Journal of Intelligent Manufacturing
https://doi.org/10.1007/s10845-021-01860-6

Pipeline for ontology-based modeling and automated deployment of digital twins for planning and control of manufacturing systems

Amon Göppert[1] · Lea Grahn[1] · Jonas Rachner[1] · Dennis Grunert[2] · Simon Hort[2] · Robert H. Schmitt[1,2]

Received: 4 January 2021 / Accepted: 9 October 2021

Abstract

The demand for individualized products drives modern manufacturing systems towards greater adaptability and flexibility. This increases the focus on data-driven digital twins enabling swift adaptations. Within the framework of cyber-physical systems, the digital twin is a digital model that is fully connected to the physical and digital assets. A digital model must follow a standardization for interoperable data exchange. Established ontologies and meta-models offer a basis in the definition of a schema, which is the first phase of creating a digital twin. The next phase is the standardized and structured modeling with static use-case specific data. The final phase is the deployment of digital twins into operation with a full connection of the digital model with the remaining cyber-physical system. In this deployment phase communication standards and protocols provide a standardized data exchange. A survey on the state-of-the-art of these three digital twin phases reveals the lack of a consistent workflow from ontology-driven definition to standardized modeling. Therefore, one goal of this paper is the design of an end-to-end digital twin pipeline to lower the threshold of creating and deploying digital twins. As the task of establishing a communication connection is highly repetitive, an automation concept by providing structured protocol data is the second goal. The planning and control of a line-less assembly system with manual stations and a mobile robot as resources and an industrial dog as the product serve as exemplary digital twin applications. Along this use-case the digital twin pipeline is transparently explained.

Keywords Digital twin · Ontology · Manufacturing · Deployment · Planning · Control

Introduction and motivation

Mass customization and other trends towards shorter life cycles and reduced lot size are driving manufacturing systems towards increased adaptability and flexibility (Hu 2013). Results of the increasing flexibility are complexity and shorter required reaction times, which is especially impacting assembly systems as those challenges accumulate in this final step of the manufacturing value chain (Wiendahl et al. 2004). Upcoming alternative assembly forms strive away from dedicated assembly lines to increase adaptability and flexibility (Greschke et al. 2014; Qamar et al. 2018). In such flexible assembly systems, the complexity of planning and control increases due to the higher degrees of freedom in the material flow (Hüttemann et al. 2017). The task of such production planning and control is the continuous adaptation of the current system status to an optimized state (VDI 2018). For this purpose, information must be collected and exchanged between system assets. Cyber-Physical Production Systems (CPPS) contain digital entities and the surrounding physical world (i.e. the real manufacturing system) (Acatech, 2011; Monostori, 2014). A major challenge for interaction between entities of the CPPS is interoperability, so that continuous communication can take place without human intervention (Hoffmann, 2017).

A core concept to tackle this challenge in CPPS are interoperable digital twins (DT) (Hildebrandt et al., 2018). A high effort is involved in the deployment of digital twins (Madni et al., 2019). To utilize the interoperability advantages of a digital twin, the implementation efforts must be reduced. Most research on digital twins focuses on the ontology-based definition and modeling with the support of a use-case (Hildebrandt et al., 2020; Liu et al., 2020). Only a few publications

✉ Amon Göppert
a.goeppert@wzl.rwth-aachen.de

1 Laboratory for Machine Tools and Production Engineering (WZL), RWTH Aachen University, Aachen, Germany

2 Fraunhofer Institute for Production Technology IPT, Aachen, Germany

focus on the deployment of digital twins or on a workflow that describes a methodology to completely create a specific digital twin and deploy it in an operating environment (Steinmetz et al. 2017). Therefore, the first objective and contribution of this paper is the design of an end-to-end workflow that shows practitioners and researchers the three main steps of ontology-based definition, standardized modeling and automated deployment of digital twins and the respective intermediate steps. This workflow is coined as the digital twin pipeline, following the concept of AutoML pipelines that aim to reduce the effort in creating machine learning models with automation (cf. AutoML pipelines: Yakovlev et al. 2020; Truong et al. 2019).

The effort for the deployment of digital twins is further increased because data is modeled differently for different systems. For example, different types of models exist for simulation and data transfer (Hoffmann, 2017; Lee & Riddick, 2010; Rodič, 2017). Thus, there is a lack of consistency and uniform models for use in different systems and different use-cases. Minimizing complexity can be achieved by standardizing the ontology-based description model of digital twins (Boss et al., 2020). Especially repetitive deployment for various subsystems can be automated with a standardized data model basis. Thus, the second objective of this work is to derive a concept for the automated deployment of digital twins based on data model standardization. This objective specifically addresses the third step of the digital twin pipeline but distinguishes from the first objective by focussing on the model-based automated deployment and not on the design of the overarching pipeline workflow. The two main objectives seek to contribute to lowering the threshold of bringing digital twins into an application.

After introducing and motivating this work, the foundations, consisting of reference architectures, semantic and syntactic data modeling, the definition of digital twin and model, and communication standards and protocols, are explained in "Foundations" section. In "State of the art" section the state of the art is presented and assessed regarding the publication's contribution to each of the three steps of the digital pipeline. Consequently, required developments on the basis of the gap in the state of the art are derived. The main "Digital twin pipeline" section gives an overview on the digital twin pipeline and subsequently describes in detail each step of the pipeline by giving practical examples and use-cases for a clear understanding. Lastly, in "Conclusion and outlook" section a conclusion of the results and the presented work, and an outlook with further research opportunities is given.

Foundations

This section lays the foundations for the design of the pipeline for creating digital twins by taking established reference architectures from smart manufacturing as a frame for further explanations. The basic tools of ontology-based semantic and syntactic data modeling are described for the standardized creation of digital models. Data modeling is especially relevant for the definition and modeling phase of the developed pipeline. The difference between digital models and digital twins is defined to build an understanding of the deployment tasks. Finally, relevant communication standards and protocols are outlined as they are a central aspect for enabling digital twins and their automated deployment.

Smart manufacturing: reference architectures

Reference architectures define the basic framework for the digitalization of production systems. They create a common understanding of Industry 4.0 through standardization and integration of essential components into a framework. Essential components are the representation of assets and their integration of data, communication technologies and the use of established standards to ensure interoperability not only within a production system but across networks.

A multitude of initiatives has published frameworks and guidelines for their interpretation of smart manufacturing (an overview is given by: Bader et al., 2019; Moghaddam et al., 2018; Weyrich & Ebert, 2016). In Germany, the initiative 'Plattform Industrie 4.0' developed the Reference Architecture Model Industrie 4.0 (RAMI4.0) with the goal to sufficiently precise the description of an asset or a combination of assets over the entire product life cycle, whereas an asset is defined as an object that has a value to an organization (DIN SPEC 91345 2016). RAMI4.0 is a layer model in which the complex correlations of an asset are structured in such a way that every relevant aspect is represented at any point in the life cycle. Further details on RAMI4.0 can be seen in the referring DIN SPEC (DIN SPEC 91345 2016).

Besides RAMI4.0, the US-based 'Industrial Internet Consortium' (IIC) developed the Industrial Internet Reference Architecture (IIRA) which provides guidance and assistance in the development, documentation, communication and deployment of unique Industrial Internet of Things (IIoT) systems as an architectural template and assists in achieving a common understanding to enhance system interoperability across industrial sectors (Industrial Internet Consortium 2019). Approaches to align both reference architectures IIRA and RAMI and the concepts of the digital twin and the asset administration shell (AAS) were made in cooperation by both initiatives (Boss, 2020). The core message is that IIRA emphasizes broad applicability and interoperability across industries while the service-oriented RAMI 4.0

reaches deeper in describing models for the digitalization of manufacturing. Moreover, the concept of the asset administration shell meets the key requirements for digital twins as described by the IIRA and supports digital twin use-cases outside of the manufacturing domain.

Further cooperation has been made with the Chinese Intelligent Manufacturing System Architecture (IMSA) and the Industrial Value Chain Reference Architecture (IVRA) by the Japanese Industrial Value Chain Initiative (IVI). While the Hierarchy Levels axis and the Layers axis of RAMI are very similar to the defined axis in IMSA, both approaches differ in the definition of life cycle phases (Sino-German Industrie 4.0 2018). At the center of IVRA is the three-dimensional model of a smart manufacturing unit. It contains the axes Activity View, Management View and Asset View, which shows that information is arranged differently than in RAMI (Industrial Value Chain Initiative 2018). There still is a need for harmonization and compatibility of new and existing reference architectures to develop a cross-industry and cross-country understanding of industry 4.0 components (Standardization Council Industrie 4.0 2020). All frameworks have in common that consistent modeling based on a digital representation of the physical objects of production systems is a necessary foundation for smart manufacturing. The introduced examples show the variety and quantity of different reference architectures in the context of Smart Manufacturing and IoT. However, the results of this research are based on RAMI4.0 to secure standardized exploitation. For the communication standards and protocols, the IIoT Connectivity Stack Model by IIRA is introduced in "Communication standards and protocols" section.

Ontology-based semantic and syntactic data modeling

Fundamental to the automated modeling of a digital twin is a thorough definition of the domain-specific assets. Ontologies enable a common understanding of a domain and facilitate communication between humans and systems (Andrew, 2004). Therefore, ontologies include both the conceptual description of a domain (Semantics) and the specification of this conceptualization in terms of a formal description (Syntax) (Gruber, 1995). Lemaignan et al. (2006) distinguish between ontologies at conceptualization level and operational level. At the conceptualization level, ontologies investigate and formalize a company's knowledge. The main purpose at the operation level is to ensure interoperability in the data exchange of heterogeneous systems. For both purposes, ontologies study the categories of entities and their relationships with each other. For the present domain of production systems, a multitude of ontologies exist that follow the PPR principle, in which production systems can be described by the three main components (1) product, (2) process and (3) resource (Cao et al., 2019; Cutting-Decelle et al., 2007). Accordingly, a domain-specific ontology includes a catalog of all components and assemblies that are used and the product structure of products to be manufactured. Additionally, a catalog of all resources and their structure is necessary. The product structure provides information about the composition of a product from components, assemblies and individual parts (Schuh, 2014). The resource structure comprises on the one hand the description of the capabilities of a resource, on the other hand, information about their spatial design. These three assets are interrelated, as resources (e.g. drilling machine) are used to execute a process (e.g. drilling) on a product (e.g. metal sheet) (Martin & D'Acunto, 2003). In addition to the three main assets, ontologies include the factory structure definition by the spatial arrangement of the resources (VDI, 2016). Further ontologies for specific shop floor related information (e.g. walls, buildings) exist (Balaji et al. 2016; Brickschema, 2020; Linked Building Data Community Group (LBD) 2020; Rasmussen et al. 2019). Existing ontologies can be divided into comprehensive (e.g. Lemaignan et al., 2006; Usman et al. 2011) and specific ontologies for products (e.g. Panetto et al., 2012) and resources (e.g. Weser et al., 2020; Järvenpää et al., 2019; Wan et al. 2018). They can be distinguished according to their field of application (e.g. simulation: Lee & Riddick, 2010), degree of detail and the modeling language used.

The majority of recent ontology developments are modeled using the Web Ontology Language (OWL) (Antoniou & van Harmelen, 2004). OWL is a specification of the World Wide Web Consortium (W3C) originally designed for web applications. OWL extends the underlying the Resource Description Framework (RDF) syntax by the semantic definition of relationships ("*oneOf*", "*unionOf*"). For this purpose, it distinguishes between classes, properties and instances. Classes stand for concepts that can have properties. Instances are individuals of one or more classes. To formally describe entities and their relationships, one of three OWL specifications (OWL Lite, OWL DL and OWL Full) can be used depending on the application (McGuinness et al. 2004).

Regardless of the domain under consideration, meta-models serve to formalize the modeling of existing assets and their relationships. The meta-model of the asset administration shell is a standardized information model for the application in Industry 4.0 (see Fig. 1) (Boss, 2020). The concept differentiates between asset types and asset instances. Asset types consist of information from the technical datasheet while instances include actual instances of measured data. Asset instances are linked to asset types. The relationship allows a continuous forwarding of updates of the type to the asset instances automatically or on-demand (Tantik & Anderl, 2017). A shell consists of a header and a body. The header contains information to identify the corresponding asset (e.g. ID) and the body is composed of submodels and

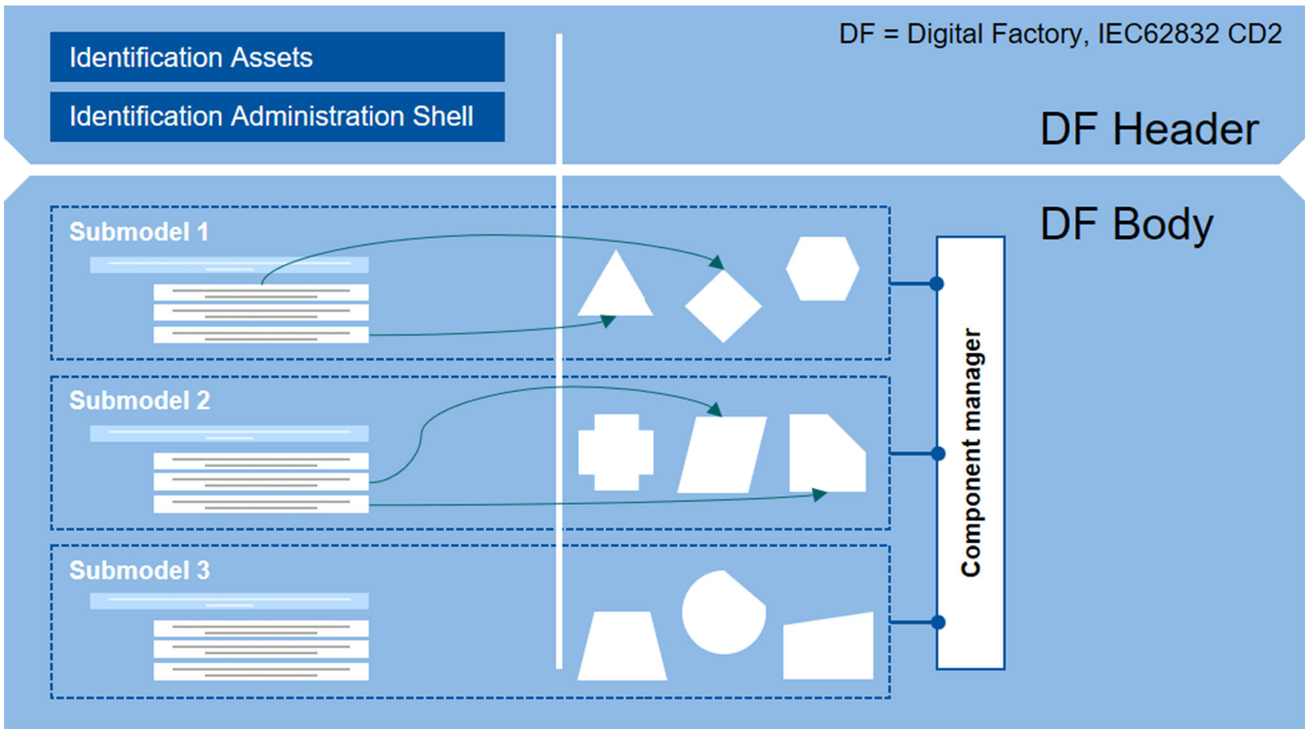


Fig. 1 Asset administration shell concept with header and body (based on: VDI, 2016)

contains information about the asset. Submodels are predefined properties to be used as semantic references for asset modeling (Boss, 2020). Using domain-specific submodels allows to create properties with a reference to existing property definition and therefore facilitates interoperability (e.g. ecl@ss Hepp, 2005). A communication manager allows updates and queries of information as a service (VDI, 2016, Boss et al. 2020). Its purpose is to facilitate data exchange along the entire value chain by providing a standardized digital representation for intelligent and non-intelligent assets. Interoperability is ensured through a standardized communication interface (Tantik & Anderl, 2017).

SOIL (SensOr Interfacing Language) is a meta-model specifically designed for sensor systems, but its underlying information model is applicable to other domains as production systems. The modeling of an asset using four basic building elements is a foundation for generic modeling that enables standardized communication interfaces. Assets (e.g. sensor) are defined through (1) components, (2) functions, (3) parameters and (4) measurements (see Fig. 2). All elements are inherited from a base class that includes an ID, a human-readable name and a description. Measurements are dynamically changed information (e.g. temperature). Parameters cover static data that is only changed through external interaction (e.g. calibration data). Functions consist of a set of input arguments and a set of return parameters. Components are the structural representation of an asset, including an arbitrary number of children elements (Bodenbenner et al., 2020).

In addition to an underlying meta-model for semantics, a formalized syntax for modeling and machine-to-machine communication is necessary. JSON (JavaScript Object Notation) is the mainly used format for information exchange in web applications. It is based on JavaScript data types and establishes a precise protocol for receiving and answering API (Application Programming Interface) requests and responses over the HTTP protocol. Due to the simple syntax, the format is both machine-readable and human-readable. The JSON syntax consists of key-value pairs (Pezoa et al., 2016). JSON Schema is a standard to specify a schema for JSON documents. It is used to validate JSON files by comparing the information they contain with the referenced schema. The schema is divided into two parts: a mandatory schema section and an optional definition section to facilitate schema reuse (see Fig. 3). A JSON file satisfies a schema if all required keywords are included and the defined patterns are considered (Pezoa et al., 2016).

Definition of digital twin and model

The described asset administration shell supplies a meta-model for creating digital description models and digital twins. Often the discrimination between such a model and a digital twin is not clearly defined (Tao et al. 2018; Wright & Davidson, 2020). To define the components and the context

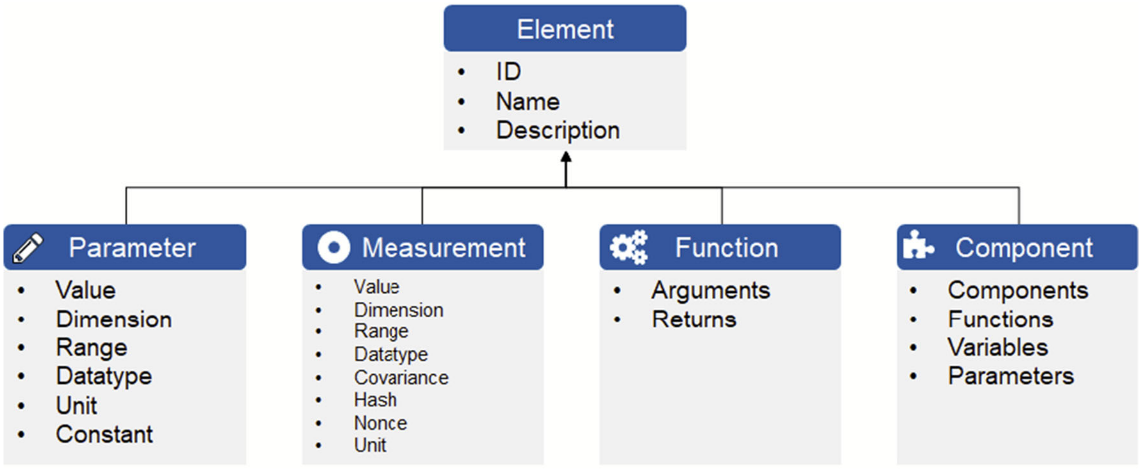
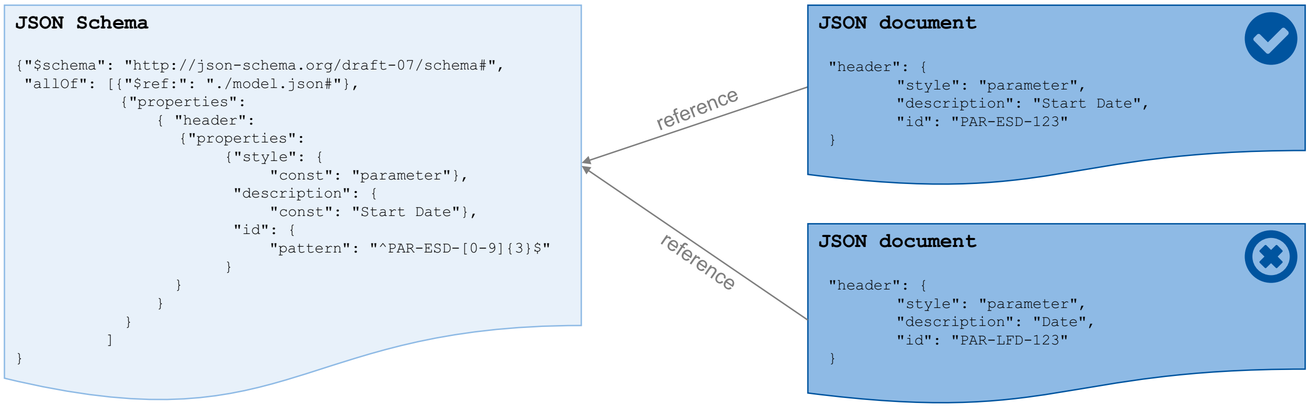


Fig. 2 SOIL meta model (Bodenbenner et al., 2020)



```
{"$schema": "http://json-schema.org/draft-07/schema#",
 "allOf": [{"$ref:": "./model.json#"},
           {"properties":
               { "header":
                  {"properties":
                       {"style": {
                            "const": "parameter"},
                        "description": {
                            "const": "Start Date"},
                        "id": {
                            "pattern": "^PAR-ESD-[0-9]{3}$"
                        }
                   }
               }
           }
         ]
}
```

```
"header": {
        "style": "parameter",
        "description": "Start Date",
        "id": "PAR-ESD-123"
}
```

```
"header": {
        "style": "parameter",
        "description": "Date",
        "id": "PAR-LFD-123"
}
```

Fig. 3 JSON Schema defining required properties for parameter of earlies starting date

of a digital twin, embedding it within the framework of cyber-physical systems (CPS) is often applied (Rosen et al., 2015; Stark & Damerau, 2019; Uhlemann et al., 2017; Weyer et al., 2016). Cyber-physical production systems (CPPS) are part of the Industry 4.0 context, enforcing flexibility of automation pyramid-based manufacturing production systems (Iarovyi et al. 2016). CPPS contain digital entities that are in intensive connection with the surrounding physical world (i.e. the real manufacturing system), its objects and its ongoing processes, providing and using, at the same time, services (Acatech, 2011; Monostori, 2014). In other words, a data flow between digital entities and physical objects exists. The approach of the CIRP Encyclopedia of Production Engineering by Stark and Damerau (2019) to define a digital twin supports this description. Kritzinger et al. (2018) state a digital twin as a digital and physical object with fully integrated and automatic data flow. This implies that a state change of a digital object leads to a change in the physical object and analogous a state change of a physical object leads to a change in the digital object as referred to as a twinning cycle by Jones et al. (2020). Such a coupling of objects allows for applications such as online simulation, production control, user operation guidance and real-time state-monitoring (Tao et al. 2018).

In contrast, the digital model contains a low degree of data flow integration, i.e. the data flow is just possible by manually transferring data from one object to another (Kritzinger et al., 2018). Figure 4 shows the difference, where the level of integration in terms of the automation of data flow is depicted. Stark and Damerau (2019) provide levels of connectivity between DT and its environment. Level 0 and Level 1 are for uni-directional and bi-directional connection, respectively. A DT in level 3 has an automatic connectivity with context-aware and self-directed communication. This level 3 DT supports the definition of a fully connected automatic data flow of Kritzinger et al. (2018).

An enhancement of this definition is nesting the digital model within the digital twin proposed by Cimino et al. (2019). This decouples the data acquisition and transfer from the digital model and defines the digital model itself as a digital object within the digital twin. Figure 5 shows this integration of the digital model within the digital twin.

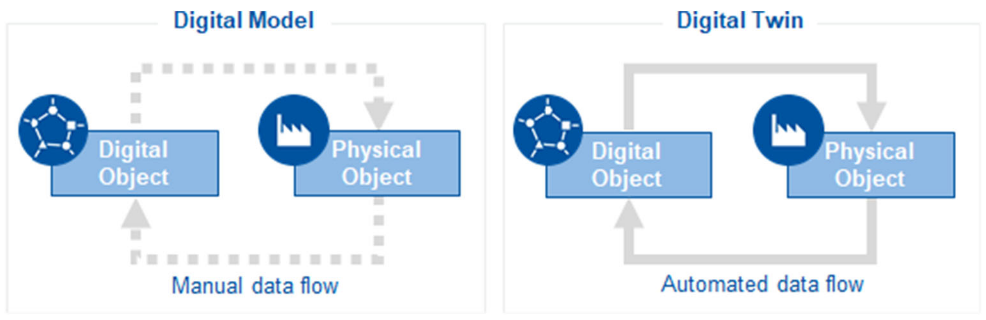


Fig. 4 Differentiation between a digital model and digital twin, based on data flow automation (based on Kritzinger et al., 2018)

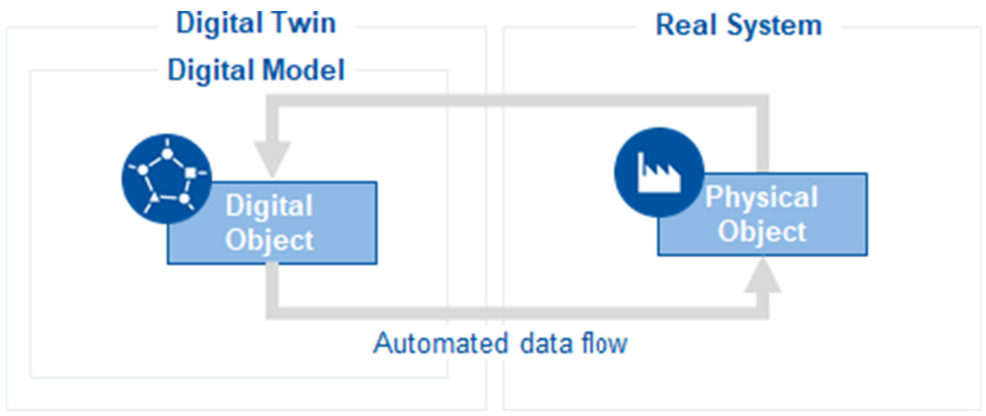


Fig. 5 Nested digital model within a digital twin (based on Cimino et al., 2019; Kritzinger et al., 2018)

In addition to the definition of a digital twin and a digital model, the digital shadow is coined as a term for an automated data flow from the physical to the digital side, but manual connection vice versa (Kritzinger et al., 2018). Since digital shadows are therefore mainly used for verification via simulation (Wohlfeld et al., 2017), the definition of the digital shadow is not utilized but described here for the sake of completeness.

In "Ontology-based semantic and syntactic data modeling" section the asset administration shell (AAS) meta-model with the concept of asset types and asset instances was introduced. For the development of DTs, Stark and Damerau (2019) propose a digital master as a DT prototype analogue to the AAS type. In Stark and Damerau (2019) an instantiation creates living DTs on the basis of the digital master prototype. In the AAS meta-model an instantiation leads to an instance, linked to its type. This comparison of AAS and the ideas of Stark and Damerau (2019) shows the suitability of AAS concepts for DT creation. As a consequence, in the later descriptions of the definition and modeling of DTs within these pipeline phases, the AAS concept is incorporated as a meta-model.

Communication standards and protocols

The Industrial Internet Consortium (IIC) proposed the IIoT Connectivity Stack Model as an enhancement of the existing layer models Open Systems Interconnection (OSI) and TCP/IP (Joshi et al. 2018). The layered architecture classifies communication standards and protocols (CSP) for the connection of endpoints in an IIoT environment (see Fig. 6). These CSP are shown in the connectivity part, which is subdivided into the framework and the transport layer. While the latter exchanges messages, the framework layer is responsible for the exchange of structured data, such as states, events and streams. The connectivity part builds on the capabilities of the network part below, which enables the exchange of packets (e.g. Internet Protocol (IP)), frames and bits (e.g. Ethernet, 5G). The information part, in turn, is responsible for the semantic interoperability and relies on the syntactic and technical interoperability of the framework and transport layer.

A connectivity transport standards build the foundation of the connectivity frameworks by defining messaging protocols, communication modes as well as endpoint addressing. Besides the widely used and applied protocols TCP and UDP, the Constrained Application Protocol (CoAP) protocol was specifically developed for IoT applications as a lightweight and efficient alternative based on HTTP and RESTful archi-

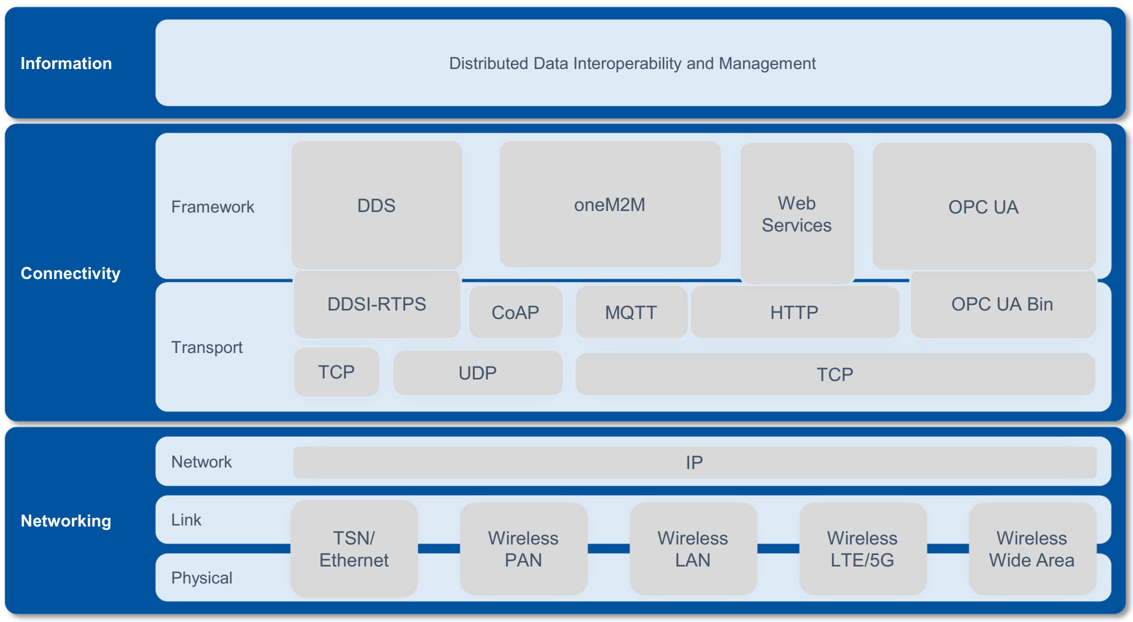


Fig. 6 Connectivity Stack Model as an enhancement of the existing layer models Open Systems Interconnection (OSI) and TCP/IP (based on Joshi et al., 2018)

tectural style. The Message Queuing Telemetry Transport (MQTT) protocol is also a lightweight protocol realizing a many-to-one data collection, which can be applied in large networks of small devices (e.g. sensors). The devices in the network (MQTT clients) are connected to a server (MQTT broker), where they can subscribe to certain topics or publish on them. Is a message published on a topic, it is sent to all MQTT clients subscribed to it. Even though the broker-based architecture prevents the real-time capability the messaging protocol realizes a small code footprint and a minimal network bandwidth. Because of that, the protocol is increasingly used in industrial use-cases such as automotive, logistics and manufacturing (Joshi et al. 2018; Shelby et al. 2014; OASIS MQTT Technical Committee 2019).

As shown, there are protocols the connectivity frameworks can operate with on the bottom layers, but there are few standards above the connectivity frameworks assuring semantically interoperability. To gain semantically interoperability, process knowledge in the form of description models and ontologies has to be integrated into the IIoT networks. Especially the integration of these models in all layers of the IIoT architecture requires time-ooconsuming human interaction. To optimize this process and aim for an automated integration of unambiguously, machine-readable process knowledge, ontologies as well as CSP need to be brought together. First approaches are developed mainly using the standards Open Platform Communications Unified Architecture (OPC UA), MQTT and oneM2M. Bunte et al. (2018) integrate common OWL ontologies into the OPC UA standard and only discovered few limitations (e.g. type restriction, Semantic Web Rule Language (SWRL) implementation). The SemOPC-UA lowers manual interaction while enhancing OPC-UA with Semantic Web Services (SWS) via OWL-S in order to generate a flexible orchestration plan which can be adapted to unforeseen events in production (Katti et al. 2018). Perzylo et al. (2019) developed a software tool that automatically transfers OPC UA NodeSets (graph-based data structures that comply with XML) to OWL models. An extension of the MQTT protocol in an oneM2M architecture is developed by Kim et al. (2019). Using semantic web technology in combination with the MQTT publish/subscribe messaging pattern devices can be automatically recognized and provide services autonomously. With the tool SemSub (Semantic Subscriptions), Piller and Khelil (2020) enable MQTT clients in an IoT network to autonomously subscribe without knowing the available topics. The topic is determined by analyzing keywords with semantic services.

Concluding remarks

The previous section lays the foundations for the design of the pipeline for creating digital twins. In the beginning, established reference architectures from smart manufacturing were presented as a frame for further development. Ontology-based concepts (e.g. PPR) were introduced to provide a framework for the definition and modeling phase of the pipeline. Specific presented modeling syntax (JSON) and semantic (SOIL) for data modeling will be applied in the pipeline. The difference between digital models and digital twins was defined to build an understanding of the deployment tasks. Finally, relevant communication standards and protocols were outlined as they will be a central aspect in the pipeline deployment phase.

State of the art

Automated ontology-based creation of digital twins

The core goal of this work is to design an end-to-end pipeline for creating digital twins in the manufacturing domain. The pipeline can be structured along the three phases ontology-based definition, standardized modeling and automated deployment. The state of the art was researched and analyzed based on a required contribution to the understanding of at least two phases of the DT pipeline. Another aim of the state-of-the-art research was to identify publications with an end-to-end methodology comparable to the DT pipeline approach.

Starting with the ontology-driven definition of a digital twin, Hildebrandt et al. (2018) describe an ontology building method in the context of CPS, including ontology requirements and a formal ontology. The building methodology seeks to ensure highly reproducible and interoperable ontologies by applying methodological building blocks and standardized ontology requirement specifications. The application of the methodology lies in the machine and process data communication in CPS. In 2020 Hildebrandt et al. further developed reusable design patterns for ontology building with industrial use-cases. In both works Hildebrandt et al. show a clear structural methodology for building ontologies that can be seen as a basis for interoperable digital twins. Hence, it is a strong contribution to the first definition phase and also supports standardized modeling, but does not touch the deployment phase. The structural ontology building is a required foundation for the automated generation of digital models and deployment of digital models, but the actual automation methodology is not part of the works of Hildebrandt et al. (2018).

Bao et al. (2020) demonstrate an ontology-based modeling of a digital twin. An ontology is created as a preparation for the modeling of the digital twin, although the actual method for the ontology building is lined out in a short description of seven steps towards an ontology. The focus of the paper lies on the modeling itself and the information sources during the modeling phase.

Wan et al. (2018) present an ontology-based method for semantic modeling of reconfigurable manufacturing resources. The context of the application is a manufacturing cyber physical system. In the modeling phase a manipulator serves as a practical example. The deployment of the cyber-physical manufacturing resources is not described.

Liu et al. (2020) describes digital twin-based CPS nodes as a hardware-software integrated agent for a physical manufacturing resource. In the course of creating these nodes an excerpt of the custom ontology is presented, and the ontology development process in the ontology editor Protegé (2020) is shown. The ontology-based modeling of the nodes is outlined for a geometric and kinematic description of a manipulator. The deployment of the digital twin-based CPS nodes is described in function modules, where ontology information is loaded by an ontology manager. Liu et al. apply web communication protocols such as MQTT for the data exchange between the CPS and the CPS node in the deployment phase. Although the deployment of the CPS nodes contains a seamless integration via the web communication tools, the deployment is manual and the communication structure is not integrated in the digital model.

Erkoyuncu et al. (2020) developed an ontology-based design framework for adaptive DTs. The focus of the framework is the interoperability of DT software and the possibility to update existing DTs with the ontology-based design framework. The results of the modeling is shown with a mobile robot use-case and a gearbox demonstrator. But, the methodology for modeling is not described. A deployment of the DT is not shown.

AboElHassan et al. (2021) describe a framework for deploying DTs into a manufacturing execution system or an enterprise resource planning system. The differentiation between DT and digital shadow is used as a basis for the role definition of a deployed DT. The framework bases on a generic description of DTs without ontologies as a standardization basis. The framework focuses on a message-oriented communication between the DT components. An approach for the automation of DT deployment is missing. Nevertheless, the framework partly contributes to the deployment of DTs.

Kousi et al. (2019) present digital modeling techniques in hybrid human and robot production systems. The DT in this case is a dynamically updated virtual representation of the shopfloor, combining real time sensor data, resource data and CAD models. A unified data model represents the infrastructure and its components. The data model lays the basis for a repeatable deployment of software components. Although the data model uses the unified modeling language, the modeling itself is not standardized as it is not following established meta-models or ontologies, which limits the transferability and interoperability between systems. The robot operating system (ROS) framework for managing resources and sensors is deployed automatically with configuration files, supplied by the DT. The communication and integration layer for the existing agents follows a publish-subscribe pattern and is implemented manually on top of the ROS framework. As a conclusion this paper partly contributes to the modeling phase and although the communication layer is not, the ROS framework is actually deployed automatically, which results in a full contribution to the last phase of the DT pipeline.

Park et al. (2020) developed a virtual representation for a DT application, based on the asset administration shell meta-model. The virtual DT representation to describe assets is aligned with the RAMI 4.0 reference architecture. Four stages of designing a DT in a conceptual map are described. Firstly, the DT definition has to fulfill a set of technical requirements. Among vertical and horizontal integration, indicator generation and simulation capabilities an outstanding and relevant requirement is the automatic DT creation with a configuration data library. The library contains a base model, logic and meta-data, which comprise sufficient information for creating a DT automatically. A further stage is the asset administration shell inheritance by, among other steps, implementing the type and instance concept and following a service-oriented architecture. An optional stage is the improvement of existing asset descriptions, which aims to fulfill the requirements in the first two stages. Finally, for the type and instance stages a set of technical functionalities such as planning and scheduling, monitoring and logistics design are addressed as a design stage. In conclusion Park et al. (2020) contribute to a meta-model-based modeling and deployment of DTs. An automatic deployment is not explicitly described, but the creation of DTs for described along an industrial use-case in an extensive manner.

Steinmetz et al. (2017, 2018) propose a tool to use ontologies for complex applications from different domains that need to share interpretable information for a common manufacturing purpose. The tool aims to facilitate the creation of Internet of Things (IoT) applications (Steinmetz et al. 2017) and CPS entities (Steinmetz et al., 2018). The facilitation utilizes information extraction from the ontology and generative programming techniques. The information extraction is implemented as an add-on for the common Protegé (2020) ontology editor to develop a model. A specific IoT lite ontology serves as a basis. The lite ontology as well as an excerpt from the model are presented in the publication. A code generation tool uses the model to automatically create a java class. The generated class is an interface for a device to interact with a middleware and thus helps with the automation of device integration.

The outline of the presented publications and the overview in Table 1 show that publications on the creation of DTs mostly deal with the ontology-based definition, which results from the relevance of interoperability of components, devices and devices in CPS. Only publications that involved at least two phases were included in the research. Several publications differentiate between lite and heavy ontologies for the sake of various applications of the derived models. In addition, the inheritance from domain independent meta-models and overarching ontologies such as those from the W3C is outlined in a clear and repeatable manner. The ontology editor Protegé (Noy et al. 2000; Protegé, 2020) appears in multiple publications making it a standard tool for ontology creation and editing. As standardized modeling of DTs is a core task in the creation process, all analyzed publications treat this topic. A full contribution to this pipeline phase contains a description of the modeling procedure and the supply of exemplary syntactic and semantic code. Exemplary code is shown in use-cases exhibiting products or manufacturing resources in CPS. The final phase of the DT pipeline, the automated deployment, is rarely found in the analyzed literature. When publications such as Park et al. (2020) or AboElHassan et al. (2021) describe a deployment of DTs, they focus on the information flow and integration framework, but they do not take the automation of DT deployment in CPS with code generation into account. Only Steinmetz et al. (2017, 2018) and Kousi et al. (2019) present concepts for the automatic generation of code for the facilitation of DT deployment.

Especially Liu et al. (2020) and Park et al. (2020) published methods guiding through the different phases of creating and deploying DTs in an extensive and clear manner. But these methods do not take the interface between modeling and deployment phase into account. The approaches lack a detailed description of how to bring a DT model into operation.

To conclude, the potentials for further work and research are an incomplete description of an end-to-end workflow for creating DTs in CPS and a concept for the automated deployment by code generation.

Derivation of required developments

Based on the revealed potentials in the state-of-the-art, the following required developments to contribute to the creation of DTs are derived. A comprehensive DT pipeline is required that shows the path from definition over modeling to deployment. The required interoperability in CPS must be ensured by inheriting from established ontologies and meta-models. Since the documentation of the DT modeling is mostly done in a generic manner, an exemplary use-case must be used to clarify how an ontology is used to create a digital model. As shown in “Definition of digital twin and model” section on the definition of digital model and twin, the difference lies in the automatic data flow between the physical and cyber objects. In other words, data communication between the digital model and the real world must be established by the generation of code. To ensure efficient repeatability of DT deployment in a CPS the code generation must be automated. The gap in the automation of DT deployment in the literature leads to the required development of suitable methods.

Table 1 Overview on the state-of-the-art for three phases of the digital twin creation pipeline

● Full contribution ○ Partial contribution	Ontology-based definition	Standardized modeling	Automated deployment
Hildebrandt et al. (2020)	●	●	
Bao et al. (2020)	●	●	
Wan et al. (2018)	●	●	
Liu et al. (2020)	●	●	○
Erkoyuncu et al. (2020)	●	○	
AboElHassan et al. (2020)		○	○
Kousi et al. (2019)		○	●
Steinmetz et al. (2017, 2018)	●	●	●
Park et al. (2020)	●	●	○

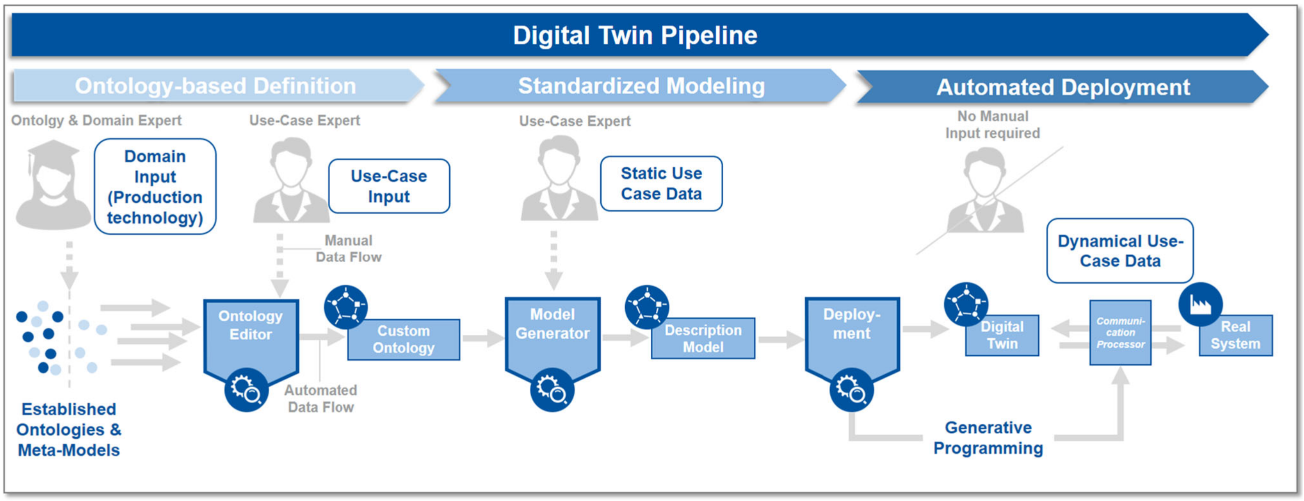


Fig. 7 Three phases of the end-to-end digital twin pipeline for the application in cyber-physical systems

Digital twin pipeline

Overview

A consistent workflow that includes the necessary stakeholders, phases, steps and data flow supports both practitioners and researchers to sustainably create and also deploy DTs into operation CPS environments. The proposed workflow is structured as an end-to-end pipeline, in the style of well-established workflows for machine learning models. Figure 7 shows the digital twin pipeline with three phases.

First, to ensure interoperability and reusability of DTs a meta-model and ontology-based standardization of the description model is required. Established ontologies and meta-models are pre-selected, according to domain and ontology knowledge input. This input must be supplied by an ontology and domain expert, so that all relevant state-of-the-art ontologies are taken into account. The domain knowledge is required to ensure a selection that fits the needs of a use-case within a specific domain. Due to the complexity of domains and the rare occurrence and necessity this selection process and the information input is kept manual. The selected ontologies and meta-models are implemented as a basis for creating a custom ontology in an ontology editor (e.g. Protegé, 2020). A use-case expert uses the selected ontologies and meta-models for creating a custom ontology according to digital twin requirements from a specific use-case. The ontologies and meta-models from the definition phase serve as a schema to standardize the creation of description models. A use-case expert transfers the use-case knowledge into the description model by filling out the provided schema. This information comprises static use-case data that can be supplied before start of operation. In the following Git repository exemplary schemata and data models are provided for reference and understanding: https://github.com/Project-AIMFREE/description_model

Since the use-cases are even more specific, analogous to the definition phase the input is provided manually. The result of the standardized modeling phase is a description model that has no data flow or simply manual data flow between digital objects (i.e. description model) and the real world (i.e. physical objects). The aim of the deployment is the establishment of a connection between digital and physical objects with automated machine-to-machine data flow. With communication standards and protocols (e.g. MQTT, see "Communication standards and protocols" section) the deployment can be automated, under the precondition that the required protocol components (e.g. topics, messages) are formally described in the standardized modeling. In contrast to defining schemata and creating description models, the deployment of digital twins is a repeatable task that is suitable for automation. Generative programming uses the formal descriptions to automatically create a communication master that manages the automatic data flow. Two applications are used to show automated DT deployment. First, the control of a line-less assembly system and second the planning with simulation-based scenario analysis.

Definition phase

In the definition phase, established ontologies and meta-models are pre-selected according to a domain input. Domain knowledge is used for pre-filtering to select relevant ontologies and meta models. Figure 8 shows this exemplary for the domain production systems. As an example, for the domain of production systems the presented ontologies on the domain level and meta models on the structure level can be used to create a domain-specific ontology in an ontology editor. The input can be divided into domain and structure level. The domain level includes ontologies that describe the semantics and connections between elements for the specific domain. In this case, MASON (Lemaignan et al., 2006) is used as a comprehensive ontology that describes products, processes and resources because of its general applicability for production systems and BOT (Rasmussen et al. 2019) is used to describe the general structure of the shopfloor and the factory building itself. Meta Models for the fundamental and consistent structure of the description model and the syntax for execution are described in the structure level. In this case, the concept of the asset administration shell and SOIL (cf. "Ontology-based semantic and syntactic data modeling" section) are used. Since the ontologies may partly overlap, the appropriate features are selected in an ontology editor. The creator may only use specific parts of an ontology and combine them (e.g. use one ontology for robot description and another one for capabilities and services). All information is collected in the ontology editor and a new, domain-specific ontology is created. However, if a well-fitting ontology with a consistent meta model already exists, it can be used straight away.

This results in a domain-specific ontology in which all syntax and semantic for the description model representing the digital twin is being described. All parts of the meta-model shown in Fig. 9 implement the superior header, which describes the defining information for every submodel. It defines the style, a unique ID, name and description of the described submodel. Following the paradigm of the asset administration shell, every submodel has a header with those properties and a body with individual information represented in further submodels. The main purpose of the header is the identification of every model. Identifiers are locally unique and become globally unique by prepending the parents' identifiers successively. Based on the SOIL model, the style of a submodel can be either a parameter, variable, function or component. A component again consists of further components, parameters, variables and functions, enabling complex systemic relationships to be depicted. The differentiation between parameters and variables allows the differentiation into time-dependent characteristics, which is relevant for communication and database structure.

The semantics of the derived ontology are depicted in Fig. 10. An order consists of different products that have to be produced. All assets can be differentiated between Types and Instances, as described in RAMI 4.0. As an example, a product matches a product type, which means that its submodels are applied. In addition, the described process step types and part types are applied, instantiated and assigned to the product. According to MASON, resources can be divided into Geographical Resources that in this case are represented by the BOT ontology, Human Resources and Machine Resources that can be either mobile or stationary. The matching process by a control system can be done by matching a certain process step to a resource on which it will be executed. The combination of resources (besides machines also operators and tools can be required) and a process step result in the process. The displayed objects are all components, which themselves consist of sub-models such as components, variables, parameters and functions. This is shown as an example for order and product where an order contains products, variables like the current lateness and parameters like the corresponding customer and the earliest start date (ESD). The included component product itself has a header with representing characteristics and a body with submodels like the included parts and process steps as components and the current position as a variable.

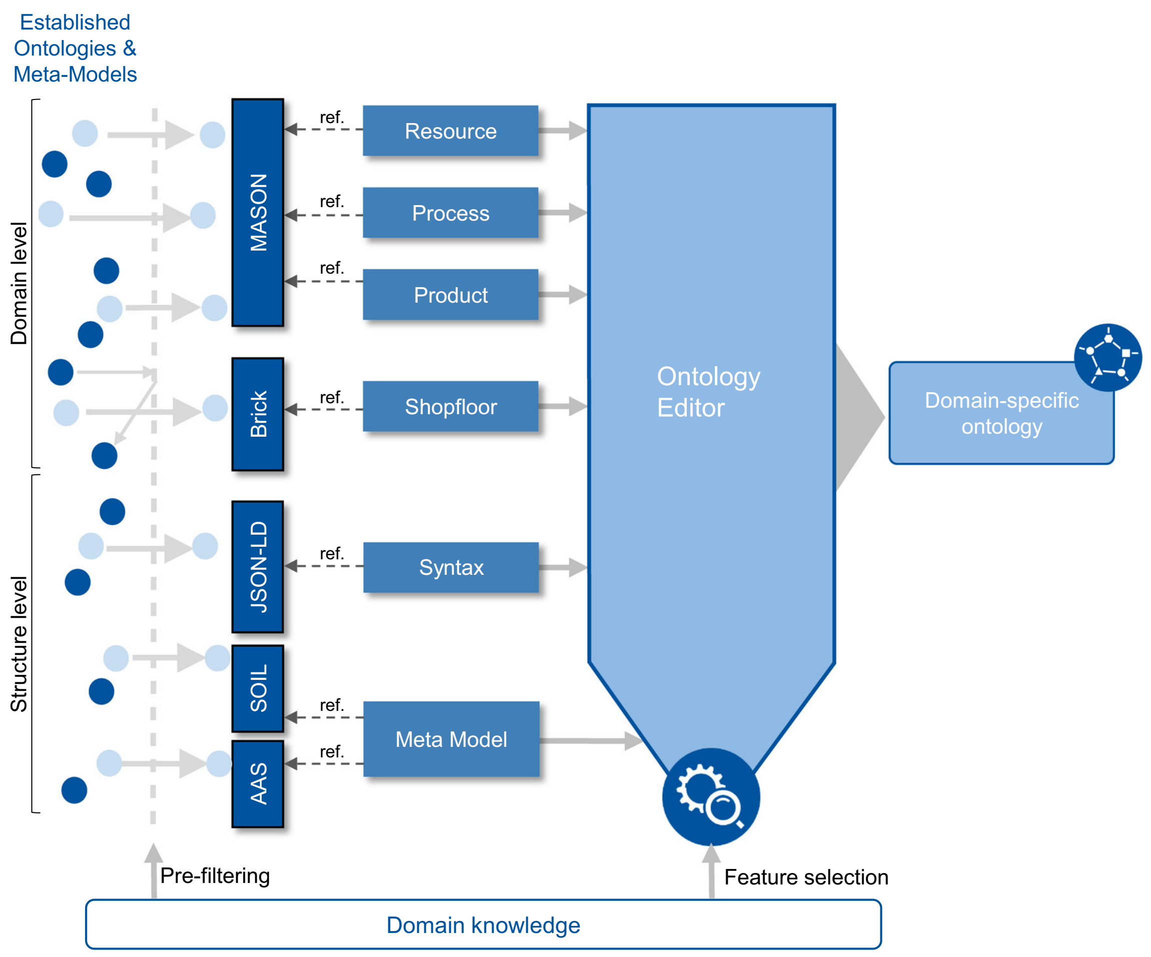


Fig. 8 Procedure of definition phase of the proposed pipeline

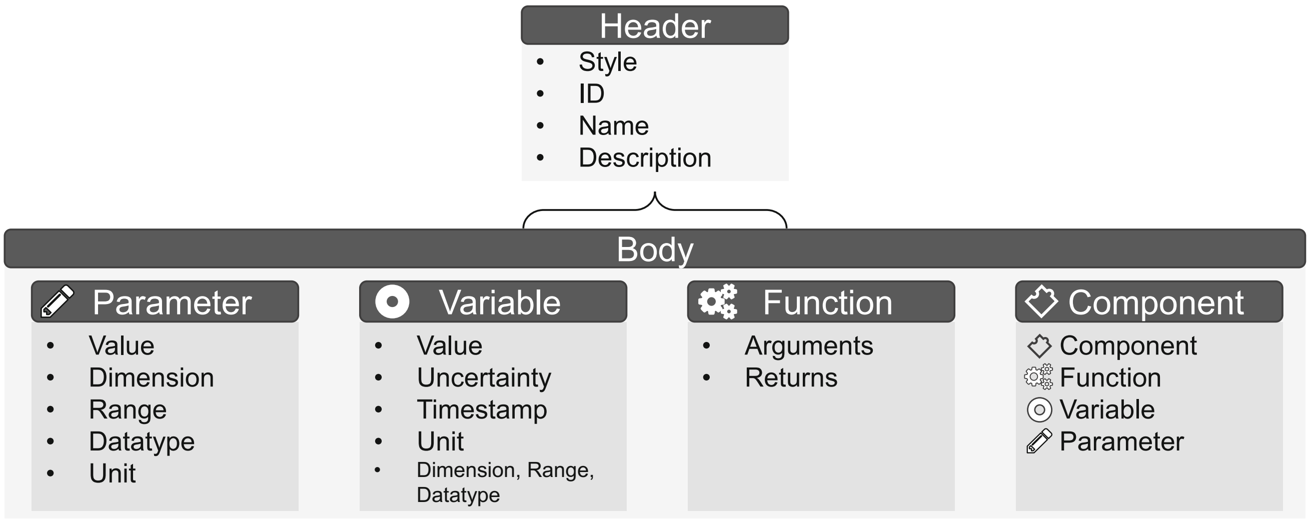


Fig. 9 Meta-model of the proposed description model combining aspects of SOIL and the asset administration shell

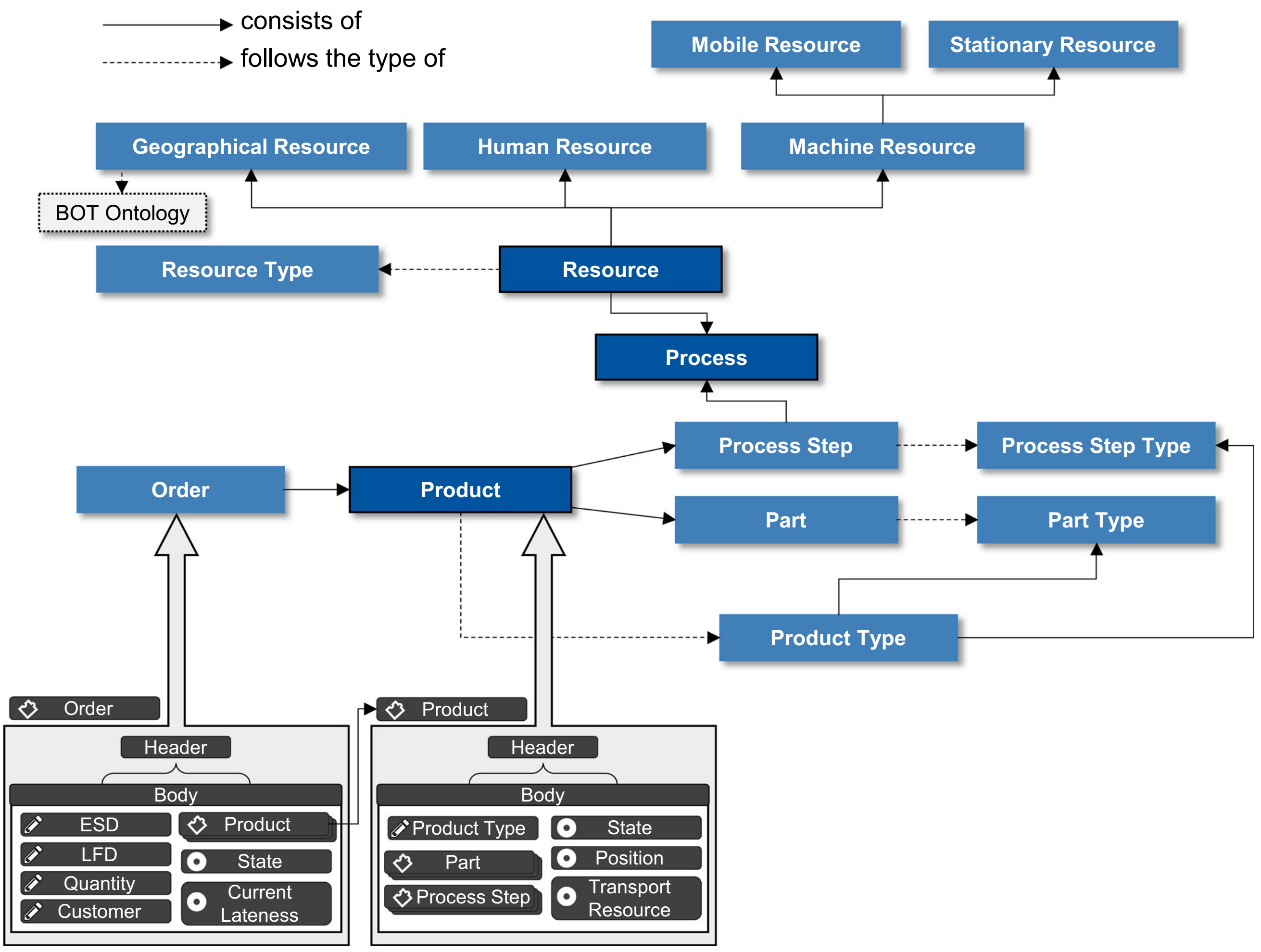


Fig. 10 Extract of the derived domain-specific ontology incorporating established ontologies and meta-models

Standardized modeling phase

During the modeling phase, a use-case-specific description model is created using a model generator (e.g. Bodenbenner et al., 2020, admin-shell-io 2020). The output of the definition phase is used as input information. Additional input is provided by static, use-case-specific information. Specific instances of the use-case are modeled according to the predefined ontology. The ontology provides the necessary and optional parameters and variables to describe the object of investigation. The predefined links between individual components serve as a blueprint for the links between use-case-specific instances. By using a priori defined submodels for parameters, variables and components a consistent modeling for the whole system is ensured. Similar parameters for different instances follow the same structure. By modeling the submodels in the JSON Schema standard, constant information such as the structure of the ID, the unit or the expected bandwidth of the measured value is predefined and can not be violated. Dynamic information, e.g. the actual measured value, is assigned during the deployment phase (Fig. 11).

To evaluate the proposed modeling, an excerpt of the modeling of a simple assembly system is presented in this section. The system is described following the PPR principle regarding its product, process steps and resources (see Fig. 12). The assembly product is an industrial dog consisting of several parts that are assembled following a precedence graph (see Fig. 11). The assembly priority graph illustrates the process flexibility of the product to be assembled. The assembly of the head and tail can be realized at the end, at the beginning as well as in the middle of the process chain. The assembly of the front and rear legs can also be flexibly thawed in their sequence. Only when assembling the rear parts, the joints have to be mounted before the legs. The assembly takes place at four manual stations, which are connected by a mobile robot. Due to the fact that the assembly stations are not rigidly linked (e.g. with roller conveyors) and every process step can be realized at every station, a flexible sequence

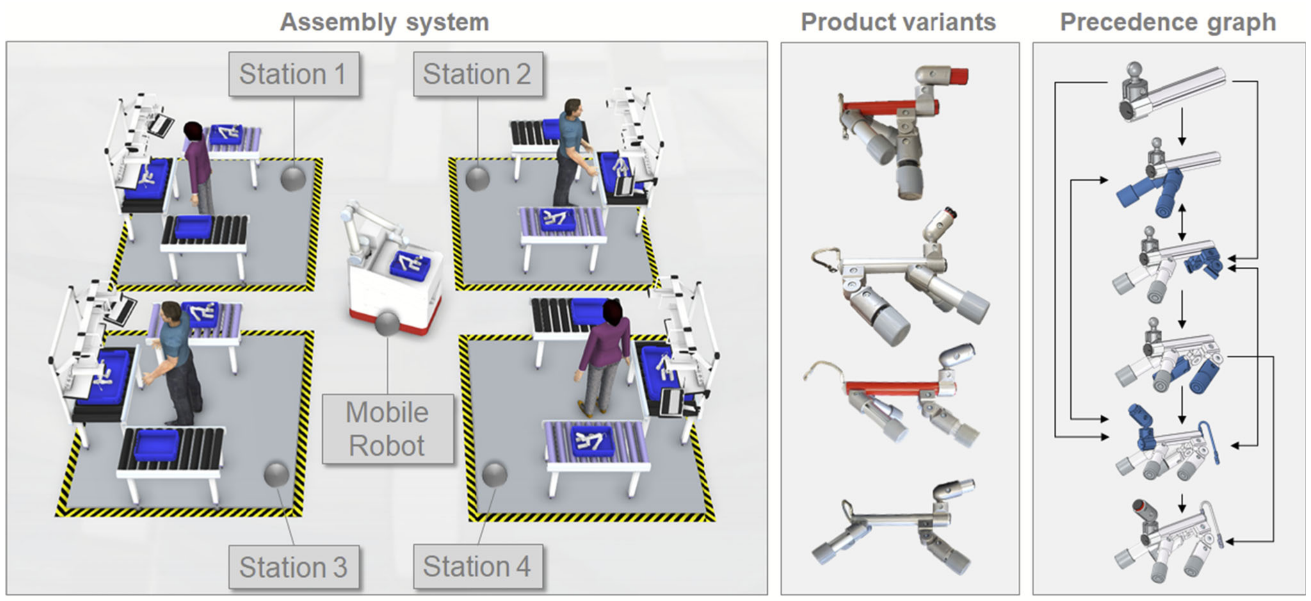
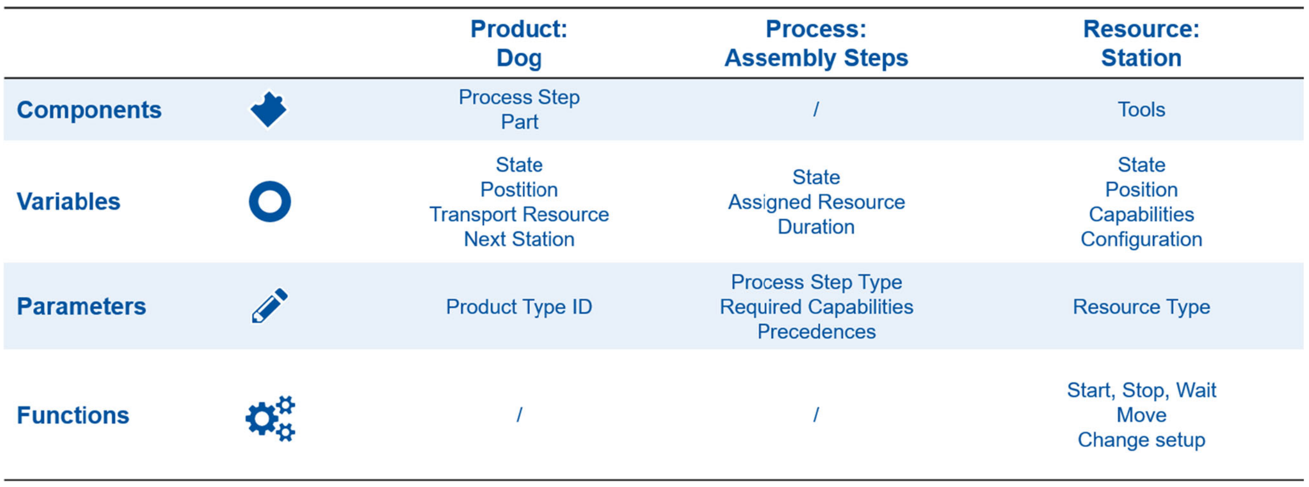


Fig. 11 Manual assembly system consisting of 4 stations interconnected with a mobile robot for assembly of four product variants of an industrial dog



	Product: Dog	Process: Assembly Steps	Resource: Station
Components	Process Step Part	/	Tools
Variables	State Postition Transport Resource Next Station	State Assigned Resource Duration	State Position Capabilities Configuration
Parameters	Product Type ID	Process Step Type Required Capabilities Precedences	Resource Type
Functions	/	/	Start, Stop, Wait Move Change setup

Fig. 12 Model components for evaluation use case following the PPR principle

of the individual process steps is feasible. The assembly system outputs four variants of the product (#1-#4) that differ in color of assembled parts and the dog's leg position.

Following the custom ontology the product variant #2 is modeled as shown in Fig. 13. The description model follows the predefined schema "producttype". Its subcomponents and parameters are modeled following a priori defined submodels "weight", "size" and "process_step_type". Within the header of the submodel "producttype" the style and id pattern is defined. The submodels likewise contain a header with a corresponding definition of the id pattern. The body of the submodel instances follows the schema structure of the submodels while including static information values (e.g. weight of producttype #2).

Automated deployment phase

The last step in the pipeline is the deployment phase. Here the previously determined models are created according to the specific requirements and distributed to the DT as well as all agents. Generative programming enables the automatic generation of specific interfaces based on the same models.

This is where the advantage of the pipeline becomes apparent: Based on the same data and the model, a digital representation is generated and stored in the DT. At the same time, the digital representation and all related data becomes accessible to all systems involved via the DT. Finally, the interfaces between the individual systems, agents and the DT are also created automatically. For example, both plan-

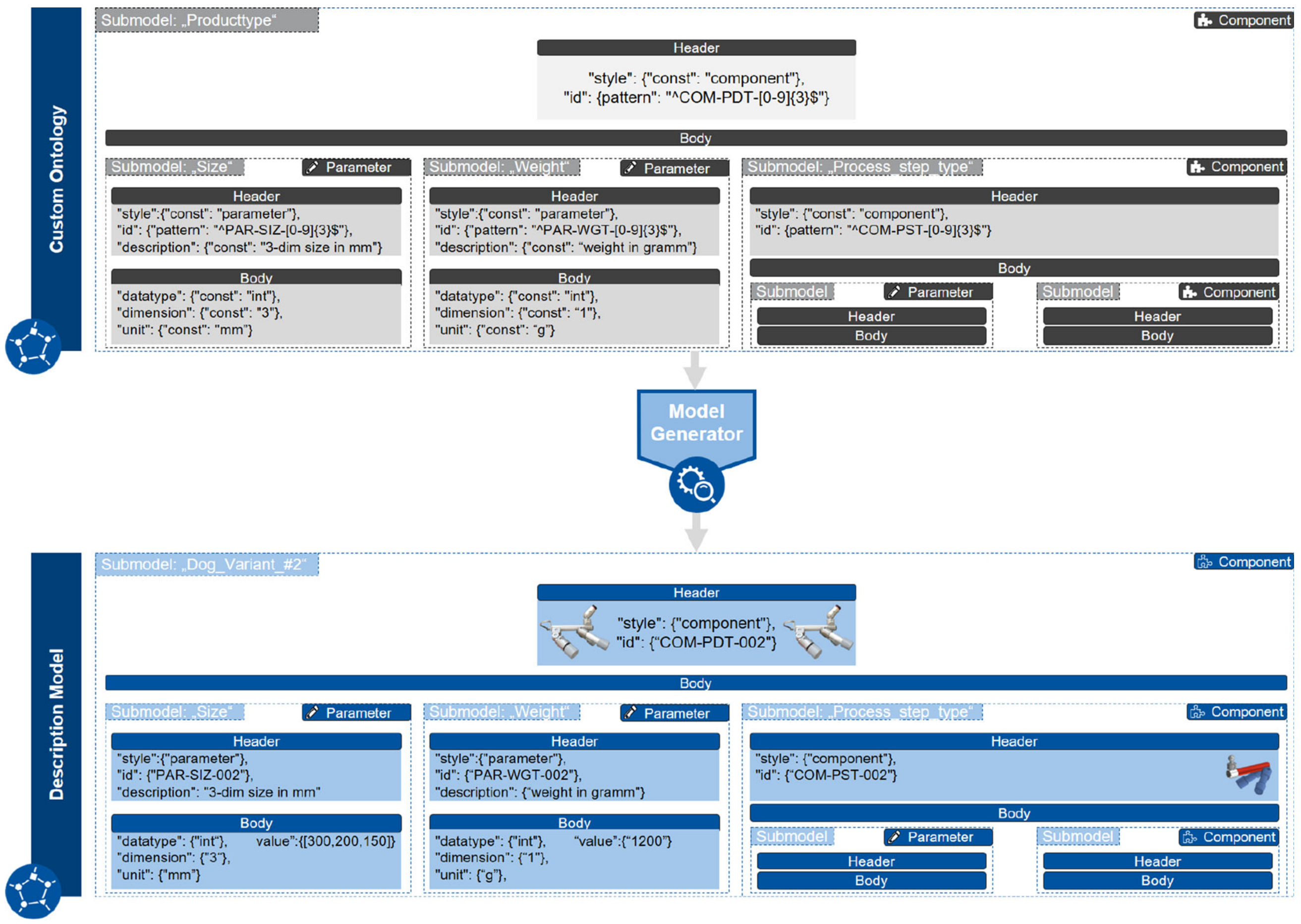


Fig. 13 Model generation for product variant description model based on custom ontology schema

ning and control can operate with the same models and the same data. The communication between the individual assets is also standardized and automatically generated.

Based on the digital twin model, a control agent is initialized for a product when it is introduced by ERP. The initialization is performed by a data processor and uses the ontology-based modeling structure. The control agent automatically implements relevant publishers, subscribers and services. Detailed knowledge of the user regarding transfer protocols is not necessary. The topic names on which relevant information is shared and received follow a defined structure. The concatenation of the Unique ID ensures the automated recognition of relevant information from other assembly participants (see Fig. 14). Using globally unique identifiers avoids wrong subscriptions. By analyzing the message header, communicating assets ensure the correct connection in a handshake. A client callback function handles possible connection errors, unsubscribing when needed.

The description model enables agent generation for different communication protocols based on a publish-subscribe pattern (e.g. MQTT or ROS). The decision for a specific protocol depends on the use case requirements regarding message size, latency, security, reliability, etc. Regarding security, client authentication with credentials and certificates on the application level as well as Transport Layer Security encryption on the transport level and Virtual Private Networks on the network level are suitable and established measures. Due to limited message sizes and no particular requirements on latency and security, MQTT was chosen to evaluate the pipeline within the example assembly system. Using MQTT wildcards, the control system subscribes to all topics following the structure of COM-Product-#/VAR-CurrentStatus-#/. Therefore, it can listen to the status of all products, e.g. product 1 in the system. If the state of a product changes because an assembly step is completed, the control system carries out the disposition and scheduling. It first assigns a new processing station to the product. The corresponding message includes the description of a station i.e. its header and is published to a topic following the structure COM-Product-#/VAR-NexStation-#/. Then, the control system generates a transport order for the mobile transport robot. This robot has subscribed to the corresponding topic and thus automatically receives information about the transport order

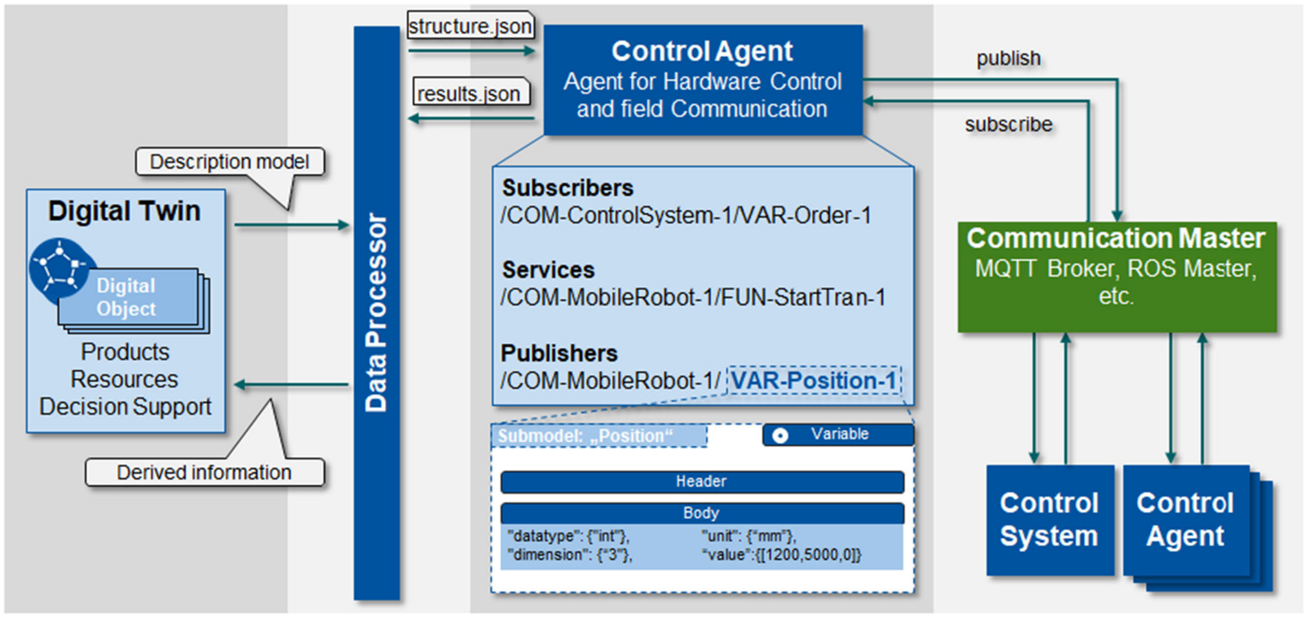


Fig. 14 Deployment of the control agent with a communication master

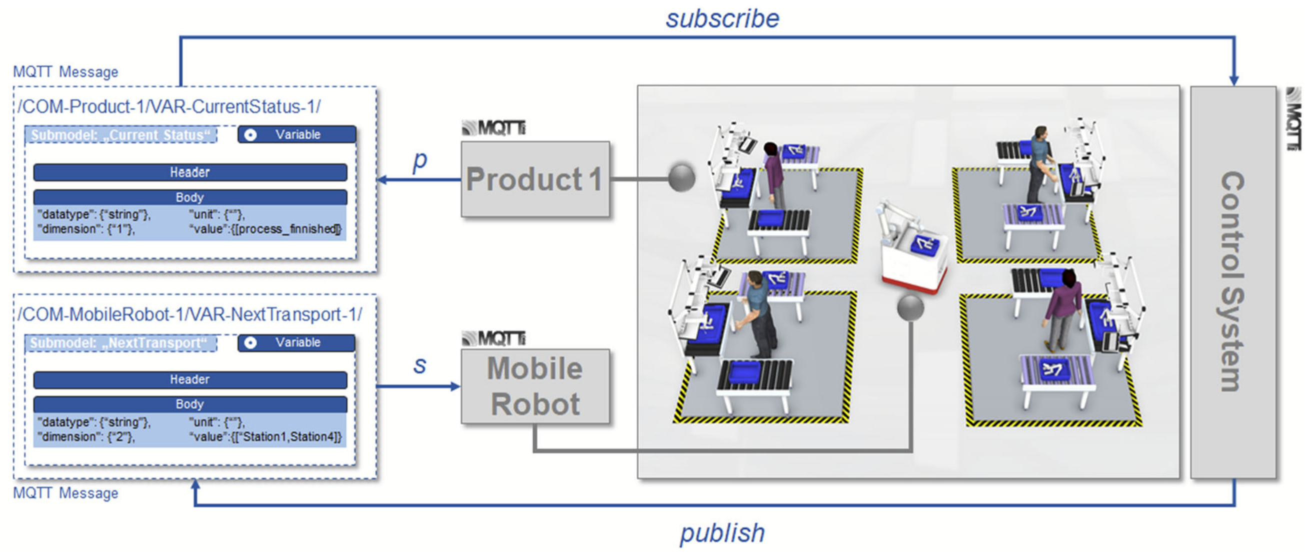

subscribe
MQTT Message
/COM-Product-1/VAR-CurrentStatus-1/
Submodel: „Current Status"
Variable
Header
Body
"datatype": {"string"}, "dimension": {"1"}, "unit": {""}, "value":{[process_finnished]}
p
MQTT
Product 1
/COM-MobileRobot-1/VAR-NextTransport-1/
Submodel: „NextTransport"
Variable
Header
Body
"datatype": {"string"}, "dimension": {"2"}, "unit": {""}, "value":{["Station1,Station4]}
s
MQTT
Mobile Robot
MQTT Message
Control System
MQTT
publish

Fig. 15 MQTT communication messages connecting products and resources with a control system

to be carried out. The message content follows the semantics and syntax of the pre-defined sub-model (see Fig. 15).

In the past, simulation models mainly were standalone solutions with a limited scope and lifetime. With the use of the digital twin as a standardized data basis, a new simulation modeling paradigm is introduced in which simulation models can be generated automatically during the whole product life cycle. The high effort for creating simulation models manually can be skipped and simulation models are used recurrently (Harrel & Hicks, 1998; Rodič, 2017).

In the deployment phase of the digital twin pipeline, scenario analyses can be used either for initial potential analyses on how the system should be configured but also as a decision support in which various alternative actions and scenarios are evaluated. For this purpose, the components of the digital twin are aggregated in a data processor to form a file that describes the entire system in its current state (see Fig. 16). The digital twin is continuously updated by the previously described control agent and therefore contains static (e.g. number of work stations) and dynamic (e.g. progress of job) data that is abstracted in the simulation. Based on user input or historical data various alternative system configurations and actions are modeled into a variety of production scenarios. For each scenario, a simulation model is automatically cre-

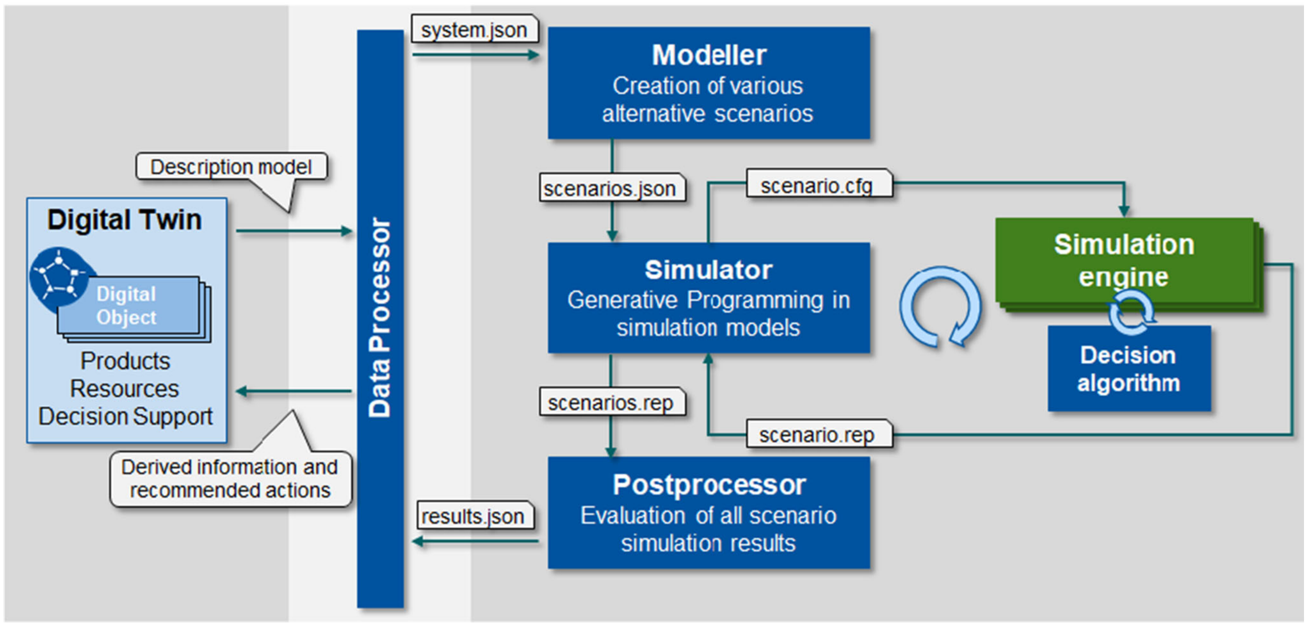


Fig. 16 Use of the digital twin for automated generation of simulation models in comprehensive scenario analyses

ated using the user input and the static and dynamic data from the digital twin, which makes it possible to start the simulation from the current time when jobs are already in the system. It is then simulated using a discrete-event-simulation engine with the use of external decision algorithms. The results of the simulation studies are summarized and evaluated in a postprocessor to enable data-based action recommendations (decision support; e.g. new machine, layout change) or automatically perform actions (e.g. rescheduling). To sum up, simulation models can be generated, simulated and evaluated automatically when using the digital twin pipeline. The consistent description model enables the standardized generation without further need for expert knowledge.

Conclusion and outlook

The required reference architectures, the ontology semantic and syntax and the communication standards and protocols were presented for developing a digital twin pipeline. The definition of the digital twin was founded on an automatic two-way data flow between digital and physical objects. The state-of-the-art analysis showed that multiple publications that incorporate multiple phases of the DT pipeline mostly contribute to the ontology-based definition and modeling of DTs. Besides a range of publications describing the application of DTs only a few focus on the actual deployment of DTs into application and even fewer treat the automation of deployment. The analysis also showed a research gap for an end-to-end workflow that describes the full pipeline from defining over modeling to deploying a DT. Such a workflow brings together expertise from researchers that focus mostly on defining and modeling DTs and expertise from practitioners that apply and deploy DTs in a manufacturing environment. Therefore, a DT pipeline with the phases ontology-based definition, standardized modeling and automated deployment is presented. A customized ontology that uses established ontologies and meta-models was presented. The subsequent modeling was shown with the example of a toy-case. Through the automated deployment of a standardized Digital Twin, a wide range of other applications can be developed and connected in a modular way. For the control of assembly systems, the creation of a connection between control agents and a communication processor was shown as an example. For planning of such systems an automated scenario analysis including simulation was presented.

An outline on a generic DT pipeline is given. The examples to clarify the steps of the pipeline are taken from the manufacturing domain and more specifically from an assembly use-case with the application of planning and control. This helps the understanding but limits the transferability to other domains or to other manufacturing use-cases. For example, for the definition phase, other ontologies might be relevant. The application is crucial for the design of the deployment automation. As planning and control were presented, the requirements of these applications were taken into account in the formal definition of the DT. For other applications, various other requirements and boundary conditions are needed. Another future work could be a thorough mapping of tools for the practical implementation of every DT pipeline phase. Examples were given with protégé for ontology definition or the model generator for the standardized modeling phase. Further efforts could also be spent on the research and development towards a tool for automated deployment using generative programming for the presented applications of simulation and control or other applications. As the ontology definition part and the modeling phase are rather well covered due to the similarity of DT creation across applications and domains, the biggest challenge lies in the development of application-specific tools due to a lack of a scaling effect.

Acknowledgements This work is part of the research project "AIM-FREE" that is funded by the German Federal Ministry for Economic Affairs and Energy (BMWi) within the indirective on a joint funding initiative to fund research and development in the field of electro-mobility (funding number: 01MV19002A) and supported by the project management agency German Aerospace Center (DLR-PT). The authors are responsible for the content.

Funding Open Access funding enabled and organized by Projekt DEAL.

Availability of data and material The following Git repository provides exemplary schemata and data models for reference and understanding: https://github.com/Project-AIMFREE/description_model

References

AboElHassan, A., Sakr, A., & Yacout, S. (2021). A framework for digital twin deployment in production systems. In P. Weißgraeber, F. Heieck, & C. Ackermann (Eds.), *Advances in Automotive Production Technology – Theory and Application* (pp. 145–152, ARENA2036). Berlin, Heidelberg: Springer. https://doi.org/10.1007/978-3-662-62962-8_17.

Acatech. (2011). Acatec position paper: Cyber-physical systems. Driving force for innovation in mobility, health, energy and production. 1. Aufl. Hg. v. Hellinger, A., and Seeger H. National Academy of Science and Engineering (2).

Andrew, A. M. (2004). Ontologies: A silver bullet for knowledge management and electronic commerce. *Kybernetes, 33*(9/10), 1544–1546. https://doi.org/10.1108/03684920410556142

Antoniou, G., & van Harmelen, F. (2004). Web ontology language: OWL. In S. Staab & R. Studer (Eds.), *Handbook on ontologies. International handbooks on information systems* (pp. 67–92). Springer.

Bader, S. R., Maleshkova, M., & Lohmann, S. (2019). Structuring reference architectures for the industrial Internet of Things. *Future Internet, 11*(7), 151.

Balaji, B., Bhattacharya, A., Fierro, G., Gao, J., Gluck, J., Hong, D., et al. (2016). Brick: Towards a Unified Metadata Schema For Buildings. In *BuildSys '16: The 3rd ACM International Conference on Systems for Energy-Efficient Built Environments* (pp. 41–50). New York, USA. https://doi.org/10.1145/2993422.2993577.

Bao, Q., Zhao, G., Yong, Y., Dai, S., & Wang, W. (2020). Ontology-based modeling of part digital twin oriented to assembly. *Proceedings of the Institution of Mechanical Engineers, Part b: Journal of Engineering Manufacture*. https://doi.org/10.1177/0954405420941160

Bodenbenner, M., Sanders, M. P., Montavon, B., & Schmitt, R. H. (2020). Domain-specific language for sensors in the internet of production. In J. P. Wulfsberg, W. Hintze, B.-A. Behrens, A. Brosius & S. Ihlenfeldt (Eds.) *Production at the leading edge of technology. Proceedings of the 10th Congress of the German Academic Association for Production Technology (WGP), Dresden, 23–24 September 2020. Lecture Notes in Production Engineering*, 1st edn 2021 (pp. 448–456). Berlin: Springer.

Boss, B., Malakuti, S., Lin, S., Usländer, T., Clauer, E., Hoffmeister, M., et al. (2020). Digital Twin and Asset Administration Shell Concepts and Application in the Industrial Internet and Industrie 4.0: An Industrial Internet Consortium and Plattform Industrie 4.0 Joint Whitepaper. Retrieved March 1, 2021 from https://www.plattform-i40.de/PI40/Redaktion/DE/Downloads/Publikation/Digital-Twin-and-Asset-Administration-Shell-Concepts.pdf?__blob=publicationFile&v=9.

Brickschema (2020). Retrieved March 1, 2021 from https://brickschema.org/ontology/1.1.

Bunte, A., Niagemann, O., & Stein, B. (2018). Integrating OWL ontologies for smart services into automation ML and OPC UA. In: *2018 IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA)*, (pp. 1383–1390). Turin:IEEE. https://doi.org/10.1109/ETFA.2018.8502593

Cao, Q., Zanni-Merk, C., & Reich, C. (2019). Ontologies for manufacturing process modeling: A survey. In D. Dao, R. J. Howlett, R. Setchi, & L. Vlacic (Eds.) *Sustainable design and manufacturing 2018, Bd. 130, smart innovation, systems and technologies* (pp. 61–70). Springer.

Cimino, C., Negri, E., & Fumagalli, L. (2019). Review of digital twin applications in manufacturing. *Computers in Industry, 113*, 103130. https://doi.org/10.1016/j.compind.2019.103130

Cutting-Decelle, A. F., Young, R. I. M., Michel, J. J., Grangel, R., Le Cardinal, J., & Bourey, J. P. (2007). ISO 15531 MANDATE: A product-process-resource based approach for managing modularity in production management. *Concurrent Engineering, 15*(2), 217–235. https://doi.org/10.1177/1063293X07079329

DIN SPEC 91345. (2016). *Referenzarchitekturmodell Industrie 4.0 (RAMI4.0) (ICS 03.100.01; 25.040.01; 35.240.50)*. Berlin: Beuth Verlag GmbH

Erkoyuncu, J. A., del Amo, I. F., Ariansyah, D., Bulka, D., Vrabič, R., & Roy, R. (2020). A design framework for adaptive digital twins. *CIRP Annals, 69*(1), 145–148. https://doi.org/10.1016/j.cirp.2020.04.086

Greschke, P., Schönemann, M., Thiede, S., & Herrmann, C. (2014). Matrix structures for high volumes and flexibility in production systems. *Procedia CIRP*, *17*(1), 160–165.

Gruber, T. R. (1995). Toward principles for the design of ontologies used for knowledge sharing? *International Journal of Human-Computer Studies, 43*(5–6), 907–928. https://doi.org/10.1006/ijhc.1995.1081

Harrel, C. R., & Hicks, D. A. (1998). Simulation software component architecture for simulation-based enterprise applications. In *Proceedings of the 1998 winter simulation conference* (pp. 1717–1721).

Hepp, M. (2005). eClassOWL: A fully-fledged products and services ontology in OWL. In: *Poster Proceedings of ISWC2005. Galway*.

Hildebrandt, C., Kocher, A., Kustner, C., Lopez-Enriquez, C.-M., Muller, A. W., Caesar, B., Gundlach, C. S., & Fay, A. (2020). Ontology building for cyber-physical systems: Application in the manufacturing domain. *IEEE Transactions on Automation Science and Engineering, 17*(3), 1266–1282. https://doi.org/10.1109/TASE.2020.2991777

Hildebrandt, C., Törsleff, S., Caesar, B., & Fay, A. (2018). Ontology building for cyber-physical systems: A domain expert-centric approach. In *2018 IEEE 14th international conference on automation science and engineering (CASE)* (pp. 1079–1086).

Hoffmann, M. (2017). Adaptive und skalierbare Informationsmodellierung zur Ermöglichung autonomer Entscheidungsfindungsprozesse für interoperable, reaktive Fertigungenautonomer Entscheidungsfindungsprozesse. Dissertation. RWTH Aachen, Aachen.

Hu, S. J. (2013). Evolving paradigms of manufacturing: From mass production to mass customization and personalization. *Procedia CIRP*, *7*, 3–8.

Hüttemann, G., Göppert, A., Lettmann, P., & Schmitt, R. H. (2017). Dynamically interconnected assembly systems. In R. Schmitt & G. Schuh (Eds.), *WGP-Jahreskongress Aachen* (Vol. 7, pp. 261–268). Aachen: Apprimus Wissenschaftsverlag.

Iarovyi, S., Mohammed, W. M., Lobov, A., Ferrer, B. R., & Lastra, J. L. M. (2016). Cyber-physical systems for open-knowledge-driven manufacturing execution systems. *Proceedings of IEEE, 104*(5), 1142–1154. https://doi.org/10.1109/JPROC.2015.2509498

Industrial Value Chain Initiative. (2018). Strategic implementation framework of industrial value chain for connected industries. Hg. v. Industrial Value Chain Initiative. Tokyo, Japan

Industrial Internet Consortium. (2019).The industrial internet of things: volume G1: Reference architecture.

Järvenpää, E., Siltala, N., Hylli, O., & Lanz, M. (2019). The development of an ontology for describing the capabilities of manufacturing resources. *Journal of Intelligent Manufacturing, 30*(2), 959–978. https://doi.org/10.1007/s10845-018-1427-6

Jones, D., Snider, C., Nassehi, A., Yon, J., & Hicks, B. (2020). Characterising the digital twin: A systematic literature review. *CIRP Journal of Manufacturing Science and Technology, 29*, 36–52. https://doi.org/10.1016/j.cirpj.2020.02.002

Joshi, R., Didier, P., Jimenez, J., & Carey, T. (2018). The industrial internet of things volume 5G: Connectivity framework

Katti, B., Plociennik, C., & Schweitzer, M. (2018). SemOPC-UA: Introducing semantics to OPC-UA application specific methods. *IFAC-PapersOnLine*, *51*, 1230–1236. https://doi.org/10.1016/j.ifacol.2018.08.422.

Kousi, N., Gkournelos, C., Aivaliotis, S., Giannoulis, C., Michalos, G., & Makris, S. (2019). Digital twin for adaptation of robots' behavior in flexible robotic assembly lines. *Procedia Manufacturing, 28*, 121–126. https://doi.org/10.1016/j.promfg.2018.12.020

Kritzinger, W., Karner, M., Traar, G., Henjes, J., & Sihn, W. (2018). Digital twin in manufacturing: A categorical literature review and classification. *IFAC-PapersOnLine, 51*(11), 1016–1022. https://doi.org/10.1016/j.ifacol.2018.08.474

Lee, Y.-T. T., & Riddick, F. H. (2010). Core manufacturing simulation data: UML Model Standard. Retrieved March 1, 2021 from https://www.nist.gov/publications/core-manufacturingsimulation-data-uml-model-standard?pub_id=907352.

Lemaignan, S., Siadat, A., Dantan, J.-Y., & Semenenko, A. (2006): MASON: A proposal for an ontology of manufacturing domain. In *IEEE workshop on distributed intelligent systems: Collective intelligence and its applications (DIS'06). Prague, Czech Republic, 15–16 June 2006* (pp. 195–200). IEEE.

Linked Building Data Community Group (LBD). (2020). Building Topology Ontology (BOT). Retrieved March 1, 2021 from https://w3clbd-cg.github.io/bot/..

Liu, C., Jiang, P., & Jiang, W. (2020). Web-based digital twin modeling and remote control of cyber-physical production systems. *Robotics and Computer-Integrated Manufacturing, 64*, 101956. https://doi.org/10.1016/j.rcim.2020.101956

Madni, A., Madni, C., & Lucero, S. (2019). Leveraging digital twin technology in model-based systems engineering. *Systems, 7*(1), 7. https://doi.org/10.3390/systems7010007

Martin, P., & D'Acunto, A. (2003). Design of a production system: An application of integration product-process. *International Journal of Computer Integrated Manufacturing, 16*(7–8), 509–516. https://doi.org/10.1080/0951192031000115831

McGuinness, D. L., van Harmelen, F., et al. (2004). OWL web ontology language overview. *W3C recommendation*, *10*, 2004.

Moghaddam, M., Cadavid, M. N., Kenley, C. R., & Deshmukh, A. V. (2018). Reference architectures for smart manufacturing: A critical review. *Journal of Manufacturing Systems, 49*, 215–225.

Monostori, L. (2014). Cyber-physical production systems: Roots, expectations and R&D challenges. *Procedia CIRP, 17*, 9–13. https://doi.org/10.1016/j.procir.2014.03.115

Noy, N. F., Fergerson, R. W., & Musen, M. A. (2000). The knowledge model of Protege-2000: Combining interoperability and flexibility. In R. Dieng & O. Corby (Eds.), *International Conference on Knowledge Engineering and Knowledge Management* (pp. 17–32). Springer.

OASIS MQTT Technical Committee. (2019). MQTT. https://mqtt.org/mqtt-specification/. Accessed 22 Dec 2020

Panetto, H., Dassisti, M., & Tursi, A. (2012). ONTO-PDM: Product-driven ONTOlogy for product data management interoperability within manufacturing process environment. *Advanced Engineering Informatics, 26*(2), 334–348. https://doi.org/10.1016/J.AEI.2011.12.002

Park, K. T., Yang, J., & Noh, S. D. (2020). VREDI: Virtual representation for a digital twin application in a work-center-level asset administration shell. *Journal of Intelligent Manufacturing, 32*, 1–44

Pezoa, F., Reutter, J., Suarez, F., Ugarte, & Mart. (2016). Foundations of JSON schema. In *Proceedings of the 25th International Conference on World Wide Web*, (pp. 263–273)

Piller, T. C., & Khelil, A. (2020). SemSub: Semantic subscriptions for the MQTT protocol. In *IEEE 6th World Forum on Internet of Things (WF-IoT)*, (pp. 1–6). New Orleans, LA, USA: IEEE. https://doi.org/10.1109/WF-IoT48130.2020.9221477

Protegé. (2020). The board of trustees of the Leland Stanford Junior University. Version v4.0.2. Online verfügbar unter https://webprotege.stanford.edu/, zuletzt geprüft am 06.12.2020

Qamar, A., Hall, M. A., & Collinson, S. (2018). Lean versus agile production: flexibility trade-offs within the automotive supply chain. *International Journal of Production Research*, *56*(11), 3974–3993.

Rasmussen, M. H., Lefrançois, M., Schneider, G. F., & Pauwel, P. (2019). BOT: The building topology ontology of theW3C linked building data group. *Semantic Web, 12*, 143–161.

Rodič, B. (2017). Industry 4.0 and the new simulation modelling paradigm. *Organizacija, 50*(3), 193–207.

Rosen, R., Wichert, G., Lo, G., & Bettenhausen, K. D. (2015). About the importance of autonomy and digital twins for the future of manufacturing. *IFAC-PapersOnLine, 48*(3), 567–572. https://doi.org/10.1016/j.ifacol.2015.06.141

Schuh, G. (2014). *Produktkomplexität managen. Strategien; Methoden; Tools*. Carl Hanser Fachbuchverlag.

Shelby, Z., Hartke, K., Bormann, C., & Frank, B. (2014). Constrained application protocol (CoAP). Retrieved December 22, 2020 from https://tools.ietf.org/html/rfc7252

Sino-German Industrie 4.0. (2018). *Alignment report for reference architectural model for industrie 4.0/intelligent manufacturing system architecture*. Federal Ministry of Economic Affairs and Energy.

Standardization Council Industrie 4.0. (2020). DIN/DKE German Standardization Roadmap Industrie 4.0. Berlin, Germany.

Stark, R., & Damerau, T. (2019). Digital twin. In S. Chatti, & T. Tolio (Eds.) *CIRP Encyclopedia of production engineering* (pp. 1–8). Springer.

Steinmetz, C., Rettberg, A., Ribeiro, F. G. C., Schroeder, G., & Pereira, C. E. (2018). Internet of Things ontology for digital twin in cyber physical systems. In *2018 VIII Brazilian symposium on computing systems engineering (SBESC). VIII Brazilian symposium on computing systems engineering (SBESC). Salvador, Brazil*, 05.11.2018–08.11.2018 (pp. 154–159). IEEE.

Steinmetz, C., Schroeder, G., dos Santos Roque, A., Pereira, C. E., Wagner, C., Saalmann, P., & Hellingrath, B. (2017). Ontology-driven IoT code generation for FIWARE. In *IEEE 15th international conference* (pp. 38–43).

Tantik, E., & Anderl, R. (2017). Integrated data model and structure for the asset administration shell in industrie 4.0. *Procedia CIRP, 60*, 86–91. https://doi.org/10.1016/j.procir.2017.01.048

Tao, F., Cheng, J., Qi, Q., Zhang, M., Zhang, H., & Sui, F. (2018). Digital twin-driven product design, manufacturing and service with big data. *The International Journal of Advanced Manufacturing Technology, 94*(9–12), 3563–3576. https://doi.org/10.1007/s00170-017-0233-1

Truong, A., Walters, A., Goodsitt, J., Hines, K., Bruss, C. B., & Farivar, R. (2019). Towards automated machine learning: Evaluation and comparison of automl approaches and tools. In *IEEE 31st International Conference* (pp. 1471–1479).

Uhlemann, T.-J., Lehmann, C., & Steinhilper, R. (2017). The digital twin: Realizing the cyber-physical production system for industry 4.0. *Procedia CIRP, 61*, 335–340. https://doi.org/10.1016/j.procir.2016.11.152

Usman, Z., Young, R. I. M., Chungoora, N., Palmer, C., Case, K., & Harding, J. (2011). A manufacturing core concepts ontology for product lifecycle interoperability. In W. van der Aalst, J. Mylopoulos, N. M. Sadeh, M. J. Shaw, C. Szyperski, M. van Sinderen, & P. Johnson (Eds.) *Enterprise Interoperability, Bd. 76. Lecture Notes in Business Information Processing* (pp. 5–18). Springer.

VDI. (2016). Digital Factory - Data management and system architectures (4499 Blatt 3).

VDI. (2018). Agentensysteme in der Automatisierungstechnik - Anwendung (2653 Blatt 3).

Wan, J., Yin, B., Li, D., Celesti, A., Tao, F., & Hua, Q. (2018). An ontology-based resource reconfiguration method for manufacturing cyber-physical systems. *IEEE/ASME Transactions on Mechatronics, 23*(6), 2537–2546. https://doi.org/10.1109/TMECH.2018.2814784

Weser, M., Bock, J., Schmitt, S., Perzylo, A., & Evers, K. (2020). An ontology-based metamodel for capability descriptions. In *Proceedings of the IEEE international conference on emerging technologies and factory automation (ETFA), Vienna, Austria.*

Weyer, S., Meyer, T., Ohmer, M., Gorecky, D., & Zühlke, D. (2016). Future modeling and simulation of CPS-based factories: An example from the automotive industry. *IFAC-PapersOnLine, 49*(31), 97–102. https://doi.org/10.1016/j.ifacol.2016.12.168

Weyrich, M., & Ebert, C. (2016). Reference architectures for the Internet of Things. *IEEE Software, 33*(1), 112–116.

Wiendahl, H.-P., Gerst, D., & Keunecke, L. (2004). *Variantenbeherrschung in der Montage*. Heidelberg: Springer, Berlin Heidelberg.

Wohlfeld, D., Weiss, V., & Becker, B. (2017) Digital shadow: From production to product. In M. Bargende, H.-C. Reuss, & J. Wiedemann (Eds.) *17. Internationales Stuttgarter Symposium. Proceedings* (pp. 783–794). Springer Fachmedien Wiesbaden.

Wright, L., & Davidson, S. (2020). How to tell the difference between a model and a digital twin. *Advanced Modeling and Simulation in Engineering Sciences, 7*(1), 1–13. https://doi.org/10.1186/s40323-020-00147-4

Yakovlev, A., Moghadam, H. F., Moharrer, A., Cai, J., Chavoshi, N., Varadarajan, V., et al. (2020). Oracle AutoML. *Proceedings of the VLDB Endowment*, *13*, 3166–3180. https://doi.org/10.14778/3415478.3415542.