Comparison between Digital Twin platforms

LEO RUIZENDAAL, University of Twente, The Netherlands

Digital twins are widely seen as a technology to manage the complexities of cyber-physical systems as they grow in size. This has been accelerated due to the widespread adoption of IoT devices, allowing for large quantities of devices to have a network interface. Digital twin platforms provide the ability to systematically create models for physical objects, enabling continuous monitoring, simulation and optimization by utilizing networks to collect data from internet connected devices. The effectivity of an implementation is dependent on the platform chosen to create and operate the system. There are several platforms available for both local deployment and using online service providers. Two widely used online service providers are Microsoft Azure and Amazon Web Services which provide digital twin platforms Azure Digital Twin and IoT TwinMaker respectively. Both of these platforms differ significantly in pricing structure and provide different service infrastructure, integrated with their existing services. These differences are of importance when regarding low-latency systems, due to their unusual requirements. This research evaluates the pricing structure and ease of integration of both platforms in the context of a low-latency use case.

Additional Key Words and Phrases: Digital twin, Digital model, Digital thread, Real-time control

1 INTRODUCTION

The fourth industrial revolution, commonly referred to as Industry 4.0 revolves around data, machine learning, robotics and internet of things devices. As resources such as sensors, data and computing power proliferate, the need for interconnected, cooperative and autonomous production systems has grown significantly. Advances in information and communication technologies have accelerated the development of manufacturing, fostering rapid progress in computer-aided technologies[7]. This transformation enables intelligent management of production facilities and products by leveraging historical data, continuous sensor input and extensive information-processing capabilities[9]. To enable these insights Digital Twins are used, they represent a physical object in a virtual environment using bidirectional communication.

To contribute to the innovation and research of cyber-physical systems, the University of Twente has established the IoT CyberLab. This lab emphasizes the security and privacy of IoT devices while providing state-of-the-art infrastructure for students and researchers. Its offerings span data collection, safety analysis and system testing[14]. To enhance accessibility, the IoT CyberLab is developing a remote development system featuring a physical rover interconnected with a virtual model via bidirectional data exchange. This system will allow users to remotely execute, modify and simulate programs.

Digital twin platforms present a promising solution for implementing such systems without the significant development effort, hosting costs and maintenance burdens associated with custom-built systems. Given the variety of digital twin platforms available, selecting the most suitable option requires a thorough comparison based on ease of integration and cost-effectiveness. This research evaluates two leading online hosting services, Amazon Web Services (AWS) and Microsoft Azure, both of which offer robust digital twin modeling capabilities.

TScIT 42, January 31, 2025, Enschede, The Netherlands

© 2025 University of Twente, Faculty of Electrical Engineering, Mathematics and Computer Science.
Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

2 PROBLEM STATEMENT

As Digital twins are used to model and manage complex systems, the choice of platform becomes an important factor deciding the effectiveness of implementations. Each platform offers distinct features, creating uncertainty about which one strikes the best balance between cost and ease of integration. The absence of a comprehensive comparison poses significant challenges for developers, potentially leading to inefficient models or difficulties during design, deployment or scaling of a Digital twin systems. Thus, a comparative evaluation of a selected of these DT platforms can serve as a valuable resource to guide developers in making well-informed decisions.

2.1 Research question

This research aims to explore the differences in implementation and operation cost between two widely used Digital Twin platforms: AWS IoT TwinMaker and Azure Digital Twin. The study is structured around the following sub-questions:

RQ1: How are the selected DT platforms structured?
RQ2: What are the key functional differences between the platforms?
RQ3: How do these differences impact cost and ease of integration?

3 BACKGROUND

3.1 Existing Implementation

The existing digital twin implementation consists of a rover accepting commands via a REST HTTP service. A unity virtual environment is responsible for the visualization of the rovers position. The telemetry of the rover and the commands from the virtual environment are processed using an external server that forwards the messages.

The rover is a DiddyBorg V2 robot, consisting of a frame, 2 sets of 3 motors, a thunderborg motor controller, a single board computer, a battery pack and a camera[11]. The computer provides a wireless WiFi connection for communication with the remote server. The motor controller provides a battery voltage sensor input, command of the motors and a configurable status indicator while the computer provides a camera feed and internal conditions regarding processor temperature and usage.

3.2 Digital Models

The process of digitization of physical assets has been long ongoing. The first 3D models dating back to over 40 years ago[12]. At this

time the development was focused on digital models, these are handmade models often used for designing products. A commonly used tool is CAD software, where computers aid in creating a detailed design digitally. These models are often used for creating production and construction documentation alongside evaluating design options and considering possibilities with greater ease and without the need for a physical model.
In the process of creating a digital model data flows only when manually transferred, where the digital model can be a simple approximation or a highly detailed model that can provide all production instructions directly without further modification. The models purpose dictates the amount of detail and the focus of attention. Where a model used for calculation might contain a lot of information about material properties of different elements. A model used for demonstration can contain more spatial resolution to give a clear indication of the visual appearance of the end product[12]. Figure 1a shows a physical object and its digital model with only manual data flow.
Applying this concept to a rover, the model could be manually sculpted to replicate a physical object or designed digitally from scratch. Once created, the model can serve various purposes, including production, demonstration or reference.

3.3 Digital shadows

The progression from digital models leads to the concept of a digital shadow. Unlike a static digital model, a digital shadow is a dynamic representation of a physical object that is continuously updates with real-time data. While digital models are primarily used for production, digital shadows enable ongoing simulation, predictions and analyses. By integrating data from various heterogeneous sources, digital shadows facilitate complex management and provide insights throughout the life-cycle of an asset [5]. Figure 1b illustrates a physical object paired with its corresponding digital shadow, demonstrating a unidirectional automated flow of data.
For instance, augmenting the digital model of a rover with continuous telemetry data from sensors on the physical rover would result in a digital shadow. This digital shadow could be utilized for remote monitoring, or it could generate a model of the surrounding environment using LiDAR data. These capabilities make digital shadows invaluable for tasks requiring real-time updates and data-driven decision-making.

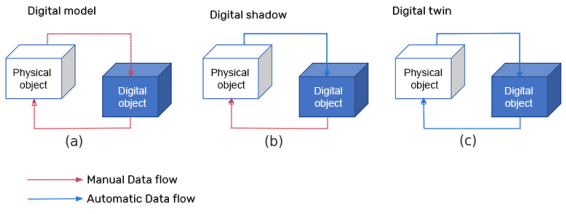


Fig. 1. Different levels of automation (adapted from [12])

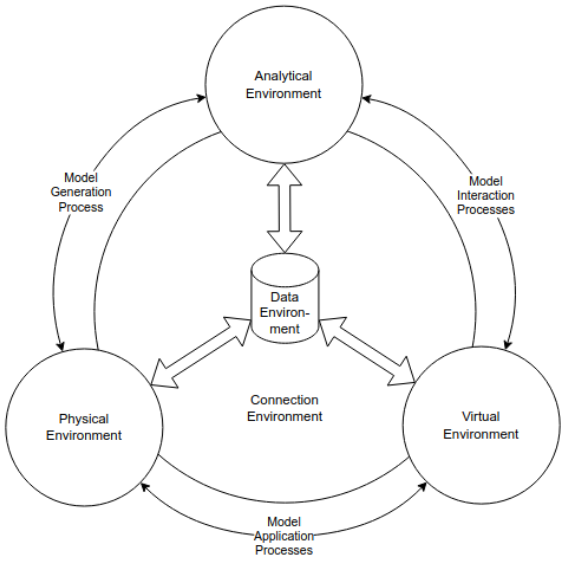


Fig. 2. Components of a Digital Twin[6]

3.4 Digital Twins

Digital Twins(DTs) are an advanced integration of models, services and real-time data designed to interact with and monitor cyber-physical systems or processes. By leveraging both current and historical data, DTs are utilized for various purposes, such as optimizing production processes, detecting faults, scheduling preventative maintenance and gaining deeper insights into system behavior as a whole.
A Digital Twin comprises two main components: the virtual environment (VE) and the physical environment (PE). The VE serves as a digital representation of the PE, and it resides within a digital twin platform, while the PE consists of the actual physical process, equipment or asset (see Figure 2).
The term "twin" reflects the continuous, bidirectional interaction between the VE and the PE. Changes in the physical environment are mirrored in the virtual environment, while control signals and simulations from the VE are sent back to influence the PE. This synchronization is depicted in Figure 1c, which shows the digital and physical objects interconnected with an automatic bidirectional data flow.
For example, when the digital shadow of a rover is enhanced with automated processes that relay data to and from the physical rover, it evolves into a digital twin. This transformation enables advances simulation, the validation of operations before implementation and the calculation of physical movements using complex algorithms that integrate multiple data sources.

3.5 Digital Twin Platforms

Digital Twin Platforms provide a suite of services designed to assist developers in creating, managing and interacting with digital twin models. These platforms offer specialized tools and processing capabilities that facilitate data handling, including the necessary

calculations and transformations for both incoming and outgoing information[10]. Many platforms aim to provide an end-to-end solution, integrating their digital twin services with additional offerings such as storage, data processing, communication and interactive dashboards[2, 4].
For this research, Microsoft Azure Digital Twin and Amazon Web Services (AWS) IoT TwinMaker were selected as the platforms of interest due to their popularity and their integrated services for hosting and managing digital twins. Both platforms offer a set of essential services required to deploy a fully functional digital twin system, with each platform requiring its own distinct set of components.

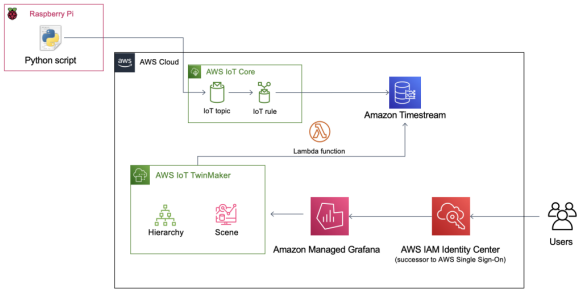


Fig. 3. Structure of required AWS services

AWS IoT TwinMaker. AWS IoT TwinMaker provides a comprehensive ecosystem for digital twin solutions. The platform requires several core services for its operation:

- **IAM Role**: A role with the necessary permissions to access other AWS services.
- **Storage**: S3 buckets are used for storing historical and configuration data.
- **Data Routing**: Data is routed from edge devices using AWS Lambda functions and Timestream Telemetry Store for time-series data storage.

```
"propertyDefinitions": {
  "telemetryAssetType": {
    "defaultValue" : { "stringValue": "Rover" }
  },
  "temperature": {
    "dataType": { "type": "DOUBLE" },
    "isTimeSeries": true,
    "isStoredExternally": true
  },
  "battery_voltage": {
    "dataType": { "type": "DOUBLE" },
    "isTimeSeries": true,
    "isStoredExternally": true
  }
```

Listing 1. Section of AWS digital twin sensor definition

A detailed overview of the AWS IoT TwinMaker arrchitecture, showing the integration of these components and their interactions, is provided in Figure 3.
The TwinMaker defines the virtual environment through entities that can contain components. Entities represent the physical objects represented in the virtual environment containing components that connect certain functions to selected data sources. AWS IoT TwinMaker uses json formatted files to define entities and components, listing 1 shows a snippet of the definition of the sensor component for the rover.

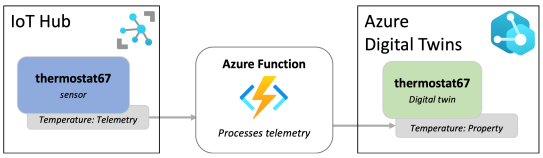


Fig. 4. Structure of required Azure services

Azure Digital Twin. Azure Digital Twin offers a robust set of services, with the core components required for its operation including:

- **IoTHub**: Facilitates the transfer and communication of data with external devices and systems.
- **Functions**: A function that manages the transportation of data between the digital twin and the IoTHub, which handles external communication.
- **IAM Role**: A role with required permissions for accessing data within services.

A complete overview of the Azure Digital Twin architecture, detailing how data flows between services can be found in Figure 4
Azure Digital Twin uses Digital Twin Definition Language (DTDL) to define a digital twin. Listing 2 shows a snippet from the definition of the virtual rover detailing the sensors and telemetry. Azure Functions are used to transfer data from the IoT Hub and digital twin, this allows for processing of the data and handling different formats.

```
"@type": "Component",
  "name": "Sensors",
  "schema": {
    "@type": "Object",
    "fields": [
      {
        "name": "temperature",
        "schema": "double"
      },
      {
        "name": "batteryVoltage",
        "schema": "double"
      }
```

Listing 2. Section of Azure DTDL sensor definition

Table 1. Publications in google scholar with search term "Digital Twin" (checked January 2025)

Year	Publications
2018	33400
2019	39400
2020	46600
2021	54000
2022	55800
2023	52800
2024	82400

4 RELATED WORK

There is an increasing trend regarding publications referencing digital twins, Table 1 shows results returned by google scholar. The growing number of publications aligns with the increase in IoT devices used in industry[9]. In Adamenko et al. several digital twin platforms are compared[1]. Differences in ease of development and cost are discussed and concludes with none of the tested tools presenting a perfect solution for their chosen application as all showed disadvantages. The paper concludes by suggesting limitations in computing power, advanced software and infrastructure are potentially holding digital twin applications back.

In Singh et al. the viability of developing digital twins using Unity and Gazebo is compared[13]. The paper compares the implementation and performance of these tools. The tools differ on performance and functionality, but can be matched with application specifications if carefully chosen. Further the latency of systems involving these software solutions indicate the potential for directly coupled systems without local on-device processing. Cost is discussed and indicates a preference for free software trials for development purposes and concludes Unity as a cost-effective option provided its specifications are required. As Gazebo is free it can be an alternative if the application requires different functionality.

Pfeiffer et al. discusses the use of the label DT and the associated concept in comparison to the older concept of object oriented modeling[10]. In the paper multiple different DT platforms are compared to shed light on what makes DT different from preexisting methods of modeling. This paper provides insight into the differences between the definition languages of different platforms. With indications how these dissimilarities influence the functionality and integration of platforms.

5 METHODOLOGY

A Digital Twin consists of three primary components: physical, digital and communication layers. To ensure a fair comparison the physical layer remained unchanged, and the functionality of the digital twin was kept identical. The Digital Twins were implemented using AWS IoT TwinMaker and Azure Digital Twin, with the communication layer designed using python scripts running on the rover. These scripts were adapted to accommodate the distinct data ingestion requirements of each platform. This approach isolates platform-specific integration challenges and cost considerations from variations in the physical setup.

The physical setup utilized a DiddyBorg V2 rover, as previously deployed in a project at the IoT CyberLab (see Section 3.1). Its python libraries facilitated rapid development of the communication programs.

The architecture created to develop the DT in AWS IoT TwinMaker consists of a communication program on the rover, a Timestream database to store the telemetry, a lambda function to transfer the data to the IoT TwinMaker service. To connect the telemetry and control to the rover a python program was written that relays the data via REST HTTP requests on the rover side and MQTT on the IoT Core side.

For the Azure Digital Twin implementation an architecture consisting of a communication program on the rover, a IoTHub instance for device communication and Azure Functions for data transfer. The telemetry and control were relayed between the rovers REST HTTP endpoints and the IoTHub using a python library.

6 DISCUSSION

6.1 Ease of integration

Both AWS IoT TwinMaker and Azure Digital Twin build their Digital Twin systems using separate modular services, enabling independent scaling and the flexibility to integrate diverse services within a single platform.

Azure adopts a structure where data is initially send from the device to the IoTHub. From there, Azure Functions transfer the data to the Digital Twin. This architecture provides significant flexibility, enabling resource-intensive tasks to be offloaded to web services, thereby reducing the embedded system's workload. Azure supports multiple programming languages for writing functions and offers data transfer options via HTTPS, MQTT and AQMP, enhancing versatility. For simpler use cases Azure does not demand extensive role and permission management through IAM, but this feature can be enables for complex systems where security is stricter.

AWS TwinMaker relies heavily on AWS CloudFormation templates for tool deployment and uses S3 buckets for data storage. The usage of CloudFormation makes the system more complex to setup as individual parts have to be initialized simultaneously. Data is typically streamed to AWS IoT Core using HTTPS or MQTT. From there, AWS Lambda functions optionally processes the data and transfers it to the TwinMaker instance. These functions can be written in a variety of programming languages, easing the development process. AWS requires detailed role-based access control for service to service interaction, which becomes complex and cumbersome to manage.

6.2 Cost

For cost calculation the device is assumed to send messages of a total size of 100 bytes, this accounts for all message information and headers. To facilitate comparison all costs are noted in USD.

Azure Digital Twin calculates cost for Digital Twin usage, Function usage and IoTHub messages separately[8].

- **Digtal Twin**: Usage is calculated using a flat rate of $1.30 per million api calls or messages processed.
- **Functions**: The cost is based on the flat rate of $0.20 per million messages processed.

- **IoTHub**: Cost consists of a monthly fee based on a tiered system with prices increasing with usage (see Table 2).

Table 2. Azure IoTHub pricing

Message limit	Monthly Cost
400000	$25
6000000	$250
300000000	$2500

The limits for messages is 4 KB providing the opportunity to aggregate 40 updates into a single message, sacrificing latency for cost optimization.

AWS IoT TwinMaker charges seperate rates for usage of TwinMaker, IoT Core and lambda functions[3].

- **IoT TwinMaker**: Usage costs $1.65 per million operations.
- **IoT Core**: Costs are based on a tiered system where price decreases as usage increases. However the first tier is valid for amounts up to 1 billion messages, therefore all of the calculated graphs use the first rate of $1.20 per million messages.
- **Lambda**: Usage incurs a fixed cost of $0.20 per million messages

The limit for messages is 5KB providing a maximum of 50 updates in a single message.

The overall cost of each platform comprises Digital Twin usage costs, function usage and communication expenses. Real-time use cases, which require high message frequency, significantly impact total expenses. To illustrate cost differences, Figures 5, 6 and 7 presents a comparison of daily operating costs against update frequency for 10, 100 and 1000 devices respectively.

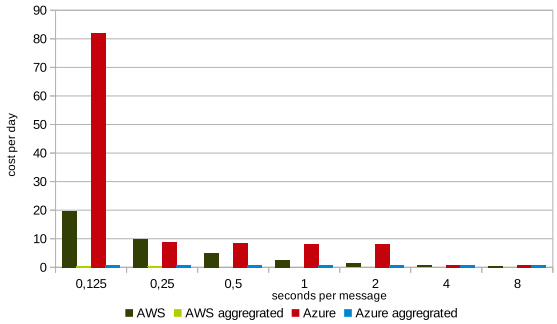


Fig. 5. Daily cost in USD operating 10 devices at different update frequencies

7 CONCLUSIONS

The Azure platform offers more user-friendly development tools, enabling stepwise implementation by allowing individual components to be configured separately and integrated later. Both Azure and AWS support seamless integration with other services provided

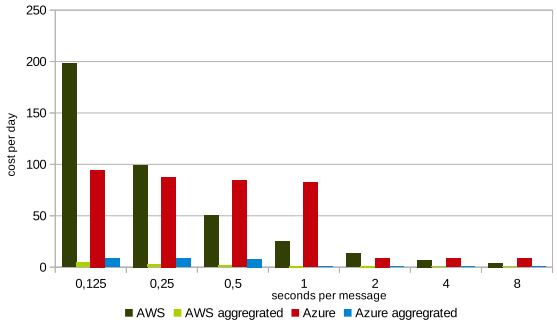


Fig. 6. Daily cost in USD operating 100 devices at different update frequencies

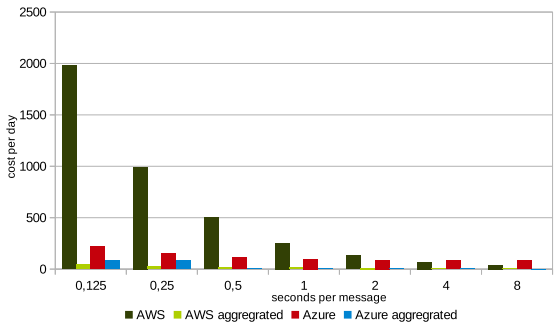


Fig. 7. Daily cost in USD operating 1000 devices at different update frequencies

by their ecosystems, accelerating development when those services are already in operation. This reduces the need for complex data transfer or automated solutions to connect disparate data sources. The cost difference between platforms is depend heavily on the number of devices and the frequency of data traffic. Azure's tiered pricing structure can significantly increase daily costs for small-scale deployments with high-frequency updates, as frequent usage quickly pushes costs into the next pricing tier. Conversely, for larger-scale deployments, Azure proves more cost-effective at high data frequencies. AWS, on the other hand, is more suitable for applications requiring infrequent updates due to its consistent cost structure. Data aggregation can substantially lower costs on both platforms by reducing message volume, but it introduces significant latency, increasing it by up to 40-50 times. While aggregation is a viable cost-saving strategy for low-frequency applications, it is unsuitable for use cases like rover control, where low latency is critical. These findings highlight that operating low-latency systems using cloud-based Digital Twin platforms is a costly approach, and leveraging local processing capabilities may provide significant advantages in terms of cost.

8 LIMITATIONS

This research encountered several limitations. Due to time constraints, no custom 3D models were developed, preventing the evaluation of the platforms' 3D capabilities. Additionally, the physical

rover lacked feedback mechanisms or advanced sensors, limiting the demonstration of the Digital Twin to simulated telemetry regarding movement data. This constraint reduced the usability of the resulting framework, particularly for scenarios requiring detailed interaction between physical and digital layers.

8.1 Future work

Future research should explore other platforms designed specifically for low-latency applications. Platforms with alternative architectures and pricing models may better meet the requirements of such use cases. Investigating offline or hybrid platforms that offer local processing capabilities could significantly reduce costs without increasing latency. Expanding the scope of evaluation to include custom 3D environments, advanced sensor feedback and real-world telemetry will provide deeper insights into the capabilities and limitations of different Digital Twin platforms.

REFERENCES

[1] Dmytro Adamenko, Steffen Kunnen, and Arun Nagarajah. 2020. Comparative Analysis of Platforms for Designing a Digital Twin. In *Advances in Design, Simulation and Manufacturing III*, Vitalii Ivanov, Justyna Trojanowska, Ivan Pavlenko, Jozef Zajac, and Dragan Peraković (Eds.). Springer International Publishing, Cham, 3–12. https://doi.org/10.1007/978-3-030-50794-7_1

[2] Amazon. 2024. Guidance for Digital Twin Framework on AWS. (2024). https://d1.awsstatic.com/solutions/guidance/architecture-diagrams/digital-twin-framework-on-aws.pdf

[3] Amazon. 2025. *AWS Product and Service Pricing | Amazon Web Services.* https://aws.amazon.com/pricing/?aws-products-pricing.sort-by=item.additionalFields.productNameLowercase&aws-products-pricing.sort-order=asc&awsf.Free%20Tier%20Type=*all&awsf.tech-category=*all

[4] Bacancy. 2024. A Guide to Azure Digital Twins - All You Need to Know. https://www.bacancytechnology.com/blog/azure-digital-twins Section: Azure.

[5] Fabian Becker, Pascal Bibow, Manuela Dalibor, Aymen Gannouni, Viviane Hahn, Christian Hopmann, Matthias Jarke, István Koren, Moritz Kröger, Johannes Lipp, Judith Maibaum, Judith Michael, Bernhard Rumpe, Patrick Sapel, Niklas Schäfer, Georg J. Schmitz, Günther Schuh, and Andreas Wortmann. 2021. A Conceptual Model for Digital Shadows in Industry and Its Application. In *Conceptual Modeling*, Aditya Ghose, Jennifer Horkoff, Vítor E. Silva Souza, Jeffrey Parsons, and Joerg Evermann (Eds.). Springer International Publishing, Cham, 271–281. https://doi.org/10.1007/978-3-030-89022-3_22

[6] Jascha Grübel, Tyler Thrash, Leonel Aguilar, Michal Morad, Julia Chatain, Robert Sumner, Christoph Hölscher, and Victor Schinazi. 2022. *The Hitchhiker's Guide to Fused Twins: A Review of Access to Digital Twins in situ in Smart Cities.* https://doi.org/10.48550/arXiv.2202.07104

[7] 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 (Jan. 2021), 346–361. https://doi.org/10.1016/j.jmsy.2020.06.017

[8] Microsoft. 2025. *Prijsoverzicht Azure | Microsoft Azure.* https://azure.microsoft.com/nl-nl/pricing

[9] László Monostori. 2014. Cyber-physical Production Systems: Roots, Expectations and R&D Challenges. *Procedia CIRP* 17 (Jan. 2014), 9–13. https://doi.org/10.1016/j.procir.2014.03.115

[10] Jérôme Pfeiffe, Daniel Lehner, Andreas Wortmann, and Manuel Wimmer. 2022. Modeling Capabilities of Digital Twin Platforms - Old Wine in New Bottles? *The Journal of Object Technology* 21 (Jan. 2022), 3:1. https://doi.org/10.5381/jot.2022.21.3.a10

[11] PiBorg. 2017. *DiddyBorg v2 - Overview.* http://www.piborg.org/blog/diddyborg-v2-overview

[12] Ludmila Seppälä. 2020. *Digital model, digital shadow, or digital twin – what is at the core of data-driven shipbuilding? / Blog / Resources.* https://www.cadmatic.com/en/resources/blog/digital-model-digital-shadow-or-digital-twin/

[13] Maulshree Singh, Jayasekara Kapukotuwa, Eber Lawrence Souza Gouveia, Evert Fuenmayor, Yuansong Qiao, Niall Murray, and Declan Devine. 2025. Comparative Study of Digital Twin Developed in Unity and Gazebo. *Electronics* 14, 2 (Jan. 2025), 276. https://doi.org/10.3390/electronics14020276 Number: 2 Publisher: Multidisciplinary Digital Publishing Institute.

[14] Carlijn Van Den Heuvel. 2024. *IoT Cyberlab officieel geopend aan Universiteit Twente.* https://www.utwente.nl/nl/eemcs/nieuws/2024/12/1881368/iot-cyberlab-officieel-geopend-aan-universiteit-twente