actuator is the actuator object with attribute mqtt_dt
Require: value is the value to write to the actuator

```
topic ← "dt/actuators/" + actuator.mqtt_dt
mqtt.publish(topic, value)
```

5.2 Monitoring adaption

Monitoring adaption does not directly impact the behaviour of the approach, but rather the quality of the extracted knowledge as it permits either observing the whole network or focusing on targeted elements. The collected information goes from network traffic to configuration elements inside the traces and derivation of communication patterns between devices, providing an understanding of which element of the system affect others and how they respond.

The logic is fundamentally executed inside SN-M, where the virtual network can be sniffed and monitored. However, part of the monitoring actions resides in SU-M in order to be analysed and processed for further decision-making and suggestions.

5.3 Decision-making adaption

Once monitoring insights are analysed in SU-M, autonomous actions and suggestions are triggered based on this knowledge. For these actions to be carried out, the supervisor must specify the specific mitigation rules.

The adaptive attribute of this logic is present in how the system launches the actions, i.e. automatically or under user supervision. The supervisor must specify which decisions over the actuators are safe to be instantly made and which require the expert to authorise the suggestion. Despite the configuration, if an action is approved, it will automatically apply for the mitigation of anomalies in the digital counterpart. Nevertheless, the effect of the actions in the real part depends strictly on the communication adaption. This approach ensures that mitigation solutions can be launched in the virtual network to observe the consequences without necessarily affecting the real assets, unless feedback communication is explicitly allowed.

```
{
    "name": "Light",
    "status": "active",
    "configuration": {
        "cycle_on": 1,
        "cycle_of": 1,
        "brightness": 100,  // Configure brightness to 100%
        "light_mode": "on"  // Toggle light mode to "on"
    }
}
```

Listing 2: Example of actuator configuration trace

The procedure to apply solutions makes use of the feature that devices present to receive configuration traces from other devices, which can be used by the supervisor to alter configurations. Listing 2 illustrates a configuration example in JSON format to adjust the parameters of an actuator.

5.4 Communication adaption

The adaption of communication refers to the control over the communication space between real and digital assets. This not only facilitates the synchronisation of counterparts by providing asset states, but also contemplates feedback mechanisms that can influence the real world based on insights derived from the digital equivalent. While real-to-digital

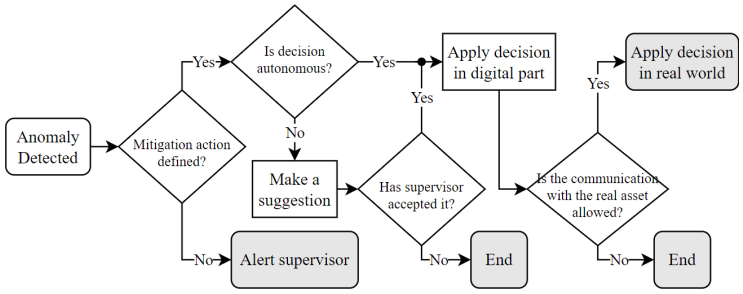

Figure 6: Anomaly mitigation flowchart

communication is set by default and just controlled by the adaptive synchronisation logic, the digital-to-real connection depends on the adaptive communication logic to administer the knowledge that can influence the real counterpart and when it can do so. This arrangement isolates critical assets for testing or deception applications, where the stage of the virtual system may differ from the real one's, and feedback actions may result inappropriate or harmful for the real system.

The aforementioned logic is primarily hosted in SU-M, where the expert can schedule the restricted communications according to the planned scenario and in concordance with other adaptive configurations. For instance, if an input asset is not synchronised and is provided with experimental values, the actions to mitigate its effects should be restricted, as this situation is not actually happening in the real world.

Figure 6 illustrates the execution flowchart of the system for a detected anomaly. It comprises adaptive configurations in SU-M such as autonomous or supervised decisions, and restricted or permissive communication with the real word assets.

6 Use Case: IoT-based Environmental Monitoring System

The IoT Environmental Monitoring System (IoT-EMS) UC addresses the deployment of an intelligent and resilient system. It is composed of autonomous IoT devices that continuously monitor and regulate environmental parameters, such as temperature, humidity, air quality, lighting, and access control. These devices operate together to ensure optimal environmental conditions while managing factors related to user comfort and safety. The system is designed to withstand adversarial scenarios in which attackers may attempt to alter environmental variables and compromise human welfare.

The architectural design of the system adopts a decentralised approach and utilizes communication protocols such as Hypertext Transfer Protocol (HTTP). This model supports modularity and scalability while enhancing robustness and fault tolerance. It also supports the execution of advanced experiments focused on real-time environmental control, security enhancement, and the evaluation of device resilience under adverse conditions. In particular, the UC supports protection strategies, adversarial deception, and testing methodologies aimed to improve the functioning of the IoT infrastructure.

For these applications, the ADTwin architecture presented in Figure 2 is adopted as a pattern, enabling the adaptive approach for this UC. Motivated by the critical nature of maintaining safe and stable environmental conditions, where deviations may directly affect human health and comfort, this UC highlights the importance of precision in monitoring variables such as air quality, toxic substance presence, and other ambient factors. Nonetheless, while IoT integration offers notable advantages in automation and continuous monitoring, it also introduces vulnerabilities related to cybersecurity, interoperability, and reliability, as discussed in Siwakoti et al. (2023).

The platform replicates an IoT network deployed in a controlled environment, where a range of sensors and actuators work together to manage the room's conditions. Environmental sensors gather data on key factors, whereas user input sensors collect human interactions such as biometric authentication and motion detection. Actuators then adjust environmental elements accordingly, including lighting, air circulation, and access mechanisms. All components interact through HTTP-based communication in a decentralized framework that promotes resilience and eliminates single points of failure as illustrated in Figure 7. The use of non-secured communication, such as HTTP, is intended to facilitate experimentation in this specific UC, allowing for improved monitoring.

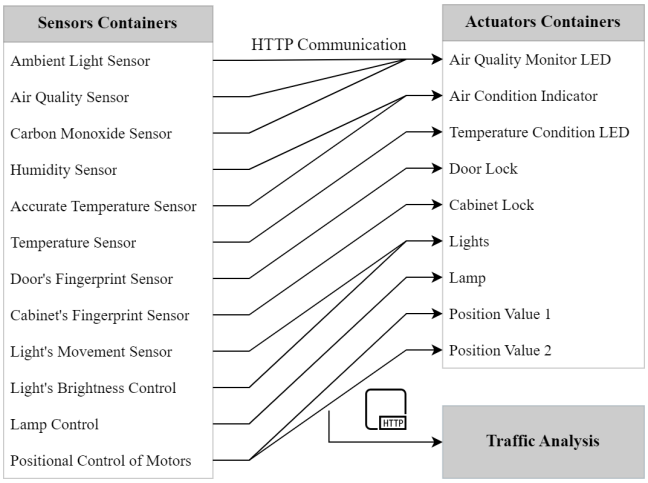

Figure 7: IoT-based environmental monitoring system

This UC facilitates the detection of threats targeting the IoT network by offering a controlled simulation of network attacks, thereby preserving the integrity of the real system. It also enhances the identification of anomalies in device behaviour by monitoring operations under real-world conditions, and supports user behaviour analysis to distinguish between legitimate and malicious interactions. Moreover, it favours laboratory testing under extreme conditions by manipulating sensor inputs in the digital environment to simulate non-usual, but critical scenarios, such as extreme temperatures or presence of toxic substances, ensuring that the system remains robust and secure.

6.1 IoT-EMS physical space

The physical space of the UC is structured around a distributed hardware infrastructure designed for autonomous environmental monitoring and control. It includes an Intel Galileo v1 board, which manages communication processes and sensor logic, and an Arduino UNO, devoted to the control of some specific actuators. Sensors, such as capacitive sensors, buttons, potentiometers, Light-Dependent Resistors (LDRs), temperature and humidity sensors, air quality sensors, and gas detectors, are deployed alongside various actuators, including LEDs, visual indicators, and servomotors. Each of these components is mapped to a specific IoT device within the network, embedding its own logic to operate autonomously and to manage its interactions with other devices through HTTP communication, which have been chosen for simplicity and better understanding of the inside-network behaviour. Table 4 enumerates the specific electronic components that have been used for the implementation of the UC and the IoT device used for.

Table 4: Sensor and actuator components used in the IoT-EMS

Component	**Specifications**	**IoT Device**
Touch sensor	TTP223-BA6	Fingerprint
Button	–	Positional
PIR Sensor	PIR Motion Sensor (120º, 0.1∼6m)	Motion
Potentiometer	(300º, 10kΩ)	Brightness
LDR	–	Light
Temp/Humidity	DHT11 (H: 20∼90%, T: 0∼50ºC)	Air condition
Temperature	TTC3A103*39H (−40∼125ºC)	Thermostat
Air quality	Air Quality Sensor v1.3	Air quality
Gas sensor	MQ9 (CO: 200∼1000ppm; CH_4, LPG: 200∼10000ppm)	Gas
Servomotor	EMAX 9g ES08A (180º)	Adjustment
LED	–	Lights, alarms, indicators

The Intel Galileo incorporates a hardware interface implementing the “mraa” library that facilitates real-time communication between the system and the physical environment, which also continuously extract and publish device states to the MQTT server that serves as the core of the communication space. Every device in the network is assigned a unique IP address, allowing for individual identification, data exchange, and reconfiguration through a user-accessible REST-API.

6.2 IoT-EMS virtual space

The virtual space is assembled within a Docker virtual network that assigns one container to the representation of each device of the system. The containers include the exact logic defined in the real devices to provide a deterministic output, coherent and consistent with the real behaviour, and are identified by the same IP address. To overcome IP collisions in the communication space, the Docker definition bridges each device to an equivalent local IP address for the communication from outside of the virtual space as exemplified in Listing 3.

```
servo1 :
    container name: servo1
    restart: on-failure
    build:
```

```
    context: ./servo1
cap_add:
    - NET_ADMIN
    - SYS_NICE
environment:
    - DISPLAY=$DISPLAY
    - XAUTHORITY=/root/.Xauthority
    - TYPE
networks:
    mac-lan:
        ipv4_address: 192.168.0.122
    h-link:
        ipv4_address: 10.0.0.122
tty: true
```

Listing 3: Example of container definition

In contrast to the real system, the virtual network does not have access to a real hardware interface. To solve this, a virtual interface interprets MQTT asset values and provides an equivalent function set that mimics "mraa" responses to keep device logic intact.

6.3 IoT-EMS communication space

The communication space in the IoT-EMS UC resides in the server side, and its principal component is precisely the MQTT server. It is launched with an eclipse-mosquitto image and exposed to both the real system and the virtual network to provide OR-M functionality. The operation of the server, as explained in the approach, is to retrieve the states of real IoT devices to be used in the virtual synchronised assets. Moreover, it also collects the digital assets states for visualisation purposes.

The communication space is also responsible for receiving the monitoring information from SN-M to analyse and provide the real system with necessary reconfigurations according to the rules established in SU-M by the expert.

For the supervisor to have the most comprehensible outlook with respect to the status of the system, VI-M is integrated, providing the supervisor with a user-friendly interface to envisage what currently occurs in both, the real assets and digital equivalents. Additionally, the interface helps the user reconfigure the synchronisation settings, deciding on the status of digital components and receiving alerts on inconsistent states of the system. As for the testing application, it is worth to mention that VI-M provides a valuable sight of the simulation, establishing a deeper understanding of the changes in the system for the experiments carried out.

7 Experiments and results in IoT-EMS

This section presents the experimental validation of the proposed ADTwin approach within the specific IoT-EMS scenario. It aims to demonstrate the feasibility and effectiveness of the developed methods in a real-world context. The following subsections detail the operation of the synchronisation mechanisms between physical and virtual devices, as well as the functioning of asynchronous modes designed for deception and testing purposes. In addition, the system capability to detect and respond to attacks is evaluated, highlighting its potential for enhancing protection in IoT environments.

7.1 Experiment 1: synchronous functioning

The proposed experiment monitors the states of both real and digital counterparts of a button and a door that unlocks for some seconds when the button is pressed (for the button, 1 indicates pressed and 0 released; for the door, 1 means locked and 0 unlocked). Before the measurement, the system is preconfigured to full-synchronisation mode (see Figure 8), which means that real and digital system are provided with identical inputs, and expected, consequently, to generate identical outputs. Figures 9 and 10 show a reading of the button and door states, respectively, in the same time window.

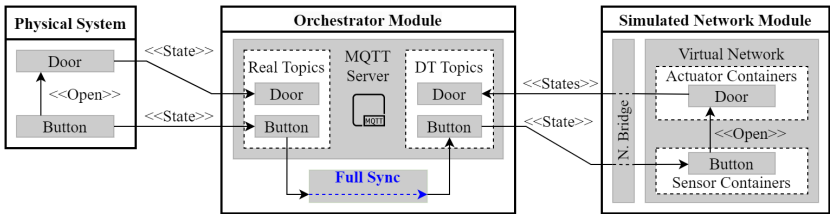

Figure 8: Structure for experiment 1 (full synchronisation)

The figures illustrate the simultaneous operation of both systems. It is important to note that the offset observed between the real and digital states in Figure 10 is due to measurements being taken on the digital counterpart, where communication from the real system introduces an unavoidable delay. Nevertheless, it can be observed that both systems respond appropriately to inputs. For example, around timestamp 16:32:07, the button registers a press, causing the door to unlock immediately for a few seconds.

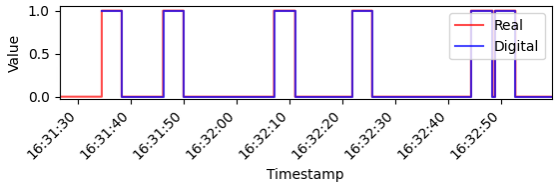

Figure 9: Comparison of states of the button with synchronisation

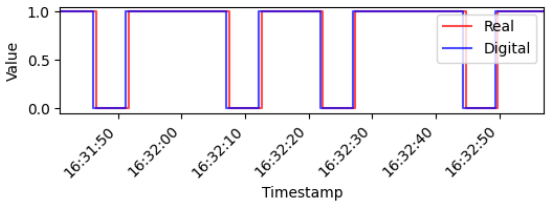

Figure 10: Comparison of states of the door with synchronisation

7.2 Experiment 2: asynchronous functioning

For this experiment, the systems are intentionally not synchronised (see Figure 11), allowing the exploration of the testing and deception approaches of PDT. The devices targeted in this scenario are, once again, the button and the door. Figures 12 and 13 illustrate the states of the button and the door, respectively. In this asynchronous mode, each system operates independently, and inputs only affect their corresponding actuators. Notably, the figures show an initial interaction with the real button, followed by a separate interaction with the digital counterpart.

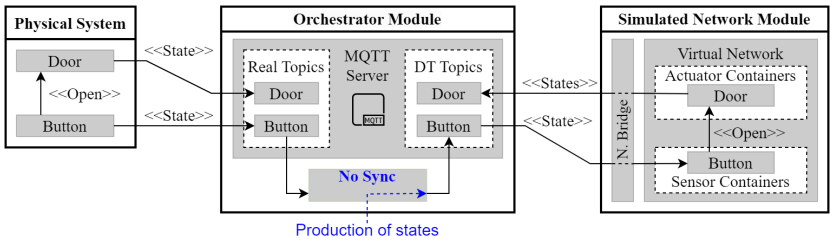

Figure 11: Structure for experiment 2 (no synchronisation)

Both interactions produce the same outputs, but at different timestamps. This separation is crucial for deception and testing purposes, as it ensures that the real system remains isolated from potentially harmful actions occurring in the digital environment.

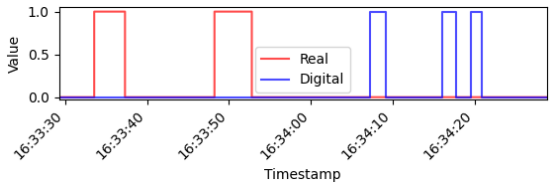

Figure 12: Comparison of states of the button with no synchronisation

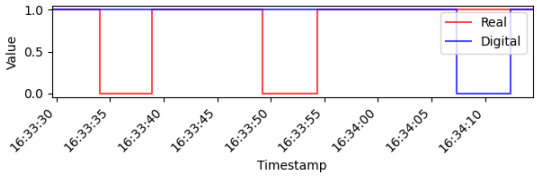

Figure 13: Comparison of states of the door with no synchronisation

7.3 Experiment 3: detection and protection against attacks

This last experiment highlights the potential of having two counterparts running the system's logic in parallel. By enabling real-time comparison between the real and digital

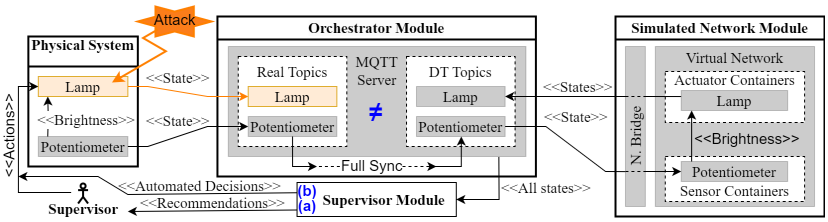

Figure 14: Structure for experiment 3 (synchronisation for protection)

systems, the DT can automatically alert or take proactive measures to ensure that the real system remains within its expected operational parameters (see Figure 14). The experiment presents the states of a potentiometer used to adjust the brightness of a lamp. During the test, attacks are launched against the lamp device in an attempt to alter its brightness value without any intervention from the potentiometer. The attack is carried out by sending an HTTP request to the device with an altered configuration, using information obtained from a genuine configuration captured in a previous network trace. The experiment allows for the evaluation of the system ability to detect and respond to unauthorised changes, ensuring that only legitimate inputs from the potentiometer can affect the lamp brightness.

Figures 15 and 16 present the reading of the system when automatic actions are not operating (flow (a) of Figure 14). The graphs highlight that both systems operate seamlessly for legitimate actions. However, a discrepancy appears when the real brightness value changes without a preceding adjustment of the potentiometer. This difference underscores the system's ability to detect unauthorised modifications. Since automatic actions are deactivated, any attack will persist until manual intervention occurs, as shown in Figure 16.

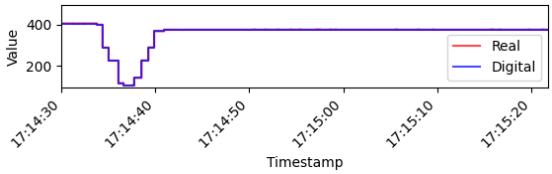

Figure 15: Comparison of states of the potentiometer under attack

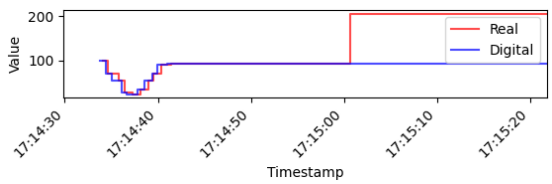

Figure 16: Comparison of states of the lamp under an attack

In contrast, when proactive actions are activated (flow (b) of Figure 14), the system is able to respond to the attacks and recover the state of the real system to the expected values in a short period of time. This is demonstrated in Figures 17 and 18, where two attempts of the same attack are detected and subsequently mitigated.

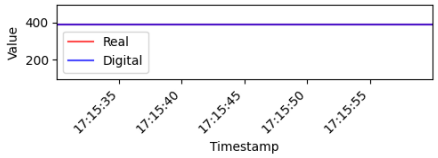

Figure 17: Comparison of states of the potentiometer with enabled protection

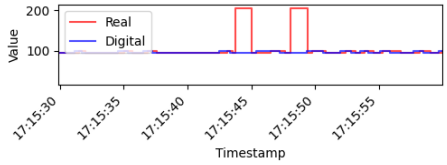

Figure 18: Comparison of states of the lamp with enabled protection

8 Validation

To evaluate the CLs defined by ITU (2023), the ITU standard also proposes six core Evaluation Dimensions (EDs). The first, *DT modelling* (ED1), focuses on structural representations of network topology and functional models that support behaviour simulation, decision-making, and control tailored to the specific requirements of the application. *Interactive mapping* (ED2) examines the synchronisation between the DT and its physical counterpart, highlighting the accuracy of data exchange and control operations. The *data service* (ED3) dimension assesses the DT ability to manage data throughout its lifecycle, from acquisition to processing, analysis, and knowledge extraction. *Intelligence* (ED4) considers the integration of AI and Machine Learning (ML), outlining the progression from localized intelligence within individual components to system-wide, adaptive, and self-optimizing functionalities. *User experience* (ED5) addresses the accessibility, clarity, and usability of the DT interface, ensuring intuitive interaction, meaningful visualisation, and effective human supervision. Finally, *trustworthiness* (ED6) evaluates the system ability to maintain privacy, resilience, and operational reliability. These six dimensions collectively provide a structured basis for assessing the development, functionality, and reliability of DT networks across various deployment contexts.

Measuring up the CLs that ADTwin offers with the mentioned EDs provides an outlook of the approach's effectiveness. ED1, which refers to the structural and functional representation, is fulfilled in SN-M through the accurate and consistent imitation of the network infrastructure, topology, and device logic. This replication remains

Table 5: Contribution of ADTwin modules to EDs

	ADTwin Modules			
ITU Evaluation Dimensions	**OR-M**	**SN-M**	**SU-M**	**VI-M**
ED1-DT modelling	-	✓	-	-
ED2-Interactive mapping	✓	-	-	-
ED3-Data service	-	✓	✓	-
ED4-Intelligence	-	-	✓	-
ED5-User experience	-	-	-	✓
ED6-Trustworthiness	✓	-	✓	-

Legend	
✓	Contributes
-	Not Applicable

faithful to the real system architecture, allowing the DT to mirror the system's behaviour without alterations. ED2, focused on interactive mapping, is supported through the integration of the communication space managed by OR-M. This component facilitates continuous synchronization between the physical and digital counterparts, alongside the application of synchronisation settings. In relation to ED3, the data service dimension is addressed within SN-M and SU-M, where data flow through a lifecycle including collection, processing and analysis, not only ensuring correct management, but advanced decision-making as well. ED4, which evaluates intelligence, is also addressed by SU-M. This module leverages AI and ML techniques to enable intelligent decision-making, allowing the system to perform autonomous reconfiguration of real-world assets in response to dynamic conditions. In respect to user experience, ED5 is promoted in VI-M. This component prioritises intuitive interaction, providing clear representations and responsive controls within the communication space to enhance human understanding and oversight. Lastly, the system contributes to ED6 (trustworthiness) through (i) the implementation of controlled deception logic mechanisms in OR-M to protect sensitive information, and (ii) communication constraints in SU-M to regulate the influence of DT decisions on the real system, thereby maintaining its functionality, reliability, and security. As illustrated in Table 5, those integrated modules demonstrate a robust alignment with the different evaluation frameworks, ensuring that ADTwin not only replicates but also enhances the monitoring, control, and resilience of complex systems.

9 Conclusions and future work

Although the high fidelity of DT is a crucial feature for testing simulations, the current direction of DT technology towards protection and deception purposes generates new security issues in connection with system exposure. While this subject remains an active area of research, specific architectures have not been sufficiently explored, leaving a gap that still needs to be addressed. Thus, this paper presents a flexible, modular and intuitive architecture that establishes the foundation for the modelling of adaptive DT systems. Concretely, the proposed framework encompasses the adaption of DT capabilities such as the degree of synchronisation, communication, monitoring, decision-making, and imitation. Different configurations with these characteristics make it easier for the system to tailor to specific scenarios for protection, deception, and testing purposes.

Additionally, the IoT-EMS scenario UC demonstrates the strength of the approach

through experimentation, showcasing the functioning of both synchronous and asynchronous modes of operation, as well as the system's response to attacks. The results presented scenarios in which autonomous proactive actions are both disabled and enabled, highlighting the system's adaptability and effectiveness under different conditions, always prioritising supervisor configurations.

Future work will focus on enhancing the system by integrating advanced detection mechanisms based on AI and ML models, with the aim of identifying and mitigating adversarial AI attacks. Additionally, facilitating the creation of adversarial AI threats within the platform will allow for rigorous testing of the targeted system and further reinforcement of the defence. The current system architecture is designed to support such extensions, providing a robust foundation for the development of future adaptive systems. This adaptability ensures that the platform can evolve to address emerging threats and leverage cutting-edge techniques for improved security and resilience.

Acknowledgements

This work has been supported by the AIAS project (GA ID: 101131292), which is funded by the European Union (EU) under HORIZON-MSCA-2022-SE-01. Additionally, this paper has received funding for open access charge by Universidad de Málaga / CBUA.

References

1. El Saddik, A., Laamarti, F., Alja'Afreh, M.. The potential of digital twins. *IEEE Instrumentation Measurement Magazine* 2021;24(3):36–41.

2. Eramo, R., Bordeleau, F., Combemale, B., Brand, M.v.d., Wimmer, M., Wortmann, A.. Conceptualizing digital twins. *IEEE Software* 2022;39(2):39–46. doi: \bibinfo{doi}{10.1109/MS.2021.3130755}.

3. Barricelli, B.R., Casiraghi, E., Fogli, D.. A survey on digital twin: Definitions, characteristics, applications, and design implications. *IEEE Access* 2019;7:167653–167671. doi:\bibinfo{doi}{10.1109/ACCESS.2019.2953499}.

4. Alcaraz, C., Lopez, J.. Digital Twin: A Comprehensive Survey of Security Threats. *IEEE Communications Surveys & Tutorials* 2022;24(thirdquarter 2022):1475 – 1503.

5. Mun, H., Han, K., Damiani, E., Yeun, H.K., Kim, T.Y., Martino, L., Yeun, C.Y.. A Comprehensive Survey on Digital Twin: Focusing on Security Threats and Requirements. *IEEE Access* 2025;.

6. ECSO WG6, . ECSO Technical Paper on Cybersecurity scenarios and Digital Twins. 2023. URL https://ecs-org.eu/ecso-uploads/2023/07/ECSO_WG6_DigitalTwin-2.1.pdf.

7. Homaei, M., Mogollón-Gutiérrez, Ó., Sancho, J.C., Ávila, M., Caro, A.. A review of digital twins and their application in cybersecurity based on artificial intelligence. *Artificial Intelligence Review* 2024;57(8):201.

8. Iqbal, M., Suhail, S., Matulevicius, R.. Deceptwin: Proactive security approach for iov by leveraging deception-based digital twins and blockchain. In: *Proceedings of the 19th International Conference on Availability, Reliability and Security*. ARES '24; New York, NY, USA: Association for Computing Machinery. ISBN 9798400717185; 2024:.

9. Suhail, S., Iqbal, M., McLaughlin, K.. Digital twin-driven deception platform: Vision and way forward. *IEEE Internet Computing* 2024;.

10. Yigit, Y., Kinaci, O.K., Duong, T.Q., Canberk, B.. Twinpot: Digital twin-assisted honeypot for cyber-secure smart seaports. In: *2023 IEEE International Conference on Communications Workshops (ICC Workshops)*. IEEE; 2023:740–745.

11. Liatifis, A., Eleftheriadis, C., Mpatzos, Z., Nanos, I., Lagkas, T., Goudos, S., Argyriou, V., Psannis, K.E., Moscholios, I.D., Sarigiannidis, P.. Sihoneypot: A digital twin-based honeypot for autonomous vehicles. In: *2024 13th International Conference on Modern Circuits and Systems Technologies (MOCAST)*. IEEE; 2024:1–4.

12. Durão, L.F.C., Haag, S., Anderl, R., Schützer, K., Zancul, E.. Digital twin requirements in the context of industry 4.0. In: *IFIP international conference on product lifecycle management*. Springer; 2018:204–214.

13. Nintsiou, M., Grigoriou, E., Karypidis, P.A., Saoulidis, T., Fountoukidis, E., Sarigiannidis, P.. Threat intelligence using digital twin honeypots in cybersecurity. In: *2023 IEEE International Conference on Cyber Security and Resilience (CSR)*. IEEE; 2023:530–537.

14. Schleich, B., Anwer, N., Mathieu, L., Wartzack, S.. Shaping the digital twin for design and production engineering. *CIRP annals* 2017;66(1):141–144.

15. Kritzinger, W., Karner, M., Traar, G., Henjes, J., Sihn, W.. Digital twin in manufacturing: A categorical literature review and classification. *Ifac-PapersOnline* 2018;51(11):1016–1022.

16. Recommendation itu-t y.3091 - digital twin network – capability levels and evaluation methods. 2023. URL https://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-Y.3091-202312-I!!PDF-E&type=items.

17. Alcaraz, C., Lopez, J.. Digital Twin Security: A Perspective on Efforts From Standardization Bodies. *IEEE Security & Privacy* 2025;23:83–90.

18. Siwakoti, Y.R., Bhurtel, M., Rawat, D.B., Oest, A., Johnson, R.C.. Advances in iot security: Vulnerabilities, enabled criminal services, attacks, and countermeasures. *IEEE Internet of Things Journal* 2023;10(13):11224–11239. doi:\bibinfo{doi}{10.1109/JIOT.2023.3252594}.