From simple digital twin to complex digital twin Part I: A novel modeling method for multi-scale and multi-scenario digital twin

Authors: Jia Wenjie, Wang Wei, Zhang Zhenzu, Wang Wei

Date: 2022-02-16T10:49:19+00:00

Abstract

In recent years, digital twins have attracted extensive attention and are becoming increasingly complex. Current digital twin implementations are predominantly confined to specific scenarios, and there remains a lack of methods for constructing complex digital twins in the face of multi-level and multi-scenario working environments, as well as model interaction and coupling. This paper proposes a standardized modeling method for complex digital twin models based on model decomposition and assembly. First, complex digital twin models are partitioned into several simple models according to the levels (Composition), scenarios (Context), components (Component), and code (Code) within the 4C architecture. Levels and scenarios enable digital twins to focus on effective elements within specific scales and scenarios. Components and code are employed to develop simple digital twin models. Second, the simple models of digital twins are assembled into complex models through information fusion, multi-scale correlation, and multi-scenario interaction. The ontology model constructs a comprehensive information repository for entities across different digital twins. The knowledge graph establishes a bridge for relationships between digital twins at various scales. Scenario iteration realizes behavioral interaction and enhances computational result accuracy. This paper provides an implementable approach for constructing complex digital twin models and supports the reuse of components and code to facilitate the rapid development of digital twins.

Full Text

Preamble

From Simple Digital Twin to Complex Digital Twin Part I: A Novel Modeling Method for Multi-Scale and Multi-Scenario Digital Twin

Wenjie Jia, Wei Wang*, Zhenzu Zhang
School of Mechanical and Electrical Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China

Abstract

In recent years, digital twin technology has attracted widespread attention as a critical enabler of digitalization and intelligence. However, as the demand for simulating multi-scale and multi-scenario phenomena in reality expands, digital twins are becoming increasingly complex. Existing digital twin implementations in the literature mostly concentrate on particular applications, while a systematic method for constructing complex digital twins that addresses all elements, variable working environments, changeable processes, and coupling effects remains lacking. This paper proposes a novel modeling method for such complex digital twins based on standardized processes for model division and assembly.

First, the complex digital twin model is divided into several simple models according to the 4C architecture: composition, context, component, and code. Composition and context enable the digital twin to focus on effective elements at specific scales and scenarios, while component and code facilitate standardized, modular development of digital twins. Second, these simple digital twin models are assembled into a complex model through information fusion, multi-scale association, and multi-scenario iteration. Ontology establishes a complete information library of entities across different digital twins, knowledge graph bridges structural relationships between different scales, and scenario iterations realize behavioral interaction and accurate calculation results. This approach provides an implementable method for constructing complex digital twin models, while component and code reuse enables rapid digital twin development.

Index Terms—complex digital twin, simple digital twin, digital twin modeling, smart manufacturing

1 Introduction

The digital twin concept was first proposed by Grieves in 2003 [1], though it did not attract widespread attention at the time. However, with advances in hardware and information technology—including computers, sensors, big data, and artificial intelligence—digital twin implementation has become feasible. NASA formally introduced the digital twin concept in 2012 [2], and subsequently, numerous universities and enterprises have invested in digital twin research. From 2017 to 2019, Gartner listed digital twins among its top ten strategic technology trends for three consecutive years. Digital twins can integrate physical and virtual spaces, making them an effective approach for realizing smart manufacturing [3].

In digital twin implementation, modeling methods have consistently represented a key technology, and scholars have explored various approaches. Grieves

et al. proposed that digital twins consist of real space, virtual space, and data/information flow links between them [4]. Tao et al. introduced a five-dimension digital twin model comprising physical entity, virtual entity, service system, digital twin data, and connection network, where the virtual entity includes geometry, physics, behavior, and rule models [5]. Zhuang et al. explored digital twin concepts and implementation approaches for product digital twins across different lifecycle stages [6]. Alam et al. proposed a digital twin reference model consisting of physical things, cyber things, and hybrid things [7]. These reference models provide fundamental structures for digital twins.

In recent years, digital twins have been widely applied across various fields, including aerospace [8,9], machinery manufacturing [10,11], medical care [12,13], shipbuilding [14], and urban construction [15,16]. Scholars have conducted further research on digital twin applications. Zhao et al. developed a digital twin workshop prototype system based on the Unity3D platform for visual monitoring of real-time data [17]. Ritto et al. employed physics-based computational models and machine learning classifiers for structural damage detection [18]. Lv et al. proposed a digital twin-driven human-robot collaborative assembly framework based on image recognition and deep reinforcement learning to improve assembly efficiency and safety [19]. Lai et al. used an improved finite element method and analytical models to monitor strain, stress, and load of a boom crane [20]. In summary, digital twin realization typically follows these steps: determine application contexts, analyze functional requirements, develop the digital twin (including hardware deployment and software development), commission, and implement. However, digital twins in the literature typically simulate behavior in only one or limited scenarios, which can be considered simple digital twins. Furthermore, even when digital twins share similar elements, their components are developed differently. Indeed, in practice, developers use different tools and platforms, leading to difficulties in model interaction.

Currently, facing multi-scale, multi-scenario, and multi-dimensional digital twin applications—which we term complex digital twins—it is challenging to initiate modeling of the whole elements, entire process, and complete business at once. This difficulty arises partly because modeling from macro to micro composition at full scale is virtually impossible with current modeling capabilities and computing power, let alone for multi-scenario, multi-dimensional applications. Some scholars are attempting to construct new standardized methods for complex digital twin modeling to simulate, predict, and optimize behavior across multiple scales and contexts. Rosen et al. proposed achieving digital twins through Modularity-Autonomy-Connectivity-Digital Twin [21]. Wang et al. introduced a new industrial internet model based on hierarchical multi-granularity digital twins, where different model types have different perception and control objects, and same-layer models cooperate to complete production tasks under higher-layer model control [22]. Pan et al. divided production logistics systems into three levels and employed a cloud-fog-edge control framework for real-time monitoring, decision-making, and control [23]. However, these works simply propose dividing digital twins into different layers, lacking clear interfaces. Con-

sidering that software is the ultimate carrier of digital twins, the development process for complex digital twins remains unclear.

Another emerging question concerns how to integrate, interact, expand, and reuse digital twin models. Information technology development has opened gates for value-added innovative services. Ontology provides a semantic model understandable by computers, enabling integration of complex model information. Bao et al. used ontology-based models to describe resources and processes for assembly workshops [24]. Dai et al. divided as-fabricated data into geometric and non-geometric information and built ontology-based information models for as-fabricated parts [25]. Singh et al. constructed an ontology model and proposed a methodology including map, define, create, convert, and populate to manage databases for digital twins [26]. Lu et al. created digital twins for all manufacturing resources of an international company by developing ontology [27]. Knowledge graphs are typically used to describe physical entity attributes and their relationships. Liu et al. built a quality knowledge model based on knowledge graphs to express product quality factors from three different scales [28]. Sun et al. used assembly-commissioning task knowledge graphs to extract action sequences [29]. Ontology and knowledge graphs can reorganize and invoke modules, providing possibilities for constructing scalable complex digital twins.

In summary, this two-part study investigates complex digital twin modeling and application. Part I proposes a novel modeling method for complex digital twins. The remainder of this article is organized as follows. Section 2 analyzes complex digital twin characteristics and presents the modeling process. Section 3 discusses dividing complex digital twins into simple digital twins through layer and context division, and building simple digital twins through functional component division. Section 4 introduces ontology models and knowledge graphs to assemble simple digital twins into complex digital twins, including information fusion, multi-scale association, and multi-context interaction. Section 5 concludes this study. Part II will illustrate complex digital twin implementation using a multi-scale and multi-scenario workshop example.

2 Characteristics and Modeling Process of Complex Digital Twins

2.1 General Characteristics of Complex Digital Twins

A digital twin represents the mapping of physical entities in virtual space. Digital twin characteristics can be considered as features that make them better reflect reality. Schleich et al. proposed four important characteristics for the ideal digital twin reference: scalability, interoperability, expansibility, and fidelity [30]. Scalability provides insight at different scales. Interoperability enables conversion, combination, and establishment of equivalence between different model representations. Expansibility allows integration, addition, or replacement of digital twins quickly. Fidelity means being very close to physical entities.

Therefore, complex digital twins have connotations of more functions, more composition scales, more dimensional data, and more scenarios. The spatial span of complex digital twins may be huge, making it impossible to express all information on the same scale. Thus, complex digital twin scalability is the ability to present correct data and hide irrelevant data at corresponding scales. Complex digital twins may comprise many simple digital twin models, so their interoperability involves not only converting and combining different model representations of the same object, but also converting and combining models of different objects. Complex digital twin expansibility is the ability to integrate, add, or replace models of different objects. Limited by current modeling methods and computing power, constructing a complex digital twin completely equivalent to a physical entity is very difficult. Therefore, complex digital twin fidelity is the ability to remain as close as possible to the physical product in required functions or behaviors.

2.2 Special Characteristics of a Complex Digital Twin Shop Floor

Taking a digital twin shop floor as an example further illustrates complex digital twin characteristics. A shop floor is a multi-scale system, and its digital twin data usually need presentation at different scales. For example, when the research object is the entire workshop, key data typically include production scheduling, product qualification rate, and energy consumption. When the research object is a machine tool, key data change to spindle speed, feed rate, and motor current. Thus, shop floor scales (layers) could include production, unit, and equipment.

Additionally, different shop floor layers include multiple contexts. For example, the shop floor layer includes contexts such as production scheduling, power consumption statistics, while the equipment layer includes contexts such as data monitoring, remote control, and fault diagnosis. As shown in Figure 1, numerous scales and contexts constitute a shop floor, which is indeed a complex system. Its digital twins should have the following characteristics:

Figure 1. Complex Digital Twin Shop Floor

1. **Scalability**: Content and data expressed in complex digital twin shop floors can adaptively change as the research object changes. When the research object is the shop floor, the digital twin shows production schedule, production efficiency, etc. When the research object is a production line, the digital twin shows machine tool status, AGV position, etc. When research focuses on certain equipment, the digital twin shows actions, health status, etc.

2. **Interoperability**: Digital twin shop floors contain virtual models of many objects, and each virtual model may comprise several models such as geometric, physical property, behavior, and rule models to realize complex functions. Interoperability means models can interact with each other, whether belonging to the same object or different objects. For example,

machine tool behavior model data can drive geometric model movement, and interaction between AGV behavior models and robotic arm behavior models can enable their cooperation.

3. **Expansibility**: On the shop floor, equipment number, type, and even location layout may be adjusted along with overall planning. Therefore, expansibility means that when physical space or functional requirements change, the digital twin can be reconfigured through simple operations to maintain synchronization with physical entities.
4. **Fidelity**: Although making virtual models identical to physical entities is currently impossible, some methods can improve model fidelity as much as possible, such as constructing models that consider more influencing factors and using more advanced algorithms.

2.3 Modeling Process of Complex Digital Twins

From the above analysis, directly constructing a complex digital twin at once is difficult partly because it covers different scales and contexts. Therefore, a division-assembly based modeling approach is proposed, as shown in Figure 2. First, a complex digital twin is divided into several simple, implementable digital twins. The initial problem is how to divide a complex model. Since complex digital twins cover different scales and contexts, they can be divided into different layers according to scales, then into different contexts. Next, these simple digital twins are assembled into a complex digital twin with more sophisticated functions. How to assemble them is another important question, as these simple digital twins may be independent. Here, ontology models can provide suitable containers to fuse related information of research objects. Knowledge graphs can describe physical entity attributes and relations, making them effective tools for describing scale associations of simple digital twins. Context interaction is also a key issue that can be realized through behavior interaction and calculation iteration between simple digital twins.

Figure 2. Modeling Process of Complex Digital Twins

3 Division of Complex Digital Twins in 4C Architecture

This paper proposes a 4C architecture for dividing complex digital twins, representing a hierarchical division method. The 4C architecture contains composition, context, component, and code. Composition divides digital twins into different layers (scales), context defines specific application scenarios, component represents the functional unit for building simple digital twins, and code provides specific component implementation. The detailed structure is shown in Figure 3.

Figure 3. Modeling Architecture of Complex Digital Twins in 4C Architecture

The 4C architecture implementation process is as follows: First, the complex digital twin is divided into several layers with different scales, determining effective expression elements and negligible details at each layer. Then it is divided into different application contexts so that each divided simple digital twin focuses on a specific function. Afterward, each simple digital twin is divided into several functional components according to specific implementation processes. Appropriate programming languages or platforms develop and encapsulate components, forming components with clear inputs, outputs, and associations. Finally, components in the same context are integrated to construct the simple digital twin.

3.1 Division of Complex Digital Twins in Composition and Context

Composition: Physical entities in complex digital twins usually belong to different contexts. Moreover, different contexts have such large spatial spans that they cannot be displayed simultaneously, making division into different scales necessary. Complex digital twins can be divided into several layers at different scales, such as system, unit, part, or subpart. Elements requiring expression differ across layers. The system-layer contains all objects and their environment in the complex digital twin, expressing elements including overall system operating conditions, operating rules, resource consumption, and product output. The unit comprises several equipment pieces to achieve certain goals, with unit-layer elements mostly related to functions such as task progress, production efficiency, and product accuracy. The part-layer is the smallest task execution unit, containing important movable parts related to equipment function that usually require monitoring or control, such as servo motors and spindles. Part-layer elements concentrate on key geometric and physical attributes like size, position coordinates, operating status, and static or dynamic attributes. The subpart-layer can include entities below part size that constitute the part, such as gears to the spindle, or microscopic effects occurring on the part, such as deformation, stress, and fluid dynamics. Therefore, modeling elements and data of the entire complex digital twin are selectively expressed at different layers, with required data obtained through sensors, equipment controllers, or additional edge controllers.

Context: Even at the same layer, digital twins cover multiple contexts, making multi-scenario digital twin construction challenging. Therefore, multi-context division is essential beyond different scale division. After layer division, digital twins may still contain many physical entities, and even specific physical entities may have many application contexts. With context division, digital twins can be simplified accordingly. In a specific simple digital twin, usually only a single or limited context is considered, and parameters not belonging to the current context can be hidden or treated as constants, simplifying the mathematical expression of mechanism models. After context division, a digital twin focusing on a single scale and single context becomes many simple digital twins. For each simple digital twin, specific implementation steps can be determined, in-

cluding input data and output results, structure, functional composition, and implementation flow.

Figure 4. Division of Different Scales and Contexts

The division of digital twins in scales and contexts is shown in Figure 4. This two-tier division method obviously transforms a complex digital twin into many simple models, enabling complex digital twins to have certain scalability. It allows different elements and data to be expressed at corresponding layers and contexts, providing insight at different scales. However, dividing a complex digital twin into many simple digital twins has disadvantages. Physical entities such as equipment, materials, and persons are individuals containing multi-dimensional information, but division into different scales and contexts separates them into many discrete parts, each recording only partial information and unable to represent the complete physical entity. Specifically, although different scale division allows data expression across four layers, it also separates attributes of some cross-scale physical entities into different layers. Additionally, after multi-context division, since only parameters of a single context are considered, mechanism model accuracy is reduced. Section 4 will discuss solutions to these problems. The following continues discussing simple digital twin implementation.

3.2 Constructing Simple Digital Twins in Component and Code

After dividing the complex digital twin, the next step is constructing simple digital twins. Digital twin development and implementation methods are diverse, with different developers using various software and platforms, leading to poor portability and scalability. However, many similar tasks exist when constructing digital twins for different application contexts. Considering that software is the carrier of digital twins, program and code reuse is also important. To reduce repetitive development work, similar tasks should be abstracted into several independent, standardized components according to function, constructing a digital twin component library that can effectively improve portability and scalability.

Figure 5. Classification of Components

Component: As shown in Figure 5, digital twin components can be divided into basic components, visualization components, and service components according to function. Basic components are mainly data-related, such as database operations (including reading and writing) and data preprocessing. Visualization components build human-computer interaction interfaces and visualize interface content. Service components analyze real-time collected data based on mechanism models and data models, realizing simulation, prediction, and optimization functions.

Although component types vary, they are basically organized according to two structures, as shown in Figure 6. The first structure, shown in Figure 6(a), is

data operation-centered, including "data input–data processing–data output," where data processing is the core of required functions. Data input sources include raw data in databases and outputs from other components. Data output includes other components in virtual space and entities in physical space. The second structure, shown in Figure 6(b), is digital twin modeling-centered, including "geometry–physics–behavior–rule." Geometry represents CAD models of physical entities. Physics includes physical attributes that can be simulated by software such as ANSYS. Behavior is the response of physical entities under internal and external factors, constructable through Markov chains, neural networks, etc. Rules are mined from data, often generated through machine learning or other algorithms.

Figure 6. The Structure of the Component

Component interfaces for changeable parameters should be reserved in advance to enable parameter modification. Component interfaces mainly include name, ID, IP, composition, context, data input, data output, other changeable parameters, and description. Name generalizes component function. ID provides unique identification in the digital twin system. IP enables communication with equipment within the network. Composition and context represent the layer and scenario of the simple digital twin in the 4C architecture, respectively. Data input and data output are interfaces for component interaction. Significantly, components have other changeable parameters, such as data acquisition frequency for data acquisition components and data dimension for data analysis components, which also require interfaces. Additionally, an interface is reserved for recording detailed component function descriptions, which can be used to analyze component correlations through artificial intelligence technologies such as text recognition in the future.

Code: Code is the final step in developing simple digital twins. Component development processes usually depend on multiple programming languages or platforms, so code in a broad sense includes not only programming language codes but also development platforms and software tools. Appropriate programming languages or platforms are chosen according to actual requirements. Based on the component structures described above, code helps develop component functions such as data input, data processing, data output, or geometric shape, physical attributes, relationship rules, and behavior, finally encapsulating component interfaces.

Componentization of functions enables complex digital twins to have certain expansibility. In the 4C architecture, components are the smallest functional units making up simple digital twins, so digital twin evolution and updates are achieved through component creation, modification, and deletion, with detailed operational processes shown in Figure 7. When creating a new digital twin, according to functional analysis and component division results, determine sequentially whether components realizing the function exist in the component library. If a component exists, it can be used directly. If not, the component should be developed and synchronized to the component library. Finally, digi-

tal twin functions are realized by connecting different component types. When modifying a digital twin, offline or online functional testing of components is required. Only components passing tests can replace original components. Simultaneously, modified components need updating to the component library, distinguished by adding version numbers and descriptions. When deleting a digital twin, because functional dependencies may exist between components, component correlation analysis is necessary to ensure component deletion will not affect system function.

Figure 7. Digital Twin Creation, Modification, and Deletion Process

4 Assembling Simple Digital Twins into a Complex Digital Twin

Simple digital twins have been divided into specific scales and contexts in the above analysis. Assembling the complex digital twin requires considering information fusion, multi-scale association, and multi-context interaction. The ontology model provides a computer-understandable semantic description that can serve as a container to fuse related multi-dimensional information. Knowledge graphs build semantic relation networks between nodes, making them effective tools for describing scale associations. Multi-context interaction can be realized through behavior analysis and iteration of related simple digital twins, though specific interaction and iteration modes need selection according to actual situations.

4.1 Information Fusion Based on Ontology Model

In Section 3, complex digital twin data could be placed into different scales, for example in four layers: system, unit, part, and subpart. Then it is divided into different scenarios according to context. After division, the same physical entity may exist in several application contexts simultaneously, with physical entity attributes scattered across many simple digital twins. However, in complex context applications, simple digital twin data and attributes need integration into a whole database to achieve high-fidelity mapping of physical entities. Therefore, a container that can fuse information from simple digital twins at different scales and contexts is needed.

Ontology is commonly used in computer science as a conceptual model describing individuals (instances), classes (concepts), attributes, and relations. The ontology model provides a data structure describing physical entities for information sharing and fusion. Extensible Markup Language, Resource Description Framework, and Web Ontology Language are common ontology languages. Protégé, XML Editor, and other development tools can build ontology models.

Figure 8. Data Composition of Ontology Model

The structure of a complex digital twin ontology model is shown in Figure 8. The ontology-based data model may come from several simple digital twins.

Data composition can be categorized into three types: basic attributes, technical attributes, and state attributes. Basic attributes provide general physical entity descriptions such as name, ID, and serial number. Technical attributes describe physical structure and technical performance including geometric size, weight, and load. Basic and technical attributes constitute static data in the ontology model, usually requiring manual entry. State attributes primarily relate to equipment real-time data, representing dynamic ontology model data such as overall operating status, position, angle, and speed. Some data can be automatically obtained by acquisition software, while others may require manual entry.

A complex digital twin ontology model contains attributes from all related simple digital twins, which can be extensible or modifiable. During information fusion, some data is redundant. For example, even across different simple digital twins, the same equipment' s name and ID should be identical, requiring only one recording. Conversely, data unique to each simple digital twin needs recording in the complex digital twin ontology model. When simple digital twin parameters change or new ones are created, ontology model data can be updated accordingly.

Figure 9. Information Fusion of the Simple Digital Twins

The information fusion process based on ontology is shown in Figure 9. However, the ontology model after information fusion has too many parameters, and updating all parameters in real-time consumes substantial computing resources, while not all parameters require real-time updates. To solve this problem, static and dynamic data are updated differently. Static data is initialized when the digital twin is created, while dynamic data is updated in real-time during digital twin operation. This approach can both meet application requirements and reduce software load. Additionally, these dynamic attributes should be modifiable to adapt to new requirements in new application contexts.

4.2 Multi-Scale Association Based on Knowledge Graph

Knowledge graphs were first proposed by Google in 2012 for information retrieval in Google Search, improving information retrieval efficiency and quality through semantic retrieval of target information. Lately, they have been widely used in natural language processing, intelligence analysis, recommendation systems, and other areas. The essence of a knowledge graph is a semantic network revealing relationships between physical entities. Knowledge graph construction usually requires multiple knowledge-related techniques such as knowledge extraction and knowledge fusion. Triples are general knowledge graph representations including two basic forms: "Entity-Attribute-Value" and "Entity-Relation-Entity."

In knowledge graph graphical representation, nodes and lines describe these triples. Nodes represent physical entities or attribute values, connected by lines (edges) representing attributes or various semantic relations.

Complex digital twins should be able to display entities at different scales as needed, with models, data, and behavior adjusting accordingly as perspective changes. Knowledge graphs can achieve this function. In Figure 10, the knowledge graph structure can be considered in two categories: class and instance. Class includes node type, subtype, and subordination. Type is consistent with digital twin layers such as system, unit, part, and subpart. Subtype details objects such as spindle, machine tool, and production unit. In subordination, a parent node usually corresponds to several child nodes, while a child node corresponds to only one parent node, so parent nodes are recorded to represent subordination.

Instance refers to specific objects in physical space, with two categories: attributes and relations. Attributes include ID, size, weight, and other basic object attributes. Relations contain IDs of nodes associated with the current node and descriptions of relations between these nodes. In this structure, class and instance attributes are supplied in the ontology model, then relations of these simple digital twins can be determined according to the knowledge graph.

Figure 10. The Data Structure of the Knowledge Graph

Figure 11 shows a knowledge graph example in a complex digital twin at different scales. The class structure records the digital twin scale and its parent node, so a knowledge graph can bridge digital twins at different scales. Additionally, complex digital twin displayed data can be selected according to the following strategy: if all child nodes of a node are included in the scenario, attributes and relations of these child nodes are ignored. Conversely, if a node has only part of its child nodes included in the scenario, then attributes and relations of the parent node are ignored.

Figure 11. The Scale Association in a Knowledge Graph

4.3 Multi-Context Interaction Based on Behavior Analysis and Iteration

Actual physical processes usually result from multi-context interactions. In complex digital twin systems, context interaction can make virtual entities more truly reflect physical entity status. As shown in Figure 12, multi-context digital twin interaction can be summarized into three types: superposition, transmission, and iteration. Superposition is direct addition of digital twin outputs. For independent digital twins without interaction, context interaction is direct addition of output results. Transmission uses one digital twin' s output as another' s input, suitable for one-way calculation. For example, if context A' s calculation needs context B' s output, context B is calculated first, then its result is used as context A' s input to obtain the final result. Iteration means two digital twins' outputs affect each other. Contexts A and B interrelate, suitable for two-way interaction. In this case, context A' s output affects context B, and context B' s output also affects A. When two contexts interact, context A' s output is calculated first and used as context B' s input, then context B' s output is used

as context A' s input. This process repeats until final results converge.

Figure 12. The Interaction of Multi-Context Digital Twins

The key to realizing multi-context simulation of complex digital twins is finding associated intermediate variables of digital twins. List digital twin inputs, outputs, and all service component outputs, then compare and analyze variables from different digital twins to find associated intermediate variables. If associated intermediate variables are absent, superposition is selected to update both digital twins' outputs. If one digital twin' s output is another' s intermediate variable, transmission is selected to update digital twin outputs. If each digital twin' s output is the other' s intermediate variable, interactions are realized through iteration. To prevent digital twin updates from falling into infinite loops, iteration termination conditions must be set. For instance, termination conditions could be adjacent output error less than 0.01 (1%) or iteration count exceeding a certain number.

5 Conclusion

To establish complex digital twins in multi-scale, multi-dimensional information, and multi-scenarios, this article proposes a novel modeling method.

The complex digital twin is first divided into several achievable simple digital twins through composition, context, component, and code in the 4C architecture. Then simple digital twins are assembled to construct a complex digital twin through information fusion, multi-scale association, and multi-context interaction. The article' s key results can be summarized as follows:

1) Four important characteristics of complex digital twins were discussed. Complex digital twin connotation must satisfy scalability, interoperability, expansibility, and fidelity. Complex digital twin requirements include: ability to express information at different scales, interaction between simple digital twins in the system, easy creation or replacement of digital twin components, and remaining as close as possible to physical entities.

2) The first step divides a complex digital twin into several achievable simple digital twins. The 4C modeling architecture (Composition-Context-Component-Code) is proposed to divide complex digital twins. Composition constrains digital twin elements within limited scales. Context allows digital twins to focus on specific behaviors and outputs. This division transforms complex systems with multiple scales and contexts into several implementable simple digital twins. Furthermore, components are constructed based on standard modularization and reusability, enabling rapid digital twin construction. Subsequently, code can be developed based on standard interfaces to realize simple digital twins.

3) Second, ontology, knowledge graph, and multi-context behavior analysis are introduced to assemble simple digital twins into complex digital twins. The ontology model integrates basic, technical, and status attributes from

simple digital twins. Effective data update methods are necessary for information fusion, treating static and dynamic data differently: static data is updated only during digital twin initialization, while dynamic data is updated in real-time during digital twin operation. The knowledge graph records necessary physical entity attributes and sets relations between simple digital twins, enabling complex digital twin expression adjustment according to specific contexts. Additionally, three types of interaction between digital twins are studied, which can improve complex digital twin behavior accuracy based on intermediate variable existence.

The modeling method proposed in this paper provides reasonable methodology and operable guidance for complex digital twin modeling in multi-scale and multi-dimensional information fusion and multi-context scenarios. Implementation processing methods for complex digital twins will be explained in Part II, also providing support for prediction and optimization based on complex digital twins.

Acknowledgment

This work is supported by the National Natural Science Foundation of China (No. 52175456), the Central Universities the Fundamental Research Funds for NSAF U1830110 and ZYGX2019J032.

References

[1] M. Grieves, Digital twin: manufacturing excellence through virtual factory replication, White paper, Melbourne: US Florida Technology, 2014.
[2] E. Glaessgen and D. Stargel, "The digital twin paradigm for future NASA and U.S. Air Force vehicles," in Proc. 53rd IAA/ASME/ASCE/AHS/ASC Struct. Struct. Dyn. Mater. Conf., 2012. [Online]. Available: https://arc.aiaa.org/doi/pdf/10.2514/6.2012-1818.
[3] D. Liu, H. Huang, B. Wang, T. Zhou, S. Luo, Operation paradigm for remanufacturing shop-floor based on digital twin, Computer Integrated Manufacturing Systems 25(06) (2019) 1515-
[4] M. Grieves, J. Vickers. Digital twin: mitigating unpredictable, undesirable emergent behavior in complex system//Trans-disciplinary Perspectives on Complex Systems. Berlin, Germany: Springer-Verlag, 2017
[5] F. Tao, M. Zhang, Digital Twin Shop-Floor: A New Shop-Floor Paradigm Towards Smart Manufacturing, IEEE Access 5 (2017) 20418-20427.
[6] C. Zhuang, J. Liu, H. Xiong, X. Ding, S. Liu, G. Weng, Connection, architecture and trends of product digital twin, Computer Integrated Manufacturing Systems 23(04) (2017) 753-768.
[7] K.M. Alam, A. El Saddik, C2PS: A Digital Twin Architecture Reference Model for the Cloud-Based Cyber-Physical Systems, IEEE Access, 5 (2017) 2050-2062.
[8] D. Guo, J. Bao, G. Shi, Q. Wan, X. Sun, H. Weng, Research on Modeling

of Aerospace Structural Parts Manufacturing Workshop Based on Digital Twin, Journal of Donghua University (Natural Science) 44(04) (2018) 578-585+607.
[9] J. Guo, H. Hong, K. Zhong, X. Liu, Y. Guo, Production Management and Control Method of Aerospace Manufacturing Workshops Based on Digital Twin, China Mechanical Engineering 31(07) (2020) 808-814.
[10] R. Soderberg, K. Warmefjord, J.S. Carlson, L. Lindkvist, Toward a Digital Twin for real-time geometry assurance in individualized production, Cirp Annals-Manufacturing Technology 66(1) (2017) 137-140.
[11] C. Zhuang, J. Liu, H. Xiong, Digital twin-based smart production management and control framework for the complex product assembly shop-floor, International Journal of Advanced Manufacturing Technology 96(1-4) (2018) 1149-1163.
[12] B. Bjornsson, C. Borrebaeck, N. Elander, T. Gasslander, D.R. Gawel, M. Gustafsson, R. Jornsten, E.J. Lee, X. Li, S. Lilja, D. Martinez-Enguita, A. Matussek, P. Sandstrom, S. Schafer, M. Stenmarker, X.F. Sun, O. Sysoev, H. Zhang, M. Benson, C. Swedish Digital Twin, Digital twins to personalize medicine, Genome Medicine 12(1) (2019)
[13] K. Bruynseels, F.S. de Sio, J. van den Hoven, Digital Twins in Health Care: Ethical Implications of an Emerging Engineering Paradigm, Frontiers in Genetics 9 (2018).
[14] P. Xu, W. Chen, L. Liao, Z. Zhang, Y. Feng, Research on digital workshop of ship pipe machining based on digital twin, Ship Science and Technology 41(15) (2019) 139-144.
[15] F. Dembski, U. Woessner, M. Letzgus, M. Ruddat, C. Yamu, Urban Digital Twins for Smart Cities and Citizens: The Case Study of Herrenberg, Germany, Sustainability 12(6) (2020).
[16] C. Fan, C. Zhang, A. Yahja, A. Mostafavi, Disaster City Digital Twin: A vision for integrating artificial and human intelligence for disaster management, International Journal of Information Management 56 (2021).
[17] H.R. Zhao, J.H. Liu, H. Xiong, C.B. Zhuang, T. Miao, J.S. Liu, B. Wang. 3D visualization real-time monitoring method for digital twin workshop. Computer Integrated Manufacturing System, 2019, 25(6): 1432-1443.
[18] T.G. Ritto, F.A. Rochinha, Digital twin, physics-based model, and machine learning applied to damage detection in structures, Mechanical Systems and Signal Processing, 155 (2021).
[19] Q. Lv, R. Zhang, X. Sun, Y. Lu, J. Bao, A digital twin-driven human-robot collaborative assembly approach in the wake of COVID-19, J. Manuf. Syst. 60 (2021) 837-851.
[20] X.N. Lai, S. Wang, Z.G. Guo, C. Zhang, W. Sun, X.G. Song, Designing a Shape-Performance Integrated Digital Twin Based on Multiple Models and Dynamic Data: A Boom Crane Example, Journal of Mechanical Design, 143 (2021)
[21] R. Rosen, G. von Wichert, G. Lo, K.D. Bettenhausen, About The Importance of Autonomy and Digital Twins for the Future of Manufacturing, Ifac Papersonline 48(3) (2015) 567-572.
[22] S.L. Wang, Y.K. Wang, B. Yang, S.B. Wang, Fog manufacturing: New

paradigm of industrial internet manufacturing based on hierarchical digital twin. Computer Integrated Manufacturing Systems, 2019, 25(12): 3070-3080
[23] Y.H. Pan, T. Qu, N.Q. Wu, M. Khalgui, G.Q. Huang, Digital Twin Based Real-time Production Logistics Synchronization System in a Multi-level Computing Architecture, J. Manuf. Syst., 58 (2021) 246-260.
[24] Q. Bao, G. Zhao, Y. Yu, S. Dai, W. Wang, The ontology-based modeling and evolution of digital twin for assembly workshop, International Journal of Advanced Manufacturing Technology (2021) 1-17.
[25] S. Dai, G. Zhao, Y. Yu, P. Zheng, Q. Bao, W. Wang, Ontology-based information modeling method for digital twin creation of as-fabricated machining parts, Robotics and Computer-Integrated Manufacturing 72 (2021).
[26] S. Singh, E. Shehab, N. Higgins, K. Fowler, D. Reynolds, J.A. Erkoyuncu, P. Gadd, Data management for developing digital twin ontology model, Proceedings of the Institution of Mechanical Engineers Part B-Journal of Engineering Manufacture, (2020).
[27] Y.Q. Lu, X. Xu, Resource virtualization: A core technology for developing cyber-physical production systems, J. Manuf. Syst., 47 (2018) 128-140
[28] S. Liu, Y. Lu, J. Li, D. Song, X. Sun, J. Bao, Multi-scale evolution mechanism and knowledge construction of a digital twin mimic model, Robotics and Computer-Integrated Manufacturing 71 (2021).
[29] X. Sun, R. Zhang, S. Liu, Q. Lv, J. Bao, J. Li, A digital twin-driven human-robot collaborative assembly-commissioning method for complex products, International Journal of Advanced Manufacturing Technology, (2021).
[30] B. Schleich, N. Anwer, L. Mathieu, S. Wartzack, Shaping the digital twin for design and production engineering, CIRP Ann-Manuf. Technol. 66(1) (2017) 141-144.

Note: Figure translations are in progress. See original paper for figures.