https://dx.doi.org/10.1016/j.rcim.2020.101958

To cite this paper: Zheng, P., & Sivabalan, A. S. (2020). A generic tri-model-based approach for product-level digital twin development in a smart manufacturing environment. *Robotics and Computer-Integrated Manufacturing*, *64*, 101958. https://doi.org/10.1016/j.rcim.2020.101958

A Generic Tri-Model-Based Approach for Product-Level Digital Twin Development in a Smart Manufacturing Environment

Pai Zheng[1*] and Abinav Shankar Sivabalan[2*]

[1.] *Department of Industrial and Systems Engineering, The Hong Kong Polytechnic University, Hung Hom, Hong Kong, China*

[2.] *School of Mechanical and Aerospace Engineering, Nanyang Technological University, Singapore*

[*] *Corresponding authors: pai.zheng@polyu.edu.hk;* abinav.sivabalan@ntu.edu.sg;

A Generic Tri-Model-Based Approach for Product-Level Digital Twin Development in A Smart Manufacturing Environment

Abstract

Smart manufacturing, as an emerging manufacturing paradigm, leverages massive in-context data from manufacturing systems for intelligent decision makings. In such context, Cyber-Physical Systems (CPS) play a key role in digitizing manufacturing systems and integrating multiple systems together for collaborative works. Among different levels of smartness and connectedness of CPS, Digital Twin (DT), as an exact digital copy of a physical object or system including its properties and relationship with the environment, has a significant impact on realizing smart manufacturing. A DT constantly synchronizes with its physical system and provides real-time high-fidelity simulations of the system and offers ubiquitous control over the system. Despite its great advantages, few works have been discussed about DT reference models, let alone a generic manner to establish it for smart manufacturing. Aiming to fill the gap, this research introduces a generic CPS system architecture for DT establishment in smart manufacturing with a novel tri-model-based approach (i.e. digital model, computational model and graph-based model) for product-level DT development. The tri-model works concurrently to simulate real-world physical behaviour and characteristics of the digital model. To validate the proposed architecture and approach, a case study of an open source 3D printer DT establishment is further conducted. Conclusions and future works are also highlighted to provide insightful knowledge to both academia and industries at last.

Keywords: digital twin; cyber-physical systems; reference model; smart manufacturing

1. Introduction

The evolution of the cutting-edge Information and Communication Technologies (ICT) have started to push manufacturing lines beyond automation towards smart-manufacturing systems [1]. According to the National Institute of Standards and Technology (NIST) **"***Smart Manufacturing systems are systems that are fully-integrated, collaborative manufacturing systems that respond in real-time to meet changing demands and conditions in the factory, in the supply network, and in customer needs"* [2]. In order to achieve that, Cyber-Physical System (CPS) plays a significant role, which is the result of a continuous evolution of embedded systems into Mobile Ad-hoc Networks (MANETs), Wireless Sensor Networks (WSN) and industrial Internet-of-Things (IoTs) [3,4]. CPS integrates a physical object or a system in real-world with its digital model, either a cyber-twin or digital twin (DT) in the virtual world through sensors, networks and computing devices [5]. Hence, it is a multidisciplinary system that relies on the combination of computation, communication and control technologies (3Cs) to digitize existing systems [6].

DT serves as one of the core technologies to realize CPS [7], which by definition is an exact virtual copy of a physical object or system including its properties and its environment [8]. A DT constantly synchronises with its physical system through continuous data transfer and provides real-time status of the system through ultra-realistic simulations, making the user feel as if operating the physical system from the virtual environment [9]. Besides monitoring, a DT provides deep integration with the control system of the physical system that warrants direct control over the physical system. It can be found that CPS and DT are confusingly similar since they share the common objectives of data digitalisation, connectivity, physical system monitoring and to a greater extent imparting smartness to objects [10]. Nevertheless, the difference lies in the fact that a CPS is a holistic system that includes the physical object, communication interfaces, computational hardware, software applications and the digital model [7], while a DT is an interconnected virtual model that represents the physical object. A CPS can achieve the aforementioned objectives without relying on a DT, while a DT would have to rely on the CPS infrastructure to realise its goal of mirroring the physical system through ultra-realistic simulations and exercising ubiquitous control.

Despite a surge in technological advancements and industry-wide discussions about CPS [11], the technical adaptation and implementation of DTs are still under exploration. Current hardware limitations that restrict real-time data processing could be one possible setback for industries to adopt DTs. Also, very few resources explain the actual infrastructure behind DTs and their implementations, pointing at another major reason for laxness in DT adoptions. Although a great part of existing research has been focusing on identifying the potential areas of DT applications and the benefits of DT in different fields, technical aspects of DT establishment are not well-established. To fill this gap,

this work aims to present a general system architecture for DT establishment by considering both hardware and software aspects, and to uncover the core technologies that develop the DT that can be used as a reference model for future product-level DT implementation. The rest of this work is organised as follows: Section 2 provides a comprehensive review of the related works in the field of DT. Section 3 introduces a generic four-layered CPS infrastructure for DT establishment. Section 4 further proposes a novel tri-model-based approach for product-level DT development. To validate the system and approach, Section 5 presents a case study of an open-source 3D printer DT establishment process. Conclusion and potential future directions are summarised in Section 6 at last.

2. Literature Review

As reported earlier, DT serves as the core part of the holistic CPS, which has successfully been adopted in many sectors including military [12], healthcare [13] and manufacturing [14]. Hence, understanding CPS infrastructure and its implementation are the fundamental basis for DT establishment. Despite those achievements in a broad view, this section restricts its scope by reviewing works related to both CPS architecture and DT implementations for smart manufacturing from existing literature to identify the gaps.

2.1 CPS architecture

In literature, the first prototype architecture for CPS was proposed by Tan et al. [6], in which a global reference time-based event/information-driven approach has been highlighted to establish a network of connected objects between the physical and digital world. Shi et al. [15] conducted a survey on the various application scenarios of CPS, and identified the key issues in CPS implementation including modelling, data transmission, security etc. Despite those generic concepts, CPS has been widely adopted in the manufacturing field. For instances, Lee et al. proposed [7,16] a 5C architecture for implementing CPS in an industrial environment and an efficient way of managing and analysing data through a clustering algorithm for prognostic health management. Meanwhile, Leitão et al. [17] identified the key challenges in implementing industrial CPS architecture by conducting the study on four major European innovation projects. Zheng et al. [18] proposed a CPS-enabled conceptual framework for smart manufacturing involving various aspects of manufacturing such as design, monitoring, control, inspection and scheduling. Liu et al. [19] introduced a augmented reality-assisted computer numerical control (CNC) machine tool based on MT-connect and OPC protocols. Zhang et al. proposed [20] an agent and CPS-based self-organizing and self-adaptive intelligent shop-floor framework to support smart manufacturing.

2.2 DT in the smart manufacturing scenario

Uhlemann et al. [21] believed that DT has a huge potential in real-world applications especially in the manufacturing sector and has a colossal role to play in Industry 4.0. Glaessgen et al. [12] elaborated how DTs can bring about Industry 4.0 transformation and assist in tracking military vehicles in the defence sector. Owing to its deeper ties with the physical counterpart, a DT can be used in various stages of the product development process for project lifecycle management [22] and the relevant works are explained sequentially below.

In the design stage, although a DT cannot make suggestions on new product development, it can provide valuable information during the next-generation of iterative product development [23]. Zheng et al. [24] introduced a data-driven cyber-physical product development approach for closed-loop design iterations and further defined the implementation strategies for value co-creation process [25]. Tao et al. [26] proposed a DT-driven product design approach to assist in the optimised redesign of existing products. Meanwhile, a flexible DT concept using a modular approach for evaluating product designs was suggested by Guo et al. [27]. DebRoy et al. [28] revealed how DT can facilitate additive manufacturing in analysing microstructure, heat transfer properties and structural rigidity of manufactured parts during rapid prototyping.

During the manufacturing phase, DT can be used to overwatch the manufacturing process and provide a status update on the scenario. Lu et al. [29] summarizes the core research issues in the DT-driven smart manufacturing field. For instances, Tan et al. [30] proposed a DT construction framework which models IoT data into a simulation model. Schluse et al. [31] introduced a DT simulation technology for manufacturing facilities, to monitor manufacturing systems throughout its entire life cycle. A reliable and efficient infrastructure for improved communication and interaction between physical and virtual shop-floors using DTs were presented by Ding et al. [32], whereas the key components towards realising a new paradigm of smart factories through a shop-floor based DT were explained by Tao et al. [33]. Moreover, Söderberg et al. [34] put forward a DT system for mass customised production in manufacturing facilities by leveraging its power in simulation to optimize the manufacturing system.

In the distribution and usage stage, for instances, Nikolakis et al. [35] developed a DT for industrial logistical application and tried to improve efficiency by enhanced planning and control. Petković et al. [36] proposed to use virtual reality digital warehouse twins to realistically simulate worker behaviours. Lee et al. [37] presented a predictive and effective system maintenance method by utilising IMS WatchDog agent for analysing data collected through a CPS infrastructure and a cloud-based DT. Haag and Anderl [38] asserted that by developing a DT alongside its physical product and to remain its virtual counterpart till the end of its product lifecycle will provide more effective and streamlined services. Tuegel et al. [39] explained how a DT equipped with a multi-physics model would inherit the characteristics of the physical model and how real-time integrated data will help in better management of the physical object through scheduled maintenance and timely replacements.

Despite the rapid development of information technologies, the industry-wide talk about CPS and DT, and the businesses' interest, the implementation seems to be far from happening [40]. Lee [10] presented various challenges that are looming CPS implementation and considers safety, robustness and agility to be the most important concerns as a physical world is very different from a controlled digital world. After reviewing the existing works, it can be inferred that most works focussed on adopting CPS and DT in a high-level view (e.g. the overall manufacturing system) [41] or leveraging the potential of DTs to reap its benefits, while the development of a generic DT at product-level has not been well-addressed.

3. Generic system architecture for DT establishment

The proposed generic architecture consists of four different layers, the *physical layer*, *data extraction and consolidation layer*, *cyberspace layer* and the *interaction layer*. A schematic representation of the proposed CPS architecture showing four different layers for the establishment of a digital twin is shown in **Figure 1**. The physical layer includes all the physical items in their working environment. The data extraction and consolidation layer act as an interface between the physical layer and the cyberspace layer by data digitalisation, transmission and reception. The cyberspace layer forms the crux of this framework and is the layer in which the DT exists and exercises control over the entire infrastructure. The final layer is the user interaction layer which presents the realised objective of DT, ubiquitous monitoring and control to the users. The proposed framework differs from previously proposed frameworks in its core, the actual DT in the cyberspace, by utilising a tri-model based approach to add more versatility. The tri-model based approach provides vast and deeper interaction with the physical model than the conventional geometrical design-based DT with cognitive intelligence by leveraging the state-of-the-art artificial intelligence.

To actualise its two constitutional purposes, a DT should have two modes of operation, a *monitoring mode* and a *control mode*. In the monitoring mode, the DT operates like a cyber-twin and mirrors the physical object like a digital shadow while retaining limited supervisory powers over the physical object. Meanwhile, in the control mode, the DT exercises complete command over the physical system and acts as the system's command centre. Throughout this paper, DT functions and behaviours are explained separately based on these two modes of operation to avoid conflict of control precedence between a command given to the physical system directly and through a DT. More details on the limited supervisory powers and control precedence are explained in the sections below to present the individual layers of the DT framework.

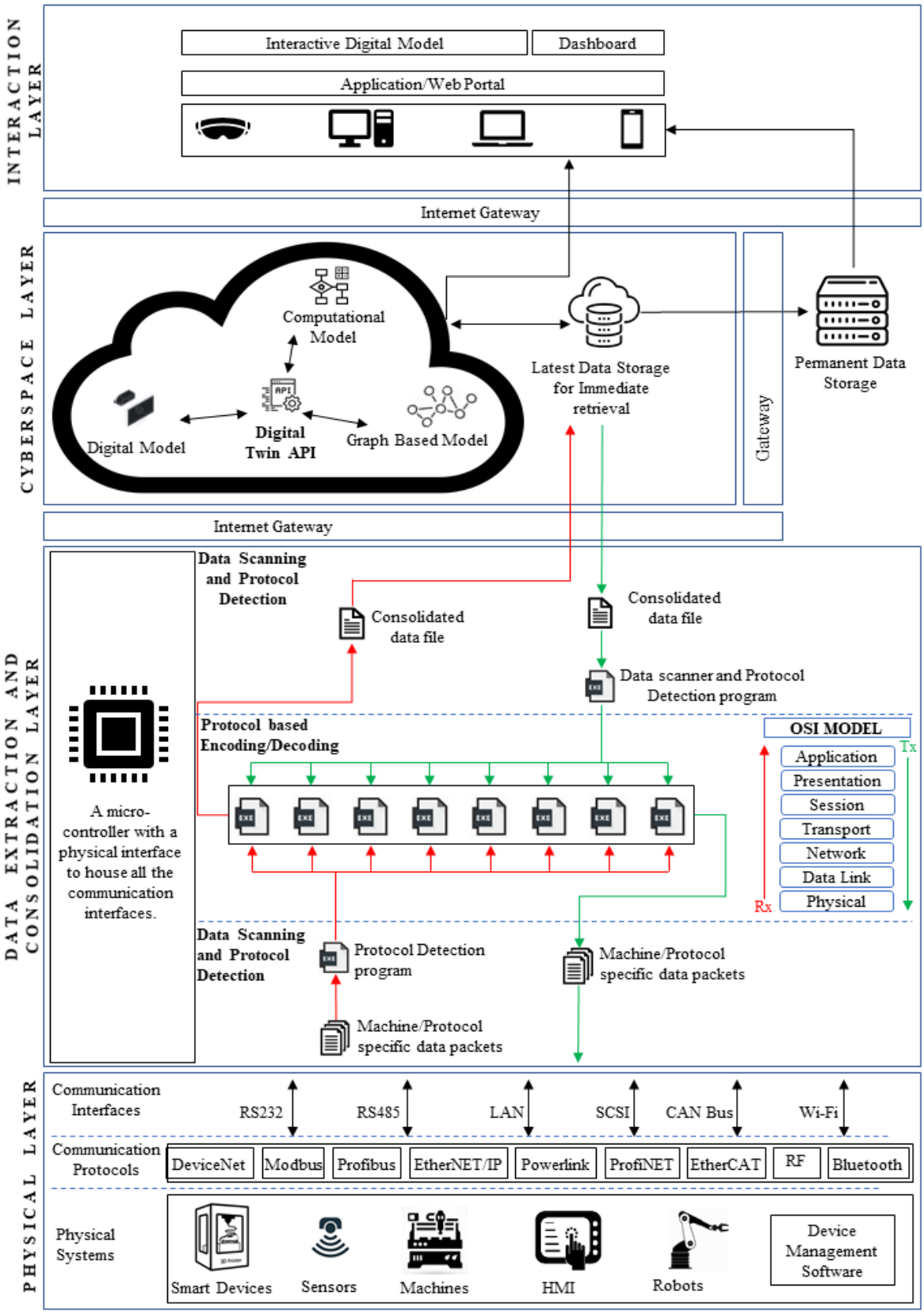

Figure 1. The proposed CPS architecture for DT development.

3.1 Physical Layer

The physical layer includes the physical system under observation along with its environment and their interaction as shown in Figure 1. The physical layer may include numerous machines each following different communication protocols as well as different communication interfaces for data transfer. For these systems to be monitored digitally, they must be digitalized and made available on the digital network. Digitalization of systems is done by acquiring complete data about the system and its environment throughout its entire product life cycle through various data acquisition methods either directly or with the help of additional sensors and devices. Depending on the communication protocol utilised, the data obtained from each system will be encoded in a standard prescribed in the communication protocol based on the OSI model. This results in an enormous amount of data packets that must be interpreted separately from one another making it difficult for further processing.

During the control mode, the physical layer is the end actuated layer as opposed to the monitor mode in which the physical layer is the initiation point. The control signals or commands that are sent as data packets from the digital twin based on user input actuates the physical model accordingly. For this, the control systems connected to the individual physical systems must be open, active and be able to act upon live commands. In short, during the control mode, the digital twin acts as a high-level SCADA software that manages the entire physical system but with and additional visual representation of the entire physical world. Also, the DT is intelligent, interactive and can continuously evolve by virtue of Machine Learning (ML) and Artificial Intelligence (AI).

3.2 Data Extraction and Consolidation Layer

Data extraction and consolidation layer act as a bridge between the physical and cyber layer by receiving all the data packets from the physical systems, processing them and converting them into a machine-readable form before passing it on to the cyber layer. The data packets that are received from the physical layer are encrypted according to the communication protocol used by the physical systems. As it is extremely inefficient to process data from every object in the physical system independently, the collected data must be consolidated in a standard machine-readable format. The module contains a processing device such as a microcontroller or computer while also providing the necessary infrastructure to connect the various communication interfaces from the physical layer. All the communication protocols follow the OSI model for data transmission and reception with each protocol using its unique method of encoding and decoding techniques. Hence, the data extraction module first runs a scanning algorithm on the incoming data to identify the communication protocol it uses. The identified data packets are then sent to the respective extraction software that decodes the seven layers of the OSI model to obtain the data passed on by the machine as shown in Figure 1. The data from different machines are presented in a machine-readable unified data format such as a .txt or .csv file. The files are then uploaded to the cloud from which the data can be accessed universally.

On contrary, during the control mode, the control signal to individual systems are sent as consolidated information from the layers above and must be distributed to the physical systems. The data from the cloud is read continuously and the module scans the incoming data, identifies the physical object to which the information is to be sent and determines the communication protocol associated with the machine. It then passes the respective data to the corresponding sub-program that enforces the seven layers of the OSI model on the data and converts them into smaller data packets in the corresponding machine-readable form and are finally conveyed over to the physical layer.

3.3 Cyberspace layer

Cyberspace layer forms the core of the DT architecture and is the layer in which DT is established. The cyberspace layer contains the proposed Tri-model DT established in the cloud and cloud storage for data handling. To provide ubiquitous access to the physical system, the digital twin is established on the cloud and the internet serves as the communication medium so that the DT can be accessed from anywhere. The DT is not just a visual replica of the physical system but a more advanced digital version of the physical object that includes all the physical and chemical properties of the system and thereby obeys the laws of science in a similar manner to the physical system. In order to make this realistic DT, the digital model is augmented by a backend graph-based model that inhibits the digital model through constraints that impose laws of science upon the digital world and a computational model that evaluates the digital model's real-time condition as well as its feasibility to perform a future task as mentioned in Figure 1. These three models are interconnected and controlled by a Digital Twin API that is also responsible for data transmission and reception to and from the cloud storage. Cyberspace contains a web data storage server that stores the raw data obtained from the data extraction and consolidation layer as well as it's corresponding processed information. The processed information is obtained from the tri-model digital twin through the digital twin API. As the amount of cloud

storage is limited and expensive, only the most recent and latest data is stored in the cloud for immediate retrieval. After which it is transferred to offline storage where it is permanently stored.

In the monitor mode, data that is transferred from the data extraction and consolidation layer is first stored in the server. The data is obtained by the DT API which passes it on to the three individual models to perform their respective task. The updated information is stored back in the server corresponding to the input data and is also forwarded to the interaction layer. In the control mode, the user input from the interaction layer is received by the DT API, first stored in the server, then processed, verified and updated visually by the computational, graph-based and digital models respectively. The information is stored once again in the server corresponding to user input while also being transferred to the data extraction and consolidation layer.

3.4 Interaction layer

Interaction layer is the final layer of the DT framework and it allows human interaction with the physical system by way of the digital twin. The existence of DT in the cloud space provides opportunities to access them from anywhere on any device, even those with low processing power as almost all the work is done in the cloud space and only the result is projected or streamed to the accessing device through a secure web portal. Hence, the interaction layer can be any device that can present visual information while recognizing user interaction. It can be as simple as a smartphone or a mixed reality headset that can provide an immersive experience as shown in Figure 1. In the monitor mode, the DT in the cloud space processes all the information and updates the digital model. This updated digital model is then presented along with the relevant real-time status of the physical system to the user in an interactive webpage or an executable program. In the control mode, the interaction layer acts as an HMI with a visual representation of the system via the digital twin. The DT provides real-time synchronisation and model update with the physical system and thereby provides the user with immersive control rather than a conventional approach. The interactive model translates the user input through trigger points that execute respective actions and passes it onto the cyberspace as command signals which are then processed by the DT API and passed on to the physical world.

4. Tri-model-based approach for product-level DT development

The DT is a complex digital object requiring huge amounts of data, with visualizing and rendering capability, computational power and intelligence, which is achieved by fusing different technologies together. The digital model can be developed by using any existing CAD modelling software through external APIs. Since the DT operates in real-time during the whole lifecycle of the physical object, a data handling software is also required to process the enormous amount of data generated. A graph-based model (NoSQL database) can establish multiple relationships among physical systems and within them considering both structured information of the objects and the unstructured data from the context (e.g. documents). Hence, it also ensures that the life-time data of the digital twin is stored which can be used for product analysis and next-generation product development. As mentioned earlier, the high-level computations required to impart real-world characteristics to the digital model are performed on a computational software and the real-world constraints are added through a graphical model.

4.1 Digital Model

Two most commonly used and prominent ways of creating a digital model include parametric modelling and solid modelling. Parametric modelling makes use of numeric values like dimensions, equations and their relations to control the geometry. This reduces the complexity in designing a system that interacts with other digital objects by means of numeric data. In a real-world scenario, there may be more than one instance of the physical object or a manufacturer may opt to provide digital twin as a service for the customers. In such cases, for every physical object made, a unique digital twin must be associated with it throughout its entire product cycle. This creates the need for creating multiple instances of the same digital twin with minor variations because of irregularity in manufacturing to be created. In order to overcome the difficulty of making individual digital models for every physical object, a parametric reference model can be made, as shown in **Figure 2**. Just like the physical systems, the environment in which the system exists and with which it interacts also plays a crucial role in determining the efficiency and service life of the system. The objects' environment may include physical objects such as other systems, human operator etc., or environmental factors such as wind, moisture etc., that has an influence on the system's performance. For example, while modelling a Digital Twin for an aircraft, the aircraft's performance varies according to the external air pressure, airstream velocity, air density and moisture content along with the temperature. In such scenarios, airflow or an extremely large wind tunnel is modelled as an environment and the parameters are updated real-time through sensor data and its interaction is controlled by the backend computational model with reference from the graph-based model

and the working conditions. Hence in order to exactly compute the status and performance of the system its surrounding is also essential. Hence, the working environment of the object must also be recreated digitally, referred to as the digital twin environment, also shown in **Figure 2,** to simulate real-world experience on the digital model.

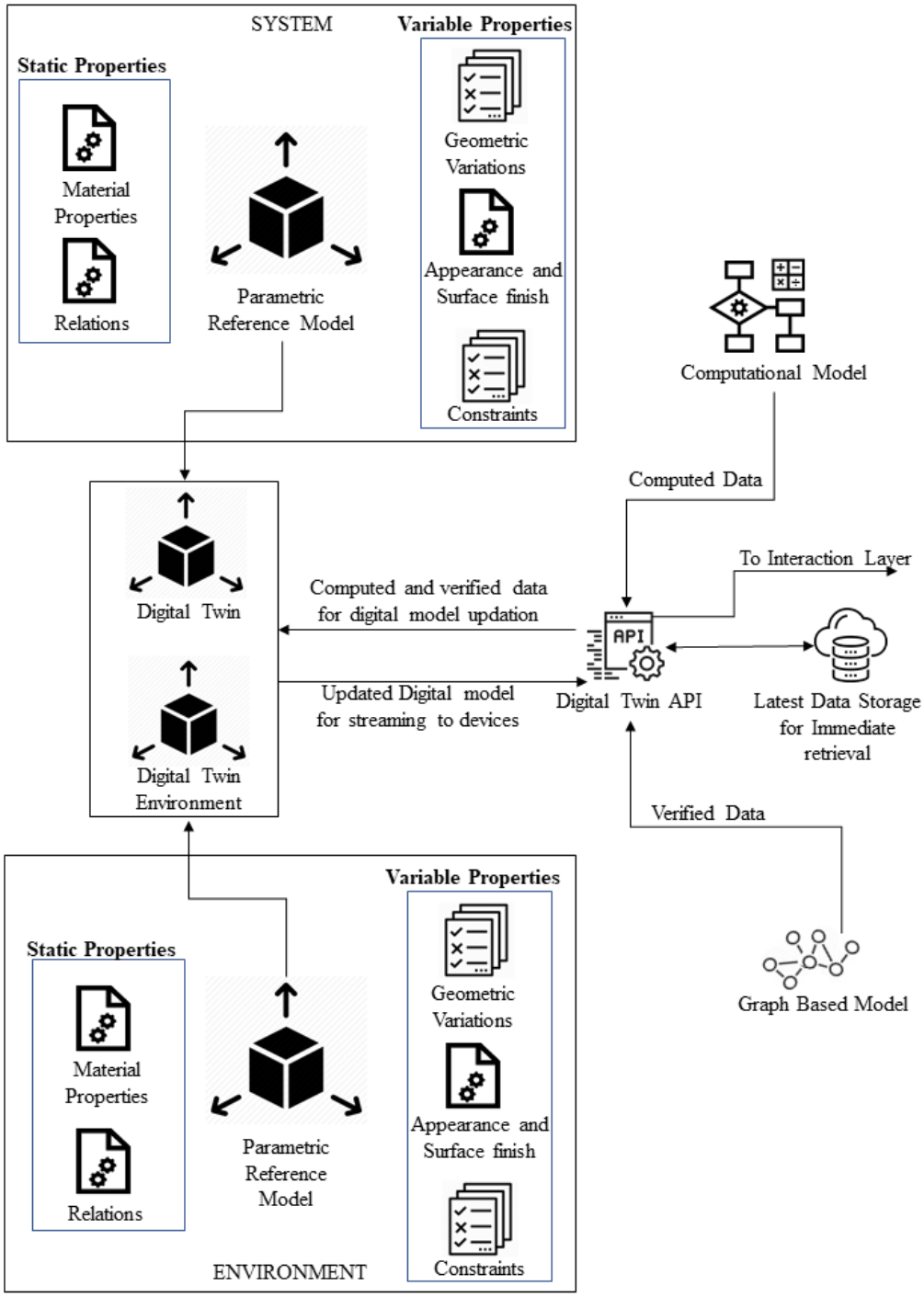

Figure 2. Digital model establishment and its interaction with other models

The reference model is a flexible model made by incorporating all the tolerance and variations the physical system can possibly have and is scalable and amendable within the prescribed limits. The reference model must include static properties that are constant among all instances of the DT such as material properties, geometrical constraints that define the degree of freedom of the system etc., of the physical system to maintain similarity and consistency with

the physical system. A knowledge base of all possible variations of the physical object including its surface properties like appearance, roughness, lustre etc., and other geometrical variations based on Design for Manufacturing and Assembly (DFMA), Geometric Dimensions and Tolerancing (GD&T) and past experiences are developed to incorporate all these values into the reference model. Once the reference model is created, a new twin can be derived as a subset of it, instead of creating one individually from the scratch saving time and manpower.

4.2 Computational Model

Besides the geometrical constraints, the laws of physics that govern the physical object must be added to a digital model to make it a 'real-twin'. It is also necessary to include these physical attributes so that the digital twin can be used in a simulated run to determine the point and cause of failure before operating the physical system. The need for a multi-physics model to capture the DT's interaction with its environment and its effect on the structure and geometry have been explained by [39]. An object's interaction to thermal loads and its effects are defined through thermodynamic models and computational fluid dynamics models, static and dynamic loads through stress-strain models and structural dynamic models, the effect of chemical agents in the environment like humidity through a chemical model and others depending on the factors involved. The conventional method of product evaluation includes solving the different analytical models separately and evaluating them individually for failure conditions. A digital twin provides a platform to combine the different analytical models together and integrate them into a single entity to perform more realistic analysis and evaluations on the object, as shown in **Figure 3**.

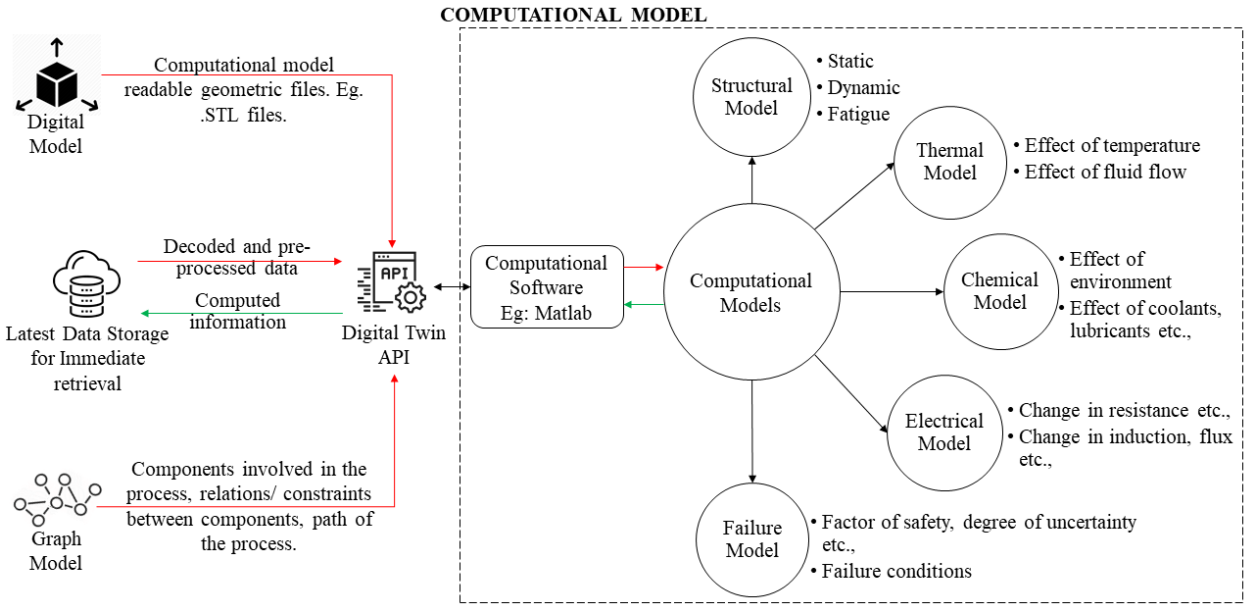

Figure 3. Computational Model architecture and its core components

The computational model only provides the numerical values regarding the status of the system based on the real-time data collected. Whenever a process is initiated in the physical system, the DT API consults the graph-based model on the components involved in the process directly or indirectly, the effects the system is subjected to and the type of analysis to be performed are all provided by the graph-based model. The entire process chart of computational model performance is presented in **Figure 4.** With this information, the data from the cloud storage server is transferred to the computational model for computation. The results are then recorded, and the status of the machine is presented to the user in the interaction layer. These analyses can be carried out in certain CAD modelling software directly with the help of built-in simulation features, but the flexibility provided by computational models to provide our own conditions, parameters and the complete availability of data from every individual step are some of the advantages of using a separate computational model. Also, this approach is the only possible way when modelling software without simulation support is utilised. Besides, the computational model unburdens a certain workload from the digital model and hence this activity is less resource-intensive and faster than the simulation approach.

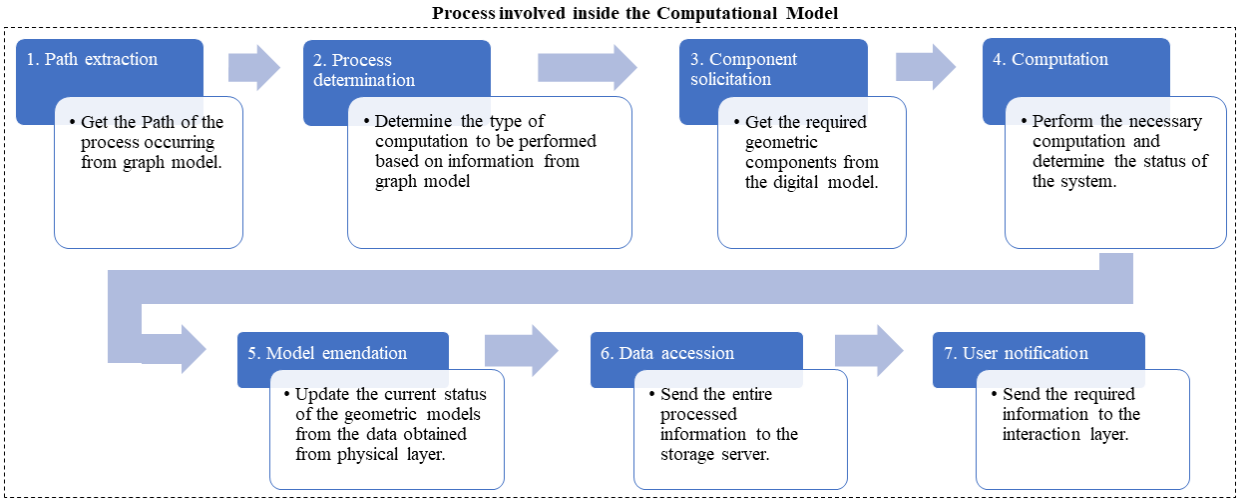

Figure 4. Working process of Computational model

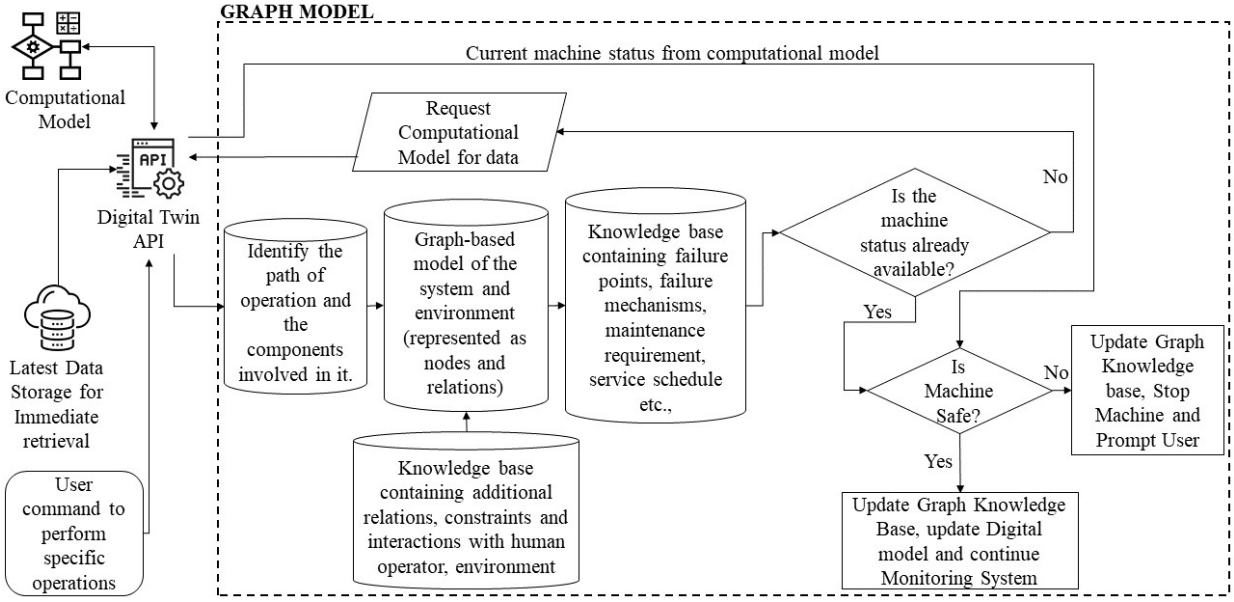

Figure 5. The underlying framework of the graph-based model.

4.3 Graph-Based Model

When a CAD environment is used to develop a digital model, the constraints and relations can be defined in the CAD program itself. Nevertheless, if the digital model is developed in a graphics modelling software such as Blender, then the amount of real-world constraints and relations that can be defined to the model is restricted. Also, the digital model includes a digital copy of the physical system and its environment. In such cases, the interaction between the environment and the system cannot be defined in the CAD environment. For example, human interaction with an HMI on the physical system, the wind blowing over an object, etc. These kinds of interactions and relations can be defined in a graph-based model, where a complex and innumerable amount of interactions and relations can be set up and stored, as shown in **Figure 5**. On top of these, the physical object is restricted from utilizing its functionality to a full extent because of reasons such as pre-built safety measures to avoid catastrophes, cycle-time optimizations, etc. These features may vary depending on the intended use of the product and impose additional restrictions on the physical system other than geometrical constraints.

A graph-based model can be represented as $G =< N, E >$, where N stands for nodes of graph and E means the edges between nodes. According to our previous work [42], there are three main types of Ns, including product

component (*P*), service component (*S*) and context information (*C*), and the edge in-between each two nodes represents the interrelationship of them. Hence, this graph-based model can help store the core information of the physical system with the ever-growing interrelations and nodes through data collection process. More importanatly, it enables the context-aware cognitions to guide the digital model and the computational model in performing the necessary tasks. For instance, the real-time variation of extrusion speed (*C*), will affect/result in the change/upgrade of the extruder module (P) or monitoring service (C). Following such manner, some artificial intelligence (AI) approaches of the computation model can be leveraged in predicting/recommending the best solution in a graph-based learning manner [43], and to make the physical product ever smarter (e.g. cognitive manufacturing).

4.4 Digital Twin API

Although each model has a pre-determined function of their own, they must be co-ordinated and the data flow among them must be facilitated in a smooth way. Hence an external program that can interact with the three models such as an API was developed. This API references the standard libraries of each modelling platform to interact with the platform externally and to impart additional functionality to the platform. By utilising the libraries of all three platforms, the API can direct those platforms by acting as the control centre of the Tri-model DT as shown in **Figure 6** below. The DT API has three sub-programs, to coordinate with each model. Each sub-program has the respective reference libraries and commands to invoke the respective software platform on which the model is built. For example, when the digital model is developed in SolidWorks, the digital model API or sub-program will contain SolidWorks reference libraries and objects to that will provide an external program to access the digital model within SolidWorks. Similarly, for the computational model, if MATLAB is used, MATLAB libraries and other sets of definitions to access and run a specific MATLAB script within the MATLAB environment and return the results are defined. Similarly, for graph-based models, the respective libraries are used. The DT API also contains its own set of the code block to coordinate and control the data flow among each DT model and stands responsible for storing and retrieving data and information to and from the storage.

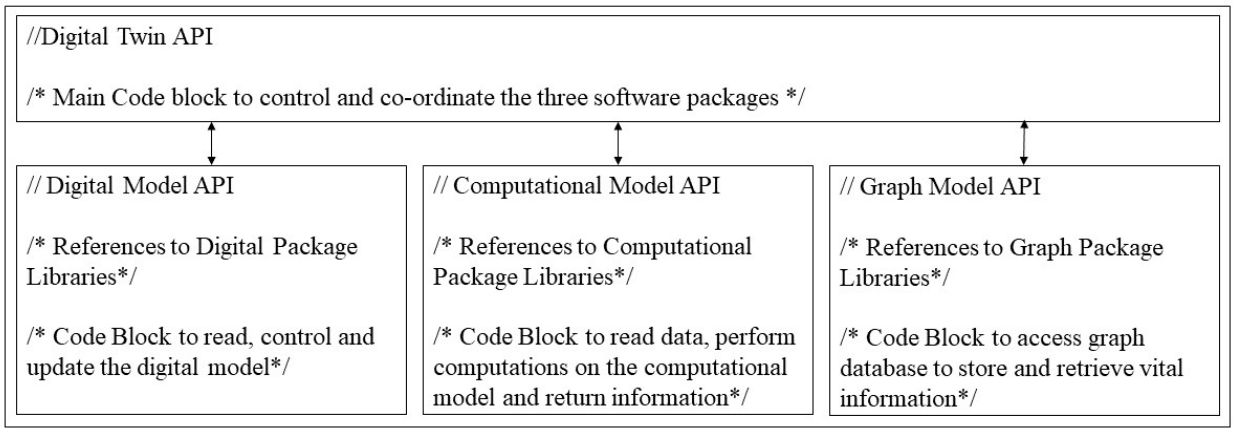

Figure 6. A simplified framework of DT API.

In the monitor mode, the DT API detects the process being carried out by the physical system through its ties with the physical systems control. The DT API requests the graph model to identify the components of the system involved in the process by pathfinding algorithm as shown in Figure 4. The nodes in the identified path become the active nodes and the components of the physical system involved in it are detected. Based on the process, the type of effect the components will be subjected to is also determined from the knowledge base. The DT API passes on this information along with the data obtained from the physical layer to the computational model and requests it to perform the necessary computations as presented in Figure 3. The DT API cross-references the results provided by the computational model with the graph-based model knowledge base to verify if the system is under the safe operating condition as mentioned in Figure 4. If the system exceeds the safety limit, the DT API intimates the user and stops the physical system immediately. Once the backend processing is done, the digital model is updated to visually represent the status of the system as shown in Figure 2. The digital model and the processed status information of the system are then forwarded to the interaction layer for user monitoring.

In the control mode, the user inputs are converted as corresponding data signals which are then received by the DT API. Based on these signals, the DT API requests the graph-based model to identify the path and the active nodes

of the graph from which the components, the effects they are subjected to are determined. The computational model then analyses and presents the results to the graph-based model through DT API. If the results are in the safe zone, the digital model is updated, and the data is passed on for storage and to the data extraction and consolidation layer. As mentioned in Figure 4, the graph-based model references the database to check if the data and corresponding processed information are already available. As the DT API stores every data and its corresponding processed information, if a match is found, the information is retrieved instead of computing again. This makes the DT faster as it evolves by collecting more data.

5. Case Study and Discussions

To demonstrate the proposed framework and approach, an open-source 3D printer has been chosen for DT implementation. The chosen 3D printer, ANET A8, is a Prusa i3 clone that operates on the RepRap platform. The ANET A8 is also an entry-level 3D printer with basic functionalities such as printing from a micro-SD card or through USB serial communication. The open nature of ANET A8 allows easy customization and upgradations. Initially, the stock firmware present in the 3D printer's motherboard was replaced by Marlin, a widely used open-source firmware that supports additional functionalities such as multiple screen output, sensor and nozzle additions, etc.

5.1 DT establishment with a 3D Printer

The DT is established in a sequential way as explained below. The physical world contains the physical system, the ANET A8 3D printer. The 3D printer uses an RS232 serial interface for data transmission. A Raspberry Pi, connected in serial communication with the 3D Printer, acts as the data extraction and consolidation module, acting as an infrastructure that bridges the physical world with the cyberspace. A backend python software has been developed to communicate with the 3D printer via serial communication. The software requests and receives encapsulated data from the 3D printer five times per second translates them into useful information and writes the data to a temporary file. Another program simultaneously uploads the data to the cloud server. In control mode, the program downloads the data from the server and the data is read, translated into G-Code (3D printers work on G-Code) and the commands are passed over to the physical printer.

On the other side, a Digital Twin API in the form of a standalone application has been developed in Visual C# using WPF App and .Net framework 4.5 in Visual Studio to act as the bridge between the three DT models. By referencing the libraries and SDK of the respective programs, the API can interact with and add additional functionalities to the programs. The API references the libraries and SDK of the program it interacts with and can be used to add any sort of additional functionalities to the programs. The API acts as a command centre for the three digital twin models that reside within three different environments and ensures smooth information flow across them as and when required. The API overwatches the operating mode of the digital twin i.e. control mode or monitor mode and assigns the priority either to the digital system or the physical system respectively. The API also downloads raw data from the storage server and uploads processed information to the server. Besides, the Digital Twin API handles all the communication to and from the interaction layer, and the schematic of the developed DT is shown in **Figure 7**.

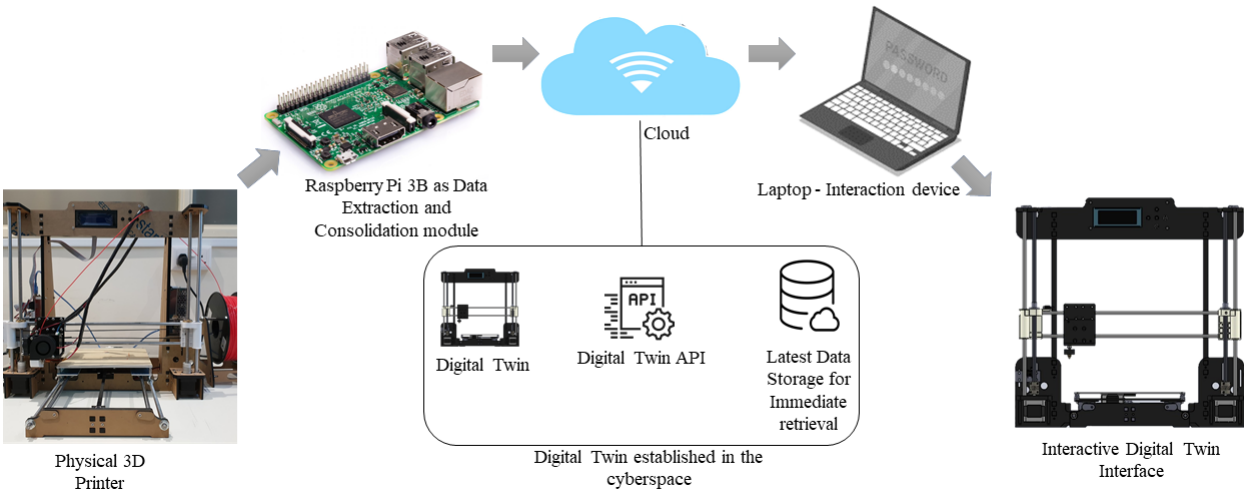

Figure 7. DT implementation for the 3D printer case.

A parametric model of the ANET A8 3D printer has been developed in SolidWorks and the assembly relations, geometrical constraints, material properties and surface features are added to the digital model to give it a realistic effect. As mentioned, it is essential to make a base reference model from which the individual digital twins can be derived. Since the 3D printer is a small tabletop 3D printer, its environment is only the desk on which it is placed and the air surrounding it. The desk has no effect on the 3D printer besides acting as a support and hence it is neglected. The surrounding air and room temperature are included in the computational model. A knowledge base of all possible geometric variations such as the tolerance limits, minor manufacturing and assembly defects that are accepted in quality check, customization features such as colour, surface finish etc., are required to incorporate all these variable parameters into the base model to develop a reference model. The digital model of the 3D printer and a corresponding front end interface developed for user interaction is shown in **Figure 8**.

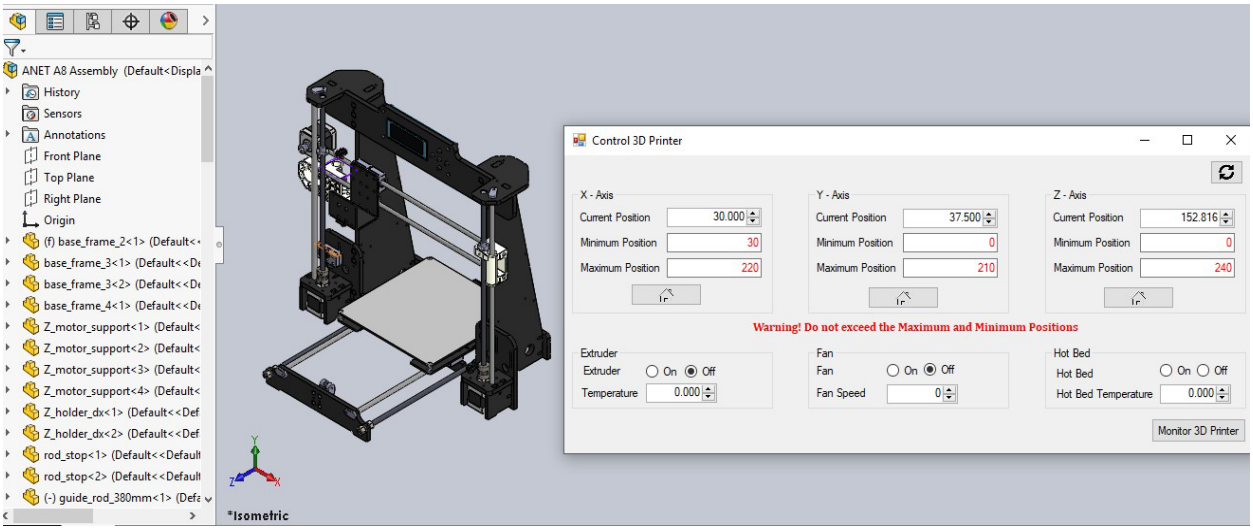

Figure 8. The interactive 3D model and control interface of the 3D printer case.

A digital twin can provide an ultra-realistic digital copy of the physical object only by the addition of its physical attributes. Due to the limited availability of sensor information from the 3D printer, developing a complete structural, physical and thermal model of the system to provide the real-time physical condition of the system was not possible. The 3D printer was only capable of sensing temperature changes in its extruder and hotbed, speed of the fan attached to the extruder and the nozzle and the x, y and z axes movements. Additional sensors could have been utilized for data collection and a more accurate simulation but there are other hardware limitations such as the Raspberry Pi which had a weak processor to handle huge amounts of real-time data. Hence the computational model containing only the thermal models of the extruder and the hotbed was considered feasible. The computational model was developed in MATLAB using MATLAB's built-in functions for thermal analyses. The parts of the digital 3D printer are made in .STEP format so that it can be read by MATLAB. Once the DT API provides information for computation from the graph-based model, the computational model obtains the necessary .STEP files and perform computation. A snippet showing the thermal model used for analysing the hotbed has been shown in **Appendix A**.

A graphical model of the ANET A8 3D printer developed in Neo 4J is shown below in **Figure 9**. The graphical model is essentially a knowledge base that acts as a rulebook for the digital twin and makes sure the digital model operates within acceptable or correct limits. The graphical model and computational model work in conjunction to determine the status of the system and warns the user if the system is critical or when it is subjected to peak load. As mentioned earlier, the DT stores incoming data and its corresponding processed information. Hence, as the amount of data collected increases over time, the DT will become faster and will be able to provide instantaneous results for simulations. When combined with the power of ML and AI, the graph model can be evolved to provide customized suggestions and a study on combining DT with AI is being carried out and will be presented in future work.

As mentioned earlier, the three models of the digital twin are coordinated by the Digital Twin API that acts as a command centre for the Digital Twin. The DT API receives the transmitted data from the cloud, cross-references it with the Neo4J graphical model, asks the MATLAB computational model to perform computations if needed and finally passes the processed information to the SolidWorks digital model. The processed information from MATLAB is once again cross-checked with the Neo4J to see if the operations are safe. If the results yield an unsafe value the DT stops the system and prompts the user in the interaction layer. The information is also stored in the cloud corresponding to the data. For example, the extruder nozzle is continuously heated during the 3D printing process. The thermal computational model computes the heat map and if the thermal stress exceeds the limit of the copper extruder, the process is stopped. Instead of the thermal stress, if the temperature value exceeds the maximum value (due to material blockage or insufficient airflow), the process is halted by the information from graph-based model even before computing. The Digital Twin requires extensive processing power because of the incredible amount of real-time data being handled and the computation involved. Hence, it is set up on the cloud instead of the end-user platform and an interactive dashboard has been developed that can stream contents of the digital twin to the end-user. This allows the end-user to utilize the power of digital twin from any device with an internet connection making it ubiquitous. When the user commands the DT from the interaction layer, the same process is carried out in the DT API.

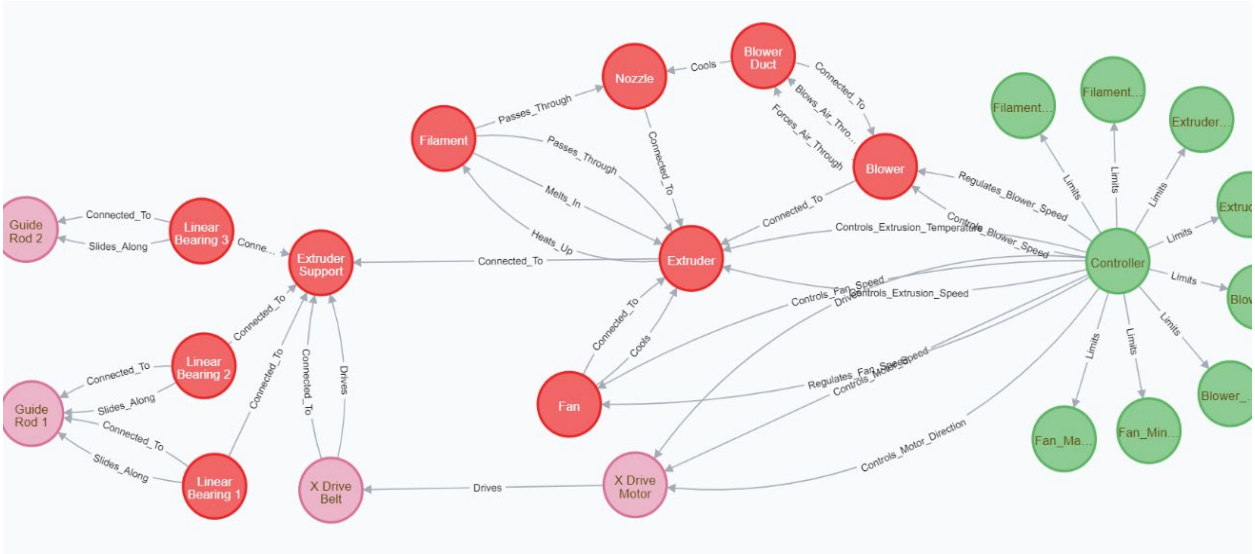

Figure 9. An example of graph-based model of the 3D Printer extruder developed in Neo4j.

5.2 Discussions

As mentioned, the limitation of the proposed approach of DT is the extensive processing power required for computing. The time taken to process the data in the DT takes 2-3 seconds in average which results in a 2-second delay for updating the digital model in case of monitor mode and a 2-second delay to update both the digital and physical model while in control mode. Since the data from the physical world and user inputs from the interaction layer are captured real-time, the delay in processing causes the DT API to skip data that has been collected in the meantime. It must also be noted that this is a simple system with limited data from both the physical world and the interaction layer. In a highly complex setup, this problem will be more significant. Nevertheless, the research provides the benefit of having a DT which can simulate real-world behaviour of the physical system and the added intelligence of graph-based and computational model to provide a significant improvement over DTs with just the digital model. The computational model of the 3D printer helps in regulating the thermal stress and keeps the system temperature low by halting the process thereby reducing the failure rate of extruder and hot-bed. The computational model along with the graphical model serve to improve the quality of the system rather than improving productivity or efficiency. However, in the future work will be done to incorporate ML into the system, which, with the data can regulate the system to operate within preferred parameters.

The proposed DT establishment framework can serve as a reference model for DT implementations including the necessary CPS infrastructure required to set up a functional DT. It also provides a more compelling model for data collection and utilisation compared to conventional DAQs. The tri-model based approach justifies the added cost of DT implementation by providing actual information on product failure and helps in failure prediction besides

providing remote monitoring and ubiquitous control. Furthermore, the graph model can provide information on the most used features and the highly utilised components based on query requests, which facilitate next generation product improvement.

6. Conclusion and Future work

Smart manufacturing has been the hot topic of the decade and ever-increasing industries are implementing smart production lines to manufacture the products. Although CPS is widely adopted, the real implementation of DT has often been overlooked. Despite most existing high-level discussions, little considers the product-level reference model, as a generic approach, to establish the DT. To fill these gaps, this work first introduced a generic 4-layered CPS system architecture, including the physical layer, the data extraction and consolidation layer, the cyberspace layer and the interaction layer, serving as the fundamental basis for DT establishment. Moreover, a tri-model-based reference model for product-level DT development was proposed, where the core scientific contributions are summarised below:

1) *Digital Model* provides not only a virtual representation of the physical objects, but also the ubiquitous access and control over it, where high-fidelity simulation can be performed.

2) *Computational Model* helps to predict the system behaviour and health condition through computation and ensures safe operating conditions of the system.

3) *Graph-based Model* supports the intelligent decision makings and stores the ever-growing user and system generated data, thereby reducing the workload on the computational system. It also helps in identifying the most significant process and component of a system to facilitate future product improvement.

By leveraging this model, the product-level DT can be simulated in a more realistic way and it can be used for predictive analysis for successful completion of a task or accurate scheduling of maintenance, repair and etc. A case study of 3D printer DT implementation has been further conducted to demonstrate the proposed framework and approach. However, there are some limitations in the current implementation. For instance, the processing time is significant and the digital model update and the results in the interaction layer suffer from large latency. Nevertheless, despite the hardware limitations, it did not affect the generic adoption of this DT reference model for industries moving towards smart manufacturing. Moreover, it is suggested that future works should focus on optimising the DT for real-time computing and reduced latency (e.g. cloud-edge computing infrastructure), and the incorporation of AI into the DT (e.g. machine learning techniques). It is envisioned that the power of a DT largely relies on the amount of data that it can extract and process into meaningful context-aware values to both manufacturers and end-users.

Acknowledgement

The authors would like to acknowledge the financial support of the Start-up Fund for New Recruits (1-BE2X, Project ID: P0031040) from the Hong Kong Polytechnic University, Hong Kong.

Reference

[1] H.S. Kang, J.Y. Lee, S. Choi, H. Kim, J.H. Park, J.Y. Son, B.H. Kim, S. Do Noh, Smart manufacturing: Past research, present findings, and future directions, Int. J. Precis. Eng. Manuf. - Green Technol. 3 (2016) 111–128. https://doi.org/10.1007/s40684-016-0015-5.

[2] NIST, Product Definitions for Smart Manufacturing, (2018).

[3] F.J. Wu, Y.F. Kao, Y.C. Tseng, From wireless sensor networks towards cyber physical systems, Pervasive Mob. Comput. 7 (2011) 397–413. https://doi.org/10.1016/j.pmcj.2011.03.003.

[4] S. Basagni, M. Conti, S. Giordano, I. Stojmenovic, Mobile Ad Hoc Networking, IEEE, 2004.

[5] S.P.A. Datta, Emergence of Digital Twins - Is this the march of reason?, J. Innov. Manag. 5 (2017) 14. https://doi.org/10.24840/2183-0606_005.003_0003.

[6] Y. Tan, S. Goddard, L.C. Pérez, A prototype architecture for cyber-physical systems, ACM SIGBED Rev. 5 (2008) 1–2. https://doi.org/10.1145/1366283.1366309.

[7] J. Lee, B. Bagheri, H.A. Kao, A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems, Manuf. Lett. 3 (2015) 18–23. https://doi.org/10.1016/j.mfglet.2014.12.001.

[8] E. Negri, L. Fumagalli, M. Macchi, A Review of the Roles of Digital Twin in CPS-based Production Systems,

Procedia Manuf. 11 (2017) 939–948. https://doi.org/10.1016/j.promfg.2017.07.198.

[9] A. Parrott, L. Warshaw, Industry 4.0 and the digital twin, Deloitte Univ. Press. (2017) 1–17.

[10] E.A. Lee, Cyber Physical Systems: Design Challenges, Berkeley, 2008.

[11] D. Lucke, C. Constantinescu, E. Westkämper, Smart Factory - A Step towards the Next Generation of Manufacturing, Manuf. Syst. Technol. New Front. (2008) 115–118. https://doi.org/10.1007/978-1-84800-267-8_23.

[12] E. Glaessgen, D. Stargel, The Digital Twin Paradigm for Future NASA and U.S. Air Force Vehicles, 53rd AIAA/ASME/ASCE/AHS/ASC Struct. Struct. Dyn. Mater. Conf. AIAA/ASME/AHS Adapt. Struct. Conf. AIAA. (2012) 1–14. https://doi.org/10.2514/6.2012-1818.

[13] K. Bruynseels, F.S. de Sio, J. van den Hoven, Digital Twins in health care: Ethical implications of an emerging engineering paradigm, Front. Genet. 9 (2018) 1–11. https://doi.org/10.3389/fgene.2018.00031.

[14] X. Sun, J. Bao, J. Li, Y. Zhang, S. Liu, B. Zhou, A digital twin-driven approach for the assembly-commissioning of high precision products, Robot. Comput. Integr. Manuf. 61 (2020) 1–14. https://doi.org/10.1016/j.rcim.2019.101839.

[15] J. Shi, J. Wan, H. Yan, H. Suo, A survey of Cyber-Physical Systems, 2011 Int. Conf. Wirel. Commun. Signal Process. WCSP 2011. (2011). https://doi.org/10.1109/WCSP.2011.6096958.

[16] B. Bagheri, S. Yang, H.A. Kao, J. Lee, Cyber-physical systems architecture for self-aware machines in industry 4.0 environment, IFAC-PapersOnLine. 28 (2015) 1622–1627. https://doi.org/10.1016/j.ifacol.2015.06.318.

[17] P. Leitão, A.W. Colombo, S. Karnouskos, Industrial automation based on cyber-physical systems technologies: Prototype implementations and challenges, Comput. Ind. 81 (2016) 11–25. https://doi.org/10.1016/j.compind.2015.08.004.

[18] P. Zheng, H. Wang, Z. Sang, R.Y. Zhong, Y. Liu, C. Liu, K. Mubarok, S. Yu, X. Xu, Smart manufacturing systems for Industry 4.0: Conceptual framework, scenarios, and future perspectives, Front. Mech. Eng. 13 (2018) 137–150. https://doi.org/10.1007/s11465-018-0499-5.

[19] C. Liu, S. Cao, W. Tse, X. Xu, Augmented Reality-assisted Intelligent Window for Cyber-Physical Machine Tools, J. Manuf. Syst. 44 (2017) 280–286. https://doi.org/10.1016/j.jmsy.2017.04.008.

[20] Y. Zhang, C. Qian, J. Lv, Y. Liu, Agent and cyber-physical system based self-organizing and self-adaptive intelligent shopfloor, IEEE Trans. Ind. Informatics. 13 (2017) 737–747. https://doi.org/10.1109/TII.2016.2618892.

[21] T.H.J. Uhlemann, C. Lehmann, R. Steinhilper, The Digital Twin: Realizing the Cyber-Physical Production System for Industry 4.0, Procedia CIRP. 61 (2017) 335–340. https://doi.org/10.1016/j.procir.2016.11.152.

[22] K.Y.H. Lim, P. Zheng, C.-H. Chen, A state-of-the-art survey of Digital Twin: techniques, engineering product lifecycle management and business innovation perspectives, J. Intell. Manuf. (2019). https://doi.org/10.1007/s10845-019-01512-w.

[23] P. Zheng, T.-J. Lin, C.-H. Chen, X. Xu, A systematic design approach for service innovation of smart product-service systems, J. Clean. Prod. 201 (2018) 657–667. https://doi.org/10.1016/j.jclepro.2018.08.101.

[24] P. Zheng, X. Xu, C.-H. Chen, A data-driven cyber-physical approach for personalised smart, connected product co-development in a cloud-based environment, J. Intell. Manuf. (2018) 1–16.

[25] P. Zheng, Y. Lin, C.H. Chen, X. Xu, Smart, connected open architecture product: an IT-driven co-creation paradigm with lifecycle personalization concerns, Int. J. Prod. Res. 57 (2019) 2571–2584. https://doi.org/10.1080/00207543.2018.1530475.

[26] F. Tao, F. Sui, A. Liu, Q. Qi, M. Zhang, B. Song, Z. Guo, S.C.Y. Lu, A.Y.C. Nee, Digital twin-driven product design framework, Int. J. Prod. Res. 7543 (2018) 1–19. https://doi.org/10.1080/00207543.2018.1443229.

[27] J. Guo, N. Zhao, L. Sun, S. Zhang, Modular based flexible digital twin for factory design, J. Ambient Intell. Humaniz. Comput. 10 (2018) 1189–1200. https://doi.org/10.1007/s12652-018-0953-6.

[28] T. DebRoy, W. Zhang, J. Turner, S.S. Babu, Building digital twins of 3D printing machines, Scr. Mater. 135 (2017) 119–124. https://doi.org/10.1016/j.scriptamat.2016.12.005.

[29] Y. Lu, C. Liu, K.I.-K. Wang, H. Huang, X. Xu, Digital Twin-driven smart manufacturing: Connotation, reference model, applications and research issues, Robot. Comput. Integr. Manuf. 61 (2020) 101837.

https://doi.org/10.1016/j.rcim.2019.101837.

[30] Y. Tan, W. Yang, K. Yoshida, S. Takakuwa, Application of IoT-Aided Simulation to Manufacturing Systems in Cyber-Physical System, Machines. 7 (2019) 2. https://doi.org/10.3390/machines7010002.

[31] M. Schluse, M. Priggemeyer, L. Atorf, J. Rossmann, Experimentable Digital Twins-Streamlining Simulation-Based Systems Engineering for Industry 4.0, IEEE Trans. Ind. Informatics. 14 (2018) 1722–1731. https://doi.org/10.1109/TII.2018.2804917.

[32] K. Ding, F.T.S. Chan, X. Zhang, G. Zhou, F. Zhang, Defining a Digital Twin-based Cyber-Physical Production System for autonomous manufacturing in smart shop floors, Int. J. Prod. Res. 0 (2019) 1–20. https://doi.org/10.1080/00207543.2019.1566661.

[33] F. Tao, M. Zhang, Digital Twin Shop-Floor: A New Shop-Floor Paradigm Towards Smart Manufacturing, IEEE Access. 5 (2017) 20418–20427. https://doi.org/10.1109/ACCESS.2017.2756069.

[34] R. Söderberg, K. Wärmefjord, J.S. Carlson, L. Lindkvist, Toward a Digital Twin for real-time geometry assurance in individualized production, CIRP Ann. - Manuf. Technol. 66 (2017) 137–140. https://doi.org/10.1016/j.cirp.2017.04.038.

[35] N. Nikolakis, K. Alexopoulos, E. Xanthakis, G. Chryssolouris, The digital twin implementation for linking the virtual representation of human-based production tasks to their physical counterpart in the factory-floor, Int. J. Comput. Integr. Manuf. 32 (2019) 1–12. https://doi.org/10.1080/0951192X.2018.1529430.

[36] T. Petković, D. Puljiz, I. Marković, B. Hein, Human intention estimation based on hidden Markov model motion validation for safe flexible robotized warehouses, Robot. Comput. Integr. Manuf. 57 (2019) 182–196. https://doi.org/10.1016/j.rcim.2018.11.004.

[37] J. Lee, E. Lapira, S. Yang, A. Kao, Predictive manufacturing system - Trends of next-generation production systems, IFAC Proc. Vol. 46 (2013) 150–156. https://doi.org/10.3182/20130522-3-BR-4036.00107.

[38] S. Haag, R. Anderl, Digital twin – Proof of concept, Manuf. Lett. 15 (2018) 64–66. https://doi.org/10.1016/j.mfglet.2018.02.006.

[39] E.J. Tuegel, A.R. Ingraffea, T.G. Eason, S.M. Spottswood, Reengineering aircraft structural life prediction using a digital twin, Int. J. Aerosp. Eng. 2011 (2011). https://doi.org/10.1155/2011/154798.

[40] B. Ślusarczyk, Industry 4.0 – Are We Ready?, Polish J. Manag. Stud. 17 (2018) 232–248. https://doi.org/10.17512/pjms.2018.17.1.19.

[41] J. Cheng, H. Zhang, F. Tao, C.F. Juang, DT-II:Digital twin enhanced Industrial Internet reference framework towards smart manufacturing, Robot. Comput. Integr. Manuf. 62 (2020) 101881. https://doi.org/10.1016/j.rcim.2019.101881.

[42] Z. Wang, C.H. Chen, P. Zheng, X. Li, L.P. Khoo, A novel data-driven graph-based requirement elicitation framework in the smart product-service system context, Adv. Eng. Informatics. 42 (2019) 100983. https://doi.org/10.1016/j.aei.2019.100983.

[43] Z. Wang, P. Zheng, C.H. Chen, L.P. Khoo, A graph-based requirement elicitation approach in the context of smart product-service systems, Int. J. Prod. Res. 2018-Decem (2019) 1–17. https://doi.org/10.1080/00207543.2019.1702227.

Appendix A

A snippet of computational model used to perform thermal analysis on hot-bed

```
%Thermal Analysis of Hot Bed
thermalModel=createpde('thermal','steadystate');
importGeometry(thermalModel,'Hot Bed.stl');
pdegplot(thermalModel,'Facelabels','on','FaceAlpha',0.5);

generateMesh(thermalModel,'Hmax',1);
thermalModel.Mesh;
pdemesh(thermalModel);

k1=205; %Thermal COnductivity of Aluminium in W/mK
hc=20; %Heat Transfer coefficient of air at 25C (Natural Convection) W/m2K
T1=288.15; %Ambient Temperature in K (25C);
T=60; %Hot Bed Temperature in C;
T2=273.15+T; %Hot Bed Temperature in K;

thermalProperties(thermalModel,'Cell',1,'ThermalConductivity',k1);
thermalBC(thermalModel,'Face',1:thermalModel.Geometry.NumFaces,'ConvectionCoefficient',hc,'AmbientTemperature',T1);
thermalBC(thermalModel,'Face',32,'Temperature',T2);
thermalBC(thermalModel,'Face',33,'Temperature',T2);
thermalBC(thermalModel,'Face',34,'Temperature',T2);
thermalBC(thermalModel,'Face',35,'Temperature',T2);
thermalBC(thermalModel,'Face',36,'Temperature',T2);
thermalBC(thermalModel,'Face',37,'Temperature',T2);

pdeplot3D(thermalModel,'ColorMapData',results.Temperature);
```