Articulating Data and Control Planes for the Composition and Synchronization of Digital Twins

Dan Freeman Mahoro, Thomas Ledoux, Thomas Hassan, Thierry Coupaye

Articulating Data and Control Planes for the Composition and Synchronization of Digital Twins

Dan Freeman Mahoro
danfreeman.mahoro@orange.com
Orange Innovation
IMT Atlantique
Rennes, France

Thomas Ledoux
IMT Atlantique
Nantes, France
thomas.ledoux@inria.fr

Thomas Hassan
thomas.hassan@orange.com
Orange Innovation
Rennes, France

Thierry Coupaye
thierry.coupaye@orange.com
Orange Innovation
Grenoble, France

Abstract

Digital twins have been a trending concept in recent years. As a result, the number of digital twins has been steadily growing, and their reuse and composition are open questions. For example, *The Thing In The Future* (Thing'In) platform gives the possibility to represent a complex system as a Property Graph of Digital Twins (DT)s. However, the representation of the graph hinders understanding of the system and raises the question of modeling more complex digital twins and synchronizing the graph with the system it represents.

In this paper, we propose an approach for creating Composite Digital Twins (CDT)s of complex systems based on a graph of Digital Twins (DT)s. The approach is articulated between a Data Plane (DP) which hosts the graph representation of a complex system and a Control Plane (CP) that synchronizes the DP and the Physical Plane (PP) hosting the complex system. In our solution, we provide mechanisms to create user-defined CDTs that are a composition of other DTs. We also propose transparent mechanisms to ensure the synchronisation of CDTs with the physical world. We explain the main steps of our approach illustrated by an example and discuss possible use cases.

CCS Concepts: • **Computer systems organization** → **Embedded and cyber-physical systems**; **n-tier architectures**; • **Theory of computation** → *Streaming models*; • **Software and its engineering** → **Abstraction, modeling and modularity**.

Keywords: Design of Composite Digital Twin, Synthetic properties, Synchronisation, Composition

ACM Reference Format:
Dan Freeman Mahoro, Thomas Ledoux, Thomas Hassan, and Thierry Coupaye. 2023. Articulating Data and Control Planes for the Composition and Synchronization of Digital Twins. In *1st International Workshop on Middleware for Digital Twin (Midd4DT '23), December 11–15, 2023, Bologna, Italy*. ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3631319.3632300

Midd4DT '23, December 11–15, 2023, Bologna, Italy

ACM ISBN 979-8-4007-0461-1/23/12…$15.00
https://doi.org/10.1145/3631319.3632300

1 Introduction

Digital Twins (DT) have (re)gained momentum in recent years both in research and industry[1][2]. A Digital Twin is a virtual/digital representation (a model or a set of models) of a real/physical entity (e.g., objects, devices, machines, systems such as buildings, factories, cities, telecommunication, electrical or transport networks, humans or animals, etc.) or process which is synchronized with the real world (more or less in real-time as a function of the nature of the use cases).

The adoption of the digital twin concept in particular in complex systems such as found in the Internet of Things and in Industry 4.0 or Smart Cities, brings new opportunities, yet entails numerous challenges to overcome [3].

The composition of Digital Twin is crucial for scaling the DT capabilities to multiple systems, particularly in establishing and overseeing a System of Systems (SoS)[4][5][6]. Instead of handling each system independently, it streamlines the process by employing a single entity – the Composite Digital Twin. This significantly improves the comprehension and management of intricate, interconnected systems.

The aim of this work is to propose an abstraction of complex systems that provides a global and synthetic view by semi-automatically aggregating the data from the entities composing a complex system. This approach involves creating atomic digital twins representing the entities making up the original system and gradually synthesizing these atomic

DT into more readable and understandable structures called Composite Digital Twins (CDT). This gradual composition respects the existing hierarchy of subsystems in the original complex system.

Our work is based on the Thing in The Future (Thing'In) experimental digital twin platform developed at Orange [7]. Digital twins are represented by Property Graphs (PG)s. They are used to model both the structure and the semantics of real-world systems in such a way that it can be efficiently accessed and linked to these systems in real time and bidirectionally.

The purpose of this work is to find an approach for creating a digital twin from smaller digital twins represented by Property Graphs in Thing'In. However, as Property Graphs are standard graph structure, the proposed approach is generic and be can applied to others digital twins systems.

Based on this analysis, we consider the following two Research Questions (RQs) in the remainder of the paper.

RQ1: *"How can we effectively analyze a property graph to improve our understanding of a complex system?"* We make the following hypotheses for providing an answer to RQ1:

- H1: Exposing a hierarchical structure with different hierarchical levels in a Property Graph representing a complex system (system, system of systems, etc.) results in better understanding of the system.
- H2: New nodes with synthetic properties can be created using the properties of nodes on the same hierarchical level and/or a lower hierarchical levels.

If hypotheses H1 and H2 are validated, it is necessary to propagate updates efficiently between the different hierarchical levels, as the new nodes are not explicitly linked to the real system. This raises a second research question: RQ2: *"How can a property graph representing a physical system be updated while remaining consistent with the system it represents?"* To answer to RQ2, we make the following hypothesis:

- H3: Dependencies between nodes on different hierarchical levels can be exploited to propagate updates.

In this paper, we propose a solution based on software architecture articulating different *planes* to promote the separation of concerns (e.g., composition *vs* synchronisation). We also propose to automatically generate some potential *synthetic properties* for the new composite digital twins. In this way, we improve reusability and make it easier for users to design new digital twins.

This paper is organized as follows. Section II presents the approach and an illustration to better understand the contribution. Section III gives some user stories based on the illustrative example. Finally, Section IV discusses the related work while Section V concludes the paper by opening on future research perspectives.

2 Related work

The "Digital Twin Capabilities Periodic Table"[5] by the Digital Twin Consortium can be viewed as a catalog of functions or "capabilities" (organised in "capability families"), which can be chosen from and assembled in building digital twins for specific use cases. Three identified capabilities can be linked to the topics discussed in this paper: 'ontology management', 'digital twin integration' and 'digital twin orchestration' — but composite and hierarchical twins as discussed in this paper is not (yet) covered (same for industrial digital twin platforms by Microsoft or Amazon for instance).

Several research projects have been conducted on digital twins of complex systems based on graph representations, as well as on the composition of digital twins. With regard to DT based on semantic web technologies, a notable project that aligns with our own approach is the "World Avatar Project" introduced by Lim et al. [8]: the World Avatar is a dynamic Knowledge Graph (dKG)[9] accessible through a platform called "J-park Simulator" (JPS)[10]. JPS follows the linked data principle [11] to construct a cross-domain knowledge graph, and ontologies are leveraged to build the dKG. The dKG is continuously updated and modified by a fleet of agents (i.e., software, methods, services, etc.) that use semantic web technologies and operate on the knowledge graph to fulfill defined objectives.

With regards to the composition of DTs, there are works in the literature exploring the idea of interconnected DTs. For instance, Reiche et al. [12] introduce the Digital Twin of a System (DTS) based on a network of Digital Twins: the authors propose a DTS implementation based on an Asset Administration Shell[13] implementation. Another approach to the composition of digital twins is presented by Jia et al.[14]. They introduce the concept of a complex Digital Twin which is a composition of a "simple digital twins". Their approach consists of "division" and an "assemble" phase. Their starting point is a complex system which they decompose into smaller, simpler subsystems; "simple DTs" of such subsystems are created and reassembled in the "assemble" phase.

Below is a more focused analysis on previously mentioned works and their specificities regarding composition of Digital Twins, comparably to our approach developed in further sections. In the world avatar project[8], complex systems are represented with a graph, similarly to our approach. However, their method lacks the concept of DT composition, as the entire graph is considered a single DT. Reiche et al. [12] primarily focuses on implementation of the composition of DTs. However, it lacks a detailed description of the relationships between the individual DTs. Jia et al.[14] also share a similarity with our approach as they use semantic web technologies to describe the relationships between the individual "simple DTs". However, the main difference lies in their use of a pre-established hierarchical graph representation, which requires prior knowledge of the system to be twinned. In contrast, our approach involves a separate phase to discover the hierarchical structure in the graph representation.

3 Proposed approach and illustrative example

3.1 Conceptual Architecture

We propose a three-plane architecture to host the implementation of the composite digital twins: the Physical Plane (PP), the Control Plane (CP) and the Data Plane (DP) [1](see Figure 1).

First, in the PP, we find the real complex system that is represented by a graph in the DP. Physical entities in the PP are equipped with sensors and actuators. A sensor (e.g., camera) monitors the state of a physical entity and signals when changes occur. An actuator (e.g., a mechanism to close and open a door) receives a signal and performs an action (e.g., closing a door). The physical system in the PP and its graph representation in the DP are synchronized.

in that regard, the ADTs are consistent with the definition of Digital Twins Instances (DTI) as defined by Grieves and Vickers in [15] : "this type of Digital Twin [the DTI] describes a specific corresponding physical product that an individual Digital Twin remains linked to throughout the life of that physical product". An ADT can be whatever the user decides : for instance, if the user chooses to represent a whole system by a single node, that node would be an ADT of that system.

After applying our approach, the PG should include a new type of node called *Composite Digital Twins (CDT)s*. A CDT is a composition of two or more digital twins (ADT or CDT). For example, in Figure 3, a CDT of the front wheel is composed of the ADTs of the tyre, inner tube and rim.

Thus, we consider the concept of Digital Twin Property Graph (DTPG), that we define it as an extension of a regular Property Graph, as follows:

Definition 3.1. Let O, L, K and V be respectively, a set of Objects, a set of Labels, a finite set of property Keys names and a set of Values, we assume that these sets are pairwise disjoint. Given a set X, we assume that P(X) is the set of all finite subsets of X, excluding the empty set.

A DTPG is a tuple $G = \{N, E, A, C, \lambda, \theta, \mu\}$ where:

- $N \subset O$ is a finite set of objects called Nodes;
- $E \subset O$ is a finite set of objects called Edges;
- $A \subseteq N$ is set of nodes called ADTs that are synchronized at a set frenquency f with a physical entity;
- $C \subseteq N$ is a set of nodes called CDTs that are a composition of ADTs and/or CDTs;
- $\lambda : E \rightarrow N \times N$ is a function assigning to each edge an ordered pair of Nodes (i.e., a source node and a target node);
- $\theta : O \rightarrow$ P(L) is a function assigning to each object a finite set of labels;
- $\mu : O \times K \rightarrow V$ is a partial function assigning values for properties to objects;

Such that $N \cap E = \emptyset$ and the domain of μ is finite

For example, in Figure 2, we have a DTPG representing a graph where the nodes are ADTs of bike parts and the edges show which parts are linked (label *"is_attached_to"*).

This DTPG is a representation of a complex real-world system that we would like to digitize. The synchronization between the Physical Plane and the Data Plane is achieved initially by means of *Atomic Controllers*, themselves contained in a third plane: the Control Plane (see Figure 1). We call an Atomic Controller a software element present in the CP, which ensures that an ADT is bidirectionally synchronised with its physical counterpart.

After applying our approach, the CP should contain a new type of controllers called a *Composite Controller*, a software element, responsible for updating a CDT. The composite controller receives notifications from all the DTs making up the CDT whenever a change occurs in the PP or in the DP.

Our approach consists of three main phases that follow hypotheses H1, H2 and H3 as explained in Section 1. The first and the second phases focus on the DP and the third phase on the CP.

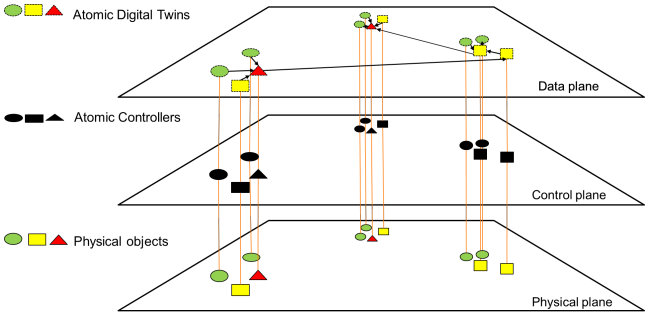

Figure 1. Three-plane architecture

3.2 Phase 1: Exposing a hierarchical structure in the DP with different levels

We tend to think of complex systems as hierarchical systems made up of sub-systems, which can be nested in arbitrary ways. In fact, according to [16], "complexity frequently takes the form of hierarchy".

Intuitively, we want to turn the graph into a tree, where the root is a central node to which other nodes converge. For instance, in a bike, that node could be the frame (see Figure 2).

To achieve that vision, we tranform the DTPG into a directed tree structure or several directed trees (a *polytree*[17]). First, we identify the roots. We arbitrarily choose the roots to be *sink nodes* (a node with no outgoing edges [18]). We place the rest of the nodes in hierarchical levels where each level corresponds to the distance to the sink nodes (sink nodes are at level 0). A DT hierarchical level is defined as follows:

[1]Please note that our control plane and Data Planes are to distinguish from the control plane and Data Planes in (Software Defined) Networks.

Definition 3.2. Given a graph G, let SI be the set of sink nodes and $D(N_i, SI)$ a function that gives the shortest distance amongst distances of a node N_i to all the nodes in SI. A DT hierarchical level is a set of nodes $H \in G$ such as $\forall N_i, N_j \in H \implies D(N_i, SI) = D(N_j, SI)$

However, the edges of the graph do not necessarily form a tree structure, and to break up possible lattices, we duplicate nodes that have more than one ancestor. After this phase, we should have a tree structure that resembles the Figure 3 with duplicated nodes such as node 10 and node 10'. The nodes and their duplicates are controlled by the same controller in the CP.

It is worth noting, however, that not every graph necessarily has a hierarchical structure. In fact, notions such as trophic inconsistency [19] can be used to determine if nodes of a given graph can be classified into hierarchical levels.

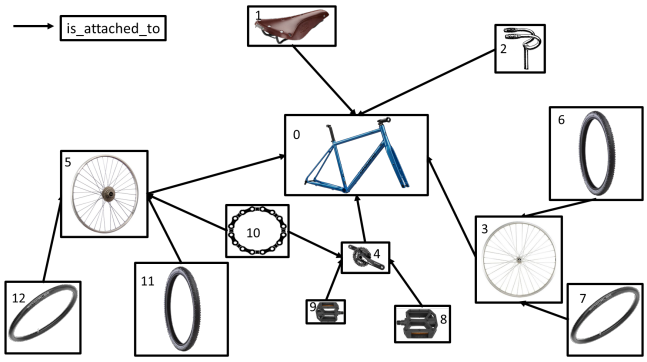

Figure 2. A graph representing a bike in the DP where each rectangle is a ADT node.

3.3 Phase 2: Composite Digital Twins and Synthetic properties

In this section, we present a two-step process for creating CDTs and their *synthetic properties* (see Algorithm 1). It relies on sub-algorithms. First, we call *CDT_creation*(L_i, L_{i-1}) which takes as input two adjacent hierarchical levels and returns a set $SCDT$ of newly created CDTs with references to their components (see Definition 3.3). Then, we call *Connection_to_graph*($SCDT$, L_{i-1}) which will add relevant edges on the new CDTs, namely edges from the components to their respective CDT. Finally, *Synthetic_prop_computing*($SCDT$) will compute new synthetic properties based on properties of the components of each CDT.

3.3.1 Composite Digital Twin creation.

The first step in creating a CDT is to identify the nodes on which the CDTs depend. We call these nodes *components* and define them as follows:

Definition 3.3. (components) Given a hierarchical level L_i ($i > 0$), components are a set of nodes $S_i = \{N_i\} \cup S_{i-1}$ where $N_i \in L_i$ and S_{i-1} is a set of nodes N_{i-1} such as $\forall N_{i-1} \in S_{i-1} \implies N_{i-1} \in L_{i-1}$ and there is an edge between N_i and N_{i-1}.

Algorithm 1 Main algorithm

Input: $levels = \{L_0, L_1, ..., L_n\}$

```
i ←n
while i < 0 do
    SCDT ← CDT_creation(L_i, L_{i-1})
    Connection_to_graph(SCDT, L_{i-1})
    Synthetic_prop_computing(SCDT)
    i ← i − 1
end while
```

For each set of components identified, we create a new Composite Digital Twin node in L_{i-1}. For each newly created CDT, we create edges labeled *"is_component_of"* from each component to the corresponding CDT (see Figure 3).

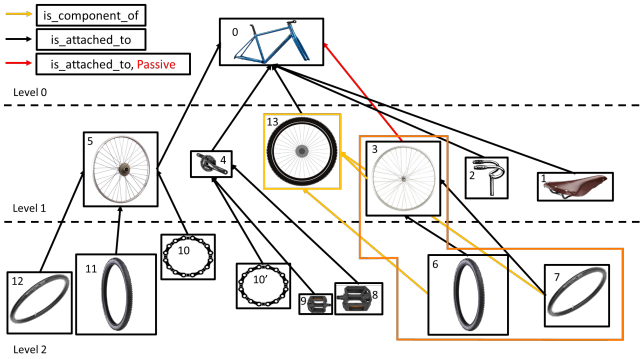

Figure 3. Nodes are classified in hierarchical levels. Node 13 is a CDT and the orange polygon shows which DTs are components of CDT 13.

For the algorithm *Connection_to_graph*($SCDT$, L_i), we proceed as follows. Let C_i be a CDT from the set $SCDT$ of a hierarchical level L_i, N_i be a component of C_i at the same hierarchical level, and N_{i-1} be a node of the higher hierarchical level L_{i-1} such that there exists an edge (N_i, N_{i-1}). In the algorithm, we create an edge from C_i to the node N_{i-1} and we add a label "passive" to the edge (N_i, N_{i-1}). The "passive" label is added so that it in the next iteration of Algorithm 1, CDT are used instead of their components (see Figure 4).

3.3.2 Synthetic Property computing.

Synthetic properties are properties of a CDT that are the result of a function of one or more properties of components of the CDT.

Given a CDT composed by the component set $S = \{N_0, ..., N_i\}$. If all nodes from S share a property P of data type integer, it is possible to recommend to the user, several synthetic properties "S_P" in the CDT such as aggregation operations (e.g., S_P can be Sum (P), Max (P), Min (P), Avg(P)), a composition of these aggregation functions or even user-defined functions

on P. When a synthetic property is defined, the specification of the property (aggregated properties, the involved nodes, etc.) is stored in the memory of the corresponding controller in the next phase. The operations proposed depend directly on the data type of the properties concerned. We leave it up to the user to choose the function they wish to use for their synthetic properties.

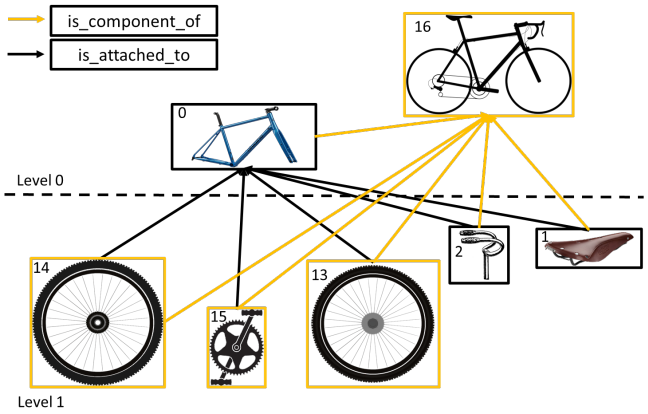

Figure 4. The second iteration of our algorithm, CDTs are in orange rectangles, ADTs are in black rectangles.

For instance, for the bike (see Figure 3), the node 13, representing the front wheel can have a "wear-level" property that can be the average "wear-level" of the rim (node 3), the inner tube (node 7) and the tyre (node 6). This "wear-level" property of the front wheel (node 13) must be updated whenever the "wear-level" changes in nodes 3, 6 or 7.

3.4 Phase 3: Synchronisation

Once a CDT has been created and synthetic properties defined, we proceed to create *Composite Controllers* in the CP based on the specifications of synthetic properties stored in the previous phase.

Composite controllers oversee the updates of the CDT. When an event is captured in any of the controllers controlling a DT involved in a synthetic property. Composite controllers also send notifications to relevant controllers when a synthetic property is updated in the DP.

In fact, synchronization mechanisms that guarantee the consistency of the CDT with real-world are of two types: from ADT to CDT and from CDT to ADT. Both mechanisms are based on a publish/subscribe model [20].

3.4.1 From ADT to CDT. Given a CDT C_i and its set of components CS, the corresponding composite controller CC_i subscribes to all the atomic controllers of components of the CS set. This means that, whenever a modification occurs (from the PP or the DP) that changes the value of a property P of a component of C_i, a notification specifying the property P that is being updated and its new value is sent to CC_i and updates C_i.

3.4.2 From CDT to ADT. Given a CDT C_i and its corresponding composite controller CC_i, all atomic/composite controllers of C_i's components subscribe to C_i. When a modification is made to a synthetic property S_P in C_i, CC_i sends a notification to a controller of all involved nodes in S_P. The notification contains the updated property and its value. C_i stores the knowledge of affected nodes, properties, and values when creating the synthetic property. After sending the notification, C_i's component controllers update the system based on the synthetic property computation. For example, in complex cases, such as cooling a house to a specific temperature, each room's digital twin adapts its strategy to achieve the desired temperature.

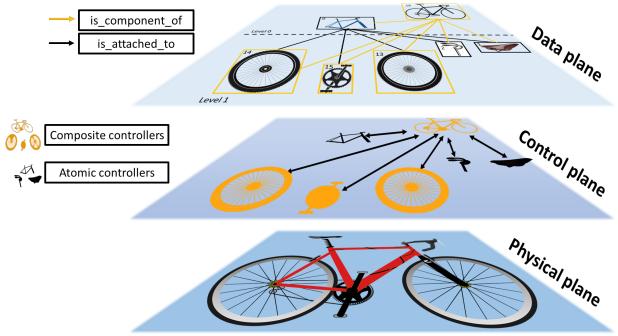

Figure 5. Composite controllers in the Control Plane

4 Illustrative use case

To demonstrate the value of a digital twin composition framework, we present an illustrative use case based on the bike example. This use case is not meant to be realistic. Actual use cases for DT composition may be found in diverse domains (e.g. Industry 4.0, Network Management, Logistics, etc), some of which are discussed in the following section.

Consider a bike rental company operating a large fleet of bikes in a city. With digital twins of each bike, the rental company can track the number of bikes in real time. This information enables efficient restocking of bike depots by redistributing bikes from one location to another.

Besides, digital twin composition offers even more advanced capabilities. By representing bikes as Composite Digital Twins (CDTs), which model and manage each bike as a composition of its components and sub-components, the rental company gains additional benefits. They can remotely assess the state (starting from localization) of each bike and its individual components.

For instance, it becomes possible to evaluate the overall dynamic condition of a bike, even if certain components are not in optimal condition, such as a deflated tire, a missing saddle, a broken brake. This assessment, based on synthetic

properties defined earlier, helps determine whether a bike requires immediate repair, such as a flat tire. Knowing the composition of each bike and the state of its components opens avenues for optimized spare parts inventory management and bike maintenance.

Going one abstraction level higher, the rental company may cluster bikes by administrated areas. Each bike cluster/fleet may be monitored as a single CDT, which will enable to compute synthetic properties of the fleet (e.g. global state, user habits for each area), thus enabling a better view of each fleet and easier anticipation of repair/replace costs in the long term.

5 Conclusion and future work

This paper introduces an approach for creating and managing *Composite Digital Twins (CDT)* of complex systems. CDTs are hierarchical compositions of digital twins, starting from *Atomic Digital Twins (ADT)* which represent the finer granularity in the modelling of entities chosen for a given considered target physical system. CDTs are recursively composed of ADTs and CDTs.

The approach defines and articulates a *Data Plane (DP)* which hosts the (property) graph representation of a complex system and a *Control Plane (CP)* which controls the composition of digital twins in the DP and the synchronisation with the *Physical Plane (PP)*, i.e. the real complex physical system.

The paper has two main contributions. First, it introduces, in the Data Plane, a hierarchical structure within an initial graph of ADTs ("a flat graph model") to embody composite digital twins. At each hierarchical level, composite digital twins are generated with synthetic properties which aggregate properties from other lower (hierarchical) levels. The second contribution is an implementation architecture, in the CP, of software elements called composite controllers, dedicated to the bi-directional synchronization of the CDTs defined in the Data Plane.

The next stages of our ongoing work are to implement our approach. Specifically, with regard to the generation of synthetic properties, one of our perspectives focuses on implementing mechanisms to specify update propagation strategies based on the dependencies between a composite twin and its individual components. Afterwards, we will validate the proposed hypotheses on important real-life use cases, mainly focused on the domain of telecommunication networks and digital infrastructures.

References

[1] David Jones, Chris Snider, Aydin Nassehi, Jason Yon, and Ben Hicks. Characterising the digital twin: A systematic literature review. *CIRP Journal of Manufacturing Science and Technology*, 29:36–52, 2020.

[2] Enric Escorsa. Digital twin: A glimpse at the main patented developments. https://www.ificlaims.com/news/view/blog-posts/digital-twin-patent.htm, 2018. (accessed: 18.07.2023).

[3] James Moyne, Yassine Qamsane, Efe C. Balta, Ilya Kovalenko, John Faris, Kira Barton, and Dawn M. Tilbury. A requirements driven digital twin framework: Specification and opportunities. *IEEE Access*, 8:107781–107801, 2020.

[4] Nataliya Shevchenko. An introduction to model-based systems engineering (mbse). Medium, DataDrivenInvestor (blog), Dec 2018. Accessed: 2023-november-15.

[5] van Schalkwyk Pieter. Digital twin capabilities periodic table : A digital twin consortium user guide. Digital Twin Consortium, Mar 2022. Accessed: 2023-november-15.

[6] Ginestra Bianconi, Alex Arenas, Jacob Biamonte, Lincoln D Carr, Byungnam Kahng, Janos Kertesz, Jürgen Kurths, Linyuan Lü, Cristina Masoller, Adilson E Motter, et al. Complex systems in the spotlight: next steps after the 2021 nobel prize in physics. *Journal of Physics: Complexity*, 4(1):010201, 2023.

[7] Thierry Coupaye, Sébastien Bolle, Sylvie Derrien, Pauline Folz, Pierre Meye, Gilles Privat, and Philippe Raïpin-Parvedy. A graph-based cross-vertical digital twin platform for complex cyber-physical systems. In *The Digital Twin*, pages 337–363. Springer, 2023.

[8] Mei Qi Lim, Xiaonan Wang, Oliver Inderwildi, and Markus Kraft. The world avatar—a world model for facilitating interoperability. In *Intelligent Decarbonisation: Can Artificial Intelligence and Cyber-Physical Systems Help Achieve Climate Mitigation Targets?*, pages 39–53. Springer, 2022.

[9] Jethro Akroyd, Sebastian Mosbach, Amit Bhave, and Markus Kraft. Universal digital twin-a dynamic knowledge graph. *Data-Centric Engineering*, 2:e14, 2021.

[10] Andreas Eibeck, Mei Qi Lim, and Markus Kraft. J-park simulator: An ontology-based platform for cross-domain scenarios in process industry. *Computers & Chemical Engineering*, 131:106586, 2019.

[11] Xiaochi Zhou, Andreas Eibeck, Mei Qi Lim, Nenad B Krdzavac, and Markus Kraft. An agent composition framework for the j-park simulator-a knowledge graph for the process industry. *Computers & Chemical Engineering*, 2019.

[12] Leif-Thore Reiche, Claas Steffen Gundlach, Gian Frederik Mewes, and Alexander Fay. The digital twin of a system: A structure for networks of digital twins. In *2021 26th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 1–8. IEEE, 2021.

[13] Sebastian Bader, Erich Barnstedt, Heinz Bedenbender, Bernd Berres, Meik Billmann, and Marko Ristin. Details of the asset administration shell-part 1: the exchange of information between partners in the value chain of industrie 4.0 (version 3.0 rc02). *Federal Ministry for Economic Affairs and Energy: Berlin, Germany*, 2022.

[14] Wenjie Jia, Wei Wang, and Zhenzu Zhang. From simple digital twin to complex digital twin part i: A novel modeling method for multi-scale and multi-scenario digital twin. *Advanced Engineering Informatics*, 53:101706, 2022.

[15] Michael Grieves and John Vickers. Digital twin: Mitigating unpredictable, undesirable emergent behavior in complex systems. *Transdisciplinary perspectives on complex systems: New findings and approaches*, pages 85–113, 2017.

[16] Herbert A Simon. The architecture of complexity. *Proceedings of the American philosophical society*, 106(6):467–482, 1962.

[17] Sanjoy Dasgupta. Learning polytrees. In *Proc. of the 15th conference on Uncertainty in artificial intelligence*, pages 134–141, 1999.

[18] Giannis Moutsinas, Choudhry Shuaib, Weisi Guo, and Stephen Jarvis. Graph hierarchy: a novel framework to analyse hierarchical structures in complex networks. *Scientific Reports*, 11(1):13943, 2021.

[19] Robert S MacKay, Samuel Johnson, and Benedict Sansom. How directed is a directed network? *Royal Society open science*, 7(9):201138, 2020.

[20] Patrick Th Eugster, Pascal A Felber, Rachid Guerraoui, and Anne-Marie Kermarrec. The many faces of publish/subscribe. *ACM computing surveys (CSUR)*, 35(2):114–131, 2003.

Received 30 September 2023; revised 30 October 2023; accepted 20 November 2023