Preparation phase report: A digital twin of a microbrewery—a case study investigating DT model integration and orchestration techniques

Ander Lee
a.lee@student.tue.nl
Version 2.3.0
April 2022

CONTENTS

1 INTRODUCTION

The digital twin (DT) model was first introduced by Grieves in a 2002 presentation [1]. At the time it was referred to as a concept for Product Lifecycle Management (PLM). The model asserts that systems are dual in nature, that is, a system has a physical twin and a virtual twin. The premise of the model is that the two sides are bound together via a data link. This link is bidirectional that it transfers data from the physical space to update the virtual space, and vice versa.

The association to PLM implies that a DT is a dynamic model that adapts to specific stages in the product's lifecycle. The stage changes made in either space are promptly translated to the subsequent outcomes in the other space. An example in practice could be, an operator—a real human or an autonomous decision making program—makes a design tuning to the virtual mock-up, this change consequently modifies the behaviors of the manufactured physical product. The physical product which possesses the latest information received from the virtual space, is deployed and operates in the intended context. From then the physical product collects data about the system itself or the reactions arising from its interactions with the environment and sends them back to the virtual space. These behavioral characteristics of DTs make a great distinction from simulations, which are commonly confused with DTs. To simply put it, we consider that DTs take a step further by using simulations to influence physical space. This idea is particularly helpful to the products with very broad scale lifecycle, both in terms of timeline and complexity. As we can see soon this is a reason why DTs have gained successe in many industries.

After Grieves' proposal of using DTs for PLM, the early adopters of this concept were aerospace engineering organizations such as NASA and US Air Force [1]. It is not difficult to realize this is more than just a coincidence. The lifecycle of aircraft is broad, on average, between 20 to 35 years [2].The design phase alone can already take many years, much less the operation and maintenance phase, which can last up to several decades. Even if narrowed to just a snapshot in time, an aircraft is a complex composition of multi-disciplinary outputs. For instance, the jet engine is a high performance mechanical system; the fuel system is largely based on the understandings of chemical reactions; and the communication equipment is the results of electrical and computer designs. Therefore, DTs become one of the best suited methods in this type of circumstance. As time passed, other industries also started to participate in this trend of multi-disciplinary and long-term product lifecycle. Production factories these days use dedicated software to analyze the efficiency of operations and optimize the plant on-the-go [3]. Hospitals also use digital assets to aid experiments on drugs and patients [4]. As a result, the study of DT development becomes more and more important.

As we have seen the design choices of DT development have grown exponentially in the first two decades of 20th century. In the physical space, this is partially enabled by wireless communication technologies such as remote sensors and cellular networks. On the other hand, in the virtual space more diverse modeling methods are available commercially. A notable progress is shown in big-data models and artificial intelligence models [5]. Under this premise we start to notice a problem of how to assemble and manage all the available assets together in a DT in order to support the product lifecycle. It is reasonable to assume most tools are not created by the same party, and even if they are, the DT developers will be facing great restrictions due to vendor lock-in. Hence to collaborate using heterogeneous tools we need integration and orchestration. They give attention to aspects as focused as data format merging and unit conversion, to bigger considerations like inter-model scheduling. In this study, we want to investigate what is considered good practice for integration and orchestration.

As we mentioned before that DTs ultimately aim to support the services in a complex product lifecycle. In this project we also want to verify our findings of integration and orchestration techniques in a case study. We choose a microbrewery as the use case. A brewery has many steps of procedure in order to produce a beer. Among them, a key process that impacts the quality of the beverage is fermentation. The process can be tracked by existing sensors which detect, such as, the alcoholic content and temperatures. Moreover, the conditions of the process are relatively mild, i.e., low temperature, medium high pressure, which allows easy observations. The process is a well studied phenomena, making it easy to construct and evaluate. Therefore, we reckon the construction of microbrewery as a suitable proof-of-concept for DTs. In addition, models for bio-chemical processes often vary significantly in terms of properties and behaviors. Some models might deal with chemical reactions that take place continuously, while other models might deal with discrete observations which only occur periodically. Due to this vast variety, we think it can highlight the integration and orchestration efforts which will be incorporated in our DT services.

In Section 1.1, we will describe the growing attention to Industry 4.0 and the role of DTs as part of this trend. After that, the DT domain will be introduced in Section 1.2, in which the term DT will be defined and a reference DT model will be elaborated. Section 1.4 examines the current state of the art in DT technologies.

1.1 Context

As claimed by market analysts [6] [7], the emerging trend of Industry 4.0 is characterized by widespread digitalization. Hence DTs are becoming more widely adopted as industry moves toward a digital transformation which sets out to improve the transfer of information across the value chain. This will allow stakeholders to share knowledge about the designs, conditions, and logistics of their products and operations through digital artefacts.

Having a DT, particularly in the manufacturing context, helps to accelerate the design process as it facilitates the use of abundant sensor data and accurate modeling of the physical world to provide deep insights to the engineers. Modeling in DTs has improved significantly in recent years because the concept has evolved from mere simulation to sophisticated mechanisms

that can optimize the physical entity based on the data collected. Besides, services in a DT can also offer consistent monitoring of the product throughout its life cycle, allowing optimization of the product's performance during its lifetime. Other benefits DTs can offer include predictive maintenance and risk assessment. By facilitating the virtual mock-up of the real product, one can overcome the temporal and spatial limitations of experimenting in the physical world, thus greatly reducing the overall cost.

The notion of DT is often associated with the model-driven system engineering (MDSE) paradigm. In MDSE, a complex system is treated holistically. It is described in terms of the interaction between systems, or better known as 'system of systems.' The system architecture is represented as models to support requirements, design, analysis and verification activities throughout the lifecycle. The use of modelling helps drive a consistent specification without significantly increasing costs as the system depth extends. Furthermore, DTs aim to capture a comprehensive physical and functional description of the target system. Specifically, virtual entities monitor data from the physical entities, use it to fulfill services which are requested by the user, and may optimize the physical entity. Hence a closed loop of operation is formed. It is found in many studies that DTs have the potential to bring the following benefits [8]–[13]:

- Reduce maintenance costs during lifecycle.
- Reduce errors and inconsistencies across multiple iterations.
- Improve multi-disciplinary collaboration, i.e., engineers from different job functions are able to quickly grasp the high-level overview of the design.

These benefits can lead to a more seamless digital transformation in the industry. Hence we argue that further investigations of DT development is regarded important to propel the new wave of industrial transformation.

1.2 Domain overview

This section explains in further detail the concept of DT and its classification. Following that, we describe the challenges of interest for this project.

1) Digital twin: Although the term 'digital twin' has been mentioned in much literature, such works seldom share the same definition [14]. In some works the definition has close resemblance to an integrated simulation, which contains multi-physic, and multi-scale simulations working together [15]. However, in some other studies it is defined as a high fidelity digital replica of the physical asset [16]. In order to select a suitable definition for the biomanufacturing context, and to keep the later discussions consistent, hereafter we will adopt the classification proposed by Kritzinger et al. [17]. According to the classification, 'digital twin' can be roughly divided in three categories based on the level of data integration between the physical asset and the virtual counterpart:

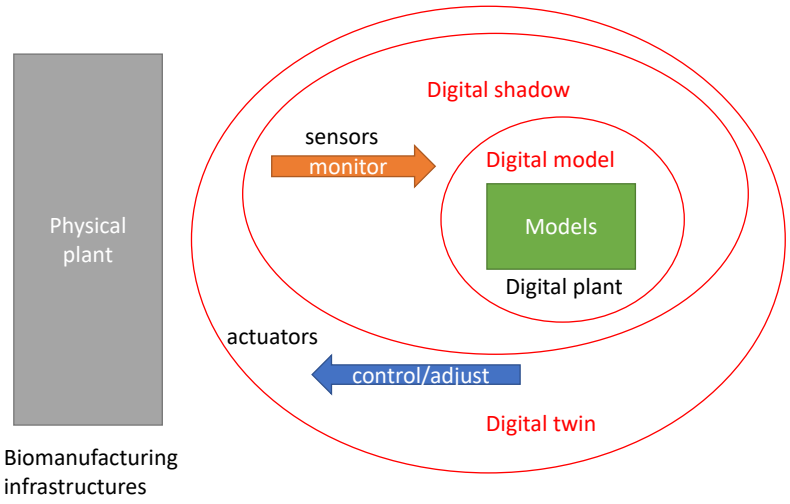

Fig. 1: Classification of digital twin

- **Digital model**: The virtual object does not involve data exchange with the physical object in an automated fashion. Hence the changes in one part have no direct effect on the other unless updates are performed manually.
- **Digital shadow**: There exists an automated one-way data communication from the physical object to the virtual one.
- **Digital twin**: An automation of bi-directional data exchange between the physical object and the virtual object. Changes made in the physical object will be reflected in the virtual object and vice versa.

In a manufacturing plant setting, Data delivered from the physical world to the virtual world is used for monitoring, usually through various types of sensors. The reverse direction—data delivery from the virtual world to the physical world—amounts to adjusting actuators. It is important to note that adjusting or controlling an actuator in a DT denotes a slightly different action than in a traditional control system. In a simple control logic, once a computation produces feedback, the signal will be sent to the actuator to activate certain operations, whereas in a DT, the adjustment data that is sent back to the actuators from the model in the virtual world is meant to calibrate and tune the parameters of an ongoing set of operations. Figure 1 illustrates the concept of DT that is used throughout this project.

Although Kritzinger's view has an emphasis on a fully automated version of DT, in practice, especially in bio-chemical applications, Udugama et al. [18] [19] recognize the unskippable need for operator's intervention in many scenarios. Therefore, a more relaxed version of DT can be described, where the digital model automatically generates its results to a human machine interface (HMI) firstly; then the operator will make a decision based on the given data, and update the actuators parameters through the HMI. A similar concept called human-in-the-loop is demonstrated in [20] as part of the design of an athlete DT, where the coach acts as the operator in managing the fitness plan for athletes. Since this project mainly concerns with a microbrewery which is also a biological fermentation process, the relaxed definition of DT is considered relevant.

2) DT model: The definition in Section 1.2.1 provides a clear view of DTs in term of the data exchange between physical and virtual worlds. A DT model is deemed useful for articulating the management of the interconnections between different parts. Tao et al. [21] propose a five-dimensional view of DT model as shown in Figure 2. Each dimension is described as follows:

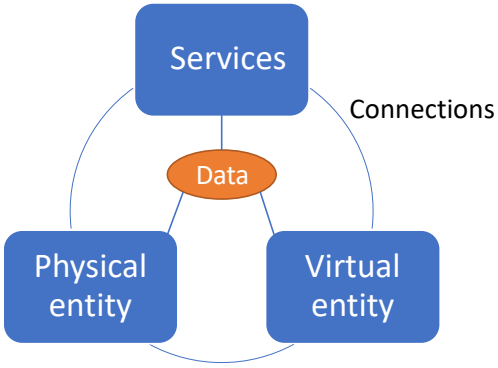

Fig. 2: 5D view of DT model

- **Physical entity** (PE): A collection of sensors and equipment working collaboratively to collect real-time data for the intended services, meanwhile receiving control orders from the virtual world.
- **Virtual entity** (VE): A digital mirror, that contains models simulating the physical counterpart with high fidelity. Calibration strategies are generated through comparing the models with entities to support models' evolution.
- **Services**: Provides various services to support the management and control of PEs as well as the operation and evolution of VEs. Services are usually requested by the users to fulfill certain functionalities.
- **Data**: A shared storage consisting of the raw data from PEs, VEs, and services. It may also be responsible for merging data from various sources to a fused format.
- **Connections**: The connections for all four of the above-mentioned dimensions. Relevant concepts include communication protocols, access ports, etc.

The 5D view is arguably a helpful reference model to follow when introducing new technological assets toward the incremental development of DTs. Even more generally, it guides stakeholders to quickly survey the basic composition of a DT.

1.3 Challenges in the domain

As DTs become increasingly powerful and used for complex systems, diverse and heterogeneous tools and models inevitably must be used to construct DTs covering all aspects of such systems. In this step, the issue of integration and orchestration arises. The orchestration, in short, can be regarded as the management of the models' schedule and event ordering. There has not been much research on systematic frameworks to manage and coordinate the tools and models that come from various vendors along with different paradigms. For instance, many physic-based simulation tools have a continuous-time computational model, while the modeling of information traffic is often based on discrete events. Also, models may differ in their fidelity and use of units,

requiring attention in integration and orchestration. It is crucial to establish a robust practice for integration and orchestration of models in DTs in order to effectively use the models and achieve a suitable DT as a whole. This suitability concerns accuracy and robustness aspects during the early phases of the lifecycle, that is the design phase and manufacturing phase, and shifts toward qualities such as maintainability and scalability in the late stages of service and retiring.

1.4 State of the art

We examine the technologies that constitute the development of DTs, concentrating on the modeling in the virtual space and its interactions—monitoring and controlling—with the physical entities.

1) Monitoring: In DT development, the monitoring of the physical world is normally performed by sensors. The subjects of measurement can be a variety of physical, chemical and biological variables. Clusters of sensing nodes may work collaboratively as enabled by internet of things (IoT) technologies.

Hard sensors normally refer to the electronic instruments that detect and collect measurements directly from their vicinity, for instance, thermostat and pressure gauge. More advanced instruments like spectroscopy—based on the interaction of electromagnetic waves and molecular bonds to measure biomass—are gaining ground [22]. By virtue of their computing capability of properties that are obscured or out of reach to human perceptions, these sort of advanced sensors can be placed in harsh environments such as an agitated bioreactor to provide process data while maintaining non-invasive to the system. In general, high accuracy and low latency are two crucial factors which are required to generate reliable online data.

Soft sensors are the inferential sensing technology that estimate process variables that cannot be directly measured, by combining data collected from hard sensors into equations and a mathematical model. Based on the techniques, they can be categorized roughly to two classes, namely model-driven or data-driven [19]. Model-driven soft sensors are built upon first principles models, which rely on in-depth knowledge of the target process, and that implies a high degree of complexity. The advantage of a model-driven approach is that it has a solid foundation based on physical laws and theories, and that allows an easier generalization of the process provided the operator also possesses sufficient know-how of the field [23]. Alternatively, data-driven models are fed massive data quantities in order to generate predictions. Common approaches include black-box models such as MultiVariate Data Analysis (MVDA), a form of statistical model that processes multiple variables simultaneously and eventually aims to decrease the dimensionality. Another data-driven approach that is rapidly gaining popularity is artificial neural networks (ANN), thank to the wide availability of hardware support like graphical processing units (GPUs), and software library support such as TensorFlow [24], and PyTorch [25].

2) Modeling: Rather similar to the classifications of soft sensors described above, modeling approaches can be sorted by their level of abstraction [26] [27]. At the higher level, techniques like first principles and mechanistic models in biochemical domains; or computer-aided design (CAD) and topological models used in mechanical domains [28] are considered knowledge-based, i.e., more demands in domain expertise, hence requiring less online data to construct. In contrast, toward the bottom level of abstraction, artificial intelligence (AI) related techniques rely more heavily on empirical data.

Upon combining the observations and the model computations, the operation is commonly known as data assimilation. Recursive parameter estimation is an approach for data assimilation. It refers to taking old estimates (old model parameters) obtained from fitting one set of data points to generate new estimates when new data points (new observations) are added to the original data set [29]. Other technique within the same category, such as extended Kalman filter is another broadly used implementation of state estimation [30] [31].

3) Controlling: We can consider controlling as the last piece in DTs that 'closes the loop' between the virtual world and and the physical world. Proportional–Integral–Derivative (PID) controllers have been used in industrial control for decades. Their popularity can be attributed the robustness and simplicity in wide range of industrial settings [32]. On the other hand, the advancement of high volume data acquisition enables growing adoption of model predictive control (MPC), which goes beyond merely a reactive mechanism as PID, also taking into account the contextual knowledge of the underlying complex process. Hong et al. [33] examine the advantage of a hybrid scheme that combines adaptive model-based feedback with direct PID control for optimizing startup, changeover, and shutdown. The study also discusses functionally partitioning components to improve flexibility and reduce operation cost for constructing hybrid models.

2 Related work

We will firstly introduce some recent efforts that address the topic of integration and orchestration. They include tools, methods, and standards which are developed by various multidisciplinary initiatives sharing a common goal of making better collaborations of models. After that, real case studies of DT development will be described as to show the current outlook of DT applications.

2.1 Integration methods

Integration is a key step of models' communication and execution. In a bioprocessing plant, communication infrastructures primarily comprise the following aspects [22]:

- Supervisory control and data acquisition (SCADA) system that handles the interfacing of monitor and control operations. Its components include remote terminal units (RTU) for processing commands; programmable logic controllers (PLC) for processing control signals; HMI for aiding the operator in various actions.
- Data standards such as XML, JSON that deliver string-value pairs of the monitoring/controlling variables. It may also contain information about the computational pipeline of the process.
- Communication protocol stacks that describe each Open Systems Interconnection (OSI) layer. Common patterns such as client/server in TCP/IP, or publish–subscribe in message queuing telemetry transport (MQTT) are well documented in IoT applications.

Apart from the generic communication elements mentioned above, standards for data transfer between models are especially important in a DT context. In this section we introduce two notable open source efforts, namely Computer-Aided Process Engineering (CAPE)-OPEN [34], and Functional Mock-up Interface (FMI) [35] [36].

In short, CAPE-OPEN is a standard of a component-based approach to process simulation [37], especially addressing the chemical manufacturing domain. The standard can be broadly seen as two parts. The first is Process Modelling Components (PMC), they represent functionally separated building blocks such as thermodynamic and physical properties engines, or numerical solvers that compute highly nonlinear equations which arise from the flowsheet. The second is Process Modelling Environment (PME); it is essentially a flowsheet—a common diagram used by chemical engineers to indicate the general flow of plant processes and equipments—that utilizes services from PMCs, and supposed to handle the related connections seamlessly. CAPE-OPEN compliant simulation programs from different vendors, are able to maintain consistent interoperability without 'glue codes' or other manually coded wrappers.

While CAPE-OPEN primarily pertains to the chemical industry, FMI is applicable to more general cyber-physical systems (CPS). FMI handles the interfacing of functional mock-up units (FMU), which is the encapsulation of a model in XML format. The XML schema could contains model variables, time-step information, etc. The exact APIs of the FMI and the exact specifications of the FMU depend on the choice of interface types. There are currently three types, which are model exchange, co-simulation, and scheduled execution. Since the third type is relatively new—introduced in FMI 3.0—it has not been fully supported by many development environments, and we will focus on the first two types. They are briefly described as follows (see also Figure 3):

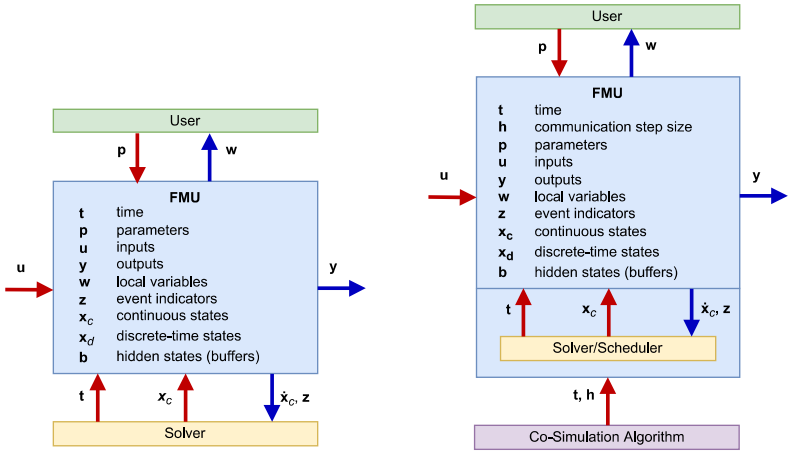

Fig. 3: FMI for model exchange (left), and FMI for co-simulation (right) [38]

- Model Exchange: exposes a numerical algorithm (e.g. ODE) to an external solver of an importer (the simulation environment). Using this solver, the FMU is evaluated at a specific time instant.

- Co-Simulation: implements not only the model algorithm, but also the required solution method. The data exchange between FMUs is restricted to discrete communication points, thus the co-simulation algorithm (serving as the master algorithm) is shielded from how individual FMUs advance time internally.

2.2 Orchestration methods

This section covers three distinct orchestration approaches which are frequently encountered in our literature survey. Respectively, they are Ptolemy II [39], Systems Modeling Language (SysML) [40], and an approach that is gaining more popularity in the cloud service domain, roughly known as DTs with service-orientated architecture (SOA).

Ptolemy II is a framework that coordinates actors of various models of computation (MoC) while maintaining strong semantics of each individual model [41]. The notion of MoC refers to an abstract collection of rules that govern the interaction between components in a design. It is analogous to the 'laws of physic' that are used to describe a given system. A few examples of MoC are shown as follows:

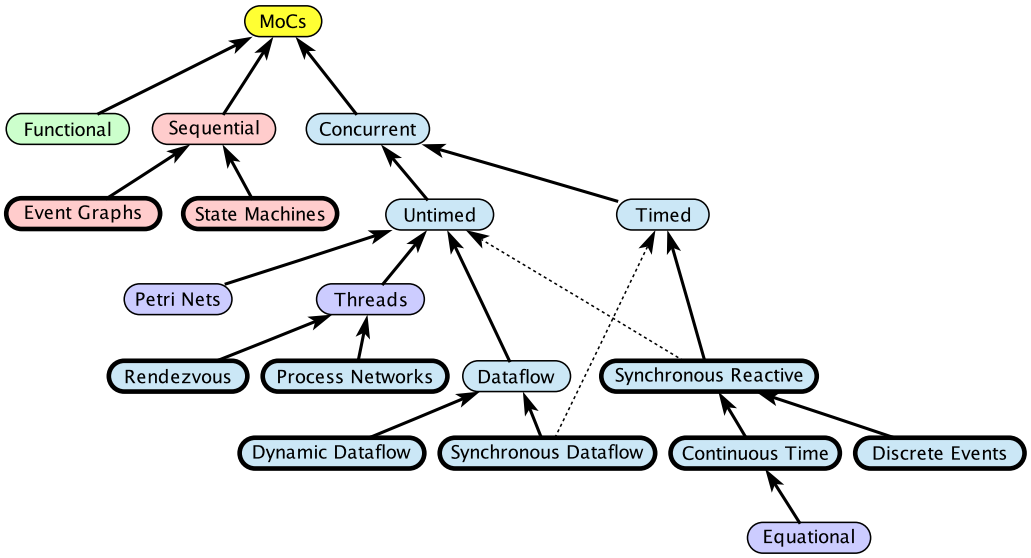

Fig. 4: MoCs supported in Ptolemy II [41]. Bold arrows indicate primary associations. Some MoCs have additional dotted arrows which indicate alternative associations if they can be used in two different modes (timed or untimed).

- Process network (PN) [42] : scheduling for concurrent distributed processes. It has a benefit of determinacy as long as there is a unique solution to the balance equations.
- Finite state machine (FSM): captures control-dominated behaviors.
- Discrete event (DE): suited for modelling the behaviors of complex systems over time, e.g., queuing systems.
- Continuous time: useful for modeling ODE.

Figure 4 shows a summary of the relationship between MoCs (denoted with bold outlines) that are implemented as 'domains' in Ptolemy II.

SysML is a dialect of Unified Modeling Language (UML) for MBSE applications. In contrast to Ptolemy II, SysML emphasizes providing rich static and dynamic behavioral information in the form of diagrams. The static diagrams can represent the system structures in varying degrees of transparency, for instance, Block Definition (black-box), Internal Block (white-box), or Requirement (declarative). The behavioral diagrams can be considered as counterparts to the MoCs in Ptolemy II. The taxonomy of SysML diagram type is summarized in Figure 5.

Lastly, alongside the trend of cloud computing, there is an increasing number of studies [44]–[46] that look into constructing DTs in line with SOA design pattern. In specific, this approach regards models in DTs as independent cloud-native microservices. The DT is deployed as containers, and orchestrated with off the shelf applications such as Kubernetes [47]. This view of DTs provides the benefits of rapid deployment, as well as auto monitoring, scaling, and load balancing among other features which are found in commercial cloud services. This practice is also suitable for continuous integration (CI). It is important to recognize that since the majority of orchestration is off-loaded to cloud space, the demands of an efficient and reliable networking setup and integration also become significantly higher.

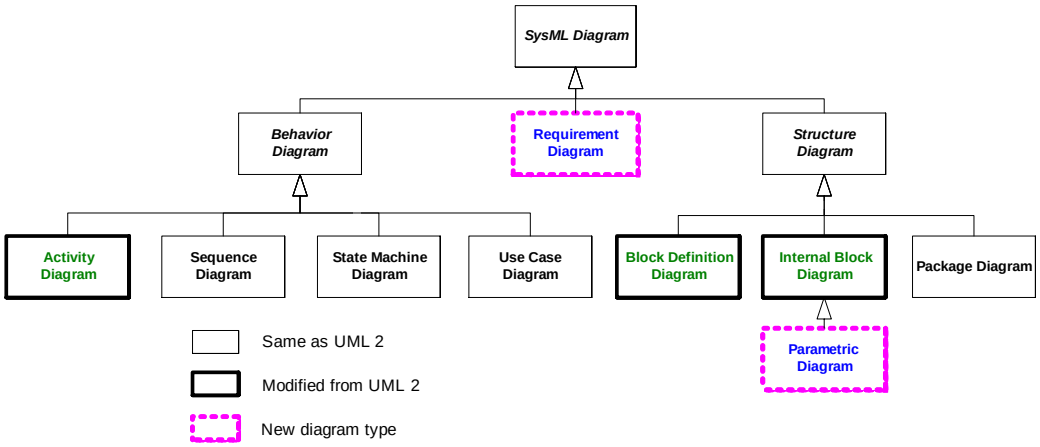

Fig. 5: SysML diagram taxonomy [43]

2.3 DT in bioprocessing

In this section a number of case studies will be reviewed. Among these works, the common objective is to construct a DT in order to optimize the bioprocessing production. These studies also demonstrate many of the technologies outlined in Section 1.4.

A study of enzyme production [48] proposes a model-based strategy to maximize the final fill of a fed-batch process. The strategy is presented in a cascaded configuration, which is separated by a supervisory layer and a regulatory layer. The supervisory layer implements a mechanistic model that calculates the required start fill. This is the model-based batch planning for initial conditions. The model's parameters are re-fitted using a least squares approach. As for the regulatory layer, the calculated feed rate is adjusted by a PID controller. The model is implemented in MATLAB and the collected data are accessed from an open platform communications (OPC) server. The study demonstrates a multi-layer approach of orchestration.

Lopez et al. [49] propose a DT of ethanol fermentation. The researchers use a data-driven soft sensor that takes the online spectroscopy measurements to compute the glucose concentration—referred as process variable (PV)—in real-time. PV is then used as the input to a PID algorithm in order to generate a final control signal—referred as manipulated variable (MV)— that adjusts the feed rate of the controlled pump. The PV-MV transformation in this example showcases how monitoring models and controlling models can be integrated.

In their manufacturing platform for antibodies, Feidl et al. [50] manage to build a process-wide control with a SCADA system. The system collects unit-relevant data streams from each process unit, then converts to a centralized data storage, which contextualizes and adds a timestamp to each data point, in which the data is transformed to process-relevant. Afterward, the SCADA system is able to send newly determined setpoints to the respective local control units. Hence, an automated end-to-end integration of the supervisory control with the data acquisition system is achieved.

Eppinger and the fellow colleagues from Siemens [51] design a DT for ketchup production. The control objectives are evaluated by a set of Key Performance Indicators (KPI). The DT firstly obtains the model parameters from historical data through machine learning based intelligent analysis. A hybrid model that combines equation-driven model and data-driven model is then developed before being applying a model order reduction process, such that it can be made compatible with the real-time hard sensors. Once the reduced models are generated, the soft sensors—referred as virtual sensors—can be synthesized and be used to predict the KPIs. Finally, given all available information, the agent of the reinforcement learning algorithm executes actions toward the physical plant and tunes itself with respect to target KPIs by using feedback. This study explains a method to orchestrate hybrid models under time critical constraints.

3 Problem description

Section 3.1 states the purpose of study, specifically, explaining why the problem is important to the DT domain. In Section 3.2, we progressively develop the concise definitions of integration and orchestration. The proposed services of the microbrewery DT will be discussed in Section 3.3, followed by the scope of the project given in Section 3.4. Lastly in Section 3.5, we will derive the research questions based on the reasoning so far.

3.1 Significance of the problem

As explained in Section 1.3, the problems of integration and orchestration in DTs are what we aim to explore in this research. Described in the work of van den Brand et al. [52], despite there exists commercial frameworks that handle the integration and orchestration of heterogeneous models, they are mostly restricted to specific modeling tools only. The study indicates there is a need to find a suitable framework and extend it to support integration and orchestration across models expressed in different formalisms and tools.

Another study produced by Negrin et al. [53] investigated the integration and orchestration for heterogeneous models concerning the autonomous driving of a container truck at a distribution center. This study signifies the importance of considering the specific DT's purpose and use case context, while extracting the universal properties of integration and orchestration which can be applied in other domains.

As the number of models and their interactions increases, the complexity of control and data flows will grow significantly. It is a purpose of this project to investigate the ways to reduce this complexity, in consequence, to allow a more methodical design of DTs.

3.2 Problem definition

Figure 6 illustrates the main ideas which are covered by integration and orchestration respectively. On the right side, it is seen that integration comprises of two components, namely encapsulation and interface. Encapsulation refers to the abstraction of a model, as well as discerning the relevant inputs and outputs from the rest of internal operations of the model. This allows the interaction with other models to take place without redundancy while still retaining all necessary information. Integration, on the other hand, is responsible for building a bridge of communication that are unanimously agreed by all involving models, such that correctness of data—both numerically and semantically—can be maintained reliably and conveniently throughout the entire operation of DTs.

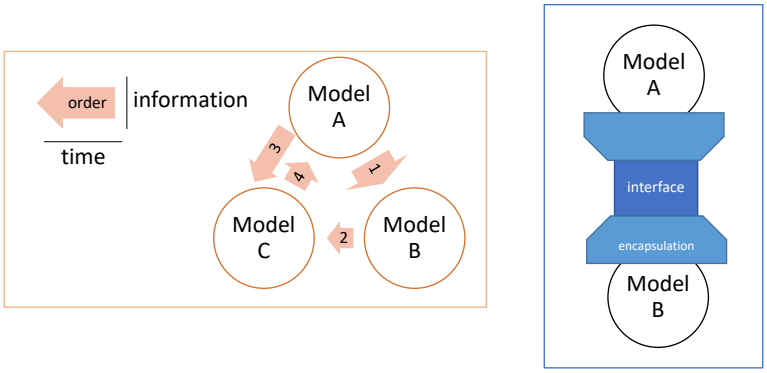

Fig. 6: Integration (right) and orchestration (left)

On the left side, the concept of orchestration is shown. it encompasses two main sub-concepts. The first one is the scheduling relation between different models; in the figure it is represented by arrow shapes. The numbering in the arrows indicates the execution order. Note that during execution, various time duration requirements may be needed for different pairs (or sets) of models interactions. This is represented by length variety of the arrow shapes. Furthermore, different types of information may also be passed down from one model to another, whether it is an event-triggering signal, a continuous stream of data, or in any other manner. The widths of the arrow shapes are used to portrait this diversity. The second sub-concept is more subtle. It has been described in [41] as the policy to control how event instances are accessed in the event queue. In here the event is referred to the internal tasks of the model that are triggered since the beginning of execution of that model. While the scheduling relation

treats individual models as unitary objects and creates a feasible schedule to group those individuals, the handling of events focuses on arranging the task sequence within the model in order to ensure a deterministic outcome. this is particularly imperative in the situations, for instance, when several models are scheduled at the same moment of time, whereas their respective events might arrive and leave in different order. Consequently, a clear and definite event policy is required to orchestrate the behaviors appropriately.

In short, we can summarize the above discussions of integration and orchestration as follows:

- **Integration** couples different models by encapsulating their properties and operations, followed by exchanging the representations in a formatted communication style.
- **Orchestration** dictates models' sequence of executions and arranges the event queue for the event instances that are triggered by the respective models.

3.3 Proposed services

We propose four services which our case study—a microbrewery DT—aims to support. They are shown as follows:

- **Production prediction (S1)**: predict the properties of end-product and whether its quantity and quality will meet the demand based on the given materials and resources.
- **Production control (S2)**: organize the production schedules and regulate the process such that the utilization of resources is optimized.
- **What-if scenarios (S3)**: create a hypothetical situation and predict its effect on the production in order to generate variants of the production schedule.
- **Predictive maintenance (S4)**: using the data stream from the plant and physical-based modelling to generate a prognosis of the remaining lifetime of plant components.

In order to assess the services and understand the key actors and stakeholders, it is important to recognize their respective phases in the lifecycle. Figure 7 shows a four-phase lifecycle view [14] from an industrial perspective and where each proposed service belongs in the cycle.

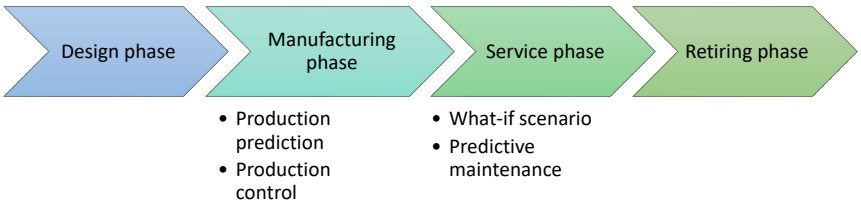

Fig. 7: Services of microbrewery DT in different product lifecycle phases

The *design phase* encompasses the designs of product, process, and plant, in increasing order of scale. The *manufacturing phase* concerns goods production, and the internal logistical affairs involved. S1 and S2 primarily deal with the questions of artifact quantity as well as the strategies to optimize the quantity. Therefore, they are considered to be in the manufacturing phase. The *service phase* includes external logistics, user experiences; the detection of anomalies, and repairs. S3 and S4 are deemed to belong in this category. In the *retiring phase*, decommissioning of the product is dealt with. Valuable data and parts can be obtained from this recycling action so as to improve the future lifecycles.

3.4 Project Scope

This is an exploratory project of the possible integration and orchestration techniques, supported by the implementation of the microbrewery DT as case study. The fermentation process models in the DT are given, they are developed by a domain expert in biochemical processes. Hence the details of simulation algorithms are not in the scope. A comparative study of the three selected frameworks—detailed in Section 5.3—will be conducted in parallel to the DT services development.

The considered services belong to the manufacturing phase and service phase of the lifecycle, as discussed in Section 3.3. Therefore, the design phase and retiring phase will not be in the scope of this study. Despite the services being influenced by the bioprocessing theme, we will try to maintain universality in the analysis, such that the findings can be applied in other domains as well.

Functional and system level testings of the DT are also part of the implementation scope in order to evaluate the performance outcomes.

3.5 Research questions

Based on the information given so far, we can specify a couple of objectives for this project. The first objective is to provide answers to the following research questions (RQs):

1) *What are the key ingredients for integration and orchestration of models in different services for a microbrewery DT?*
2) *How can these ingredients be generalized to benefit other industry fields?*
3) *What framework is suitable to support these ingredients?*

The second objective which inclines more toward the practicality, is to test the developed DT system according to the given requirements and KPIs.

4 METHODOLOGY

In Section 4.1, the approach taken to conduct this project will be described. Section 4.2 elicits the steps to actualize the aforementioned approach. In that section we also argue the prioritization of services based on the simplicity and the usefulness to highlight each framework. Section 4.3 introduces a five-step workflow that can be used to evaluate the DT maturity regarding implementation.

4.1 Research approach

In this project we will construct a microbrewery DT. We focus on the integration and orchestration aspect of the models which enable the considered services. The construction begins with defining requirements and KPIs of the services. They are crucial as to measuring the effectiveness of the resultant DT quantitatively and qualitatively. To satisfy the said requirements and KPIs, we leverage the selected frameworks and platforms to implement the services. The frameworks will be compared with each other, and supplementary components, such as database, drivers,...etc, will be built if required. The construction is concluded by the testing of the DT to ensure it functions correctly. Observations will be recorded throughout the entire construction, they will be used to derive the outcomes that answer the research questions.

Figure 8 illustrates the approach of this project. The blue blocks in the center represent the DT and its constituents. The gray blocks at the top and at the bottom are tasks that support the DT development. The red oval shaped block on the left is the expected outcome arising from the construed DT. The outcomes are broken down to bullet points which address each of the research questions respectively. The arrows represent the causal relations of of the blocks, i.e., a block's role to enable the other block by certain actions. Considered altogether, this figure gives an executive summary of the inputs (tasks, actions) and outputs (DT, insights) of this project.

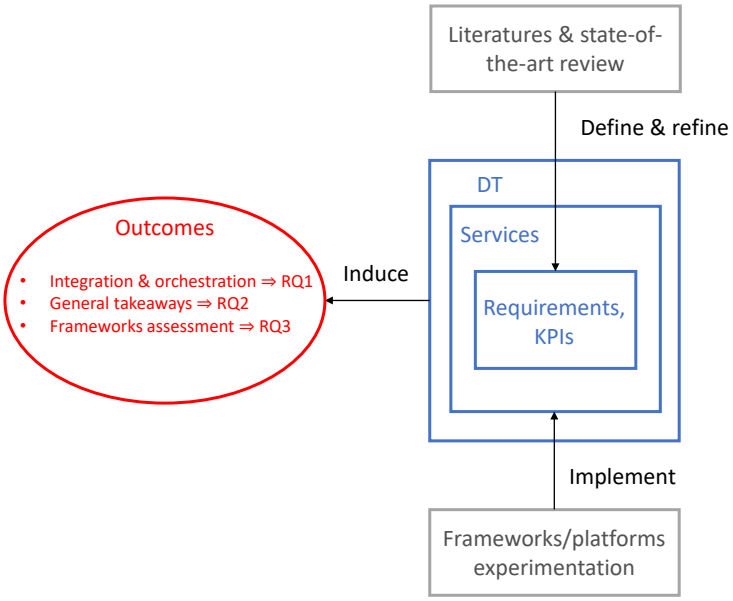

Fig. 8: Research approach

4.2 Project steps

Section 4.1 provides a holistic impression of how to approach the research. This section will continue to elaborate the exact steps to follow. We design a nine-step pathway to complete this project. The steps are divided to two phases, namely preparation phase, and implementation phase. The emphasis of the former phase is to conduct survey, and planning. The latter focuses on the software development to build the DT as well as to answer the research questions. Figure 9 shows the steps.

The reason we want to split the services to the '1+3' arrangement in step 4 is because we want to examine and compare all the proposed frameworks in the first place before committing to one. As a result, the prioritized service has to be simple

to prototype within a short period. Furthermore, that service shall highlight the capabilities of each framework with respect to handling integration and orchestration.

Hereby we decide to choose 'S1: production prediction' as our first service to implement. This decision is based on the two factors, simplicity and completeness, mentioned earlier. For the first factor, we realize S2 and S4 pose a higher threshold of technical effort in order to develop successfully. In S2, the requirements (see Section 5.2) ask to handle the triggering of models and actuators promptly, while maintaining a stable calibration. This implies a great endeavor is required for tuning the actuators in order to adapt to different frameworks. Hence we will not use S2 as a rapid prototype in step 5. S4 follows a similar reasoning as its requirements require a flexible data ownership management, which is highly dependent on the sensors calibrations and significant adaptations under different frameworks. Although S3 is not as technical demanding as S2 and S4, however, semantically speaking, 'what-if scenarios' is considered an extension to the 'production prediction' service. Functionality-wise, S3 is based on many premises of S1 as well. Therefore, we think it is unnatural to implement S3 before S1. We argue that S1 is also qualifying in the completeness factor by looking at how each framework would utilize its features to fulfill S1 requirements and KPIs—detailed in Section 5.3.

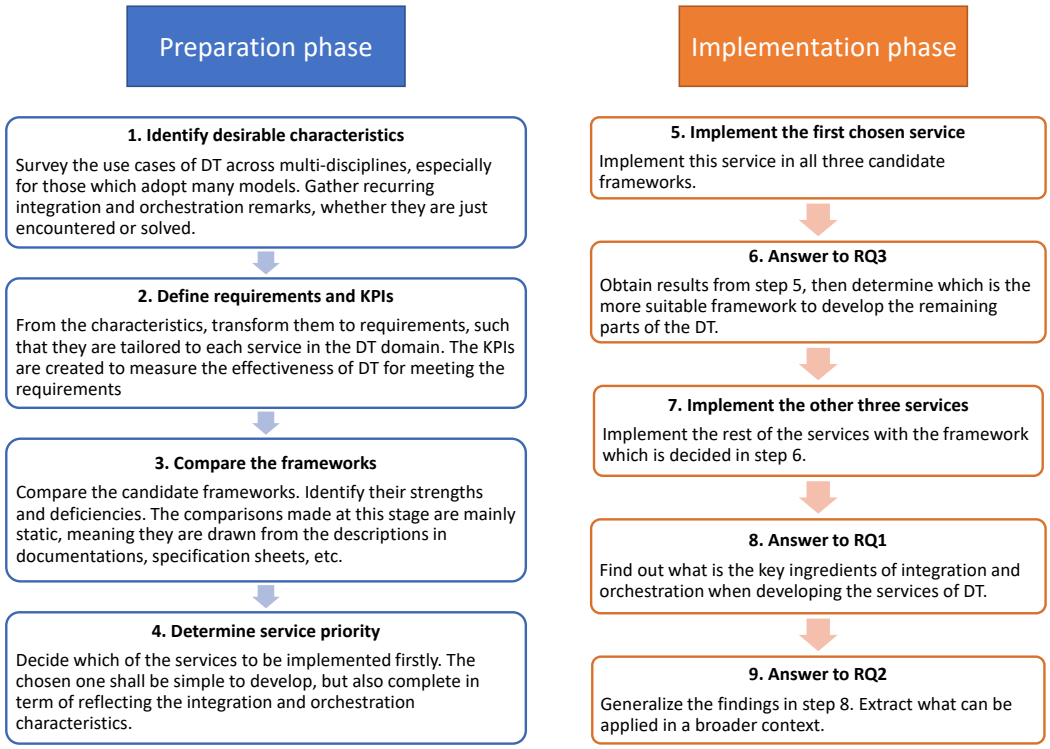

Fig. 9: Project steps

4.3 DT implementation workflow

During the implementation phase (project step 5 to 9), having setpoints helps us to judge whether the DT has met an acceptable level of development. One way of doing that is to check the corresponding requirements and KPIs of the services. From another angle, one can examine the DT by comparing it against a reference workflow to see what level it is at in term of complexity. Here we introduce a workflow which has been tailored for the bioprocessing theme.

Tao's 5D reference model introduced in Section 1.2.2 provides a glance of entities management in DTs. However, it does not address the varying maturities of the models in different attempts at developing a DT. For this purpose, Udugama et al. [19] propose a five-step workflow to implement a DT in bioprocessing. The workflow progresses in increasing order of mathematical complexity and functional requirements.

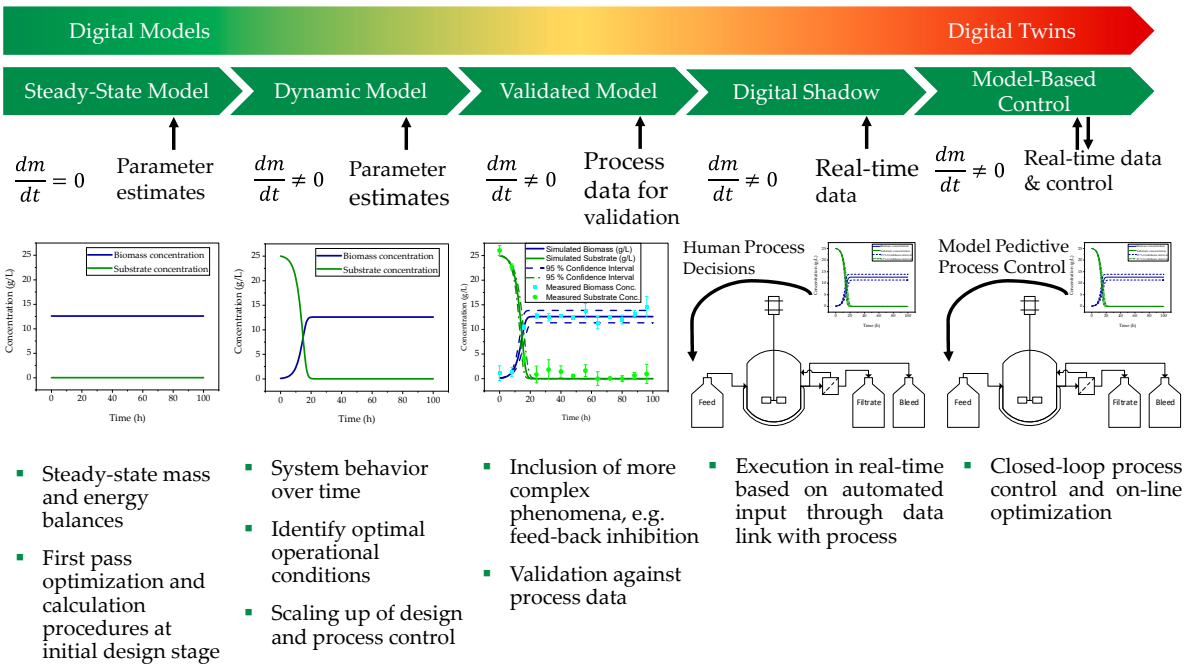

Fig. 10: The five-step implementation. This approach has been demonstrated in a design of DT for monoclonal antibodies manufacturing [54]

1) **Steady-state model**: consisted of a mass and energy balance of the different key compounds in a reaction. These models are mathematical expressions of a process that are not time-dependent and hence carry no accumulation term.
2) **Dynamic model**: contains time-based derivative terms on all variables of interest.
3) **Validated model**: extends the capabilities of dynamic process models, such that it needs to be validated against process data obtained from an actual physical process.
4) **Digital Shadow**: consistent with the definition given in Section 1.2.1; An one-way real-time monitoring model.
5) **Digital Twin**: consistent with the definition given in Section 1.2.1; A two-way real-time monitoring and control model.

As illustrated in Figure 10, the five-step view offers a systematic method in DT building, and a way to measure the maturity of a given DT. For the specific DT in this project, we aim for the 'level 5' DT—a bidirectional model-based monitoring and control scheme—whenever applicable to the proposed services.

5 Preliminary Outcomes

Following the project steps proposed in Section 4.2, we present the preliminary results for step 1, characteristic identifications in Section 5.1; step 2, requirements and KPI definitions in Section 5.2; and step 3, frameworks comparison in Section 5.3.

5.1 Characteristics identifications

Prior to deciding the key ingredients of integration and orchestration for certain services, it is helpful to firstly survey what characteristics are generally considered desirable across various use cases. These characteristics reoccur in the remarks which are found in the gathered studies. We classify them into either integration or orchestration based on the definitions given in Section 3.2. Table I presents the found results.

Label	Integration	Label	Orchestration
I1	Configurability of parameters and time advancement	O1	Control workflow and execution sequence
I2	Automated code/data generation	O2	Ontology checking
I3	Data exchange consistency	O3	Managing mixed fidelity/granularity
I4	Plug & play modularity		

TABLE I: Identified characteristics

Statement I1 can be further broken down to two parts. First is the adjustability of initialization of the subsystems. As Tolksdorf et al. [55] point out, upon the convergence of sub-models into one flowsheet, process engineers often face the challenge of guessing sensible initial values as the sub-models no longer are transparent to them, and a poor guess can easily lead to underperforming models. Therefore, it is argued that the accessibility to critical parameters throughout the whole process is essential to integration. The second part is related to the importance of multirate modeling with dynamic time management. In [56], the researchers experiment with varying execution step-sizes for a vehicle DT, showing that to a certain degree, distributed components—referred as federates—can be ran with different clocks rate and still produce matching results. The ability to parameterize time advancement extends flexibility in the platform under design.

I2 refers to the required automation upon combining models in order to reduce human errors. As chemical process optimizations are rarely accomplished by one single program, manually interfacing simulation packages from multiple vendors becomes impractical as soon as the system grows large [57]. A systematic method to generate 'glue codes' and data adaptation is an ideal solution. In practice, a hybrid scheme with varying levels of automation to trim down the hand-tuning effort is considered relevant in most cases.

I3 suggests in the case when a variable is transferred from one model to another, it shall retain its structure and its dependency relation with other objects. An important implication of this property occurs when several solvers are working on shared data. If information about algebraic dependencies between outputs and inputs are supplied, the importing tool is able to detect and handle algebraic loops automatically [36].

The plug and play property in I4 implies not only the ease of use for the operator, but also the support for templates and instantiation. Actor-oriented design—also used in Ptolemy II—orthogonalizes component definition and component composition, allowing them to be considered independently [58]. It promotes modularity which in turn reduces the associated design cost.

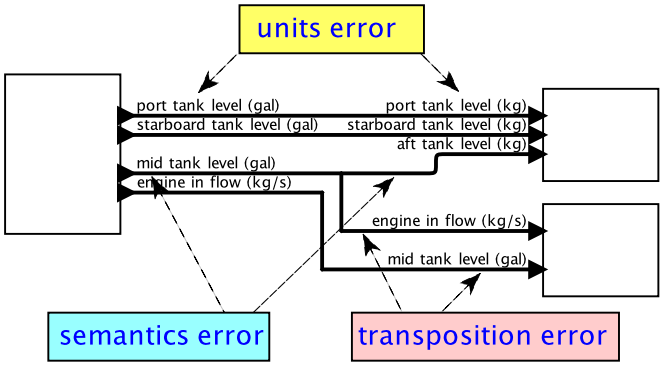

Fig. 11: Errors that can be detected by ontology checking [41]

As described in the problem definition in Section 3.2, O1 reinforces the importance of effective scheduling. One of the key factors that influences the workflow between models is coupling [59], that is, how tightly are models intertwining with one another. A loose coupling relation generally requires less complex schedules. One may use strategies such as using an external framework to de-couple a pair of inherently coupled models. Other considerations, such as atomicity, might be required should there be concurrent processes.

Statement O2 discusses ontology, which is the explicit organization of constituent concepts and the relations between those concepts. More specifically, ontology can help to express the intended use of a model. In [41], it is indicated that errors may arise from ontology inconsistencies, as illustrated in Figure 11.

In contrast to the emphasis on time step configurability in I1, O3 raises attention to the time synchronization and convergence across different models. For instance, merging discrete-time and continuous-time models by techniques of sampling and zero order hold; another example, signaling between time and untimed models. In the case of FMI, an event instant may be driven by a predefined time event or at the transition of state event indicators [36]. This is to allow numerical robustness, and is a part of the FMI feature which is known as 'hybrid ODE'.

5.2 Requirements and KPIs

The requirements are defined such that they should entail the integration and orchestration characteristics found in Section 5.1, in the meantime be tailored to the semantic goal of individual services. We must also keep in mind the 5D model of DTs, in which each requirement can be mapped to one or more connections between the other dimensions—namely VE, PE, service, and data—in the DT.

In addition, several KPIs are set out to evaluate the effectiveness of the DT in terms of how it satisfies the requirements. In principle, each KPI should be a measurable metric that reflects a certain aspect of system performance or non-functional quality. The KPIs are defined in relation to the specific requirements they address. In other words, we develop the KPIs by analyzing the corresponding requirements and capture the measurable properties in them. Subsequently, we proceed to find a closely matching indicator, and use it as the measurement of those properties.

1) Production prediction: In Table II, the top part shows the proposed requirements for S1, and the corresponding integration and orchestration (I & O) implications they address to. The bottom part of the table shows the KPIs and their related requirements. R1 and R3 are associated to the VE-PE connection, whereas R2 relates to the parameterization of the VE-Service connection. Prior to the process startup, it is important to recognize exactly which parameters need to be configured and how often this is needed. Lastly, R4 represents the VE-Data connection.

Requirements		
Label	**Requirement description**	**I & O**
R1	The model state variables shall be validated with real-time empirical data.	I3, O2, O3
R2	The operator shall be able to configure the initial model parameters based on the given materials and resources.	I1
R3	The model parameters shall be automatically updated based on the latest real-time data.	I2, I3, O1
R4	The model may be interrogated for its past instances in order to optimize its present parameters.	I1, I4, O1

KPIs			
Label	**Indicator**	**Description**	**Relevance**
KPI1	Rate of convergence	How fast the error of prediction against real-world data reaches an accepted lower threshold.	R1, R4
KPI2	Latency	How far is the model time behind real-world time.	R3
KPI3	Data availability	What is the amount of historical data that is useful in performing prediction.	R4
KPI4	Reconfigurability	How many parameters need to be set at startup time, and how many can be adjusted during runtime.	R2

TABLE II: S1 requirements and KPIs

2) Production control: Table III shows the requirements and KPIs for S2. It can be argued that R1 and R2 are complementary, as they represent the two way connection of VE-PE, which implies the condition of one trigger direction affect to that of another. The implication of R3 is that the model possess the contextual information of the physical process. In other words, the behavioral characteristics and information of the actuators are taken into account of the model computation, such that the transient behaviors can be kept to minimum.

3) What-if scenarios: Table IV shows the requirements and KPIs for S3. That being said, R1, R2, and R3 all address the VE-service connection. KPI3 highlights the human-in-the-loop idea mentioned in Section 1.2.1. As the end goal of what-if scenarios is to enable the operator to explore alternatives so they can gain a better assessment of the decisions and risks, delivering the scenarios to the operator smoothly becomes a crucial factor to the success of the operation.

Requirements		
Label	**Requirement description**	**I & O**
R1	The model optimization may be triggered on the detected disturbance, time, or plant-wide performance.	O1, O2, O3
R2	The controller calibration may be triggered on time, or state variables deviation.	I2, O1, O2, O3
R3	The model shall ensure that the calibrations comply with the controller's desired operating range.	I1, O1, O2

KPIs			
Label	**Indicator**	**Description**	**Relevance**
KPI1	Stableness of transitory behaviors	How long is the overshoots or undershoots period occurring before steady state.	R3
KPI2	System timeliness	What is the response time of triggering.	R1, R2

TABLE III: S2 requirements and KPIs

Requirements		
Label	**Requirement description**	**I & O**
R1	The scenario-dependent data shall be stored and handled separately from the master data.	I4
R2	The scenarios shall be readily applied, reset, and removed without affecting the master schedule.	I1, I2, I4, O1
R3	The iterations of scenario may be managed under version control.	I2

KPIs			
Label	**Indicator**	**Description**	**Relevance**
KPI1	Scalability	To what extent are the additional space and time bounded.	R1, R2, R3
KPI2	Accessibility	Are the model instances of the scenarios easily accessible to the operator.	R1, R3
KPI3	Modularity	Is changeover of the setup for each scenario easily adaptable.	R2

TABLE IV: S3 requirements and KPIs

4) Predictive maintenance: Table V shows the requirements and KPIs for S4. We argue R1 and R2 represent the VE-PE connection. R3 accounts for the chained connection of VE-PE-Data. The term data ownership in R3 refers to the participation in the generation, processing, value addition or analysis of data [23]. In DTs this can be complicated as there is a great diversity of involving parts across physical to virtual space. Data ownership management consists mainly of three actions: engaging, disengaging, and transferring. A smooth transition between these actions is regarded crucial especially for data-driven predictive models.

Requirements		
Label	**Requirement description**	**I & O**
R1	Dynamic data acquisition shall used to update the static attributes in the predictive model	I2, I3, O1, O2, O3
R2	Real-time measurements and simulated results shall be used in the retrofitting of the predictive model	I1, I2, O1, O2,
R3	Data ownership shall have high degree of flexibility and be transferable within the DT in order to perform efficient data prepossessing	I2, I3, O3

KPIs			
Label	**Indicator**	**Description**	**Relevance**
KPI1	Accuracy	What is the error of the predictive model against real-world measurement	R1, R2
KPI2	Impact of missing data	How missing data may affect the overall performance of predictive model	R1, R2
KPI3	Data quality	How much volume, variety, and veracity of the data is lost as the result of transferring and fusion	R3

TABLE V: S4 requirements and KPIs

5.3 Frameworks comparison

We have mentioned actor-orientated architecture—adopted by Ptolemy II—and SOA used as the approaches for orchestration in Section 2.2. In this section, we proceed to investigate the possibility of adopting them as development framework for the microbrewery DT. The key selection criteria for deciding a development framework are that the development environment should support the proposed requirements for the services. Firstly, an emerging methodology known as TwinOps will be introduced. Secondly, we will describe a flavor of frameworks based on SOA, called LwM2M network. After that, Ptolemy II will be reviewed once again, this time with extra attention given to the modeling infrastructure provided by it.

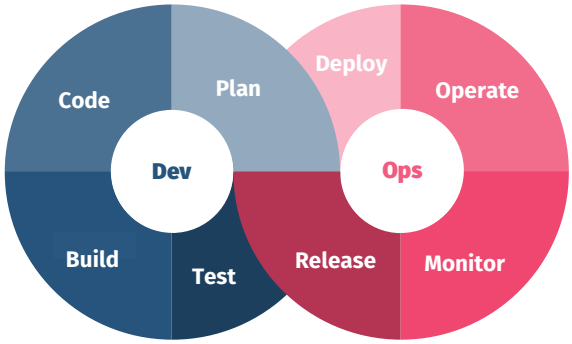

Fig. 12: DevOps concept portrayed by the infinite loop

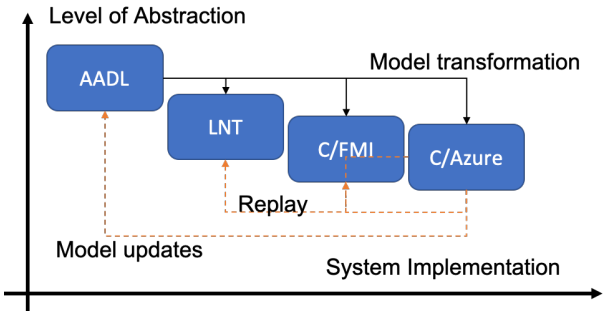

Fig. 13: TwinOps forward and feedback loop [60]

Proposed by Hugues et al. [60], TwinOps borrows the concept of DevOps [61] from software engineering, where the development and operation are merged into one continuous loop of forward delivery and feedback (see Figure 12). In TwinOps, the 'Dev' part transcends to DT models integration and target codes generation; and the 'Ops' part is overloaded with data collection and analysis in DTs. As illustrated in Figure 13, the black arrows are the code generation forwarded to various targets, and the orange arrows represented the data analytic feedback. In the work of Hugues, the exemplified case study of a monitoring system for a building utilizes the technology stack consisting of AADL, LNT, C, FMI, and Azure to build a pipeline. The AADL toolchain is responsible for specifying the modeling architecture as well as requirements. The LNT target enables model-checking. The C/FMI target supports modeling and simulation of the virtual entity. Finally the C/Azure target leverages containerized cloud system to deploy execution command on the physical entity.

M2M—short for machine-to-machine—network is a popular topic in IoT applications. One of the open source frameworks is called LightweightM2M (LwM2M) which is created by the Open Mobile Alliance (OMA) with the emphasis on efficient bandwidth [63]. It is essentially a protocol stack that supports various operations from bootstrapping and registration, to device and service management within sensor networks. Figure 14 illustrates the overall deployment view of the LwM2M architecture. When applied to DT construction, one can map the physical entities to LwM2M client objects, and the virtual entities to LwM2M server objects. The idea is to leverage the protocol to perform the necessary integration and orchestration.

In [41] it presents a meta model that describes the abstract syntax of Ptolemy II software architecture, as seen in Figure 15. Every component in a Ptolemy II model is an instance of the `NamedObj` class. There are four subclasses of `NamedObj`.

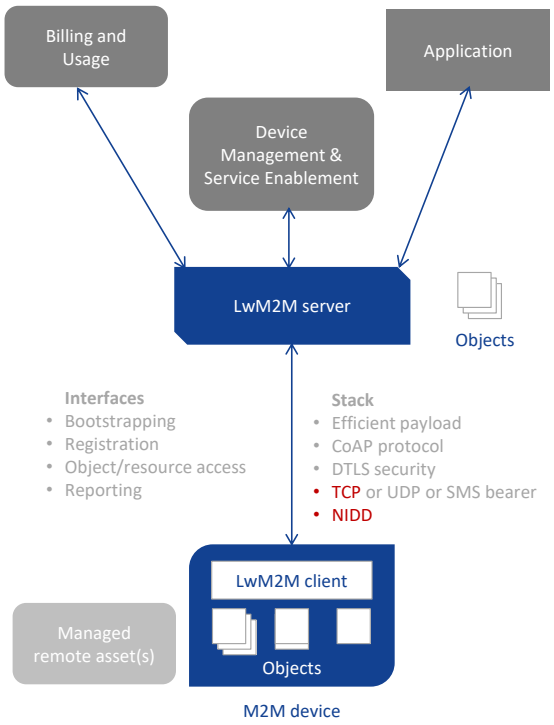

Fig. 14: LwM2M deployment [62]

		S1				S2			S3			S4		
	Strength	R1	R2	R3	R4	R1	R2	R3	R1	R2	R3	R1	R2	R3
TwinOps	Bridging Dev and Ops				O	O	O	O				O		O
	Multiple targets code generation		O						O	O				
	Traceable activities via containerization									O	O		O	
LwM2M	Bootstrapping and registration		O		O						O			
	Data communication	O		O		O	O	O				O		
	Objects and resource management					O	O							O
Ptolemy II	Pre-defined MoC	O										O		
	Support for time model	O				O	O					O		
	Actor-orientated modelling							O						

TABLE VI: Development framework comparison

These are called `Attribute`, `Entity`, `Port`, and `Relation`. The `Relation` class represents communication path between entities. The `Port` class has links to `Relation`, and it also hosts the `Receiver` interface that implements methods relating to data transmitting. Together they can be used to manage the model integration in DTs. On the other hand, for orchestration, it can be achieved in the `Executable` interface which is implemented by `Director`. The interface coordinates the iterations among actors based on the rules of the selected MoC.

In summary, shown in Table VI is a tentative assessment of each framework choice that has been introduced so far. In the table, we examine what service requirements could be benefited from the respective framework's strengths. Besides the inherent strengths of the framework's features, it is also necessary to discuss about the practicality side of developing with these frameworks. Here we recognize some deficiencies that might occur during the implementation phase. Remedies to mitigate the deficiencies will also be proposed.

The first deficiency is the ability to capture the high-level relationship of the DT dimensions, and to ensure its semantic

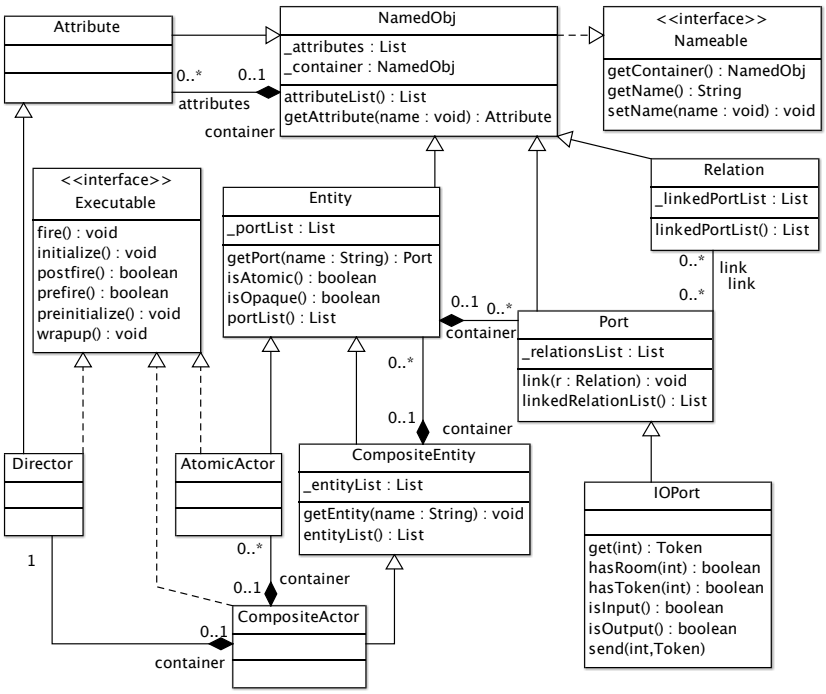

Fig. 15: Meta model of Ptolemy II

soundness. In Ptolemy II it is enabled by its self-explanatory graphical modeling environment, along with some limited capability of model error checking. In an use case of TwinOps, SysML is adopted for describing the models architecture. Further, it leverages on AADL to refine the more detailed component parts. In LwM2M there is no specified way of achieving model description. One solution can be using MBSE instruments such as SysML or UML to support preliminary analysis.

Another deficiency we predict that might occur in one or more of the frameworks is related to the methods of schedulability analysis and implementing schedules. In Ptolemy II there are built-in MoCs and sophisticated receiver queues, for instance, FIFO or LIFO implementations, that are able to support this aspect. In TwinOps it utilize the FMI approach, such that the master algorithm is made in AADL, which then be used to orchestrate the FMUs. LwM2M has described a (layered-)control scheme and the option to place objects in logical group in its protocol specification. The approaches in these frameworks are all considered valid, although they can be further enhanced by using sequence diagram and data flow diagram to formalize the control and data flow respectively.

The third deficiency is concerned with data communication pattern. Ptolemy II has predefined ‘relation’ and ‘port’ object to handle it, however, a potential shortcoming is that entities must be transformed to actors in order to work compatibly. In TwinOps, the detailed connection paths are modeled in AADL, then the pipeline will generate containerized binaries to specific targets. In LwM2M one can adopt various common common IoT strategies such as client/server (e.g., HTTP), pub/sub (e.g., MQTT) to achieve data communication. For the DT domain, it is advisable to highlight on the automation and scaling aspects.

The fourth deficiency relates to artifacts management such as data storage. In Ptolemy II there are few remarks about logging. TwinOps describes the usage of Azure IoT platform [64] for the distribution of containers. In LwM2M it is not specified, but cloud database servers are known to be advantageous in various M2M applications [65]. Building an offline or cloud database system is a sound approach, nevertheless in the DT context automation of version control is considered one of the focuses.

6 PROJECT PLANNING

In this section we propose a roadmap for this research project. The implementation phase is divided into two stages, just as shown in the project steps. In the first stage, a service—in this case, production prediction—from the four available DT services will be selected and implemented with the three frameworks outlined in Section 5.3. This is due to completion in mid-June. After that, an analysis with respect to integration and orchestration will be made. For the second stage, we will decide the most suitable framework out of three based on the previous experimentation and comparison, thus implementing the other three services with that framework. The expected completion date for the second stage is mid-August. All implementations will be tested throughout the project. The proposed timeline is presented in Figure 16.

Month-Week	Project steps (implementation phase)	Documentation
M5-W1		
M5-W2	Step 5 (build one service) begins	
M5-W3		
M5-W4		
M6-W1	Step 5 testing	
M6-W2	Step 5 completed	
M6-W3	Step 6	Report draft 1
M6-W4	Step 7 (build other services) begins	
M7-W1		
M7-W2		
M7-W3		
M7-W4		
M7-W5	Step 7 testing	
M8-W1		
M8-W2	Step 7 completed	Report draft 2
M8-W3	Step 8	
M8-W4	Step 9	
M9-W1	Buffer, in case of unexpected delay	Report draft 3
M9-W2	Buffer, in case of unexpected delay	Minor revisions
M9-W3	Buffer, in case of unexpected delay	Minor revisions
M9-W4	Buffer, in case of unexpected delay	Minor revisions

Fig. 16: Project timeline

REFERENCES

[1] M. W. Grieves, "Virtually intelligent product systems: Digital and physical twins," *Complex Systems Engineering: Theory and Practice*, pp. 175–200, 1 2019. [Online]. Available: https://arc.aiaa.org/doi/pdf/10.2514/5.9781624105654.0175.0200

[2] Boeing, "Key findings on airplane economic life," 3 2013. [Online]. Available: https://www.boeing.com/assets/pdf/commercial/aircraft_economic_life_-whitepaper.pdf

[3] P. Zheng, H. wang, Z. Sang, R. Y. Zhong, Y. Liu, C. Liu, K. Mubarok, S. Yu, and X. Xu, "Smart manufacturing systems for industry 4.0: Conceptual framework, scenarios, and future perspectives," *Frontiers of Mechanical Engineering 2018 13:2*, vol. 13, pp. 137–150, 1 2018. [Online]. Available: https://link.springer.com/article/10.1007/s11465-018-0499-5

[4] H. Kwon, S. An, H.-Y. Lee, W. C. Cha, S. Kim, M. Cho, and H.-J. Kong, "Review of smart hospital services in real healthcare environments," *Healthcare informatics research*, vol. 28, pp. 3–15, 1 2022. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/35172086

[5] Y. Duan, J. S. Edwards, and Y. K. Dwivedi, "Artificial intelligence for decision making in the era of big data – evolution, challenges and research agenda," *International Journal of Information Management*, vol. 48, pp. 63–71, 10 2019.

[6] PwC, "Industry 4.0 - publications." [Online]. Available: https://www.pwc.nl/en/publicaties/industrie-4-0.html

[7] IBM, "What is a digital twin?" [Online]. Available: https://www.ibm.com/topics/what-is-a-digital-twin

[8] S. Boschert and R. Rosen, "Digital twin-the simulation aspect," *Mechatronic Futures: Challenges and Solutions for Mechatronic Systems and Their Designers*, pp. 59–74, 1 2016. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-32156-1_5

[9] M. Macchi, I. Roda, E. Negri, and L. Fumagalli, "Exploring the role of digital twin for asset lifecycle management," *IFAC-PapersOnLine*, vol. 51, pp. 790–795, 1 2018.

[10] K. Y. H. Lim, P. Zheng, and C. H. Chen, "A state-of-the-art survey of digital twin: techniques, engineering product lifecycle management and business innovation perspectives," *Journal of Intelligent Manufacturing 2019 31:6*, vol. 31, pp. 1313–1337, 11 2019. [Online]. Available: https://link.springer.com/article/10.1007/s10845-019-01512-w

[11] F. Ansari, S. Nixdorf, and W. Sihn, "Insurability of cyber physical production systems: How does digital twin improve predictability of failure risk?" *IFAC-PapersOnLine*, vol. 53, pp. 295–300, 1 2020.

[12] H. Cai, J. Zhu, and W. Zhang, "Quality deviation control for aircraft using digital twin," *Journal of Computing and Information Science in Engineering*, vol. 21, 6 2021. [Online]. Available: https://asmedigitalcollection.asme.org/computingengineering/article/21/3/031008/1102047/Quality-Deviation-Control-for-Aircraft-Using

[13] D. G. Broo, M. Bravo-Haro, and J. Schooling, "Design and implementation of a smart infrastructure digital twin," *Automation in Construction*, vol. 136, p. 104171, 4 2022.

[14] M. Liu, S. Fang, H. Dong, and C. Xu, "Review of digital twin about concepts, technologies, and industrial applications," *Journal of Manufacturing Systems*, vol. 58, pp. 346–361, 1 2021.

[15] M. Shafto, M. Conroy, R. Doyle, E. Glaessgen, C. Kemp, J. LeMoigne, and L. Wang, "Modeling, simulation, information technology and processing roadmap," *National Aeronautics and Space Administration*, vol. 32, pp. 1–38, 2012.

[16] M. Schluse, M. Priggemeyer, L. Atorf, and J. Rossmann, "Experimentable digital twins-streamlining simulation-based systems engineering for industry 4.0," *IEEE Transactions on Industrial Informatics*, vol. 14, pp. 1722–1731, 4 2018.

[17] W. Kritzinger, M. Karner, G. Traar, J. Henjes, and W. Sihn, "Digital twin in manufacturing: A categorical literature review and classification," *IFAC-PapersOnLine*, vol. 51, pp. 1016–1022, 1 2018.

[18] I. A. Udugama, C. L. Gargalo, Y. Yamashita, M. A. Taube, A. Palazoglu, B. R. Young, K. V. Gernaey, M. Kulahci, and C. Bayer, "The role of big data in industrial (bio)chemical process operations," *Industrial and Engineering Chemistry Research*, vol. 59, pp. 15 283–15 297, 8 2020. [Online]. Available: https://pubs.acs.org/doi/full/10.1021/acs.iecr.0c01872

[19] I. A. Udugama, P. C. Lopez, C. L. Gargalo, X. Li, C. Bayer, and K. V. Gernaey, "Digital twin in biomanufacturing: challenges and opportunities towards its implementation," *Systems Microbiology and Biomanufacturing*, vol. 1, pp. 257–274, 2021. [Online]. Available: https://doi.org/10.1007/s43393-021-00024-0

[20] B. R. Barricelli, E. Casiraghi, J. Gliozzo, A. Petrini, and S. Valtolina, "Human digital twin for fitness management," *IEEE Access*, vol. 8, pp. 26 637–26 664, 2020.

[21] F. Tao and M. Zhang, "Digital twin shop-floor: A new shop-floor paradigm towards smart manufacturing," *IEEE Access*, vol. 5, pp. 20 418–20 427, 9 2017.

[22] C. L. Gargalo, S. C. de Las Heras, M. N. Jones, I. Udugama, S. S. Mansouri, U. Krühne, and K. V. Gernaey, "Towards the development of digital twins for the bio-manufacturing industry," *Advances in biochemical engineering/biotechnology*, vol. 176, pp. 1–34, 2020. [Online]. Available: https://link.springer.com/chapter/10.1007/10_2020_142

[23] A. Rasheed, O. San, and T. Kvamsdal, "Digital twin: Values, challenges and enablers from a modeling perspective," *IEEE Access*, vol. 8, pp. 21 980–22 012, 2020.

[24] "Tensorflow." [Online]. Available: https://www.tensorflow.org/

[25] "Pytorch." [Online]. Available: https://pytorch.org/

[26] H. Narayanan, M. F. Luna, M. von Stosch, M. N. C. Bournazou, G. Polotti, M. Morbidelli, A. Butté, and M. Sokolov, "Bioprocessing in the digital age: The role of process models," *Biotechnology Journal*, vol. 15, p. 1900172, 1 2020. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.1002/biot.201900172

[27] T. Zhou, R. Gani, and K. Sundmacher, "Hybrid data-driven and mechanistic modeling approaches for multiscale material and process design," *Engineering*, vol. 7, pp. 1231–1238, 9 2021.

[28] H. Jiang, S. Qin, J. Fu, J. Zhang, and G. Ding, "How to model and implement connections between physical and virtual models for digital twin application," *Journal of Manufacturing Systems*, vol. 58, pp. 36–51, 1 2021.

[29] S. C. Rutan, "Recursive parameter estimation," *Journal of Chemometrics*, vol. 4, pp. 103–121, 3 1990. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.1002/cem.1180040203

[30] R. M. Oisiovici and S. L. Cruz, "State estimation of batch distillation columns using an extended kalman filter," *Chemical Engineering Science*, vol. 55, pp. 4667–4680, 10 2000.

[31] D. Krämer and R. King, "On-line monitoring of substrates and biomass using near-infrared spectroscopy and model-based state estimation for enzyme production by s. cerevisiae," *IFAC-PapersOnLine*, vol. 49, pp. 609–614, 1 2016.

[32] "PID theory explained - NI." [Online]. Available: https://www.ni.com/nl-nl/innovations/white-papers/06/pid-theory-explained.html

[33] M. S. Hong, K. A. Severson, M. Jiang, A. E. Lu, J. C. Love, and R. D. Braatz, "Challenges and opportunities in biopharmaceutical manufacturing control," *Computers & Chemical Engineering*, vol. 110, pp. 106–114, 2 2018.

[34] "the cape-open laboratories network — expanding process modelling capability through software interoperability standards." [Online]. Available: https://www.colan.org/

[35] "Functional mock-up interface." [Online]. Available: https://fmi-standard.org/

[36] T. Blochwitz, M. Otter, M. Arnold, C. Bausch, C. Clauss, H. Elmqvist, A. Junghanns, J. Mauss, M. Monteiro, T. Neidhold, D. Neumerkel, H. Olsson, J.-V. Peetz, and S. Wolf, "The functional mockup interface for tool independent exchange of simulation models," *Proceedings from the 8th International Modelica Conference, Technical Univeristy, Dresden, Germany*, vol. 63, pp. 105–114, 6 2011.

[37] J. P. Belaud and M. Pons, "Open software architecture for process simulation: The current status of cape-open standard," *Computer Aided Chemical Engineering*, vol. 10, pp. 847–852, 1 2002.

[38] "Functional mock-up interface specification." [Online]. Available: https://fmi-standard.org/docs/3.0-dev/

[39] "Ptolemy project home page." [Online]. Available: https://ptolemy.berkeley.edu/

[40] "SysML open source project - what is SysML? who created it?" [Online]. Available: https://sysml.org/

[41] C. Ptolemaeus, *System design, modeling, and simulation: using Ptolemy II*. Ptolemy.org, 2014, vol. 1.

[42] S. Tripakis and E. A. Lee, "Fundamental algorithms for system modeling, analysis, and optimization," 2014.

[43] "OMG system modeling language specification version 1.0." [Online]. Available: https://www.omg.org/spec/SysML/1.0/About-SysML/

[44] D. Preuveneers, W. Joosen, and E. Ilie-Zudor, "Robust digital twin compositions for industry 4.0 smart manufacturing systems," *Proceedings - IEEE International Enterprise Distributed Object Computing Workshop, EDOCW*, vol. 2018-October, pp. 69–78, 11 2018.

[45] A. Borghesi, G. D. Modica, P. Bellavista, V. Gowtham, A. Willner, D. Nehls, F. Kintzler, S. Cejka, S. R. Tisbeni, A. Costantini, M. Galletti, M. Antonacci, and J. C. Ahouangonou, "Iotwins: Design and implementation of a platform for the management of digital twins in industrial scenarios," *Proceedings - 21st IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing, CCGrid 2021*, pp. 625–633, 5 2021.

[46] M. H. Hung, Y. C. Lin, H. C. Hsiao, C. C. Chen, K. C. Lai, Y. M. Hsieh, H. Tieng, T. H. Tsai, H. C. Huang, H. C. Yang, and F. T. Cheng, "A novel implementation framework of digital twins for intelligent manufacturing based on container technology and cloud manufacturing services," *IEEE Transactions on Automation Science and Engineering*, 2022.

[47] "Kubernetes." [Online]. Available: https://kubernetes.io/

[48] L. Mears, S. M. Stocks, M. O. Albaek, B. Cassells, G. Sin, and K. V. Gernaey, "A novel model-based control strategy for aerobic filamentous fungal fed-batch fermentation processes," *Biotechnology and Bioengineering*, vol. 114, pp. 1459–1468, 7 2017. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.1002/bit.26274

[49] P. C. Lopez, I. A. Udugama, S. T. Thomsen, C. Bayer, H. Junicke, and K. V. Gernaey, "Promoting the co-utilisation of glucose and xylose in lignocellulosic ethanol fermentations using a data-driven feed-back controller," *Biotechnology for Biofuels*, vol. 13, pp. 1–14, 12 2020. [Online]. Available: https://biotechnologyforbiofuels.biomedcentral.com/articles/10.1186/s13068-020-01829-2

[50] F. Feidl, S. Vogg, M. Wolf, M. Podobnik, C. Ruggeri, N. Ulmer, R. Wälchli, J. Souquet, H. Broly, A. Butté, and M. Morbidelli, "Process-wide control and automation of an integrated continuous manufacturing platform for antibodies," *Biotechnology and Bioengineering*, vol. 117, pp. 1367–1380, 5 2020. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.1002/bit.27296

[51] T. Eppinger, G. Longwell, P. Mas, K. Goodheart, U. Badiali, and R. Aglave, "Increase food production efficiency using the executable digital twin (xdt)," *Chemical Engineering Transactions*, vol. 87, pp. 37–42, 7 2021. [Online]. Available: https://www.cetjournal.it/index.php/cet/article/view/CET2187007

[52] M. van den Brand, L. Cleophas, R. Gunasekaran, B. Haverkort, D. A. Negrin, and H. M. Muctadir, "Models meet data: Challenges to create virtual entities for digital twins," *Companion Proceedings - 24th International Conference on Model-Driven Engineering Languages and Systems, MODELS-C 2021*, pp. 225–228, 2021.

[53] D. A. Negrin, L. Cleophas, and M. van den Brand, "Using Ptolemy II as a framework for virtual entity integration and orchestration in digital twins," *Companion Proceedings - 24th International Conference on Model-Driven Engineering Languages and Systems, MODELS-C 2021*, pp. 233–236, 2021.

[54] H. Helgers, A. Schmidt, and J. Strube, "Towards autonomous process control—digital twin for cho cell-based antibody manufacturing using a dynamic metabolic model," *Processes*, vol. 10, 2022. [Online]. Available: https://www.mdpi.com/2227-9717/10/2/316

[55] G. Tolksdorf, E. Esche, J. van Baten, and G. Wozny, "Taylor-made modeling and solution of novel process units by modular cape-open-based flowsheeting," *Computer Aided Chemical Engineering*, vol. 38, pp. 787–792, 2016. [Online]. Available: http://dx.doi.org/10.1016/B978-0-444-63428-3.50136-3

[56] H. Neema, J. Gohl, Z. Lattmann, J. Sztipanovits, G. Karsai, S. Neema, T. Bapty, J. Batteh, H. Tummescheit, and C. Sureshkumar, "Model-based integration platform for fmi co-simulation and heterogeneous simulations of cyber-physical systems," *Proceedings of the 10th International Modelica Conference, March 10-12, 2014, Lund, Sweden*, vol. 96, pp. 235–245, 3 2014.

[57] D. Krone, E. Esche, N. Asprion, M. Skiborowski, and J. U. Repke, "Conceptual design based on superstructure optimization in gams with accurate thermodynamic models," *Computer Aided Chemical Engineering*, vol. 48, pp. 15–20, 1 2020.

[58] E. A. Lee and S. Neuendorffer, "Actor-oriented models for codesign," *Formal Methods and Models for System Design*, pp. 33–56, 2004. [Online]. Available: https://link.springer.com/chapter/10.1007/978-1-4020-8052-4_2

[59] S. Karolius and H. Preisig, "Developing simulation tools for interdisciplinary modelling," *Proceedings of The 59th Conference on imulation and Modelling (SIMS 59), 26-28 September 2018, Oslo Metropolitan University, Norway*, vol. 153, pp. 210–215, 11 2018.

[60] J. Hugues, A. Hristosov, J. J. Hudak, and J. Yankel, "TwinOps - DevOps meets model-based engineering and digital twins for the engineering of CPS," *Proceedings - 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems, MODELS-C 2020 - Companion Proceedings*, p. 668, 10 2020. [Online]. Available: https://doi.org/10.1145/3417990.3421446

[61] C. Ebert, G. Gallardo, J. Hernantes, and N. Serrano, "Devops," *IEEE Software*, vol. 33, pp. 94–100, 5 2016.

[62] OMA, "Lwm2m v1.1," 2019. [Online]. Available: http://www.openmobilealliance.org/release/LightweightM2M/Lightweight_Machine_to_Machine-v1_1-OMASpecworks.pdf

[63] ——, "Lightweight machine to machine technical specification: Core," 2020. [Online]. Available: https://www.omaspecworks.org/about/intellectual-property-rights/.

[64] Microsoft, "Azure IoT – internet of things platform." [Online]. Available: https://azure.microsoft.com/en-us/overview/iot

[65] P. K. Verma, R. Verma, A. Prakash, A. Agrawal, K. Naik, R. Tripathi, M. Alsabaan, T. Khalifa, T. Abdelkader, and A. Abogharaf, "Machine-to-machine (m2m) communications: A survey," *Journal of Network and Computer Applications*, vol. 66, pp. 83–105, 5 2016.