A Data Structure for Developing Data-Driven Digital Twins *

Oghenemarho ORUKELE[1], Arnaud POLETTE[1], Aldo GONZALEZ LORENZO[2], Jean-Luc MARI[2], and Jean-Philippe PERNOT[1]

[1] Ècole nationale supérieure d'arts et métiers, Aix-en-Provance, France
[2] Aix-Marseille University, Marseille, France

Abstract. Digital twins have the potential to revolutionize the way we design, build and maintain complex systems. They are high-fidelity representations of physical assets in the digital space and thus allow advanced simulations to further optimize the behaviour of the physical twin in the real world. This topic has received a lot of attention in recent years. However, there is still a lack of a well-defined and sufficiently generic data structure for representing data-driven digital twins in the digital space. Indeed, the development of digital twins is often limited to particular use cases. This research proposes a data structure for developing modular digital twins that maintain the coherence between the digital and physical twins. The data structure is based on a hierarchical representation of the digital twin and its components; the proposed data structure uses concepts from distributed systems and object-oriented programming to enable the integration of data from multiple sources. This enables the development of a digital twin instance of the system and facilitates maintaining the coherence between the digital twin and the physical twin. We demonstrate the effectiveness of our approach through a case study involving the digital twin of an industrial robot arm. Our results show that the proposed data structure enables the efficient development of modular digital twins that maintain a high degree of coherence with the physical system.

Keywords: Digital and physical twin · Data structure · Digital coherence.

1 Introduction

In recent years, the industrial sector has been undergoing a revolution that seeks to exploit the increase in computational power, advances in artificial intelligence, machine learning, connected devices and availability of data to improve the manufacturing or industrial processes. This revolution, known as industry 4.0, aims to improve the efficiency of production which reduces the cost and increases the profit margin for the producers, while also benefiting the customers by allowing

* This work has been supported by the French ANR PRC grant COHERENCE4D (ANR-20-CE10-0002).

for a better user experience with customization, selections and improved availability of products for the customers based on their behaviours. Some technologies have been developed to achieve the goal of industry 4.0, a notable example is the internet of things (IoT) or the industrial internet of things (IIoT), which have allowed devices to communicate with each other, use the data generated from its operation to optimize them and recognize patterns that can be used for predictive maintenance, for instance.

One of the concepts that have been developed, which is considered as a key aspect of the industry 4.0, is the Digital Twin (DT). This is a technological concept that seeks to create a high-fidelity digital representation of a physical system, and then connect this digital representation to the physical system [1,2]. This connection allows for communication between the digital representation and the physical system, remote monitoring of the physical system with the digital representation and conducting optimization in the virtual space using data from the physical system.

This paper presents a DT data structure (DTDS) for representing physical entities in the virtual space. This data structure is built with the aim of being modular and generic in order to facilitate fast developments of DTs, and also expanding existing DTs built with the proposed data structure when there is a modification to the composition of the physical system. The paper is organized as follows: section 2 reviews literature on industrial DTs and existing DTDS, section 3 proposes a DTDS, section 4 explains the implementation of the DTDS, while section 5 discusses the implementation, provides conclusions and future outlook.

2 Literature Review

There is a consensus that the concept of a DT was introduced by M. Grieves in 2003, as explained in the white paper [1], however, there is no generally accepted definition of a DT. Grieves and Vickers defined a DT in [2] as "a set of virtual information constructs that fully describes a potential or actual physical manufactured product from micro atomic level to the macro geometrical level". Another definition argues that a DT should be understood as a virtual entity that is linked to a real-world (physical) entity, which describes a planned or actual real-world object with the best available accuracy [3]. While [4] argues that a DT is "a comprehensive digital representation of a physical asset, comprising the system design and configuration parameters, the system state and system behaviour".

Despite the different definitions, there are distinct features that are present in all of these definitions, which are (1) the physical entity, (2) the virtual entity, and (3) the connection between physical and virtual entities. These features can be considered the core components of the DT. The physical entity refers to the physical asset that is to be twinned, while the virtual entity on the other hand is the digital representation of the physical entity. For the physical entity to communicate with the virtual entity there needs to be a connection between

both of them. This connection between the entities is a bidirectional automatic data connection which allows the physical entity to send data or its state to the virtual entity, and for the virtual entity to send commands or control the physical entity.

2.1 Digital Twins in Industry

The concept of DTs is used in different sectors. This work focuses on the application of the DT in the industrial sector. The function of DTs in the industrial sector can be classified into three categories: (1) process monitoring, (2) process control and (3) process optimization, simulation or planning [6].

Process monitoring is the most common or in some cases the default application of DTs in the industrial sector. As the name implies, this application involves the use of the DT to track the real-time status of specific parameters or attributes of the physical twin. In [5] a DT was capable of obtaining the real time information of the manufacturing system using the internal sensors of a CNC machine as well as external sensors used to instrument other parts of the system, and relaying the information to the end user of the DT. Fang et al [7] developed a DT for a mobile phone manufacturing system, which is used to monitor the real-time status of the production using CAD models for 3D visualization.

The process control application involves the use of the DT to control the physical twin. This application seeks to replicate any change made to parameters of the DT in the physical twin. Fonseca et al changed the position and direction of a model naval vessel using the DT of the vessel [10], and Fen Yepeng et al used a DT to control selected production parameters of an automotive camshaft production line [5].

In the process simulation, optimization or planning application, the data obtained from the physical twin is used for computation in the DT to simulate the behaviour of the physical twin when certain parameters change [8]. For example, a DT was developed for a CNC milling machine to calculate the residual lifetime of the ball screw actuator for determining when preventive maintenance should be carried out [9].

2.2 Digital Twin Data Structure

Data structures are an organization of data in a logical or mathematical model that is simple enough so that one can effectively process the data when necessary. In the context of DTs, the digital twin data structure (DTDS) refers to the organization of the data of the physical twin in the virtual space.

In the literature surveyed, there is a noticeable lack of DTDS in the different implementations of DTs. In the DTs' implementations which have a DTDS, the DTDS acts as a central data aggregator for the multi-source data from the physical entities for DT services to utilize, and to maintain the coherence between the physical twins and the DT services as shown in Fig. 1. The DTDS can be implemented with XML [14,9], python [12] or OWL [15]. The data structure in

some cases can be developed using an object-oriented approach where different classes are created for components of the physical twin [12,15].

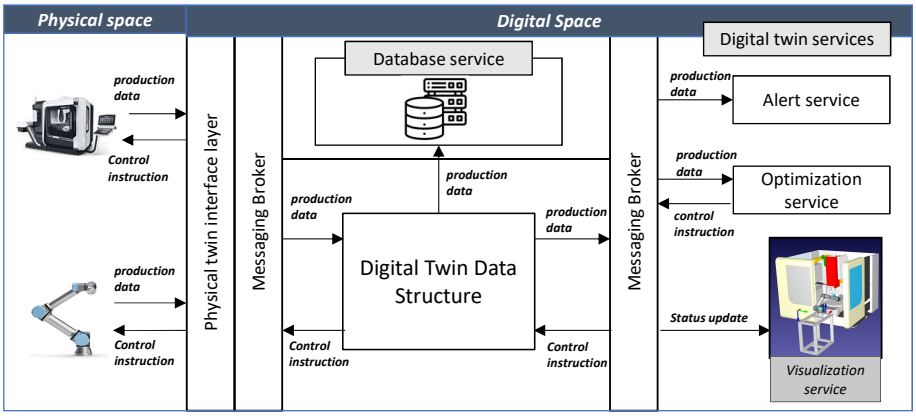

Fig. 1. Digital twin data structure in the DT concept.

Weichao Luo et al developed a data structure for the DT of CNC machine tools [9]. This data structure allowed for representation of the different components of the CNC milling machine in a hierarchical manner as well as representing the relationships between the components. In another implementation, discrete event system (DES) modelling theory was used to create a data structure for the DT of manufacturing shop floor [11]. This data structure was able to represent the flow of a manufacturing system, but lead to a complex relational structure because of the multiple components. Similarly, a DT data structure was used in the development of a DT for a mobile log crane [13], this data structure represented the different components in the system and the data state in a detailed manner. However, it had a high level of complexity due to the numerous types of classes.

The DTDS allow for the development of digital twin instances (DTI) [16], which allow for faster development of DTs of complex systems that are made up of multiple instances of the same physical entity. DTIs enable the creation of isolated instances of the DT which can be used to conduct experiments based on a test data without affecting the principal DT instance. Another benefit of the DTDS is that it allows for scaling of the DT services without having to carry out a one-to-one connection with the components in the physical entity.

Building upon the layer axis of the Reference Architectural Model Industrie 4.0 (RAMI 4.0) framework [17], the DTDS proposes an implementation for the integration and communication layers. Within the integration layer, the DTDS facilitates the digital availability of the physical twin, allowing the data exchange by various digital services that reply on them. Concerning the communication layer of RAMI 4.0, the DTDS establishes a standardized interface that enables

manufacturers, customers, or third parties to communicate with the physical twin in a manner that is independent of the underlying communication protocol [18].

The DTDS in the literature surveyed lacked sufficient genericity which makes them mostly suitable for only developing DTs for the specific use case under study or the family of systems under study in the literature. This limits the application of the DTDS in the industrial section; this paper introduces a generic DTDS that allows for modular development of DTs for various industrial systems.

3 Proposed Digital Twin Data Structure

For the development of a modular and sufficiently generic DT, it is important to develop a data structure for the virtual entity that reflect these properties as well. With this in mind, we propose a DTDS that is hierarchical, allows for multiple data connections, has standardized operations through an application programming interface (API), and implements abstraction and encapsulation to ensure data safety.

The digital twin data structure consists of four hierarchical classes: *attribute*, *component*, *system*, and *environment*. Each of these classes is discussed in detail below. This digital twin data structure idea is developed from a combination of systems programming concepts and Object Oriented Programming (OOP) principles, this approach allows for the reuse of pre-configured classes by instantiating a new instance of the class. These hierarchical classes can be used to build more complex systems, and the relationships between these classes are illustrated in Fig. 2.

The *attribute* class represents physical properties in the virtual world and functions as an abstract class. This ensures a standardized and generic design in all classes that implement it, despite any unique implementation details. The attribute class can be defined as a set $A = \{I, N, V, R, T\}$ where I is a unique identifier, N is the name of the attribute, V is the value of the attribute, R is the list of relationship of the attribute and T is the last update time of the attribute. The relationship R is a set of Ids I of the different attributes, components or systems associated with a particular class; it can be defined as $R = \{I_1, .., I_n\}$.

The *component* class consists of a collection of attribute classes that define a component in the physical twin. Standardization enables the creation of reusable component classes that can be used repeatedly. The class includes functions for accessing other classes within it. The component class consist of the following properties $C = \{I, N, F, A_c, C_s, R\}$ where I is the unique identifier of the component, N is the name of the component, F is the path to the CAD file of the component, A_c is the component attributes where $A_c = \{A_{c1}, .., A_{cn}\}$, C_s is the list of sub-components in the component with $C_s = \{C_{s1}, .., C_{sn}\}$, and R is a list of relationships of the component.

The *system* class is composed of component and attribute classes, enabling the creation of systems that consist of multiple components and attributes. It

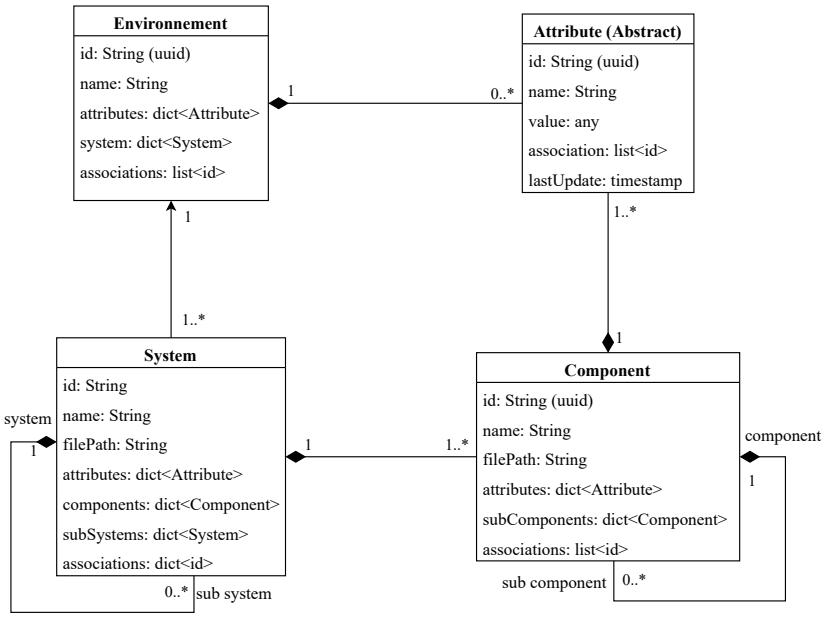

Fig. 2. Digital twin data structure classes and relationships.

features a collection of sub-systems, supporting a recursive design approach. The system class is described as $S = \{I, N, F, A_s, C, S_s, R\}$ where I is the unique identifier of the system, N is the name of the system, F is the path to the CAD file of the system, A_s is the system attributes where $A_s = \{A_{s1}, .., A_{sn}\}$, C is the system components where $C = \{C_1, .., C_n\}$, S_s is the sub-systems of the system with $S_s = \{S_{s1}, .., S_{sn}\}$, and R is the relationships of the system.

Lastly, the *environment* class serves as a wrapper for all other classes in the system, providing the starting point for accessing the digital twin data structure. It is implemented as a singleton class. Similarly, the environment class is defined as $S = \{I, N, A_e, S, R\}$ where I is the unique identifier of the environment, N is the name of the environment, A_e is the set of environment attributes with $A_e = \{A_{e1}, .., A_{en}\}$, S is the set of systems in the environment with $S = \{S_1, .., S_n\}$, and R is the relationships of the environment.

The standardized nature of the DTDS allows for algorithms to be developed for accessing the data in the data structure and sending commands to the physical twin using the DTDS. Fig. 3 presents the flow chart of an algorithm for reading the data of a specified attribute in the digital twin data structure; the algorithm consists of multiple search operations that try to find and return an attribute with the specified name in the DTDS. Similarly, other algorithms can be developed to control the physical twin using the DTDS.

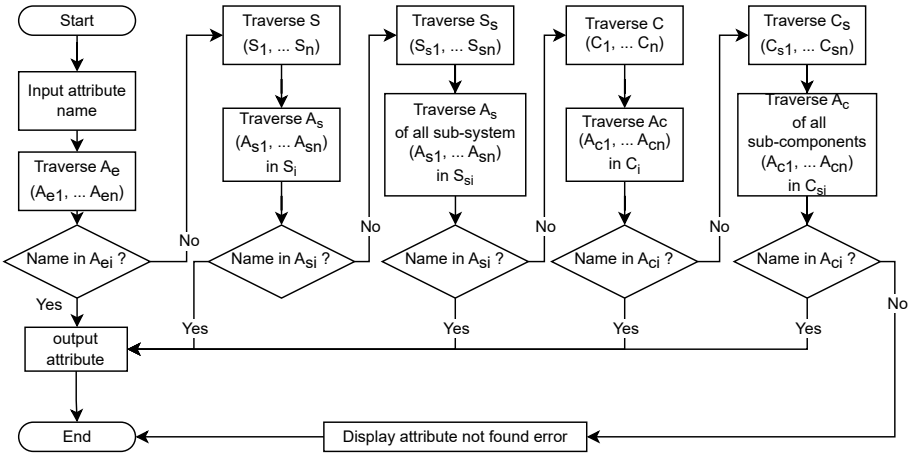

Fig. 3. Flow diagram for reading data of attribute in the DTDS.

These four classes can be used to build digital twins of complex systems if the systems have been decomposed into attributes, components and systems of interest. This approach allows for modelling a heterogeneous industrial environment made up of different systems while allowing the digital twin services access to the same parameters regardless of the implementation details of the system. This allows for a generic approach in the development of DTs.

4 Implementation of the Digital Twin Data Structure

This section presents an implementation of the DTDS in the development of a DT for an industrial robot arm. The industrial robot is a UR5e 6-axis collaborative robot that has integrated sensors which can be used to obtain data from the robot.

To create a DT, it is important to first identify the aspects of physical entity that are to be replicated in the virtual space. In this case, the aspects to be replicated are the joints (Jn) and the tool centre point (TCP) of the robot shown in Fig. 4. The joint attributes of interest include position, velocity, temperature, voltage, and current while the TCP attributes of interest include tool position and payload. These attributes were selected based on authors' preferences, additional attributes can be added as needed.

In representing the UR5e robot using the DTDS, a bottom-up approach is adopted i.e. building the DTDS from the attribute to environment class. The different attribute of interest are modelled with the attribute class. This allows for the implementation details for the data exchange of the different attributes to be implemented independently. As an example, the joint temperature is a scalar quantity while the TCP position is a vector quantity, despite the different

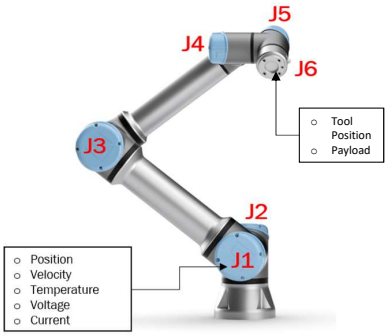

Fig. 4. UR5e robot with joints and joint attributes.

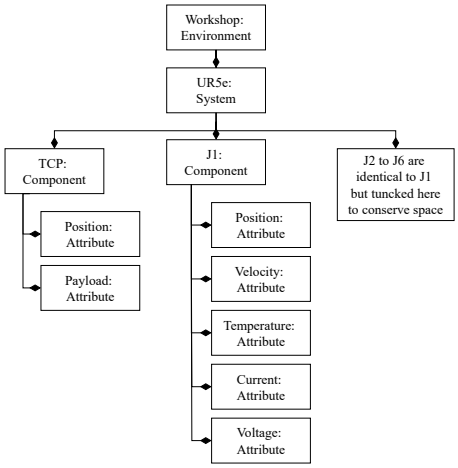

Fig. 5. Representation of the UR5e robot with the DTDS.

quantity types, they can be represented using the attribute class and integrate seamlessly into the DTDS. Within the DTDS, the joint component is represented with a component class which comprises the specific attributes associated with the joint. This facilitates the creation of five similar components for the remaining joints of the UR5e robot, with distinct names and ids assigned to each.

Likewise, the DTDS employs a component class to model the TCP of the UR5e. Subsequently, the associated attributes are incorporated into this class, effectively capturing the essential characteristics of the TCP within the DTDS framework. Subsequently, the UR5e robot is modelled within the DTDS using the system class, comprising of the six joints and the TCP components. Finally, an environment class is instantiated, serving as a container for the UR5e system within the DTDS framework. The resulting representation of the UR5e robot with the DTDS is illustrated in Fig. 5.

The process of modelling the UR5e robot with the DTDS shows how it can be applied to a system by decomposing it into attributes and components. It also illustrates the modularity of the DTDS, as seen in the modelling of a joint component with its attributes and the subsequent reusability in the instantiation of the other joints in the UR5e robot.

5 Discussion and Conclusion

5.1 Discussion

The DTDS representation of the UR5e robot facilitates the consolidation of the implementation details of data exchange between the physical and digital twins in the respective attribute classes of the DTDS, which minimizes the need for redundant implementation effort. Consequently, various DT services can seamlessly interact with the physical twin by leveraging the DTDS's API, ensuring a cohesive and standardized approach. Thus the DTDS serves as a central hub in the digital space for DT services to interact with the data from the physical twin.

Additionally, the DTDS enables the creation of multiple DTIs that can be utilized for various purposes, such as simulating the system's behaviour using test datasets, as depicted in Fig. 6. This also ensures the efficient scaling and modularity of the DT in the event that a new instance of the physical robot or an entirely new system is introduced.
As of the time of writing this paper, a digital shadow has been developed utilizing the DTDS framework to acquire real-time data from two physical entities namely the UR5e robot and a 5-axis CNC machine. The data stored within the DTDS is seamlessly integrated with a data visualization dashboard developed with Node-RED [19], alongside a 3D visualization platform facilitated by RoboDK [20].

5.2 Conclusion and Future Work

This article presents a DTDS that is capable of representing a physical entity in the virtual space. The DTDS is modular, generic and aims to fill the research gap

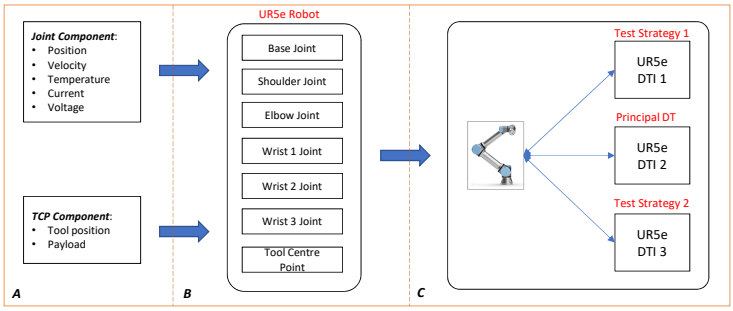

Fig. 6. Construction of digital twin instance for UR5e robot - (A) robot components (B) robot instance (C) digital twin instances of robots.

by serving as a central point for DT services to interact with the physical twin, and a generic data structure for developing digital twins of complex system. This allow for the efficient scaling of the DT services, creation of DT instances and the easy addition or removal of systems/components/attributes in the DTDS. This also allows for the development of DT for other systems in an efficient and modular manner.

The next stage of this work involves the integration of the DTDS with a decision-making service and the subsequent implementation of an interface for controlling the physical twin through the DTDS. By incorporating the decision-making service, the aim is to enable optimal and secure control of the physical twin from the virtual domain. As an integral part of validating the proposed DTDS, it will be extended to modelling the digital twins of additional industrial systems.

References

1. Michael Grieves. “Digital Twin: Manufacturing Excellence through Virtual Factory Replication”. In: (Mar. 2015).
2. Michael Grieves and John Vickers. “Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems”. In: Transdisciplinary Perspectives on Complex Systems: New Findings and Approaches. Ed. by Franz-Josef Kahlen, Shannon Flumerfelt, and Anabela Alves. Springer International Publishing, 2017, pp. 85–113.
3. Juuso Autiosalo et al. “A Feature-Based Framework for Structuring Industrial Digital Twins”. In: IEEE Access (2020).
4. Karl Hribernik et al. “Autonomous, context-aware, adaptive Digital Twins State of the art and roadmap”. In: Computers in Industry (2021).
5. Fan Yepeng et al. “A digital-twin visualized architecture for Flexible Manufacturing System”. In: Journal of Manufacturing Systems (2021).
6. Li Yi et al. “Process monitoring of economic and environmental performance of a material extrusion printer using an augmented reality-based digital twin”. In: Additive manufacturing (2021).

7. Luo Fang, Qiang Liu, and Ding Zhang. “A Digital Twin-Oriented Lightweight Approach for 3D Assemblies”. In: Machines (2021).
8. Saikiran Gopalakrishnan et al. “Integrating Materials Model-Based Definitions into Design, Manufacturing, and Sustainment: A Digital Twin Demonstration of Incorporating Residual Stresses in the Lifecycle Analysis of a Turbine Disk”. In: Journal of Computing and Information Science in Engineering (2020).
9. Weichao Luo et al. “Digital twin for CNC machine tool: modeling and using strategy”. In: Journal of Ambient Intelligence and Humanized Computing (2019).
10. Icaro Aragao Fonseca et al. “A Standards-Based Digital Twin of An Experiment with a Scale Model Ship”. In: Comput. Aided Des. (2022).
11. Jiang Haifan et al. “How to model and implement connections between physical and virtual models for digital twin application”. In: Journal of Manufacturing Systems (2020).
12. V. Zhidchenko, Egor Startcev, and H. Handroos. “Reference Architecture for Running Computationally Intensive Physics-Based Digital Twins of Heavy Equipment in a Heterogeneous Execution Environment”. In: IEEE Access (2022).
13. Yi Cai et al. “Using augmented reality to build digital twin for reconfigurable additive manufacturing system”. In: Journal of Manufacturing Systems (2020).
14. Greyce Schroeder et al. “Digital Twin Data Modeling with AutomationML and a Communication Methodology for Data Exchange”. In: IFACPapersOnLine (2016).
15. Chao Liu et al. “Web-based digital twin modelling and remote control of cyber-physical production systems”. In: Robotics and Computer-Integrated Manufacturing (2020).
16. D. Jones et al. “Characterising the digital twin: a systematic literature review”. In: CIRP journal of manufacturing science and technology (29), 36-52 (2020).
17. Baptista, Luís Filipe, and João Barata. ”Piloting Industry 4.0 in SMEs with RAMI 4.0: an enterprise architecture approach.” Procedia Computer Science (192), 2826-2835 (2021).
18. Tasnim A. Abdel-Aty et al. ”Asset Administration Shell in Manufactuing: Applications and Relationship with Digital Twins.” IFAC PapersIbkube 55-10, 2533-2538 (2022)
19. Node-Red Homepage. https://nodered.org/. Last accessed 31 Jan 2023
20. RoboDK Homepage: https://robodk.com/ . Last accessed 31 Jan 2023