VTT Technical Research Centre of Finland

On the alleviation of imminent technical and business challenges of long-lasting functional digital twins

Zeb, Akhtar; Kortelainen, Juha; Rantala, Tero; Saunila, Minna; Ukko, Juhani

Published in:
Computers in Industry

DOI:
10.1016/j.compind.2022.103701

Published: 01/10/2022

Document Version
Publisher's final version

Computers in Industry 141 (2022) 103701

Contents lists available at ScienceDirect

Computers in Industry

journal homepage: www.elsevier.com/locate/compind

On the alleviation of imminent technical and business challenges of long-lasting functional digital twins

Akhtar Zeb[a,*], Juha Kortelainen[a], Tero Rantala[b], Minna Saunila[b], Juhani Ukko[b]

[a] VTT Technical Research Centre of Finland Ltd, P.O. Box 1000, FI-02044 VTT, Finland
[b] LUT University, P.O. Box 20, FI-53851 Lappeenranta, Finland

ARTICLE INFO

Article history:
Received 7 February 2022
Received in revised form 27 April 2022
Accepted 30 April 2022
Available online xxxx

Keywords:
Business risk
Data model
Digital twin
File format
Modeling
Simulation
Standard

ABSTRACT

In this article, we discuss the technical and business risks associated with long-lasting functional digital twins, and describe different strategies for their alleviation. Functional digital twins are based on physics-based simulation models and are operated alongside the life cycle of their physical counterparts. These simulation-based digital twins are built using a simulation software. The problems with most of the commercial modeling and simulation tools are their black box nature and storing data in protective formats, leading to poor interoperability. Since the digital twins of certain assets need to be operated for a long period, even for several decades, there is a possibility that the computing infrastructure, i.e., the computing hardware and software, may not remain the same throughout the product or system life cycle. The computer hardware and operating systems are usually third-party components with limited choices for their users, whereas the selection of simulation tools is more flexible and the designer can choose from, for example, commercial, open-source, or in-house solutions. To avoid substantial costs or business disruption, the digital twin providers must be able to reproduce the underlying simulation models with up-to-date tools and adopt alternative solutions whenever needed. The findings of the study are presented in the form of propositions throughout the article.

1. Introduction

The application of computational models in engineering design has proven its value in industry. Computer-aided design (CAD), computer-aided manufacturing (CAM), and computer-aided engineering (CAE) are nowadays the mainstream approaches in industry, and numerous software applications and software systems are available for them, both for general design and for dedicated design domains and purposes. The computational models already offer insights into the performance and operation of a product or system—that is, the "design target"—in the early phase of the design process and even before any realization of the design target Boschert and Rosen (2016). They also enable efficient concurrent engineering and a fast overall engineering design process Monticolo et al. (2015).

As depicted in Fig. 1, the latest development of the digital twin (DT) concept is extending the use of engineering design models, such as the computational fluid dynamics (CFD), finite element analysis (FEA) and system simulation models, to a longer part of the design target life cycle, enabling additional services Boschert and Rosen (2016); Grieves and Vickers (2017); Boschert et al. (2018); Lim et al. (2020); Wang (2020). Grieves and Vickers Grieves and Vickers (2017) described applications for DTs in all the main life cycle phases of a product. DTs can provide valuable information for the design and development of the product and, for example, reduce the need for physical prototypes. In the production phase, DTs provide collected information about the as-built product. In the operational phase, DTs can be used for monitoring and maintenance, and for optimizing the operation of the product. At the end of the life cycle, DTs can help in disposing the product in an efficient and safe manner.

In digital design and engineering, data exchange and data interoperability have been issues for a long time Wiesner et al. (2011); Xie et al. (2013); González et al. (2007); Frechette et al. (2011); Urban et al. (1993); Fowler (1995); Gallaher et al. (2004), and standardization has been one of the main solutions to tackle such challenges Eckert (2014). However, partially due to the nature of the development of digital engineering tools, it seems that standardization cannot solve all the issues and it is lagging behind the development of digital engineering tools. The modeling and simulation tools

* Corresponding author.
E-mail addresses: akhtar.zeb@vtt.fi (A. Zeb), juha.kortelainen@vtt.fi (J. Kortelainen), tero.rantala@lut.fi (T. Rantala), minna.saunila@lut.fi (M. Saunila), juhani.ukko@lut.fi (J. Ukko).

https://doi.org/10.1016/j.compind.2022.103701
0166-3615/

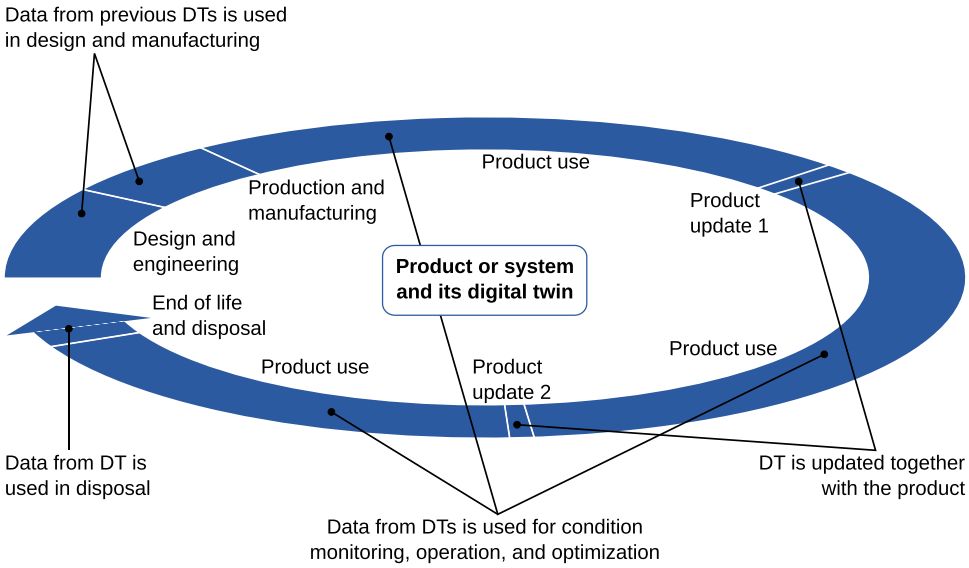

Fig. 1. Illustration of a product's life cycle, including two system upgrades.

constantly have new features and functionalities, which may not be supported by the standards. The standardization processes are also typically time consuming and expensive, thus the standards do not cover the latest developments and do not include all the details of the software solutions. In addition, software vendors' business reasons and simply lack of motivation to support existing standards and open-data specifications have been influencing the situation Wiesner et al. (2011); Schneider and Marquardt (2002); Gargiulo et al. (2014). Furthermore, commercial software production and business, like many other business fields, is balancing between new potential business opportunities and the effort and resources their implementation require. Additional features in software applications, such as standards compliance, require resources both when implemented and in the maintenance of the software, so there must be a clear need and business motivation for the additional features in the software. All these factors negatively influence the reuse of simulation information in organizations, both internally and externally, especially during subcontracting engineering design tasks.

DT simulation models may be data-driven models (e.g. based on machine learning) or simulation-based models (e.g. first principles-based simulation models); this work does not cover the former. The focus of this article is on the data management of physics-based simulation models—we call them "functional DT (FDT) models"—that serve as integral parts of the overall product or system together with the physical counterparts and need to be operational throughout their life cycles (see Fig. 1).

The management and reuse of the simulation model data of a DT during the product's or system's operational life cycle phase differ from the data management and reuse during the engineering design phase, both in the length of the data life cycle and the amount of data that is maintained. While in the engineering design phase the active time period of producing and using the data typically ranges from a few months to a couple of years, the operational time span of complex products or systems—such as aircraft, ships, defense systems, and process plants—can be as long as fifty years. Within a relatively short time span, changes in computer hardware technology, computer operating systems, and digital engineering software applications and systems do not usually need to be taken into account or the changes are manageable. On the other hand, when a DT is an integral part of the overall product or system and the expected life cycle for the system is several decades, the DT needs to be operational for a longer time than the technical design life cycle of the underlying systems (i.e., the computer hardware and operating system, and the software applications). In addition, engineering design phase models represent a whole fleet of products or systems of their kind, but by its nature, a DT is a digital representation of an individual product or system. This means that the number of DTs to be maintained is growing when new products or systems are delivered, and the overall amount of data to maintain can be remarkably larger than for engineering design models.

Many academic and industrial projects have focused on the long-term preservation of product data Brunsmann et al. (2012). However, less attention has been paid to the management of the data of DTs during their life cycles, both from technical and business points of view. The life cycle management of product data in general and DT data in particular share several common challenges. In this article, we focus on the challenges encountered in preserving the information and data needed to keep FDTs—that is, simulation models that are data connected to their real-world counterparts—operational for especially long life cycles. In addition, we study and illustrate possible strategies to alleviate the risks associated with the long-term simulation model data management of FDTs.

The article consists of several propositions and is organized as follows. In Section 2, we discuss the general concept of a DT and define an FDT, the modeling and simulation features of FDT, and the associated technical risks of long-lasting FDTs. In Section 3, some of the possible strategies for alleviating the technical risks of FDTs are discussed. In Section 4, the economic and business risks are discussed, and the means to prevent or minimize them are presented. Section 5 discusses the outcome of the work and summarizes the main findings.

2. DTs and their data life cycle management

2.1. The DT concept

2.1.1. Definitions

The concept of a DT has been actively developed in the past decade. Tuegel et al. (2011) illustrated a future DT being a detailed real-time digital model of the physical twin (i.e. the real-world system). In their vision, the DT of an airplane can simulate all the relevant phenomena, including material behavior from microstructure to the macro level, and fluid and structural dynamics. The DT is connected to the physical twin by the data measured from the physical system, and it can do prognostics of the physical twin. Grieves and Vickers Grieves and Vickers (2017) defined a "digital

twin prototype" as a DT of a prototypical physical product or a system, a "digital twin instance" as an individual DT of an individual product or system and represents it throughout the whole life cycle, and a "digital twin environment" as "an integrated, multi-domain physics application space for operating on Digital Twins for a variety of purposes." Such DTs have both interrogative and predictive capabilities, i.e., they are able to represent the current state of the physical twin and also predict the function and behavior of it. Similarly, Hartmann and Auweraer Hartmann et al. (2020) introduced the concept of an "executable digital twin" that, with its simulation tools, reuses the simulation models outside the system design and engineering tasks in the operation phase. Executable DTs would be used to integrate the simulation models originating from different parties and utilizing different types of computational solvers (system integration). However, many of the requirements (e.g., deployability, synchronicity, security, usability, reliability, and interactivity) for the realization of such DTs cannot be addressed yet and need further research.

Barricelli et al. (2019) have done a thorough study on the definition and main characteristics of DTs and the domains in which DTs and their technologies are actively developed. They define that DTs "are simulating, emulating, mirroring, or 'twinning' the life of a physical entity, which may be an object, a process, a human, or a human-related feature" and state that a DT has a single physical counterpart. The DT is more than a simulation model. It monitors, controls, and optimizes the function of its physical twin throughout its life cycle, i.e., the DT has descriptive, predictive, and prescriptive capabilities. Tao et al. (2019) illustrate that the main dimensions of the DT concept are the physical space, the virtual space, and the data and information connections between the two spaces. Their findings show that the definition of a DT is vague and depends on the context, purpose, and available technologies for its implementation, among other things.

The definition of a DT is changing, and alternative definitions are introduced regularly. In addition, the interpretations of the definitions are stretching the boundaries of earlier definitions. Some of the definitions are even unrealistically challenging while the marketing of digital solutions is especially using the ongoing hype of DTs without restraint, and in some cases, even a 3D model of the physical product or system is said to be a DT. Thus, we propose:

> Proposition: There is no clear consensus on a detailed definition of a DT, but the common elements of the definitions are the physical or real system, its digital representation, and a data connection between the two.

2.1.2. Applications

The application of a DT greatly affects the life cycle dependencies of the DT and, in fact, even the business that is done related to the product or system and its DT. When a DT is used for getting more information for engineering and design, the DT does not need to be an integral part of all the products or systems produced and delivered. The limited number of DTs can be managed, even though there may be changes in computing hardware, operating systems, and software. On the other hand, if the DT is an integral part of the overall system, the design and implementation of the DT should be done taking the possibly long life cycle into account.

From the DT application point of view, the definitions and categorization do not have much value. Definitions and categorization are important for communication, in order to both illustrate new ideas and concepts, and to emphasize the characteristic features of the DT types under discussion. For DT data life cycle management, a useful categorization is to group the implementation of DTs based on numerical simulation or other means. Another categorization criterion is the ability of a DT to describe the past and present state of the physical twin ("descriptive DT"); to predict the state, function, behavior, and dynamics of the physical twin ("predictive DT"); and to enable driving the physical twin to or towards a desired future state ("prescriptive DT").

In general, a descriptive DT would be based on the data gathered from the physical twin and the environment and on representing it in an informative way. This may include data analytics and other means of processing the data from the sensors and other sources into an intuitive form. A predictive DT would require the capabilities to estimate the future trajectories of the physical twin's function and behavior based on the current information of the system and already collected historical data. Such a DT could be composed of physics-based models (a priori knowledge of the behavior and dynamics of the target) or it could be purely based on the knowledge of behavior history and its data-predictive model (e.g., based on an artificial neural network model of the physical twin). A prescriptive DT would extend the predictive DT with optimization or some other means of defining the controls and parameters of the physical twin in such a way that the physical twin will reach the desired state in a given time or get as close to it as possible. In the last two types of DT, the fundamental element of the DT is the ability to predict the future state and behavior of the physical twin. This is inherently to do with physics-based simulation. Within this article, we refer to this kind of a DT as an FDT. Thus, we propose:

> Proposition: The value of the concept of a DT comes from its applications, not from the concept definitions. On the other hand, the definitions enable improved communication about the needs, features, and implementation of DTs.

2.2. FDTs

A DT may consist of several simulation models based on engineering data, operation data, and behavioral descriptions of the physical asset Boschert et al. (2018); Cameron et al. (2018). Fig. 2 shows the DT of an asset obtained by combining the IoT, machine learning, and data analytics with simulation models. The IoT platform brings together the physical asset data (e.g., sensor data, historical information, maintenance reports, asset and operator features), cloud computing, and big data analytics in order to extract insights and knowledge that support decision-making. The simulation platform of a DT contains the asset simulation models (e.g., FEA, CFD, and system simulation models), the simulation software for building and running the simulation models, and the computer system that operates the simulation software.

The simulation models used in DTs can be divided into being either data-driven models (e.g., machine learning models) or computational models (e.g., first principles simulation models) Erikstad (2017); Martínez et al. (2018); Liu et al. (2018). Data-driven DTs are based on black box models where the mathematical structures of the models are not explicitly available Pantelides and Renfro (2013). They are built to capture the relations between the input and output parameters of the asset and to predict its behavior or to detect anomalies to a certain extent. As these DT solutions are based on measured data from various sensors and the embedded systems of the physical counterpart, or data generated by other means, they cannot be used to predict the asset behavior that is not covered by the available collected data. Furthermore, since data-driven DTs mainly rely on data coming from the automation and monitoring systems of the physical asset, they cannot be used in situations where there is no access to the measured data (e.g., due to a

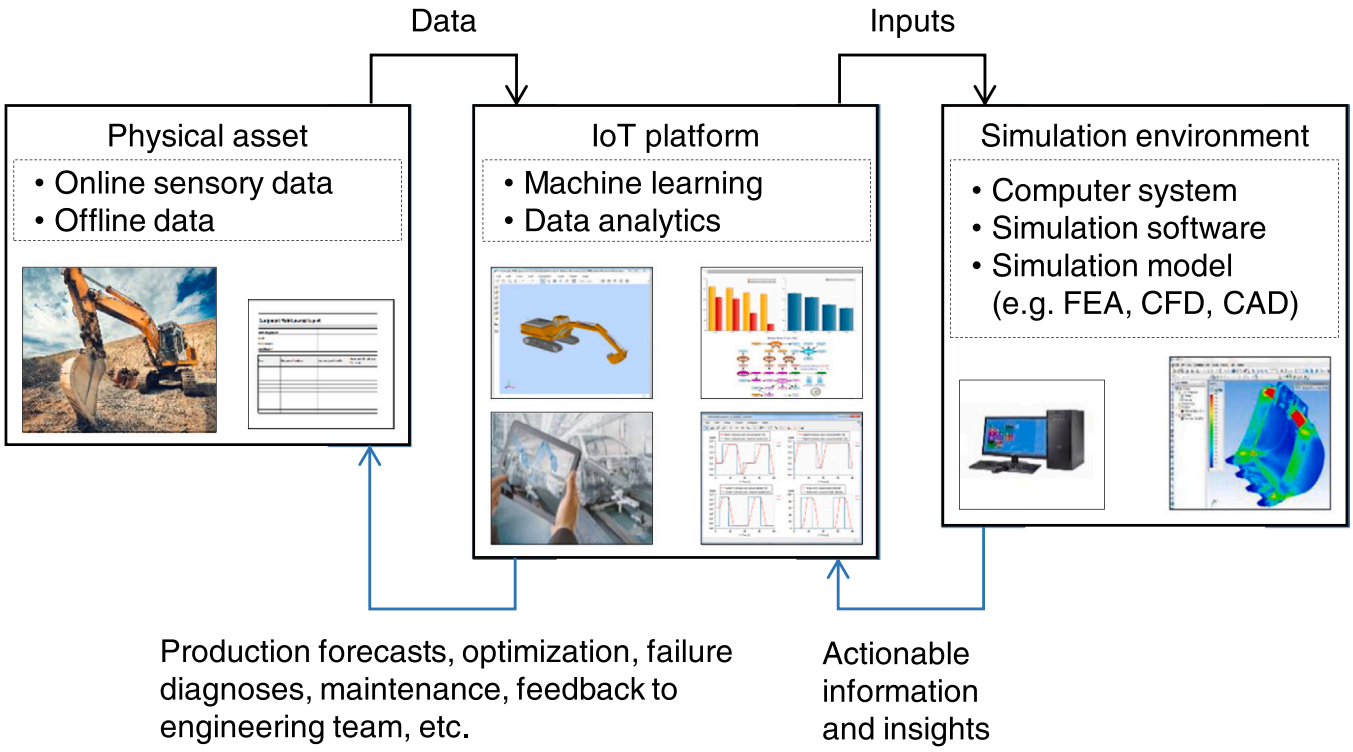

Fig. 2. DT combining the IoT, machine learning, and data analytics with simulation models.

malfunction in the data generation systems) or it is difficult to take measurements.

We define an FDT as being based on first principles simulation (physics-based simulation) and being able to predict the behavior of its physical counterpart, taking the target system and its parameters' initial values into account Pantelides and Renfro (2013); Waltrich et al. (2010); Schluse and Rossmann (2016); Magargle et al. (2017). In this case, the system's functional model can be disconnected from the physical asset and operated offline to provide inputs into the analytics systems for exploring the proposed operating conditions Magargle et al. (2017). In addition, in situations where the intended use of an asset may change, the existing functional models can be slightly modified to match the real asset conditions. These functional simulation models can be used to interpret the measurement data in different ways and, by generating different scenarios, several modes of failure can be simulated and applied for operation optimization or asset lifetime calculation Boschert and Rosen (2016); Boschert et al. (2018). Thus, we propose:

> Proposition: An FDT is based on physics-based simulation and the use of first principles models. The FDTs may be used for predicting the behavior of the real system.

The simulation models used in FDTs are similar to those used in, for example, engineering design. This means that the same general challenges with, for example, data exchange and model data management apply to FDTs as they cause problems with the models used in engineering and design.

2.3. Modeling and simulation in DTs

The field of simulation methods in engineering design is wide and contains several general methods, and numerous mathematical and numerical methods and techniques, tailored for different kinds of purposes. Even in the simulation of physical phenomena and systems, based on physical and mathematical models, the variety of methods and implementation approaches is wide. One categorization of physics-based simulation is to divide the simulation methods into the following: .

1. systemic simulation, representing systems and their overall function, performance, and dynamics;
2. continuum methods, representing continuous volumes of solids and fluids, and the phenomena involved;
3. other simulation methods, for example, methods based on probability distributions when applying Monte Carlo simulation.

The wide variety of applications and, on the other hand, simulation methods and techniques point out the main challenges that are often faced in engineering design and in the use of FDTs. These challenges are "How can the modeling and simulation data be maintained?" and "How can models between different modeling and simulation tools be exchanged?".

Physics-based simulation requires successful mastering of the physical phenomena involved in the target of the simulation—such as fluid dynamics, material physics, or mechanical dynamics—and further, the implementation of the physics in a computer software application. The stack of required knowledge and expertise to implement numerical physics-based simulation is shown in Fig. 3. Knowledge of physics lays the basis for modeling the relevant phenomena and enables reliable computational predictions of the function, behavior, and operation of the target. Running a computer simulation requires mathematical representation of the physics, such as the mathematical formulation of the equations for the conservation of mass, momentum, and energy in CFD or the relation between loads, internal forces, stress and strain, and material properties in structural analysis. The mathematical formulation of the underlying physics is often represented in the form of partial differential equations, which cannot usually be solved and integrated as they are in a simulation software application—some

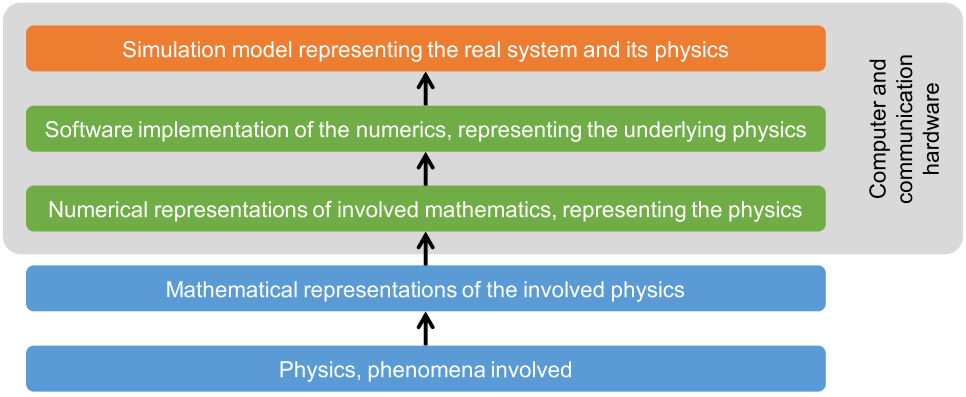

Fig. 3. The stack of required knowledge and expertise to implement numerical physics-based simulation.

numerical or approximative approach must be used. This is the numeric representation of the mathematical description of the underlying physics. To have a computer program that can run the simulation, the numerical representation must be implemented into a software application by using an appropriate software development approach and some programming language. Depending on the case, to reproduce the numerical simulation, redoing one or more of these layers is required. Quite often, the knowledge of the numerical representation of the physics is adequate to reproduce the simulation with comparative results. The layers from the numerical representation of the underlying physics in the simulation model representation stacked in Fig. 3 fall within the scope of this work. Thus, we propose:

> Proposition: The implementation of physics-based simulation requires different kind of expertise and contains several layers, the physics of the phenomena involved, the mathematical representation of the physics, the numerical representation of the mathematics, the software implementation of the numerics, and the model representation of the simulated system.

2.4. Long-lasting FDTs, technical risks, and challenges

FDTs, consisting of physics-based simulation models, rely on simulation software to produce additional information about the physical part. As shown in Fig. 4, the main building blocks of an FDT are the physical twin (the green box), the digital twin (the yellow box), and their data exchange (the black arrows). The FDT consists of the simulation software that contains the mathematical representation of the physics involved, such as the dynamics of rigid body systems or the stress-strain behavior of solid materials, a description of the details of the case in hand (i.e., the simulation model of the particular case), and the computing system, including the operating system elements and the computer hardware. In the industrial engineering design, computer hardware and operating systems are usually third-party components and either commercial or open-source solutions are used. With simulation software and depending on the application area, the solution can be commercial, open-source, or in-house software. These three main elements (i.e. computer hardware, computer operating system, and simulation software) are general and can be used for different kinds of DTs, i.e., they can be purchased or stocked and shared with many kinds of DTs. On the other hand, the simulation model is dedicated to one type of DT and may even be individual dependent, i.e., the simulation model is dedicated to one DT individual or instance.

The life cycle risks concerning these elements relate to the changes in technology in general, the availability of the elements, and changes in the construction of the physical twin. If the technology development in computing hardware continues to be fast, the base computing hardware may change so much that the current up-to-date technology is not valid after several decades. As an example, the workstation and desktop computers in the year 2000 were using 32 bit processors with only one computing core. The dominant operating systems in industry were Microsoft Windows 95, Windows NT, or proprietary UNIX systems (such as SunOS or Silicon Graphics IRIX). Since the year 2000, the mainstream Windows operating system has evolved from Windows NT 4.0 (released in 1996) to Windows XP (released in 2001), Windows Vista (released in 2006), Windows 7 (released in 2009), Windows 8 (released in 2012), Windows 8.1 (released in 2013), and Windows 10 (released in 2015) Wikipedia (2022). At the same time, the Linux operating system kernel has evolved from Version 2.2 to the current version, Version 5.6, and the number of Linux distributions and variants is now about 266 Anon (2021). A similar kind of development can be seen in simulation software. New companies and

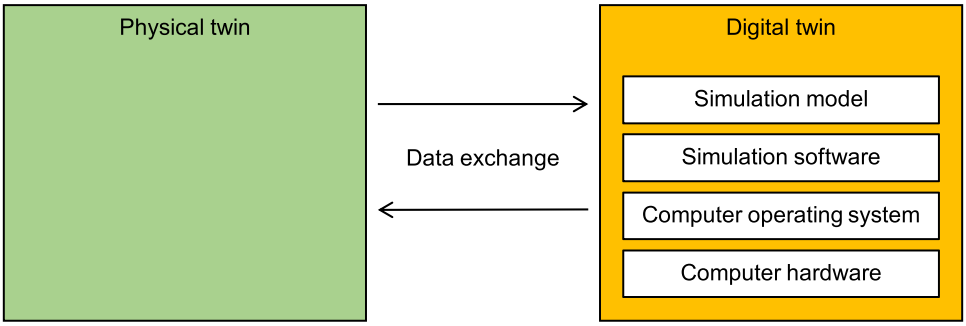

Fig. 4. The main building blocks of the FDT concept and the elements of the simulation-based DT.

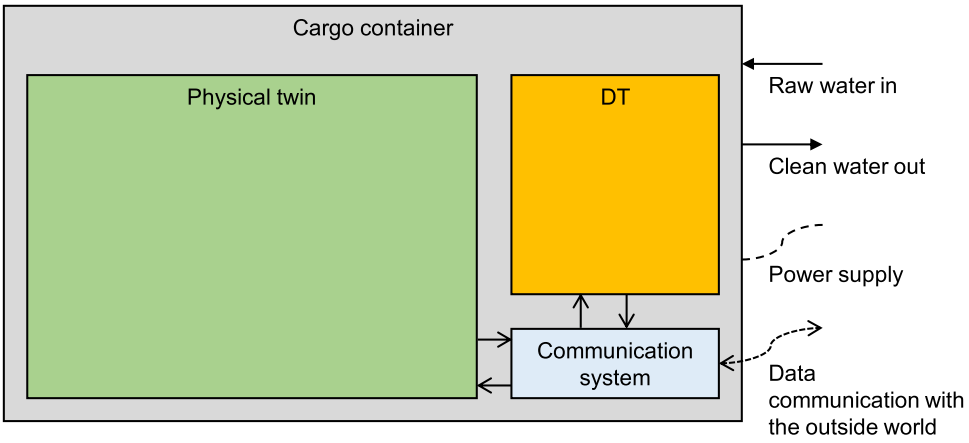

Fig. 5. Schematic illustration of the case example, a water purification system including a DT.

software applications are coming to the market, companies are acquired and merged, and new versions of software applications are introduced regularly. Estimating the same kind of development trend for the computer hardware, computer operating systems, and simulation software for the coming decades indicates that the computing infrastructure and simulation software will be remarkably different from what we are using today.

2.4.1. Case example of FDTs technical risks

With the following example of a fictive water purification system, we illustrate two scenarios of managing FDT model data and how the scenarios may affect the life cycle preservation of the FDT model data. As shown in Fig. 5, the real water purification system is built into a cargo container (the gray box) and it only requires raw water supply, electricity for its power supply, and it communicates with the outer world either with a cable network connection or with satellite communication (the light blue box). The system has a built-in DT (the yellow box), which is based on thermodynamic and one-dimensional pipe flow system simulation, together with control and automation system simulation features. The system model of the water purification process contains all the process components, pipes, sensors, and instruments that the real system has and it represents the function and behavior of the real process with reasonable accuracy. In this case, the DT of the overall system is used for the real twin's condition monitoring and model-based control of the water purification process. This means that the DT is an essential part of the overall system, and the system cannot be fully functional without the DT. In this case, the expected lifetime of the water purification system is 30 years and Company A, which is developing and producing such systems, sells 500 system units per year on average.

In the first scenario, Company A decides to use an existing system simulation model of the system's engineering design phase as the basis of the DT. The simulation model of the engineering design phase is created with a commercial system simulation software application, developed and maintained by Company B. The simulation software application provides all the necessary simulation features needed, it is easy and efficient to use, and the engineering personnel are familiar with it. The choice is natural and, at the design phase of the system, also obvious. The software is dedicated to the modeling and simulation of systems, which are similar to the water purification system, and the software contains several modeling elements that are unique to it. The software has its own binary format for saving the simulation models and the simulation results data, and it does not support exporting the model to any other similar modeling and simulation software application, which is quite common in the industrial domain. Ten years after launching the water purification system onto the market, the software company, Company B, which was developing and maintaining the modeling and simulation software, descends into financial problems and eventually goes bankrupt, and the maintenance of the software suddenly ends. Company A, producing the water purification system, has already sold about 5000 units worldwide and is responsible for providing support and maintenance for the systems, including the essential part of the system, the DT. Company A must replace the DT elements of all the sold units with a solution that can be maintained and supported.

In the scenario above, the technical risk for Company A is in the aging of the simulation software necessary to run the DT. The simulation software that is running the simulation model of the DT requires computing infrastructure. If the development in computing technology remains fast, for example, the next generations of the operating system may not be compatible with the simulation software. This will then lead to freezing the operating system, which most likely has a limited service life. Furthermore, the operating system limits options for computing hardware as the operating system only supports hardware that was available at the time when the operating system was maintained. Any final fault in the computing hardware or the operating system can break the system and disable the operation of the DT, and eventually, the whole water purification unit. The total number of 5000 sold units scales the technical risk. As the DT models are in a binary format (i.e., it is difficult to reverse engineer them) and the models cannot be exported to another format that is supported by another simulation software, Company A does not have any simple options for replacing the models of the DTs, but it must recreate them with another modeling and simulation software application.

In the second scenario, Company A applies an open simulation language to develop and represent the simulation models for the DT and uses a commercial modeling and simulation software application, developed and maintained by Company C, for the modeling and simulation work. The model data and the simulation results data are stored in an open and well-specified format. Again, 10 years later after the launch of the water purification system to the market, the software company developing and maintaining the modeling and simulation software, Company C, descends into financial problems and eventually goes bankrupt, and the maintenance of the software suddenly ends. As the simulation models of the DTs are represented with an open and well-specified simulation language and as there are several other commercial and even open-source implementations of the simulation language available, transferring the DTs into another simulation software is straightforward.

In this second scenario, the technical risk for Company A is small compared to the first scenario. Possible technical problems with

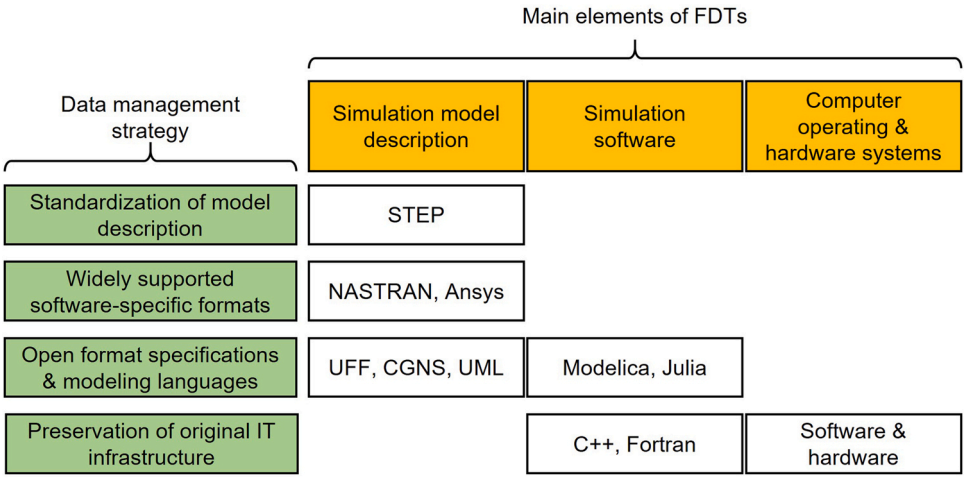

Fig. 6. Strategies for long-term data management of FDTs.

software details and possible problems with numerical solving details can cause additional work, with which the new software vendor may also be able to help.

3. Strategies for DT data life cycle management

Application software used along the product model evolution are not well integrated; different vendors use different proprietary or native file formats that lead to interoperability problems. As an example, in the United States alone industries spend billions of dollars due to the lack of interoperability between CAD tools Szykman et al. (2001). One of the reasons behind this is the business aspect. Software vendors keep product information in protective formats and do not publicly share specifications about them. The users of such application software are not only deprived of important product information but also limited in their choices of adopting alternative tools and are hence unsecured in circumstances in which these vendors halt business Brunsmann et al. (2012); Wilkes et al. (2011). Developing standardized, neutral, and open data formats for such applications will not only reduce the interoperability costs González et al. (2007) but also assist software users to keep running their simulation models along the product life cycle.

Considering that the data management of FDTs has life cycles of decades, information about the underlying simulation models needs to be preserved in formats that are independent of the original software tools. Thus, in situations where the original application software is incapable for opening or running a simulation model, the DT developer should be able to reproduce and run the corresponding model with alternative tools by utilizing the preserved information. Thus, we propose:

> Proposition: The black box proprietary file formats and the lack of interoperability among the application software increase the risks of the incapability of the computing infrastructure to keep the FDTs running throughout the physical asset's life cycle.

Fig. 6 shows a number of strategies (in green boxes) and example technologies (in white boxes) for alleviating the issues associated with the long-term data management of FDTs. For each main element of the FDT (in yellow boxes), one or more approaches are needed, for example: the further development of the STEP standard would unify the simulation model description; the widely supported formats, such as NASTRAN and Ansys, would provide flexibility in terms of simulation software selection; the open format specifications (e.g., UFF and CGNS) and open modeling languages (e.g., Modelica and Julia) would, respectively, promote common data formats and bring interoperability among software tools; and the preservation of original IT infrastructure would be required in some cases (e.g. for mission-critical applications). These strategies are discussed in more detail in the following sections.

3.1. Standardization of model description

Standardization is an influential approach to improving information exchange and interoperability. When performed by a well-established standardization authority, it provides a solid basis for stable and long-lasting information preservation. In the case of CAD, for example, the role of ISO 10303 STEP Anon (1994) (especially its application protocols AP203, AP214, and AP242) as a neutral and widely accepted geometry representation technology enables the maintenance and exchange of geometry information between different parties using different CAD software applications throughout the product or system life cycle. Well-formed standards define the semantics and syntax in such a way that the information content is preserved. Thus, the representation of product information in a standardized data model helps to Johansson et al. (2004) do the following: .

1. integrate existing and future software tools;
2. communicate and share information easily among different domains and applications without the need for customized interfaces;
3. provide users the opportunity of adopting an application that best performs a given task, provided it exchanges data with other software in a standardized format.

However, the process of developing and defining a standard is typically time consuming and demanding. For rapidly developing and evolving topics, the process of defining and agreeing upon a standard may be too laborious and slow; hence, other alternatives should also be considered. Thus, we propose:

> Proposition: Standardization improves interoperability among software applications and makes the exchange and understanding of information easier. It also improves data preservation for long product or system life cycles. However, standardization is a cumbersome process and requires remarkable efforts.

3.2. Open format specifications and open modeling languages

In engineering simulation as well as experimental research and development, the need to exchange data between software applications has led to the creation of open and common specifications of data exchange formats. An example of such a format is the Universal File Format (UFF) Anon (2021). This de facto standard format was originally developed for data exchange between CAD and CAE systems and experimental test systems. It is also used for data exchange between structural analysis software applications using FEM. Another example of an open file format is the CFD General Notation System (CGNS) Anon (2015). Widely accepted and adopted open formats enable the users of engineering software applications to change the software tool, if needed, and to preserve the efforts done for the simulation models. Besides this, the wide acceptance of the open formats increases the possibility that the format will be supported for a long time by the software vendors.

In a system simulation, the diversity of computational elements in the computations makes it difficult to use the similar kind of approach as that of the UFF. To preserve the information needed for reproducing the computation with the simulation model, the numerical solving schema should also be involved. The numerical solving schema may differ between simulation software applications, including differing in regard to how the model components are described mathematically but also regarding the overall numerical solving approach of a model. One approach to overcome this issue is the use of a modeling language. A modeling language includes the means to describe the simulation model components, their mathematical representation, and the overall approach to formulating the system model independently of the application software's implementation. Examples of such languages are Modelica Association (2020) and Simscape MathWorks (2020).

Modeling languages also enable describing a simulation or analysis case with a programming language that is designed for this purpose. A language approach enables separating the simulation model representation from the tools that are used for the modeling and simulation. The approach is similar to that of commonly used programming languages, such as C or Fortran. The specification of the language is open and independent from the implementations of the tools for the language. This gives the user, at least in principle, the freedom to choose the best tool, and even the opportunity to change the tools without losing the investments on the modeling and simulation work. Examples of open modeling languages are Modelica and Julia Anon (2021).

A simulation-domain independent approach for describing model data is to use general data modeling and, for example, metamodeling languages (such as Unified Modeling Language [UML] and its profiles mechanism) to define domain-specific concepts and features Anon (2019).

The Functional Mock-up Interface (FMI) is an open and free standard for improving the reuse and exchange of system simulation models and model components Anon (2022). The standard covers two general use cases, model exchange and co-simulation. The model exchange with FMI enables the exchange of simulation models, partial models, or model components between various modelling and simulation tools, and the co-simulation is for the coupling of simulation tools during the simulation. The FMI standard is supported by more then 170 modelling and simulation tools and the standard is maintained and its development is coordinated by the Modelica Association. When using FMI for DT model data management, the black box nature of model exchange with FMI hinders any necessary modifications of the DT simulation models if the original simulation model is not available.

Open data formats and open modeling languages are a valuable means to represent simulation model data and to exchange it between simulation software applications. Thus, we propose:

> Proposition: The DT solutions that are based on application software utilizing open formats and open modeling languages improve data exchange and ensure the information availability for reproducing the simulation models with alternative tools whenever needed.

3.3. Widely supported software application-specific formats

In some computational engineering domains, widely used software applications and their data formats can get a dominating role as the formats are increasingly supported by several software vendors. This may especially happen when the documentation of the format is open and even publicly available. Examples of such formats are NASA Structure Analysis (NASTRAN) in structural analysis (e.g., Simcenter Nastran by Siemens and MSC Nastran by Hexagon) and Ansys Fluent in CFD. Similarly to the software-independent format specifications, widely accepted software-specific formats are also expected to be supported for a long time, and the users have the option to change the software application if needed. Thus, we propose:

> Proposition: The application of widely used simulation programs and their open data formats would help in keeping the FDTs running for a long time because such solutions are supported by multiple software vendors. In the case of a solution that is becoming outdated, the DT developers would be able to adopt another compatible solution.

3.4. Software source code

Another approach to guarantee simulation model data preservation for long life cycles is to represent all the data in a source code format. This means that the simulation software needed to run the simulation is included in a source code format, such as the C++ or Fortran programming languages, and the additional data needed for the computation, such as possible model description or other input or accessory data, are provided in an accessible and editable format. Widely used and standardized programming languages are well preserved, even for very long life cycles. This approach is rather laborious to implement, but in long life cycles it provides the flexibility to, for example, update the simulation models. In addition, it does not include serious risks of third-party software or licensing changes. When using open-source software applications for simulation, this approach is rather straightforward. However, one still has to take care of the possible changes in the underlying third-party components and libraries, and other possible dependencies of the DT software. In very long timespans, caution must be taken regarding the compatibility of the computing infrastructure (i.e., the computing hardware and operating system). Thus, we propose:

Proposition: The preservation of simulation software in a source code format using a standardized programming language would enable long-term operation of FDTs. This approach naturally requires that the source code is available. This is usually not the case with commercial simulation software.

3.5. The preservation of the original IT infrastructure

The challenges associated with storing digital product or system models in native formats have been widely documented. Most of the commercial software applications store model data in their dedicated formats that are intended to only be read by the applications that originally produced them. Problems may occur when newer versions of software applications are used to open the design or engineering models for modifications.

Native models are easily accessible if the original software and hardware systems are available. Thus, it is not only the data that must be archived, the means to make sense of the data must also be preserved. For long-term access to FDT models, it would be necessary to archive the whole IT infrastructure (i.e., computer hardware, operating system, and simulation software) that was originally used for their development. However, this approach would limit the integration of the DT with other, e.g. more advanced, systems in the long term. Thus, we propose:

Proposition: Preservation of the original software and hardware systems could also be considered as an alternative in order to mitigate the risks of losing the IT infrastructure needed for FDTs' operation. This approach is usually very demanding and could be used for mission critical cases.

4. DT life cycle risks from the business point of view

4.1. Dependency and servitization risks

DTs are highlighted in servitization, where "servitization" refers to the process through which, for example, manufacturing companies complement their traditional offerings by integrating services into their business operations Rönnberg Sjödin et al. (2016). Literature reports a large variety of benefits that can be achieved with DTs, such as cost reduction and efficiency Jones et al. (2020). However, incorporating DTs into the service offerings also includes business-related uncertainties and risks, especially in a situation where DT is provided by a third party. First, there can be dramatic consequences if the third-party DT provider disappears from the market (for example, because of bankruptcy). Second, the DT provider may have serious technical issues with its services, and the main system provider has no easy way to change the DT provider, for example, because of the software and technical debt related to the development and updating of DTs Malakuti and Heuschkel (2021); Malakuti et al. (2021); Holvitie et al. (2018). In comparison to traditional industrial engineering that uses commercial, closed source tools with dedicated data models and formats, DTs are often designed to be open source for all the relevant stakeholders which increase their dependency on third-party provider. Thus, we propose:

Proposition: Dependency on the third-party provider increases the life cycle risks of a DT in the servitization business.

The main reason behind the risks is a lack of standardization. For example, Fuller et al. (2020) concluded that challenges within all forms of a DT's development relate to the modeling of such systems because there is no standardized approach to modeling. Standardized approaches increase understanding among domain experts and users while ensuring information flow between each stage of the development and implementation of a DT. In order to mitigate the life cycle business risk of a DT, the main providers may need trusted and licensed third parties that are holding a source code in addition to the forthcoming standardization. Whereas the holding of the source code may reduce the risk of the disappearance of the service provider, the possible future standardization would mitigate the business risks related to the software debt Malakuti and Heuschkel (2021); Yli-Huumo et al. (2016), of DTs. During the life cycle of a DT, the underlying software may require constant development and updating, the costs of which are usually covered by the service provider. However, if the software debt is based on the changes in the physical operation environment, it may also incur some costs for the DT utilizer. From the business perspective, the standardization of DTs may thus reduce the costs of software debt. However, it is not self-evident that a third-party provider is willing to share the source code and to use the standard modeling in all cases. High-level physical modeling is the third-party provider's core business, and sharing it requires trust building and discussion about mutual benefits. Thus, based on the notions above, we propose:

Proposition: In addition to standardization, trust building and mutual benefits mitigate the risks related to a third-party DT provider.

4.2. Human capability-related risks

Even though organizations increasingly utilize DTs to improve their competitive advantages, productivity, and efficiency through automating and digitalizing the processes of the physical counterpart composed of, for example, mechanical and electrical components as well as sensor technology for data gathering Malakuti and Heuschkel (2021), Lim et al. (2020), the human-related business risks of DTs still exist. Despite the fact that the design, construction, and implementation of the DTs require computation tools, and the fact that technology building blocks exist that reduce the development costs of DTs Alam and Saddik (2017), their implementation also requires understanding about the business processes of the organizations. In addition to the above-mentioned software debt, the development and updating of DTs involve risks related to the technical debt, for example, in a situation where the physical counterpart may require modifications for proper data acquisition Malakuti et al. (2021); Holvitie et al. (2018); Malakuti (2021). If the organization does not have capable persons to understand the business perspective of DTs and to outline the cost efficiency of updating DTs, they may face the situation in which the developed solutions never achieve payback. Not only do the updating and fixing of DTs during their life cycle cause costs, but it is also challenging to recruit and keep employees who have both computational capabilities and understanding of the computational algorithms and physical operational environment. Thus, we propose:

Proposition: The lack of understanding (the lack of skills and competences) of how to maintain DTs increases the life cycle risks of a DT.

An important characteristic of a DT is that it generates data and information in order to both support the decision-making of an organization's strategy and processes, and to forecast future business opportunities Liu et al. (2019). The interpretation and utilization of the data and information produced by DTs involve expertise that is strongly human related. If such human capabilities are lost from companies (for example, due to retirement or job changes), the people debt Malakuti (2021) related to DTs is caused that leads to challenges in interpreting and exploiting the data generated by the DTs. Based on the above, we propose:

> Proposition: Training and coordinating the personnel for data interpretation would ensure the efficient utilization of DTs through human capabilities and decrease the life cycle risk of DTs.

4.3. Coordination-related risks

Coordination complexity emerges if the relevant information about the implemented DT is not at hand for all the relevant persons of the utilizing entity. While DTs provide possibilities to generate and exchange data, for example, with suppliers, customers, and competitors, the operating of them requires coordination to ensure optimal and meaningful use. It is not only about understanding and minimizing the DT related software debt Yli-Huumo et al. (2016), technical debt Malakuti et al. (2021), and people debt Malakuti (2021) for the stakeholders, but the DT technology requires explanations at the inter-functional level so that the benefits and functioning of the DT are evident for the users. This causes risks related to the coordination of the DT along the life cycle if the simulation model is at risk of disappearing due to the aging of the required technologies caused by the technical debt, and if moving to a new modeling and simulation platform is not possible, e.g., because of the software debt or the lack of standardization. This generates possibilities to intentional or unintentional misconducts of DTs, for example to get and share data that is not intended to be shared. Thus, we propose:

> Proposition: The coordination complexity of the DT technology increases its life cycle risks.

As DTs, at their best, are efficient ways to assist visualization, promote collaboration, and support decision-making Bao et al. (2019); Kaewunruen and Lian (2019), the coordination complexity related risks require active initiatives to be avoided. One solution to alleviate such risks is to put emphasis on inter-functional management of the DT. After all, the efficient utilization of DTs requires different parties to share information with each other. The risk of coordination complexity can also be avoided by allowing parties from different departments/organizations to participate in the development of the DT. In this way, it is possible to avoid bad technical choices during the solution development phase that may cause increasing technical and software debts Holvitie et al. (2018) and risks later on. Coordination complexity risks are also likely to appear if the company invested massively in DTs in order to manage knowledge ownership but left the thoughtful introduction of their utilization to the later phase. Avoiding such a risk may require introducing DT technology gradually or being specific about the direct applications. Thus, to avoid the coordination complexity risk, we propose:

> Proposition: Inter-functional management and gradual implementation of the DT technology may reduce coordination complexity-related life cycle risks of DTs.

5. Discussion and conclusions

The progress in the evolvement of the DT concept has been fast, especially due to the rapid development in enabling technologies, such as the IoT, data analytics, big data, and AI. This is due to the promising prospects that the concept is showing, both for the users of DTs and for the service and solution providers. DTs are estimated and reported to increase productivity, efficiency, and maintainability, among other benefits. Due to the reported benefits and because of the ongoing hype in developing and applying DTs, the fast progress can blind businesses to the potential long-term risks of DTs. These risks are associated with the complexity and dependencies of the implementation, the long life cycles of the physical products and systems, and their corresponding DTs.

Our findings on the definition of a DT are: 1) there is no consensus on an explicit definition of a DT; 2) the definition of a DT is continuously changing; and 3) the definition itself does not add much value but improves the communication about DTs. One of the tasks for research is to define new concepts. Concerning the concept of a DT, there is no consensus about the definition, and the concept has been defined by many and the existing definitions have been improved continuously. What seems to be held in common is that the main elements of the concept are the physical or real system, its digital representation, and the data connection between the two. The concept of a DT is multifaceted and depends on the context, application, objectives, and needs, and also on many technical and business aspects. Due to this, there is not only one right way to apply DTs and there cannot be a clear set of steps to success. On the other hand, there are many things to be considered when applying DTs. The value of DTs comes from their applications, not from the concept's definitions.

In this work, we focused on the data management issues of FDTs that are used as integral parts of the corresponding physical asset, and FDTs have dependences on computing hardware and software technologies, which make their life cycle management challenging. To define the category of an FDT is to narrow the scope to applications where there are certain kinds of aspects involved, namely the numerical simulation of physics-based models and the long expected life cycle of the overall system. FDTs are developed and operated using specific platforms (i.e., computer hardware, software systems, and simulation tools) that evolve along with the technological progress. This trend could possibly affect the operation of FDTs, especially for long life cycle assets, and cannot be simply handled in the case of the unavailability of any of the three main elements that are required for computer simulations. As the computer hardware and operating systems are continuously evolving, the selection of the simulation approach and software remains the only choice to make for DT providers at the very initial phase that affect the long-term operation of DTs. In addition, most of the commercial simulation software utilizes black box modeling techniques; hence, the DT provider should consider simulation tools that provide detailed information about the DTs' models, including the principles of the involved physics and the implementation of the physics in numerical computing. This information should be preserved in long-lasting formats and then be utilized, whenever needed, for reproducing the DTs' simulation models by using up-to-date tools.

Table 1
Comparison of strategies for life cycle management of DT data models.

Strategy	Applicable simulation approaches	Effort	Flexibility
Standardization of model description	FEM, CFD, system simulation	High	High
Use of open format specifications	FEM, CFD	Medium	High
Use of open modeling languages	System simulation	High	Medium
Using tools with widely supported formats	FEM, CFD	Low	Medium
Preservation of original IT environment	FEM, CFD, system simulation	High	Low

The information content and the required expertise related to physics-based simulation have multiple layers. The basis for the information needed is the physical phenomenon that must be simulated. The language to describe the details of physics is mathematics, and in the computational approach, that usually means ordinary or partial differential equations together with algebraic and other formulations. Solving the mathematical sets of equations with computers requires different kinds of numerical means in order to, for example, estimate differentials and efficiently solve the equations. Implementing this in a computer software application requires that the numerical representation is programmed into the software, including any other required functionalities. All this is still on a general level and does not describe any particular simulation case. To represent the function of a particular system, its features must be described for the simulation software (e.g., in a form of a model description). Within this chain of layers for representing the information needed for the simulation, the layers from the numerical representation to the particular simulation model description are needed for reproducing the simulation model whenever needed.

In Section 3, we discussed different strategies for the life cycle management of the DT model data. Table 1 summarizes the scope, status, and flexibility of various strategies (see Section 2 for details) focusing on different simulation approaches. The added-value of these strategies requires further development as the technologies introduced in this study are still not fully mature. Moreover, the FEM and CFD are combined together because these domains rely on discretized geometry volume and applying partial different equations on the elements (FEM) or cells (CFD), whereas the system simulation has quite different bases and its challenges require different strategies.

In general: 1) standardization was found to be a strong and solid means to preserve the DT model data for long life cycles; but 2) the lack of standards in some simulation domains, especially in system simulation, was found to be a problem. Standardization has been considered an important solution for improving interoperability among the CAx systems (i.e., the CAD, CAM, and CAE systems). It benefits both the user and the vendor communities by reducing the software development, maintenance, and support costs. The standardized data files of one tool can be processed by another similar tool and the problem of vendor lock-in can be alleviated. However, due to the rapid improvements in the services and business strategies of the software vendors, standardization would require remarkable efforts. The number of required resources compared with the added value has to be evaluated from technical and business points of view. The authority requirements for information preservation are strong motivators to develop the required technologies and to enhance standardization efforts. This has been the case in the airplane industry and the development of the STEP standards family. The rising interest in DTs can serve as the tipping point for the standardization of simulation model representation. The expected active life cycle of DTs is longer than the one of engineering models, and the number of stakeholders that are influenced by the DTs can be large. These things together can have enough critical mass to initiate the effort for standardization.

Compared with standardization, the open specification of data formats and their representation was found to be an agile and workable approach. As standardization is often seen a time-consuming and burdensome process, the open data format specification is a lightweight approach to defining a standard kind of common definition. In contrast to black box modeling techniques, the application software utilizing open formats (e.g., UFF and CGNS) and open modeling languages (e.g., Modelica and Julia) provide detailed information about the simulation models. This facilitates the software users in choosing the tool that best fulfills the required purpose. The open formats and open modeling languages are expected to be available for longer periods. Thus, if one software vendor has difficultly in maintaining the FDTs' models, the user can easily adopt alternative solutions without any major business disruption. However, most software vendors may not encourage the use of open formats and open modeling languages because they want to sustain their core competences and businesses.

In some cases, transforming data from one representation to another cannot be avoided beforehand and preparing for fluent data transformation instead of statically preserving the DT model data may be a more efficient approach. An alternative approach for standardization and open format specification is the use of meta-models and meta-modeling languages. Instead of focusing on defining the information content and a common format to represent it, this approach focuses on the means to decrease the effort needed to transform the data from one representation to another. While adding a new layer of abstraction, this approach may provide improved scalability and flexibility. The meta-modeling approach still requires definition of the information content and format. Thus, it does not completely solve the issue but provides tools for decreasing the effort.

Widely supported proprietary data formats can be a cost-efficient way of preserving DT model data, but the approach has some drawbacks. Within some simulation domains, such as structural analysis and CFD, some widely used simulation tools and their file formats become commonly supported. For instance, the NASTRAN and Ansys file formats are supported by many different structural analysis tools and the disappearance of one tool can be compensated for by adopting an alternative tool that is available in the market. This may result in a certain amount of information loss; nevertheless, the files are conveniently imported and exported among the supported application software. The enabler for the cross-use of these formats is the sufficient documentation of the format and the vendor's allowing attitude regarding others using these formats.

For mission-critical applications of DTs, preserving the DT software development environment or, further, even the overall computing environment may be required. The above-discussed means may not be enough when the application of a DT is mission critical or when there is, for example, an authority requirement to have all the necessary elements preserved for DTs and no standardization or open specifications are available. In these cases, the simulation software required for running the DTs' models can be preserved in source code format using the standardized programming languages, such as C++ and Fortran. Although the standardized programming languages are expected to be supported for a long time, their compatibility with the available computer hardware and software systems cannot be fully guaranteed. In addition, the

approach may prevent the potential risks of being involved in an agreement with third parties and their changing licensing policies. However, this approach seems to be cumbersome as well as requiring in-house competences. The approach requires that all aspects of numerical simulation are covered. On the other hand, this approach fits together well with open-source software applications. Since FDTs are developed and operated using dedicated simulation software, computer hardware, and operating systems, preserving the required IT infrastructure could also help in the long-term operation of such DTs. This approach may seem straightforward, however, it would be quite difficult to manage the number of IT systems as more and more DTs are developed for every individual asset deployed in the real environment.

From the business point of view, DT life cycle risks are determined by the extent to which the DT utilizers: 1) are dependent on third-party providers; 2) lack the human skills to manage DTs; and 3) are capable of managing the coordination complexity of DTs. The business risks of DTs are extensively caused by the insufficient management of technical risks. Thus, focusing on standardization, trust building, and mutual benefits is likely to decrease the risks related to a third-party DT provider. Coordination and updating the capabilities of data interpretation decrease the risk of having insufficient human knowledge to manage the entire DT life cycle. Further, inter-functional management and gradual implementation of the DT can mitigate the risk of insufficient coordination of the DT data. Future research should investigate the extent to which in-depth ecosystem-level collaboration could assist in decreasing the life cycle risks of DTs.

While the technical enablers for implementing and applying DTs are maturing and there are new technologies, such as virtualization, that widen the possibilities to manage DTs, the fundamental challenges remain. The applications of DTs that contain technically demanding elements, such as complex physics-based simulation and the implementation of numerical methods, require that enough information is preserved for any possible future problems. In mission-critical cases, even extreme means are required to guarantee that the whole stack of technologies is available, including computing hardware and the operating system. One of the main outcomes of this work is that when the overall complexity is increasing, a certain level of standardization and open specification is needed. Similarly, as the size of the group of stakeholders affected by the operation of DTs is increasing, the motivation for initiating and implementing standardization efforts may pass the threshold for the often laborious process. In addition to standardization, other technical enablers are needed to enable the realization of the full potential of DTs. The application of DTs is not only a technical issue, it also has organizational and business dimensions.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work was supported by the DigiBuzz Research Project, funded by Business Finland[1] under grant n:o 4437/31/2019 and the participating organisations.

[1] https://www.businessfinland.fi/en

References

Alam, K.M., Saddik, A. El, 2017. C2PS: a digital twin architecture reference model for the cloud-based cyber-physical systems. IEEE Access 5, 2050–2062. https://doi.org/10.1109/ACCESS.2017.2657006. ⟨http://ieeexplore.ieee.org/document/7829368/⟩.

AnonISO, ISO 10303-1:1994 - Industrial automation systems and integration -? Product data representation and exchange -? Part 1: Overview and fundamental principles (1994). ⟨https://www.iso.org/standard/20579.html⟩.

AnonCGNS Steering Committee, A User's Guide to CGNS - Introduction (2015). ⟨http://cgns.github.io/CGNS_docs_current/user/intro.html⟩.

AnonObject Management Group, OMG Meta Object Facility (MOF) Core Specification, Tech. rep., Object Management Group (2019). ⟨https://www.omg.org/spec/MOF/⟩.

AnonUniversity of Cincinnati, Universal File Formats for Modal Analysis Testing -? Structual Dynamics Research Lab (2021). ⟨https://www.ceas3.uc.edu/sdrluff/⟩.

AnonJuliaLang.org contributors, The Julia Programming Language (2021). ⟨https://julialang.org/⟩.

AnonLWN.net, The LWN.net Linux Distribution List (2021). ⟨https://lwn.net/Distributions/⟩.

AnonFuntional Mock-up Interface, Functional Mock-up Interface (2022). ⟨https://fmi-standard.org/⟩.

Modelica Association, Modelica Language (2020). ⟨https://www.modelica.org/modelicalanguage⟩.

Bao, J., Guo, D., Li, J., Zhang, J., 2019. The modelling and operations for the digital twin in the context of manufacturing. Enterp. Inf. Syst. 13 (4), 534–556. https://doi.org/10.1080/17517575.2018.1526324. ⟨https://www.tandfonline.com/doi/full/10.1080/17517575.2018.1526324⟩.

Barricelli, B.R., Casiraghi, E., Fogli, D., 2019. A survey on digital twin: definitions, characteristics, applications, and design implications. IEEE Access 7, 167653–167671. https://doi.org/10.1109/ACCESS.2019.2953499. ⟨https://ieeexplore.ieee.org/document/8901113/⟩.

Boschert, S., Rosen, R., 2016. Digital Twin-?The Simulation Aspect. In: Hehenberger, P., Bradley, D. (Eds.), Mechatronic Futures. Springer, Cham, pp. 59–74. https://doi.org/10.1007/978-3-319-32156-1_5. http://link.springer.com/10.1007/978-3-319-32156-1_5.

S. Boschert, C. Heinrich, R. Rosen, Next Generation Digital Twin, in: I. Horváth, J. Suárez Rivero, P. HernándezCastellano, (Eds.), Proceedings of TMCE 2018, 7-11 May, 2018, Las Palmas de Gran Canaria, Spain, 2018, 209-218.10.17560/atp.v60i10.2371.https://www.researchgate.net/profile/Stefan_Boschert/publication/325119950_Next_Generation_Digital_Twin/links/5af952ca0f7e9b026bf6e553/Next-Generation-Digital-Twin.pdf.

Brunsmann, J., Wilkes, W., Schlageter, G., Hemmje, M., 2012. State-of-the-art of long-term preservation in product lifecycle management. Int. J. Digit. Libr. 12 (1), 27–39. https://doi.org/10.1007/s00799-012-0081-4. ⟨http://link.springer.com/10.1007/s00799-012-0081-4⟩.

D. B. Cameron, A. Waaler, T. M. Komulainen, Oil and Gas digital twins after twenty years. How can they be made sustainable, maintainable and useful?, in: Proceedings of The 59th Conference on Simulation and Modelling (SIMS 59), Oslo Metropolitan University, Norway, 2018, 9-16.10.3384/ecp181539.http://www.ep.liu.se/ecp/article.asp?issue=153%26article=2.

Eckert, R., 2014. 6.2.3 conflict in systems engineering product data exchange standardisation. INCOSE Int. Symp. 16 (1), 882–894. https://doi.org/10.1002/j.2334-5837.2006.tb02788.x. ⟨http://doi.wiley.com/10.1002/j.2334-5837.2006.tb02788.x⟩.

Erikstad, S.O., 2017. Merging physics, big data analytics and simulation for the next-generation digital twins. HIPER 2017, High. Perform. Mar. Veh. Zeven South-Afr. 139–149.⟨https://www.researchgate.net/publication/320196420_Merging_Physics_Big_Data_Analytics_and_Simulation_for_the_Next-Generation_Digital_Twins⟩.

Fowler, J., 1995. STEP for data management, exchange and sharing, technology appraisals. Gt. Br. ⟨http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.107.2051&rep=rep1&type=pdf⟩.

Frechette, S.P., 2011. Model based enterprise for manufacturing. Proc. 44th CIRP Int. Conf. Manuf. Syst.

Fuller, A., Fan, Z., Day, C., Barlow, C., 2020. Digital twin: enabling technologies, challenges and open research. IEEE Access 8, 108952–108971. https://doi.org/10.1109/ACCESS.2020.2998358. ⟨https://ieeexplore.ieee.org/document/9103025/⟩.

M.P. Gallaher, A.C. O'Connor, J.L. Dettbarn, L.T. Gilday, Cost Analysis of Inadequate Interoperability in the U.S. Capital Facilities Industry, Tech. rep., National Institute of Standards and Technology, GCR 04-867, Gaithersburg, MD (2004). 10.6028/NIST.GCR.04-867.

Gargiulo, C., Malandrino, D., Pirozzi, D., Scarano, V., 2014. Simulation data sharing to foster teamwork collaboration. Scalable Comput. Pract. Exp. 15 (4), 309–329. https://doi.org/10.12694/scpe.v15i4.1053. ⟨http://www.scpe.org/index.php/scpe/article/view/1053⟩.

González, M., González, F., Luaces, A., Cuadrado, J., 2007. Interoperability and neutral data formats in multibody system simulation. Multibody Syst. Dyn. 18 (1), 59–72. https://doi.org/10.1007/s11044-007-9060-8. ⟨http://link.springer.com/10.1007/s11044-007-9060-8⟩.

Grieves, M., Vickers, J., 2017. Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems. In: Kahlen, F.-J., Flumerfelt, S., Alves, A. (Eds.), Transdisciplinary Perspectives on Complex Systems. Springer International Publishing, Cham, pp. 85–113. https://doi.org/10.1007/978-3-319-38756-7_4. (Ch. Digital Tw).

D. Hartmann, H. V. D. Auweraer, DIGITAL TWINS (2020). arXiv:2001.09747v1.https://arxiv.org/pdf/2001.09747.pdf.

Holvitie, J., Licorish, S.A., Spínola, R.O., Hyrynsalmi, S., MacDonell, S.G., Mendes, T.S., Buchan, J., Leppänen, V., 2018. Technical debt and agile software development practices and processes: an industry practitioner survey. Inf. Softw. Technol. 96, 141–160. https://doi.org/10.1016/j.infsof.2017.11.015. ⟨https://linkinghub.elsevier.com/retrieve/pii/S0950584917305098⟩.

Johansson, H., Åström, P., Orsborn, K., 2004. A system for information management in simulation of manufacturing processes. Adv. Eng. Softw. 35 (10–11), 725–733. https://doi.org/10.1016/j.advengsoft.2004.03.019. ⟨https://www.sciencedirect.com/science/article/pii/S0965997804000924⟩.

Jones, D., Snider, C., Nassehi, A., Yon, J., Hicks, B., 2020. Characterising the digital twin: a systematic literature review. CIRP J. Manuf. Sci. Technol. 29, 36–52. https://doi.org/10.1016/j.cirpj.2020.02.002. ⟨https://linkinghub.elsevier.com/retrieve/pii/S1755581720300110⟩.

Kaewunruen, S., Lian, Q., 2019. Digital twin aided sustainability-based lifecycle management for railway turnout systems. J. Clean. Prod. 228, 1537–1551. https://doi.org/10.1016/j.jclepro.2019.04.156

Lim, K.Y.H., Zheng, P., Chen, C.-H., 2020. A state-of-the-art survey of digital twin: techniques, engineering product lifecycle management and business innovation perspectives. J. Intell. Manuf. 31, 1313–1337. https://doi.org/10.1007/s10845-019-01512-w. ⟨http://link.springer.com/10.1007/s10845-019-01512-w⟩.

Liu, J., Zhou, H., Liu, X., Tian, G., Wu, M., Cao, L., Wang, W., 2019. Dynamic evaluation method of machining process planning based on digital twin. IEEE Access 7, 19312–19323. https://doi.org/10.1109/ACCESS.2019.2893309. ⟨https://ieeexplore.ieee.org/document/8631019/⟩.

Liu, Z., Meyendorf, N., Mrad, N., 2018. The role of data fusion in predictive maintenance using digital twin. pp. 020023-1-020023-6. AIP Conf. Proc., Vol. 1949, Am. Inst. Phys. Inc. https://doi.org/10.1063/1.5031520. pp. 020023-1-020023-6. ⟨http://aip.scitation.org/doi/abs/10.1063/1.5031520⟩.

Magargle, R., Johnson, L., Mandloi, P., Davoudabadi, P., Kesarkar, O., Krishnaswamy, S., Batteh, J., Pitchaikani, A., Simulation-Based, A., 2017. Digital twin for model-driven health monitoring and predictive maintenance of an automotive braking system. Proc. 12th Int. Model. Conf. Prague Czech Repub. 35–46. https://doi.org/10.3384/ecp1713235. ⟨https://ep.liu.se/en/conference-article.aspx?series=ecp&issue=132&Article_No=3⟩.

Malakuti, S., 2021. Emerging Technical Debt in Digital Twin Systems. 2021 26th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), vol. 2021-Septe. IEEE, pp. 01–04. https://doi.org/10.1109/ETFA45728.2021.9613538

Malakuti, S., Heuschkel, J., 2021. The Need for Holistic Technical Debt Management across the Value Stream: Lessons Learnt and Open Challenges. 2021 IEEE/ACM International Conference on Technical Debt (TechDebt). IEEE, Madrid, pp. 109–113. https://doi.org/10.1109/TechDebt52882.2021.00021. https://ieeexplore.ieee.org/document/9463026/.

Malakuti, S., Borrison, R., Kotriwala, A., Kloepper, B., Nordlund, E., Ronnberg, K., 2021. An Integrated Platform for Multi-Model Digital Twins. 11th International Conference on the Internet ofThings (IoT '21), Association for Computing Machinery. ACM, pp. 9–16. https://doi.org/10.1145/3494322.3494324

Martínez, G.S., Sierla, S., Karhela, T., Vyatkin, V., 2018. Automatic generation of a simulation-based digital twin of an industrial process plant. Proc.: IECON 2018 - 44th Annu. Conf. IEEE Ind. Electron. Soc., Inst. Electr. Electron. Eng. Inc. 3084–3089. https://doi.org/10.1109/IECON.2018.8591464. ⟨https://ieeexplore.ieee.org/abstract/document/8591464⟩.

MathWorks, Simscape - MATLAB & Simulink (2020). ⟨https://www.mathworks.com/products/simscape.html⟩.

Monticolo, D., Badin, J., Gomes, S., Bonjour, E., Chamoret, D., 2015. A meta-model for knowledge configuration management to support collaborative engineering. Comput. Ind. 66, 11–20. https://doi.org/10.1016/j.compind.2014.08.001

Pantelides, C., Renfro, J., 2013. The online use of first-principles models in process operations: Review, current status and future needs. Comput. Chem. Eng. 51, 136–148. https://doi.org/10.1016/j.compchemeng.2012.07.008. ⟨https://linkinghub.elsevier.com/retrieve/pii/S0098135412002360⟩.

Rönnberg Sjödin, D., Parida, V., Kohtamäki, M., 2016. Capability configurations for advanced service offerings in manufacturing firms: Using fuzzy set qualitative comparative analysis. J. Bus. Res. 69 (11), 5330–5335. https://doi.org/10.1016/j.jbusres.2016.04.133. ⟨https://linkinghub.elsevier.com/retrieve/pii/S014829631630337X⟩.

Schluse, M., Rossmann, J., 2016. From simulation to experimentable digital twins: Simulation-based development and operation of complex technical systems. 2016 IEEE International Symposium on Systems Engineering (ISSE). IEEE, Edinburgh, pp. 1–6. https://doi.org/10.1109/SysEng.2016.7753162. http://ieeexplore.ieee.org/document/7753162/.

Schneider, R., Marquardt, W., 2002. Information technology support in the chemical process design life cycle. Chem. Eng. Sci. 57 (10), 1763–1792. https://doi.org/10.1016/S0009-2509(02)00075-1. ⟨https://linkinghub.elsevier.com/retrieve/pii/S0009250902000751⟩.

Szykman, S., Fenves, S.J., Keirouz, W., Shooter, S.B., 2001. A foundation for interoperability in next-generation product development systems. CAD Comput. Aided Des. 33 (7), 545–559. https://doi.org/10.1016/S0010-4485(01)00053-7

Tao, F., Zhang, H., Liu, A., Nee, A.Y.C., 2019. Digital twin in industry: state-of-the-art. IEEE Trans. Ind. Inform. 15 (4), 2405–2415. https://doi.org/10.1109/TII.2018.2873186. ⟨https://ieeexplore.ieee.org/document/8477101/⟩.

Tuegel, E.J., Ingraffea, A.R., Eason, T.G., Spottswood, S.M., 2011. Reengineering aircraft structural life prediction using a digital twin, international. J. Aerosp. Eng. 2011, 1–14. https://doi.org/10.1155/2011/154798. ⟨http://www.hindawi.com/journals/ijae/2011/154798/⟩.

Urban, S., Shah, J., Rogers, M., 1993. An overview of the ASU engineering database project: interoperability in engineering design. In: Proceedings RIDE-IMS '93: Third International Workshop on Research Issues in Data Engineering: Interoperability in Multidatabase Systems. IEEE, Vienna, Austria, pp. 73–76. https://doi.org/10.1109/RIDE.1993.281942. http://ieeexplore.ieee.org/document/281942/.

Waltrich, M., Hermes, C.J., Goncalves, J.M., Melo, C., 2010. A first-principles simulation model for the thermo-hydraulic performance of fan supplied tube-fin heat exchangers. Appl. Therm. Eng. 30 (14–15), 2011–2018. https://doi.org/10.1016/j.applthermaleng.2010.05.006. ⟨https://linkinghub.elsevier.com/retrieve/pii/S1359431110002024⟩.

Z. Wang, Digital Twin Technology, in: Industry 4.0 - Impact on Intelligent Logistics and Manufacturing, IntechOpen, 2020, 95-113.10.5772/intechopen.80974.https://www.intechopen.com/books/industry-4-0-impact-on-intelligent-logistics-and-manufacturing/digital-twin-technology.

Wiesner, A., Morbach, J., Marquardt, W., 2011. Information integration in chemical process engineering based on semantic technologies. Comput. Chem. Eng. 35 (4), 692–708. https://doi.org/10.1016/j.compchemeng.2010.12.003

Wikipedia, Microsoft Windows (2022). ⟨https://en.wikipedia.org/wiki/Microsoft_Windows⟩.

Wilkes, W., Brunsmann, J., Heutelbeck, D., Hundsdörfer, A., Hemmje, M., Heidbrink, H.-U., 2011. Towards support for long-term digital preservation in product life cycle management. Int. J. Digit. Curation 6 (1), 282–296. https://doi.org/10.2218/ijdc.v6i1.188. ⟨http://www.ijdc.net/article/view/179/248⟩.

Xie, Y., Liu, J., Liu, H., Ma, Y.-S., 2013. Features and Interoperability of Computer Aided Engineering Systems. In: Ma, Y.-S. (Ed.), Semantic Modeling and Interoperability in Product and Process Engineering, Springer Series in Advanced Manufacturing. Springer, London, pp. 143–191. https://doi.org/10.1007/978-1-4471-5073-2_6. https://link.springer.com/chapter/10.1007/978-1-4471-5073-2_6.

Yli-Huumo, J., Maglyas, A., Smolander, K., 2016. How do software development teams manage technical debt? - an empirical study. J. Syst. Softw. 120, 195–218. https://doi.org/10.1016/j.jss.2016.05.018. ⟨https://linkinghub.elsevier.com/retrieve/pii/S016412121630053X⟩.