Research and Implementation of Digital Twin Heterogeneous Data Exchange System

Peng Guqi\textsuperscript{1, 2, a}, Zhong Hengen\textsuperscript{1, b}

\textsuperscript{1}Technology and Engineering Center for Space Utilization, Chinese Academy of Sciences
\textsuperscript{2}University of Chinese Academy of Sciences, Beijing, China
\textsuperscript{a}pengguqi17@csu.ac.cn
\textsuperscript{b}zhonghengen@csu.ac.cn

Abstract—Digital twin is one of the most important technologies to realize system intelligence and automation, and it seamlessly integrates the digital and physical space. So far, the concept of digital twin has been proposed for nearly 15 years, and the importance of digital twin is increasingly recognized by academia and industry, with a large number of digital twin models applied in different industries. The interoperability of different models has become a major challenge for digital twins. In this paper, we propose a system for heterogeneous data exchange of different models, and demonstrates the data interaction prototype of the mechanical and thermal models of the electric box load parts, finally we evaluated the effectiveness of our system.

1. INTRODUCTION
Professor Michael Grives introduce the concept of "representing real physical entities equivalently with virtual digital models" in 2002, and proposed the term "digital twins" in 2011. A product, factory, or business service have a corresponding digital twin model, as shown in Figure 1.[1] Through continuous application development and refinement, the current digital twin technology mainly emphasizes two connotations, one is the mapping of physical entity models in the virtual cyberspace, and the other is the establishment of real-time data connections between models through sensors etc.. Through the interaction of virtual and physical models, the digital twin can monitor the system (or process) in real time to avoid risks before problems occur, or perform preventive maintenance to ensure system stability, while effectively reducing system validation and testing costs, serving the design and research of future products, and simplifying collaboration and unified data management among multiple systems. Today, digital twin technology has become an integral part of the digital engineering program, which will enable the connection of increasingly interconnected smart physical devices in the digital world in the future.[2]

In order to realize that the virtual system model changes in real time as the state of the physical system changes, the digital twin needs to collect and interact with the physical system to update the model.[3] The physical environment includes physical systems, sensors, communication interfaces, etc. The virtual model in the digital twin is updated by providing the operations and data associated with the physical system to the virtual environment. Therefore, the digital twin model can represent the latest physical system and accurately reflect the operating environment of the physical system. The digital
twin can also simulate tracking the state of each physical system, analytically detecting anomalous behavior and identifying new problems in advance.[4]

Digital twin models can be composed of different types of models and data, while industry-standard models are not well defined. Achieving interoperability between different system models is a major challenge for digital twin development. A homogeneous digital model (e.g. an electrical box part) is relatively easy to create, but a more complex system needs to integrate digital models from different systems to comprise a digital twin of a physical entity. This paper proposes a system model for heterogeneous data exchange, providing new ideas for building composite digital twin models, compatible with different types of models and data that exist in digital twin models, and facilitating different systems to transfer data, process data, and exchange data.

The structure of this paper is as follows: the following section introduces the theoretical foundations and related work. In Section 3, we present a heterogeneous data exchange system and explain how it works. In Section 4, we show an example that demonstrate how to implement a simple prototype through the system. Finally, in Section 5 we summarize this paper.

Figure 1. Digital twins extend in environments such as product design and manufacturing service

2. RELATED WORK
Digital twins mainly include the following related topic:

- **Digital twin model establishment**: The digital twin model of a physical product usually contains many stages and different types of models in its life cycle, such as system model, functional model, performance model, multi-physics model, manufacturing model and usage model. This requires the establishment of a global identification of the product to digitally represent each physical product, which can provide a unique identification for any physical object in any system and its entire lifecycle.[5]

- **Twin data**: Physical products will generate large amounts of data throughout their life-cycle for different scenarios and stages, including product physical entity data models, virtual models, service data, expert knowledge in various fields, historical data, etc.

- **Data management**: Product data and information are created and developed at different stages of the product life cycle. After these stages, product will generate more data, which also leads to the needs of statistical summary analysis primevally and management in the later period.[6]
- **Model data interaction:** The digital twin model can be composed of data models stored in different databases. In order to make comprehensive use of model information and make the model fully reflect the state of physical entities. It needs to unify the communication among all systems and make a standard communication protocol and format. [7]

- **Digital twin applications:** The digital twin model can provide services for users. It needs to be able to quickly retrieve and display the necessary information from different data models, and at the same time facilitate communication and interoperability between different systems. This means that the digital twin of software application platforms needs to take into account of collaboration of multiple systems.

From the above work, it can be seen that the interaction of model data is the key of digital twin models. This paper proposes a heterogeneous data exchange system to solve the problem of data interaction between different system models. Moreover, we introduce some of the previous work of data storage and exchange. Object-oriented methods are used to model and store data, but data exchange is mainly through point-to-point communications. In addition, there is also the use of ontologies as a general semantic basis for data modeling. [8,9] There have also been some works that defined the Standard for the Exchange of Product Model Data (STEP),[10] which defined product data of its lifecycle in the ISO 10303 specification. STEP is widely used in computer-aided design (CAD) and product data/lifecycle management (PDM/PLM) systems, but STEP is primarily used to store product information and cannot be used to share data between multiple systems.

A new method recently available is the Computer Aided Engineering Exchange (CAEX).[11] It is an XML-based object-oriented data architecture that describes real or logical objects in the form of data objects. Automation Markup Language (AML) uses CAEX as a framework to define a metamodel for engineering model storage and exchange, and implements an object-oriented physical and logical data storage format.[12] It includes four concepts: role class, interface class, system unit class and instance level. As shown in Fig.2.[13] It has been used to model industrial information and as a data exchange format in the production system engineering chain. The XML-based architecture makes it highly compatible and scalable. However, the modeling methods and application of models based on commercial software is already very mature, such as ANSYS, ADAMS, MATLAB / Simulink. The compatibility of interfaces between different software cannot be ignored.

![Overview of Automation Markup Language AutomationML](image)

**Figure 2. Overview of Automation Markup Language AutomationML**

### 3. **System architecture**

The purpose of this paper is to provide a systematic approach that enables data exchange between models built under different software. The main idea is, firstly, to model the physical components, then transform the model data under certain processes and rules, so that the original system models can be used to exchange information with each other.

First use the modeling tool to create a physical device model, which is used to represent the specific detailed composition of the system and the design architecture of the system.[14,15] Currently, we can not only model 3D entities, but also simulate the physical process of design in detail, and combine with
the model operation process to make the model more accurate than before. There are several program applications available for these simulations, such as Unreal Engine and Unity 3D for simulation visualization, CAD programs and SolidWorks for building physical models, dymola for Dassault Systems and Simulink for building mathematical models.

After the model is defined, the model will be used by other systems to extract information and allow different models to exchange information on modeling attributes. For each model involved, an interface (adapter) is required to exchange data between models. If you use a one-to-one model data exchange method, then it is necessary to maintain 12 adapters between the four systems to enable the four different models to communicate with each other.

A digital twin model needs to integrate multi-disciplinary, multi-physics, multi-scale, and multi-probability simulation processes, which leads to complex data exchange interfaces, and even more, if you want to implement \( f_{A \rightarrow B}(x) \) you may need to splice adapters \( f_{A \rightarrow C}(x) \) and \( f_{C \rightarrow B}(x) \), which are more complicated in specific engineering implementation. Data distortion, metadata loss and other problems will occur during the conversion. Therefore, to maintain maximum compatibility, the platform will adopt an intermediate data format, and our data exchange platform is to find a universal adapter, so that \( r \) represents the conversion of the format of the payload model under system A to the platform standard format mapping function under the digital twin framework, and \( f_{D \rightarrow B}(x) \) represents the conversion from the intermediate standard format to the target system B. The exchange model can be used as a specification for data transfer, and can also be used to define a data exchange mode. The source system must convert their data into this exchangeable format. Similarly, the target system must use the exchange format to implement their internal data structures. Therefore, the data exchange format acts as a link between different systems. As shown in Fig.3.

![Figure 3. Model-based implementation of heterogeneous data exchange under the digital twin concept](image)

Assuming that there is data interaction between System A and System B, the complete data exchange process is as follows:

a) System A and System B register with the data exchange platform center and submit \( \text{format}_A(\text{data}) \) and \( \text{format}_B(\text{data}) \) respectively, that is, they are used to generate the data schema;

b) According to the actual needs, establish a standardized data structure for the registered model data type, that is, the standard schema of the data;

c) According to the schema of both systems, generate conversion rules \( f_{A \rightarrow C}(x) \) and \( f_{C \rightarrow B}(x) \).

d) Receive the data submitted by the system A to the data exchange platform center —— \( \text{format}_A(\text{data}) \);

e) Verify that the format of \( \text{format}_A(\text{data}) \) is correct and the content is integrality;

f) According to the conversion rules, convert the platform standard format data into a format recognized by System B, that is, \( f_{A \rightarrow B}(x) \) —— \( \text{format}_B(\text{data}) \);

g) Send the generated data —— \( \text{format}_B(\text{data}) \) to System B.

There is a big difference between data exchange model O and system data models A and B. First the data exchange model is temporary, while the system data model is persistent. In addition, the purpose of
the data exchange model is to achieve real-time interaction of data, so it is usually more abstract and smaller in size. Their purpose is to transfer data. Unlike traditional model conversion, there is no need to ensure that all details of the model are lossless. However, system data is to make the business data in the system operational and directly reflect the business attributes. System data models are usually large (often with several hundred attributes or more), because they need to describe the system characteristics as accurately as possible. This means that for digital twin model, the advantages of models or software in specialized fields can be maintained, while at the same time, the necessary data information transfer can be completed. For the application of digital twin models, this approach not only reduces learning costs, but also allows for better collaboration of work under the digital twin.

In order to standardize the data transfer process, it is necessary to establish the mapping rule of each domain model to the data exchange model, so as to clarify the data attribute relationship between the models, and extract the model attributes and their data. Based on the above, considering that the data exchange model needs to meet the characteristics of multiple fields and multiple stages, AML is selected as the intermediate format for model data exchange, and the complete model data exchange process is designed as shown in the following Fig.4. Through the model element attribute mapping table, each system model is transformed into the AML model file. It is beneficial for us to realize the data interaction of the multi-domain model under the platform.

![Figure 4. Schematic diagram of the exchange process of model data elements](image)

**4. DESIGN AND IMPLEMENTATION**

Based on the above system architecture, we design a multi-system collaboration system for model interaction between different systems, and define a whole data exchange model and model mapping rules. In this paper, we use the AML conceptual model to demonstrate load model of power supply equipment in aerospace field. Due to space limitations, only a prototype tool to support the presented method and the part of developed software platform are shown.

The digital twin model involves multiple model data sources. It is expected to establish data interaction between the models. In this paper, we illustrate the process of establishing the correlation between the structure and thermal model of the power supply load with an example. We also implement the conversion of the domain model to the AML model and the data connections between the two models, thus breaking down the barriers between different system models. AML was introduced above to support the modeling of plant topology, component structure, geometry and kinematics, logical behavior, and communication networks. But AML itself does not provide formal semantics for automatic data interpretation. In practice, tools need to reach consensus on data and be responsible for semantic preservation.

As shown by the red arrows in the Fig.5, the AML conceptual model established by the system can generate the corresponding target system's model attributes class. The user first can leverage some modeling tools to generate the main AML conceptual model for the target concept, including role class, system unit class, interface class, internal elements or external interfaces to add attributes and child
elements with corresponding constraints to the model. For nested attributes and child elements, the process needs to be repeated recursively. Constraints of the nature of the model also need to generate its parent conceptual model. The reverse conversion is shown by the blue arrows in the figure, and for complex classes that already exist, these classes can be modeled by ontology experts or created using the AML editor, where the user can examine specific model attribute classes and modify them as needed, and convert the selected attribute class to an AML concept model through its AML concept tree.

![Model-based digital twin concept](image)

To demonstrate the example of the proposed method, we prototype a simplified model of the power supply electrical box load. Based on the above transformation relationship, first of all, the mapping rules of the mechanical mechanism model and thermal simulation model to the AML language should be abstracted, as shown in the table I. Through the establishment of mapping rules, the data attribute relationships between models can be clarified and the attributes of the model data can be extracted. This requires us to consider the interrelationships between various domain models when establishing mapping rules and AML file format.

| Data Element          | Example                                      | Mapping to AML          |
|-----------------------|----------------------------------------------|-------------------------|
| Hierarchy_level, Object | Heat Transfer Model Module, Heat Transfer System | Internal Element       |
| Property              | Management Attributes, Structural Attributes, Physical Attribute | Attribute               |
| Classification        | Part Model, Assembly Model                   | RoleClass               |
| Simple Relation       | Platform Assembly Relationship               | ExternalInterface with External Element for relation data |
| Directed Relation     | Cabinet Assembly Relationship                | ExternalInterface with Internal Element for relation data |
| Groups                | Board, Cold Plate, Cabinet, Electric Box     | SystemUnitClass         |

For a physical entity, to be able to effectively correspond to the same attribute in different fields, it is necessary to establish a corresponding attribute mapping relationship table in the back-end database, or establish a standard naming scheme. But no matter what method is adopted, we can encapsulate it as a mapping relationship between the model and AML element properties and complete the data exchange between heterogeneous systems.

We designed collaborative work software based on this to illustrate the feasibility of data interaction and the advantages of the digital twin model. The core interface is shown in the figure. In the digital twin framework, there is a need to maintain a single true source of model attributes and data. When one model design data changes, it can also cause multiple model attributes to change. Through this tool, a multi-domain model data relationship based on AML intermediate files is established. When a model data changes, it can also display the changes of the model data related to it, thereby ensuring the consistency of the data.
5. SUMMARY

This paper starts from the problem of data exchange of digital twin multi-domain models of aerospace payloads. Firstly, we briefly describe the relevant work content of establishing the digital twin model. On this basis, a model data exchange scheme based on AML intermediate format files is proposed, and a data exchange platform is designed. The establishment of the data exchange platform enables the maximum coordination between different heterogeneous systems and can expand the boundaries of existing systems, thereby making it possible for digital twin models to integrate multi-disciplinary, multi-physical, multi-scale, and multi-probability simulation processes, making products It can be organically combined in all aspects of each lifecycle. A brief description of the solution is also given with corresponding examples and a simple demonstration of the software platform developed. However, it should be noted that to achieve a unified data exchange platform, the formats generated by different tools need to be abstracted and unified. The models between different disciplines are very different and the conversion schemes are also inconsistent. How to build data exchange models simply and effectively, design reasonable model mapping rules, and ensure the integrity of the source system model data are issues that need to be further studied.

REFERENCES

[1] Bao J, Guo D, Li J, et al. The modelling and operations for the digital twin in the context of manufacturing[J]. Enterprise Information Systems, 2018:1-23.
[2] A. Kusiak, “Smart manufacturing must embrace bigdata,” Nature, vol. 544, no. 7648, pp. 23–25, 2017.
[3] Q. Qi and F. Tao, “Digital twin and big data towards smart manufacturing and industry 4.0: 360 degree comparison,” Ieee Access, vol.6, pp. 3585–3593, 2018.
[4] Talkhestani B A, Jazdi N, Schgl W, et al. A concept in synchronization of virtual production system with real factory based on anchor-point method[J]. Procedia CIRP, 2018, 67:13-17.
[5] Schroeder G N, Steinmetz C, Pereira C E, et al. Digital Twin Data Modeling with AutomationML and a Communication Methodology for Data Exchange[J]. IFAC-Papers On Line, 2016, 49(30):12-17.
[6] Tao F, Cheng J, Qi Q, et al. Digital twin-driven product design, manufacturing and service with big data[J]. International Journal of Advanced Manufacturing Technology, 2017.

[7] Moreno A, Velez G, Ardanza A, et al. Virtualisation process of a sheet metal punching machine within the Industry 4.0 vision[J]. International Journal on Interactive Design & Manufacturing, 2016, 11(2):1-9.

[8] R. Young, A. G. Gunendran, A.-F. Cutting-Decelle, and M. Gruninger, “Manufacturing knowledge sharing in plm: a progression towards the use of heavy weight ontologies,” International Journal of Production Research, vol. 45, no. 7, pp. 1505–1519, 2007.

[9] L. Patil, D. Dutta, and R. Sriman, “Ontology-based exchange of product data semantics,” IEEE Transactionson automation science and engineering, vol. 2, no. 3, pp. 213–225, 2005.

[10] M. J. Pratt, “Introduction to iso 10303—the step standard for product data exchange,” Journal of Computing and Information Science in Engineering, vol. 1, no. 1, pp.102–103, 2001.

[11] Danny P, Ferreira P, Lohse N. An AutomationML Model for Plug-and Produce Assembly Systems[C]// 2017 IEEE 15th International Conference on Industrial Informatics (INDIN). IEEE, 2017.

[12] Luder A, Schmidt N, Rosendahl R, et al. Integrating different information types within AutomationML[C]// Emerging Technology & Factory Automation. IEEE, 2015.

[13] Drath R, Luder A, Peschke J, et al. AutomationML - the glue for seamless automation engineering[C]// IEEE International Conference on Emerging Technologies & Factory Automation. IEEE, 2008.

[14] H. Gomaa, Software modeling and design: UML, use cases, patterns, and software architectures. Cambridge University Press, 2011.

[15] T. Stahl, M. Voelter, and K. Czarnecki, Model-driven software development: technology, engineering, management. John Wiley & Sons, Inc., 2006.