












Engineering Digital Twins and Digital Shadows as Key Enablers for Industry 4.0

Stefan Braun , Manuela Dalibor , Nico Jansen , Matthias Jarke ,
István Koren , Christoph Quix , Bernhard Rumpe , Manuel Wimmer 
and Andreas Wortmann 

Abstract

Industry 4.0 opens up new potentials for the automation and improvement of production processes, but the associated digitization also increases the complexity of this development. Monitoring and maintenance activities in production processes still require high manual effort and are only partially automated due to immature data aggregation and analysis, resulting in expensive downtimes, inefficient use of machines, and too much production of waste. To maintain control over the growing complexity and to provide insight into the production, concepts such as Digital Twins, Digital Shadows, and model-based systems engineering for Industry 4.0 emerge. Digital Shadows consist of data

S. Braun · M. Jarke

Information Systems and Databases, RWTH Aachen University, Aachen, Germany
e-mail: braun@dbis.rwth-aachen.de

M. Jarke

e-mail: jarke@dbis.rwth-aachen.de

M. Dalibor · N. Jansen · B. Rumpe

Software Engineering, RWTH Aachen University, Aachen, Germany
e-mail: dalibor@se-rwth.de

N. Jansen

e-mail: jansen@se-rwth.de

B. Rumpe

e-mail: rumpe@se-rwth.de

I. Koren

Process and Data Science, RWTH Aachen University, Aachen, Germany
e-mail: koren@pads.rwth-aachen.de

traces of an observed Cyber-Physical Production System. Digital Twins operate on Digital Shadows to enable novel analysis, monitoring, and optimization. We present a general overview of the concepts of Digital Twins, Digital Shadows, their usage and realization in Data Lakes, their development based on engineering models, and corresponding engineering challenges. This provides a foundation for implementing Digital Twins, which constitute a main driver for future innovations in Industry 4.0 digitization.

Keywords

Digital Twin • Digital Shadow • Industry 4.0 • Model-based Systems Engineering • Data Lake

1 Introduction

The fourth industrial revolution, Industry 4.0, is a fundamental driver for agile manufacturing through integration and communication of production systems. It aims at integrating Cyber-Physical Production System (CPPS) to optimize the complete value chain [74]. Initially announced by the German Federal Ministry for Education and Research in the year 2011 [6], Industry 4.0 has become an international phenomenon as the next big step towards future development and manufacturing [7, 32, 48, 52]. To this end, it leverages state-of-the-art research results from a variety of fields, including the Internet of Things (IoT), big data, and machine learning. Combining these approaches, Digital Twins are envisioned as digital duplicates of CPPS that represent, control, and monitor their physical counterpart to make better use of resources. In this vision, Digital Twins need to rely on detailed knowledge of the system, including its requirements and operation data. Since an exact digital replication of all parameters down to the atomic level is not feasible, the concept of Digital Shadows was introduced, which promotes purpose-driven compilations of production data. Moreover, while simulations approximate the behavior and effects, the calculated results often diverge from reality due to external influences such as wear, tear, pollution, or environmental impacts. Remedial actions require extensive manual effort by experienced operators performing measures on real-world counterparts. In the following, we introduce the main

C. Quix

Information Systems and Data Science, Hochschule Niederrhein, Krefeld, Germany
e-mail: christoph.quix@hs-niederrhein.de

M. Wimmer

Business Informatics – SW Engineering, JKU Linz, Linz, Austria
e-mail: manuel.wimmer@jku.at

A. Wortmann (✉)

Institute for Control Engineering of Machine Tools and Manufacturing Units, University of Stuttgart, Stuttgart, Germany
e-mail: andreas.wortmann@isw.uni-stuttgart.de

themes of this chapter, including the aforementioned engineering models, Digital Shadows and Digital Twins.

1.1 Engineering Models in Industry 4.0

Modern product development processes employ methods of model-based systems engineering (MBSE) [28, 61] and model-driven development (MDD), in which models become the primary development artifacts [15]. MBSE raises abstraction in traditional systems engineering approach by harnessing structured models over unstructured documents. MDD extends the classic model-based approach even further by establishing models as primary drivers within the development process. Adhering to explicit modeling languages, with well-defined semantics (meaning) [29], these promote understanding and transparency in development, fostering intra- and interdisciplinary communication [41], as well as automated analysis and synthesis [31] of (parts of) the system [57] under development.

Managing complexity in the interdisciplinary engineering process requires domain-specific views on the overall system, filtering essential information. Thus, by leveraging view-based modeling [18, 55], experts of participating domains are provided with the information relevant to their specific concerns in suitable modeling languages (the views), which is anchored in the overall system's context. Systems Modeling Language (SysML) [23] is a prominent collection of modeling languages, to describe the relationships between the concerns of a system and thus provides the glue between participating engineering domains and their domain-specific models.

Engineering models [8] are typically used constructively, i.e., to prescribe a system under development, and contribute directly to the CPPS development process. As CPPS development is a highly interdisciplinary effort, different models exist across different domains. These are also relevant in engineering Digital Twins since they contain essential information about the CPPS. Thus, engineering models do not only contribute to system development, but also to the development of its twin by integrating important information as well as runtime simulations.

1.2 Digital Shadows

Modern CPPSs are typically equipped with sensors that capture tremendous amounts of raw data while these CPPSs are running. These large amounts of data can no longer be transmitted in real-time or sensibly processed to gain insights into the system's state. Thus, a software system running in this context must provide mechanisms to reduce the sheer data volume and its level of detail, while also coping with obsolete and incomplete data. Data must be provided in a reduced and purpose-oriented fashion to achieve better performance and more context adaptation. To address this, we conceived a notion of Digital Shadows that

provide compact views on dynamic processes, usually combining condensed measurement data with highly efficient simplified mathematical models [42]. Therefore, we define digital twins as follows [3, 13]:

Definition 1 (*Digital Shadow*) *A Digital Shadow is a set of temporal data traces and/or their aggregation and abstraction collected concerning a system for a specific purpose with respect to the original system.*

Thus, the Digital Shadow is a set of data observed from the CPPS. This data is usually captured using sensors of various forms but also data of the state of computation, actions executed by control devices, or even input from human operators. A Digital Shadow contains purpose-oriented data for a specific point in time, provided in a transformed (e.g., reduced or augmented) form. Additionally, this data can be enriched with quality information, such as origin, fidelity, accuracy, and more. Consequently, this implies the existence of multiple Digital Shadow at different times and for a variety of objectives that may reference another. The complete history of Digital Shadows is available to a Digital Shadow, enabling temporal analyses such as predictive maintenance based on variations in the collected data.

1.3 Digital Twins

While there is some consensus that a Digital Twin is a sort of digital duplicate of a physical entity [67], there is still no generally accepted definition that specifies what this means and entails. There are numerous differing interpretations of what a Digital Twin actually is and should be capable of. Many realizations of this concept heavily depend on their specific domain or application purpose. Digital Twin applications range from simple data acquisitions across virtual models of the physical system to an integrated twin with optimization capabilities. Thus, they can serve observation and monitoring purposes purely or directly support management and controlling to support the development process of CPPSs actively. Engineering in Industry 4.0 is highly interdisciplinary, resulting in modern trends in MBSE striving for integrated models to bridge the gap between different domains. Therefore, we use a definition of the Digital Twin based on models that was developed within the German Cluster of Excellence “Internet of Production”,¹ which comprises 200 researchers in 25 departments from different domains, including mechanical engineering, electrical engineering, automation, factory construction, software engineering, systems engineering:

Definition 2 (*Digital Twin*) *A Digital Twin of a system consists of a set of models of the system, a set of Digital Shadows, and a set of services that allow using the data and models purposefully with respect to the original system.*

¹ Internet of Production: <https://iop.rwth-aachen.de>.

Hence, a Digital Twin can be engineered based on heterogeneous sets of engineering models of different domains. It relies on the Digital Shadows as highly optimized representations of cohesive, purpose-driven, data of the CPPS to conduct analyses for monitoring, decision making, prediction, and performing operations with the system. Additionally, the Digital Twin of a system provides services for handling user input, controlling its real-world counterpart, and interacting with other systems (e.g., product life-cycle management systems or other Digital Twins). We assume that each CPPS will, in the future, have its own Digital Twin and both closely interact for monitoring but potentially also for controlling purposes.

To conclude, Fig. 1 illustrates the triangular relationship between the production site, Digital Shadows, and the Digital Twin. Starting on the left-hand side of the figure, application-specific viewpoints illuminate certain aspects of the production area to collect specific data from the machines and processes. These are then collected figuratively by the Digital Shadows and flow into the Data Lake. From there, the data diffuses into the Digital Twin. Engineering models contribute to the digital replicate, releasing both analysis results back to the Digital Shadows, as well as configuration and control parameters into the production site.

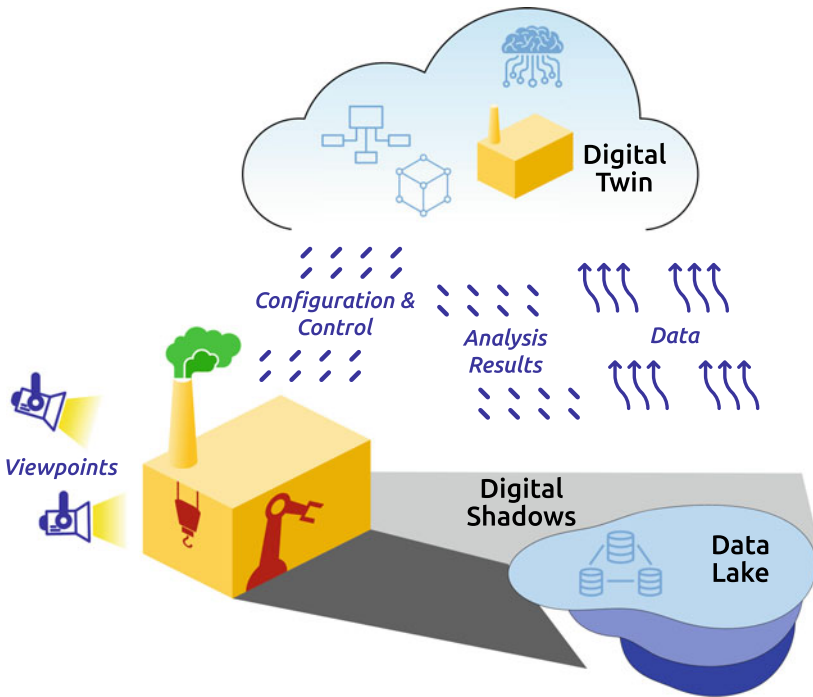


Fig. 1 Continuous Data Cycle in Industry 4.0

1.4 Outline

The remainder is structured as follows: Sect. 2 discusses challenges in engineering Digital Twins with respect to their construction, operation, and services. Section 3 introduces a conceptual structure of Digital Twins and Digital Shadows based on an underlying Data Lake infrastructure, while Sect. 4 introduces specific engineering approaches based on state-of-the-art technologies. Sect. 5 elaborates on data processing techniques (e.g., artificial intelligence) for Digital Shadows. Sect. 6 concludes with an outlook on the future of Digital Twins and Digital Shadows and their potential in Industry 4.0.

2 Challenges in Engineering a Digital Twin and Its Digital Shadows

This section discusses challenges in the model-driven engineering and operation of Digital Twins.

2.1 Challenges in Engineering Digital Twins

Integrated engineering of a digital twin and its represented system Engineering Digital Twins either coincides with the engineering of the represented system, i.e., both systems are conceived and developed together (“greenfield” engineering), or takes place after the represented system has been deployed (“brownfield” engineering) [26]. In both cases, the purpose of the Digital Twin and its information management requirements must be made explicit. However, during the greenfield engineering of Digital Twins, development of the represented system can consider these requirements, whereas in the brownfield case, the required information might not be available, demand augmentation of the represented system, or synthesis from other sources. Moreover, most Digital Twin operations demand information with specific quality requirements, such as frequency, precision, reliability and augmented with additional contextual information (e.g., time and source of origin). Where greenfield approaches can consider this during development of the represented system, brownfield approaches often cannot. This can lead to less precise, out-of-date, and unreliable Digital Twins. Consequently, brownfield approaches can severely limit the capabilities of potential Digital Twins and, to achieve effective Digital Twins, they should be considered an integral part within the development of the represented system.

The systematic identification of Digital Twin purposes in the context of available, producible, and synthesizable information is a complicated task. Consequently, its

systematic integration into the development of the represented system is a major challenge in engineering useful Digital Twins.

Data quality in context of sensor data and data streams has been addressed in various applications [20, 38]. In the context of Digital Twins, this challenge has been recognized, but not yet explicitly addressed [72]. The parallel development of a Digital Twin and its physical twin is discussed in [51], where different levels of a Digital Twin are presented.

Systematic specification and implementation of digital shadows Another quintessential challenge in engineering Digital Twins is enabling these to leverage the data retrieved from the represented system and making sense out of it. In order to retrieve data, both the aspect of data streams originating from different sources as well as heterogeneous formats of data have to be addressed. In (domain-specific) real-time and ultimately hamper Digital Twins to fulfill their purposes: On the one hand, the three Vs of big data—Volume, Velocity, and Variety—pose a challenge for handling the data. At the same time, all possibly relevant data shall be captured as one wants to keep all data for later possible usage. To mitigate this, data, information, and models of the represented system and related systems can be aggregated and abstracted prior to processing, resulting in Digital Shadows [36]. To be more exact, it might be even not possible to collect all data initially, as the volume is too large, the systems are not capable of analyzing the data at the required speed, or there is no sensor to capture specific information. As illustrated in Fig. 1, Digital Shadows are produced by different viewpoints on the represented object, i.e., only data relevant for a specific purpose or application is considered. Still, it is not possible to foresee all potential use cases in Digital Twins when designing the CPPS or the corresponding data management systems. Therefore, we propose to use a Data Lake [60] as a repository for raw data that supports functions to prepare data as required by the Digital Twins, integrates models and other sources of information, and provides the right amount, granularity, and precision of data at the right time.

For the definition of Digital Shadows, suitable data modeling and integration techniques are needed. These must enable the implementation of Digital Shadows consisting of required data integrated from multiple sources, including data transformation functions to clean, harmonize, and restructure the data into the desired format.

Some case studies that apply the concept of a Digital Shadow have been described in [42]. Data Lakes have been proposed also in [70] as a solution to the data challenges that are present in the context of Digital Twins, e.g., heterogeneity, volume.

Integration of domain expert solutions Engineering Digital Twins is an interdisciplinary effort that, depending on their purpose, can demand the collaboration of experts from automa-

tion engineering, human-machine interaction, software engineering, and more. These experts employ a variety of different solution paradigms, methods, and technologies that need to be properly integrated to produce and deploy Digital Twins. To this end, often solutions following geometric and functional paradigms, continuous and discrete perspectives, heavily front-loaded and agile methods, which employ a large variety of different implementation techniques (CAD, knowledge representation, physical modeling, state-based modeling, programming, etc.) must be integrated. Due to the required technical complexity and level of detail of the different experts' solution implementations, their integration and joint use in Digital Twins introduces another level of complexity.

MDD is a solution paradigm that aims to facilitate the integration of these different domains by overcompensating the growth in complexity of their solutions. To this end, MDD lifts abstract models (in contrast to technically detailed implementations) to primary development artifacts that can leverage terminology and concepts of the participating experts' domains. Through automated analyses and syntheses that reify domain expertise, these models can greatly facilitate the integration of solutions and their joint use in Digital Twins. Consequently, the efficient inter-disciplinary engineering of Digital Twins demands suitable modeling techniques and tools for the different domains and means to their integration. While appropriate domain-specific modeling techniques and tools have been brought forth, such as Simulink [11] or Modelica [16] for physical modeling, various CAD variants [21, 22, 43] for geometric modeling, and different software modeling techniques [14, 43, 54], means for their integration are rare and either do not consider how the different domain-specific solutions shall be integrated, nor consider the models' semantics (meaning) [15], but consider their syntactic structure [9] only.

The systematic, syntactic, and semantic integration of the Digital Twin parts contributed by experts from participating domains demands suitable and precise modeling techniques for these experts that can be integrated easily and semantically meaningfully.

Research in software language engineering [31, 39] investigates the conception, engineering, and evolution of suitable modeling techniques; on the basis of SysML [14, 75], their integration can be achieved.

Composable digital twins Systems of systems are ubiquitous in manufacturing. Through well-defined interfaces, standards, or handcrafted integration, these collaborative system groups achieve goals unachievable alone. While there are various standards on the integration of systems through joint interfaces or shared data structures in manufacturing [74], research in Digital Twins rarely considers their collaboration, integration, or composition. With Digital Twins often being data-intensive applications that conduct complex analyses

relying on (domain-specific) real-time data acquisition, operating many Digital Twins on the same data sources (e.g., Data Lakes) can lead to bottlenecks rendering the data acquisition in real-time impossible. Collaborative Digital Twin groups or composed Digital Twins that share data, reduce redundant data acquisition, could mitigate this. For achieving this, reuseability and interoperability of Digital Shadows is desirable. (8) Moreover, research on composable Digital Twins can facilitate their engineering through reusing off-the-shelf Digital Twins provided by third-party vendors.

The systematic, syntactic, and semantic integration of the Digital Twin parts contributed by experts from participating domains demands suitable and precise modeling techniques for these experts that can be integrated easily and semantically meaningfully.

Leveraging the time-honored concepts of component-based software engineering [3], such as encapsulation of functionality behind stable interfaces or substitutability of implementations, and applying these through modern architecture description languages [50] can greatly facilitate engineering Digital Twins through composition.

Measurable digital twin and digital shadow fidelity Digital Twins intrinsically serve to represent the system they observe. As such, the usefulness of a Digital Twin depends on the fidelity, i.e., the precision of this representation, with respect to its purpose. For instance, if a Digital Twin controlling an injection molding machine [3] is subject to a deviation of several millimeters, the resulting products might be rendered unusable, whereas for a Digital Twin monitoring an automated vehicle [40], this relatively small discrepancy might be tolerable. Consequently, the quality of representation and its possible degradation leading to divergence between the represented system and the Digital Twin must be considered when engineering Digital Twins. As the possible quality of representation directly depends on the quality of data in the Digital Shadow, the data quality requirements of Digital Twins need to be made explicit, such they can be considered during design, implementation, and runtime of the Digital Shadows. Moreover, Digital Twins must anticipate degradation of that quality and provide metrics to gauge that quality during runtime. Similarly, as Digital Twins are meant to strategically contribute added-value to manufacturing, these expectations should be made explicit and controlled at runtime, e.g., through simulation-based benchmarks applied to the Digital Twin. Data quality requirements, degradation metrics, and benchmarks need to be formulated relative to the heterogeneous models the Digital Twins consist of and, thus, demand suitable modeling techniques as well.

A Digital Twin and its Digital Shadows are subject to representation quality that depends on the quality of available data and aim to produce added-value through optimization. Measuring all of these must be anticipated at design-time measured to prevent diverging of the represented system, its Digital Twin, and its strategic goals.

As mentioned before, data quality issues have not been explicitly addressed for Digital Twins, but more general frameworks have been developed for representing data quality requirements for sensor data and data streams. For example, [20] proposes an ontology-based approach to model data quality requirements, dimensions, metrics, and measurements.

2.2 Challenges in Operating Digital Twins

Mind the gap between design-time and runtime Digital Twins can be used at system design-time as well as during their operation and even after their decommissioning. At design-time, they can support exploration of the design space through the rapid creation of system variants for dimensioning, testing, and simulation. To this end, the design-time Digital Twins need to operate on data and models from a system yet to be created. Both, data and models, can be provided from observations of similar or predecessor systems or synthesized from simulation of similar systems. When migrating from design-time to runtime, not only the sources of data that the Digital Twin operates upon must change, but changes in quality, precision, frequency, etc. of available real data might render observations made from design-time data invalid. For instance, a Digital Twin at runtime, controlling a real-time process, might not be able to reproduce behavior previously anticipated in design-time. Moreover, migrating from design-time to runtime might change the form of models the Digital Twin relies upon: while design-time models may be of arbitrary complexity due to the availability of sufficient computing capacity, deploying the same Digital Twin on a CPPS of less capacity might demand reductions in the used models' orders to ensure operation of the Digital Twin.

Migrating a Digital Twin from design-time to runtime demands changes in the sources of data, the form of models, and the interpretation of both. This may lead to additional engineering challenges for a Digital Twin. A careful separation of data acquisition and the conclusions drawn from design-time data can prevent this gap.

For numerical models, research in model order reduction [71] can contribute to the (partially) automated and deployment-specific reduction of design-time models into run-time

abstractions. For symbolic (knowledge bases, systems engineering models) and subsymbolic (artificial neural networks) models, this is subject of ongoing research [62].

Extensible digital twins Digital Twins are expected to chaperone their represented systems during their lifecycles. Therefore, they need to evolve with changes to the represented systems and in their environment. When new systems, subsystems, Digital Twins, sensors, or actuators become available, their data sources and models might be useful for optimizing the behavior of existing Digital Twins. Making these new data sources and models available in a Digital Shadow, without stopping the corresponding Digital Twin, and without major software engineering efforts either demands means to automatically discover and incorporate these or configuration mechanisms to adjust existing Digital Twins during their runtime. The former not only requires inventing mechanisms to discover novel data sources and models within the Digital Shadow, but also to integrate this new information meaningfully and safely without causing operational issues. The latter requires means to describe the availability of new data sources and models and their integration in Digital Shadows and Digital Twins tailored to non-programmers.

The evolution of represented CPPS requires a Digital Twin to incorporate new data sources and models during its runtime. Foreseeing this evolution during design is crucial to ensure the future extension of a Digital Twin.

For the syntactic and semantic representation of data sources, models, their interfaces and relations leveraging data structure modeling techniques, such as Unified Modeling Language (UML) class diagrams [64], and knowledge representation techniques, such as ontologies [45, 59], can facilitate engineering extensible Digital Shadows and Digital Twins. For both paradigms, (semi-)automated matching techniques that can facilitate discovering and integrating new data sources and models [1, 45] as well as service discovery mechanisms [37, 46] are available. For the non-invasive configuration of Digital Twins during runtime, low code modeling techniques [68] and results from research on models at runtime [2] might facilitate the integration of new data sources and models.

Adaptive digital twins Digital Twins are used to predict the behavior or evolution of the represented system, e.g., for predictive maintenance, future resource consumption, and more. To this end, they simulate possible behaviors based on currently available data and models. Depending on the frequency of changes of data and models as well as on the duration of computing predictions, these predictions might become outdated while being computed. Hence, computational resources are wasted and the Digital Twin's chances to react on properly predicted results are missed. If prediction is successful, but prediction and reality diverge, either the Digital Twins underlying data, models, knowledge bases, etc.

must be changed or the Digital Twins must use the predictions results to change reality. The Digital Twin's reaction to this divergence are highly application-specific and demand suitable modeling techniques to describe these reactions by domain experts.

A predictive Digital Twin must support domain-specific actions for reacting on the divergence of expectations and reality. These reactions can include applying domain-specific expertise or adjusting its models of reality.

Modeling can facilitate encapsulating domain-specific expertise in a machine-processable form that the Digital Twins can leverage. This can be continuous Simulink [63] models, discrete state-based [64] descriptions, or rule-based specifications [3]. To enable domain experts in efficiently reifying their expertise, all of these must be tailored to their domains, e.g., by applying techniques from software language engineering [31].

3 Engineering a Digital Twin and Its Digital Shadows

For engineering Digital Twins and Digital Shadows, we first have a closer look at their relationship and their components, which is depicted in Fig. 2. With respect to the definitions introduced in Sect. 1, we identify a fundamental set of conceptual constituents that most Digital Twin applications should comprise, even though concrete realizations might differ.

Modern CPPSs are usually equipped with sensors that monitor the system behavior. In addition, there are also cameras and external sensors in production plants, for example to monitor the activities of employees or to be able to react in time to smoke detection. These data are usually managed in different databases but can be combined with the help of a Data Lake.

The Digital Twin accesses the Data Lake and extracts Digital Shadows from it, combining exactly the information it requires for a specific purpose. A dedicated component within the Digital Twin, the *Shadow System*, performs this task. The Shadow System encapsulates the *Shadow Caster* and the *Shadow Controller*. The Shadow Caster processes data of the Data Lake and calculates Digital Shadows by combining insights and information of heterogeneous data sources within the Data Lake. It creates task-specific Digital Shadows, that can support the Digital Twin in performing its tasks. For example, a Digital Shadow can be the temporal evolution of a parameter within the CPPS to detect wear of CPPS components. Combined with material and maintenance information about this component, the Digital Shadow can enable predictive maintenance tasks. The Shadow Controller decides, when to create and destroy Digital Shadows. It controls the storing process of digital shadow insights and analysis results within the Data Lake and decides which data to be stored or ignored. The Shadow Controller also manages the visibility and access rights of Digital Shadows.

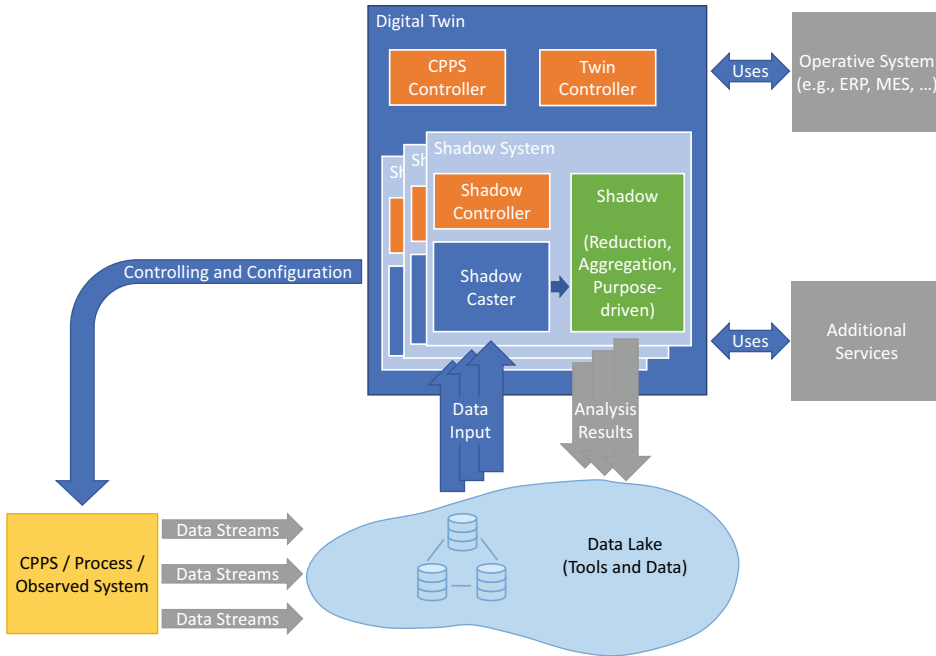


Fig. 2 Schema of the Relationship of Digital Twins and Digital Shadows

Other operative systems, such as ERP or MES, can access information provided by the Digital Twin, or leverage the Digital Twin to understand or influence the behavior of the CPPS. The same applies to other, domain-specific services. For applying reconfigurations to the system and controlling its activities the Digital Twin has a control flow to the CPPS. When the CPPS behavior needs to be adjusted and how its communication interface can be accessed is managed by the *CPPS Controller*, which is part of the Digital Twin. The *Twin Controller* encapsulates the behavior of the Digital Twin and decides which actions the Digital Twin should take, which tasks it should perform, and which Digital Shadow it requires for these tasks.

4 From Engineering Models to a Digital Twin

Engineering models are models that promote the design and construction of systems. Typical engineering models are technical or CAD drawings, circuit diagrams, or UML models. They support design and engineering activities by capturing a system's properties and giving insight about the system even before it is constructed, i.e., they prescribe properties of the things to be. Therefore, their application domain is usually in constructive domains where their prescriptive capabilities can support design decisions and design space exploration

of different variants of a system before it is constructed. Engineering models are created by domain experts, e.g., mechanical engineers, electrical engineers, software developers, administration, UI designers, etc. to master the complexity of CPPS systems. As such, they encapsulate domain knowledge that can support the realization of Digital Twins. In addition, engineering models can also support the Digital Twin at runtime, as they provide the Digital Twin the opportunity to understand and reason about the internal structure and behavior of the represented systems to detect anomalies, divergence of expected (e.g., simulated) behavior and real behavior, or other perils to operations.

Typically, engineering models are developed during the design phase of a CPPS and are of limited use after deploying the system, as they are hardly updated and, therefore, cannot represent the CPPS precisely. Their lifetime can be extended by deriving parts of the Digital Twin's functionality or providing a knowledge base for the Digital Twin. For this purpose, the essential information must be extracted from the engineering models and prepared in such a way that it can be further processed by other software systems and utilized by the Digital Twin.

4.1 Semantic Data Extraction

Each phase of a system's lifecycle contains specific activities, creates characteristic data, and relies on different services of the Digital Twin. When integrating model information, we distinguish between horizontal and vertical integration. Horizontal integration focuses on combining information from different lifecycle phases of the CPPS. For example, a CAD model of the design phase can help to evaluate whether the assembled product conforms to the specification during the implementation phase. Vertical integration combines information of models of different hierarchical levels. For example, a CPPS' SysML model consists of models representing its individual components. Vertical integration also refers to a stepwise refinement, where higher-resolution models refine a coarse system description.

Figure 3 shows the lifecycle phases of a CPPS and its Digital Twin. During *Design* the system's concept, its concrete functions and appearance are developed, based on customer requirements and market information. Data aggregated during this phase consists of customer requirements specification, often in unformalized mediums, such as plain text. However, it can also involve requirements models, e.g., specified in DOORS, where requirements are recorded and managed. Typical models that support the design phase are CAD models, describing the appearance and dimensions of the CPPS, SysML models that translate customer requirements into system functionality descriptions, and also simulations, e.g., in Matlab that evaluate the system's behavior. The Digital Twin supports the design phase by evaluating different variants of the CPPS and providing the optimal one concerning customer requirements and estimated working conditions. The *Implementation* phase combines all activities for creating the CPPS, including software development and hardware construction. During this phase, the different components of the CPPS are created, combined, and

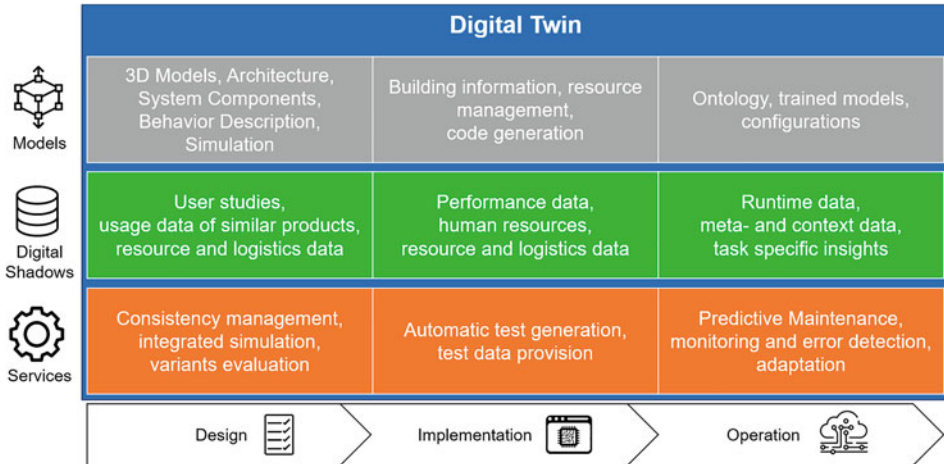


Fig. 3 The DT contains data, models, and services for every lifecycle phase of the underlying CPPS

tested to detect errors. Data in this phase include human performance, use of resources, material properties and test results. During the implementation, software architects create models specifying the internal structure of the software components and interfaces to other software systems. Also, models for Building Information Modeling (BIM) and Engineering Change Requests (ECR) are employed to support the realization of hardware components. The Digital Twin could support this phase by ensuring consistency between models of the design phase and the implementation, or by generating test cases to evaluate the implementation. During the *Operation* phase, the system is assessed and evaluated to ensure it functions as intended and does not become obsolete. In this phase, large amounts of data are created, since modern CPPS are equipped with sensors that capture changes within the system itself and its operating context likewise. The Digital Twin can monitor the system and check whether the expected parameters from the design phase conform to the real operating data. It accesses the Digital Shadows that provide information about estimated and real values and evaluate these to perform e.g., predictive maintenance.

4.2 Technologies for Connecting Digital Twins and Engineering Models

For the further use of engineering models beyond their intended life phase, special software technologies are required. During the realization of a CPPS different domains collaborate and develop individual model artifacts that in combination describe the CPPS. Combining information from these different models requires model integration mechanisms [15]. Through model integration each discipline involved in the CPPS and product development

process can develop using its own terminology and tools while the combination of these models describes the entire CPPS. Language integration [27, 44] focuses on integration of models on the language level, thus, also affects the tooling provided for describing CPPS. Language aggregation employs several artifacts of different languages to describe aspects of the target domain. The processed artifacts remain separated but describe the same CPPS. Language embedding combines models into one common artefact, while keeping the languages and tooling for these languages separated. Language inheritance also combines different languages into one artifact but in addition customizes elements of these languages [30].

Another approach for combining knowledge is tool integration [12]. Tool integration realizes the exchange of model data between specific tools. Thus, it reduces gaps during the development process and reduces inconsistencies.

Model-Driven Software Development Generating software is crucial to successfully integrate modeling in development processes [65]. Code for production systems or for test drivers can be efficiently generated and thus, improve the consistency between model and implementation and save resources. By deriving software code from engineering models, the development effort for Digital Twins can be reduced even further. Engineering models incorporate domain knowledge from which components of the Digital Twin can be derived. One challenge is the extraction of the relevant information from these models. In model-based software engineering symbols are an established way of extracting the relevant information from a model. Accordingly, it might be interesting to closely analyze engineering models according to this aspect and identify the significant symbols, that can be useful in other life phases, too.

Tagging One possibility to add required data to already existing engineering models is tagging. A tag model logically adds information to the tagged model while technically keeping the artifacts separated. Thus, the engineering model can stay clean of this additional information, and domain experts are not confused by additional model elements that do not directly correspond to the domain. A tag model could e.g., connect model elements from different life phases, thus ensure horizontal integration. It could also add information about the structural decomposition of a model to support vertical integration. It could also add data retrieval information to models to connect the engineering model with data about the physical system, its behavior or its development process.

Models at Runtime In contrast to those development methods for Digital Twins, models at runtime focuses the utilization of models during the CPPS runtime. Since engineering models encapsulate domain knowledge and information about the underlying CPPSs they can build a knowledge base for Digital Twins to rely on while they are running [4]. Thus, they support the Digital Twin to cope with new challenges and to adapt the twin to even to unanticipated changes. Runtime models are reflective, meaning that they are causally connected with the underlying CPPS. Thus, every change in the runtime model leads to a change in the reflected

system and vice versa [47]. A structural model, that describes the components of the CPPS, reflects the CPPS's constituents and can be update if new physical assets are added, or components are exchanged [24]. Physical models describe the dynamics and current state of physical phenomena [69]. The Digital Twin can rely on physical models to assure selected properties of the running CPPS. Behavioral denotes a runtime model capturing the dynamics of the systems, i.e., what the system can or will do based on its current state. Behavioral models can support run-time verification and sensitivity analysis, to determine how CPPS parameters are affected based on reconfigurations. For example, behavioral models may support the Digital Twin in predicting the CPPS's behavior or evaluating several possible reconfigurations for identifying the best solution for a present problem [17].

View-based Modeling Often, while applying model-driven development leads to redundancies and inconsistencies if models share a semantic overlap and describe the same CPPS. View-based modeling is an approach to tackle fragmentation of information across instances of different metamodels and can support the Digital Twin in combining knowledge from heterogeneous models. A view type defines the set of meta-classes whose instances can be displayed by a view [5]. Different view types can support the visualization of relevant model elements. Combined with a query language, they build a tool for describing and creating Digital Shadows and thus can also support the Digital Twin in providing insight about the current CPPS state.

Executable Metamodeling A metamodel is a model of a model and specifies the concepts and model elements that are useful for solving a specific class of problems. By means of Kermeta workbench [49], Ecore [66] meta models can also be executed. Kermeta provides an action language to implement the execution semantics of Ecore metamodels. It supports adding new methods to existing Ecore meta-classes, that define the execution semantics of the corresponding metamodel in the form of an interpreter [10].

5 Digital Shadows and Data Processing

For engineering Digital Shadows, the task of data processing is essential, as the Digital Shadow is closely linked to the data providing infrastructure. Regarding the standard data processing steps of ETL (*Extract-Transform-Load*) [34], the Digital Shadow has strong ties to the *transform* task: in the traditional sense, data is adjusted in this step, in order to be stored and used later. Digital Shadows exceed traditional data adjustment and aim to embed further intelligence in this step, making the *transform* task more sophisticated and therefore introducing a new smart data layer. For this to be possible, incoming data streams have to be steered—which is taken care of by the Data Lake—and relevant data must be identified, specified, and processed—which is the task of the Digital Shadow. Since Digital Shadows serve specific purposes, their data granularity can differ significantly; for some

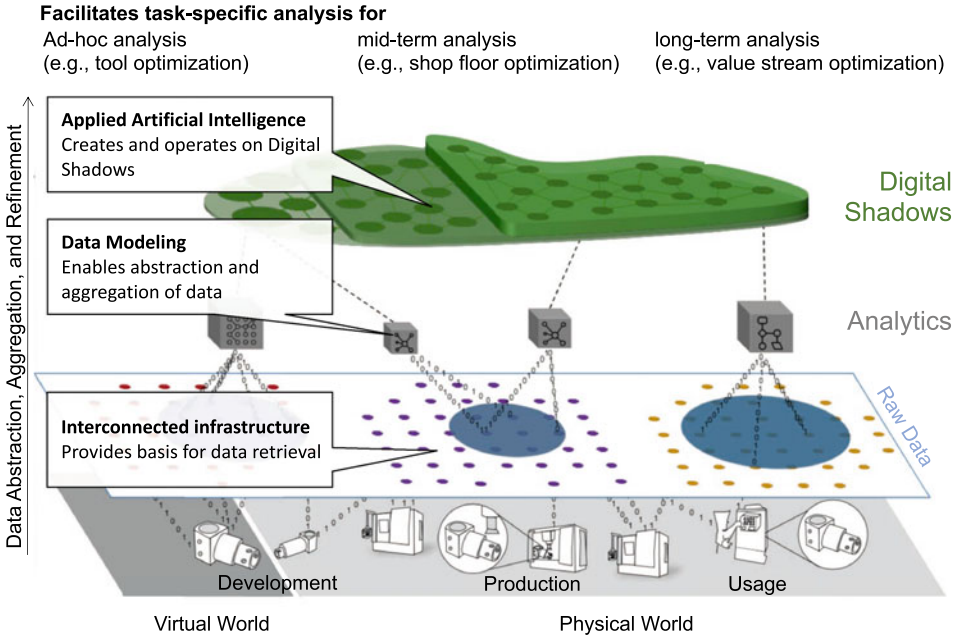


Fig. 4 Purpose-Driven Digital Shadows

purposes, a very detailed view on data is necessary, whereas for other purposes, aggregated or intelligently processed data is of interest. The structure of different granularity of Digital Shadows is depicted in Fig. 4, as envisioned in the Internet of Production [33, 36, 58].

From the bottom to the top, we observe the data abstraction, aggregation, and refinement process: In the bottom, data is in its raw format, corresponding to data gathering in the interconnected infrastructure. With each subsequent step up, the data can increasingly be refined, e.g., by employing data modeling, such that in the top we obtain task-specific Digital Shadows where artificial intelligence is applied. Depending on the task, we may need more refined or more raw data. Use cases regarding long-term analysis, e.g., value stream optimization, require more refined data, whereas, for ad-hoc analysis, e.g., tool optimization, more granular data is required. Mid-term analysis, e.g., shop floor optimization, might demand a mixture of both raw and already partly refined data.

The different levels of granularity and the different levels of data abstraction, aggregation, and refinement of Digital Shadows correspond well with the concept of Data Lakes and their levels of different data layers.

5.1 Data Lakes as a Serving Infrastructure

A Data Lake, as described in [60] is a repository where data from various sources can be stored in their original structures. In this regard, the concept of a Data Lake significantly differs to the data-handling approach of traditional data storage systems, with the most prominent concept being the data warehouse: The Data Lake introduces an *ELT (Extract-Load-Transform)* approach instead of the traditional *ETL*. Raw data is stored in its raw format without prior integration or aggregation to avoid restricting data usage and analysis to a predefined integrated schema. This especially means that we do not—as is common in data warehouses—have a schema-on-write paradigm, but delay the importance of schemas to the time of reading the data, i.e., we follow a schema-on-read approach. This also contributes to the challenge discussed in Sect. 2, that Digital Shadows should be flexible and should be able to incorporate new data sources at least semi-automatically, without requiring a software engineering process to define ETL workflows and integrated schemas.

In general, Data Lakes provide tools to extract data and metadata from heterogeneous sources, offer a data transformation engine that can transform or clean data and integrate it with other data, and provide interfaces to explore and to query data and metadata it contains. According to [60], a Data Lake is distinguishable into four components serving those purposes: *Ingestion Layer*, *Storage Layer*, *Transformation Layer*, and *Interaction Layer*. The composition of a typical Data Lake is shown in Fig. 5 and described in the following.

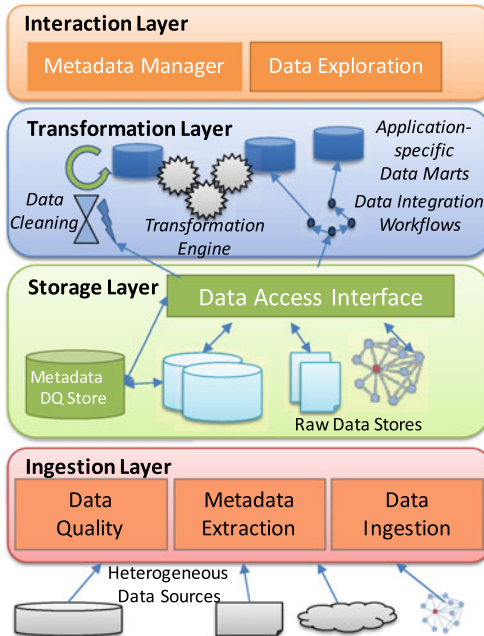


Fig. 5 Data Lake Architecture

Ingestion Layer: To make data available in Digital Shadows or Digital Twins, it is important that the data collection is as easy as possible. The ingestion of new data sources into the Data Lake should be as simple as a copy operation in a file system, as data from the sources is copied to the storage layer in its original format. To ensure sufficient data quality, the ingestion should extract metadata from the sources automatically and enforce data quality rules to prevent that data with insufficient quality pollutes the Data Lake.

Storage Layer: This layer is comprised of common big data tools, such as Apache Hadoop or Apache Spark. They provide storage and query mechanisms for heterogeneous data. Metadata management is usually limited to structural information, but should be enriched with semantic and data quality information to make the data more usable.

Transformation Layer: Raw data needs to be transformed and cleaned if it should be usable for an application. This functionality is provided by the Transformation Layer, which might also be implemented with big data tools as Apache Spark. This layer can be also considered as the Shadow Caster, as it creates Digital Shadows (or *Application-Specific Data Marts*) by aggregating, transforming, reducing, etc. the raw data.

Interaction Layer: Finally, the Interaction Layer provides means to interact with the data in the Data Lake or Digital Shadows and to make it available for a certain purpose in a Digital Twin.

An example reference model of a Data Lake System is provided by Constance, introduced in [25]. There are various benefits of Data Lakes compared to more traditional approaches. These include easier integration of heterogeneous data (e.g., relational data, JSON, XML, graphs, data streams, documents), easy exploration of heterogeneous data on data level instead of application level, and avoidance of data transformation during data ingestion. The latter has the effect of making it easier to handle data with a purpose that is unclear at design time.

On the other hand, Data Lakes pose the threat of deteriorating into data swamps. A data swamp is a Data Lake that is overwhelmed by unusable data to the degree of becoming unusable itself, because data of interest cannot be found or data cannot be interpreted anymore. A countermeasure for data swamps is to ensure metadata ingestion and data quality management when loading the data. An additional approach to ensure data comprehensibility is to adopt ontologies and other rich data models in Data Lakes. The relationship of Data Lakes and traditional data management systems is further addressed in [35].

For the use case of Industry 4.0 and the employment of Digital Shadows, the principle of Data Lakes to store data in its raw format, in particular, gives us the following advantages: The immense heterogeneity regarding the involved sensors and machinery, which often characterizes industrial production settings, can be handled well since it perfectly fits the paradigm of Data Lakes. The characteristic of Data Lakes to keep all data enables us to create multiple purpose-specific Digital Shadows based on data located in the Data Lake. In general, Digital Shadows fit well onto the idea that we have an extensive repository of data

and may select different traces of data, depending on the application of the specific Digital Shadow.

Addressing the challenges introduced in Sect. 2, we conclude: the three Vs of big data, are addressed as follows: Volume is addressed by the distributed nature of Data Lakes. Variety can be managed by the heterogeneous nature of the storage layer in Data Lakes. Velocity is most challenging aspect for Data Lakes as they are more ‘static’ repositories for long-term collection of data rather than real-time processing. However, it is also important to store streaming data in the Data Lake, which can be used, for example, for the training of machine learning algorithms. The challenges to capture all possibly relevant data and still keep all other data for later possible usage are implicitly handled by Data Lakes because all data is stored in its raw format.

With introducing a Data Lake as serving infrastructure for Digital Shadows, we face another non-technical issue: in a conglomerate of cross-company contributors, each stakeholder needs to collaborate ideally using a unified interface to access and share their data and Digital Shadows. The Data Lake provides us with such a central repository instead of many separate data silos. However, the nature of the collaboration may impose that not all data may be kept in the Data Lake. Reasons may be data access rights or policies of individual organizations. We address this issue by proposing a hybrid Data Lake approach: We introduce one central Data Lake, that everybody has access to. At the same time, we keep some data landscapes (e.g., Data Lakes) at participants’ sites where confidential data that cannot be shared with everybody resides. Still, we keep the central Data Lake and its interface as the primary instance in questions of data. This entails that the data which is not in the central Data Lake has to be retrieved from the private data repository—if allowed for the specific participant. With this, we face a new challenge—the challenge of integrating private Data Lakes. One Solution for this is to employ data virtualization. Data virtualization may take place on different levels, but for the architecture of a hybrid approach and to hide the virtualization from the users, the best choice is to employ it on the lowest level possible [73]. In this case, data governance becomes an interesting and at least partially open problem. An alliance-like approach to data governance is proposed by the Industrial Data Space initiative [56]. Concluding, we see Data Lakes as an essential component to enable Digital Shadows, already solving a lot of presented challenges.

5.2 From Data Processing to Digital Shadows

In order to engineer Digital Shadows, both application domain and computer science knowledge become a necessity resulting in cross-domain collaboration. The added value of cross-domain collaborations to engineer Digital Shadows is best illustrated by an example [36]: In [53] the combination of domain knowledge and artificial intelligence expertise provided by an interdisciplinary team allowed for the evaluation time of an engineering process simulation to be reduced by six orders of magnitude: while traditionally finite element simulation

is used to evaluate manually designed schedules for heavy plate rolling, [53] were able to adequately approximate this simulation by six reduced analytical models operating on data provided by Digital Shadows, therefore reducing the computation time from 30–240 min to less than 50 milliseconds. This example of significantly influencing a simulation’s computability demonstrates the advantages for diagnosis and predictions in domain-specific real-time if they are based on Digital Shadows.

The challenges formulated in Sect. 2 are tackled by Digital Shadows as follows: The reduction of the three Vs of big data is partially handled by the Data Lake system. Unsolved by the Data Lake is the reduction and refinement from massive data volumes to manageable ones. The Digital Shadow addresses this issue for e.g., computability reasons—as opposed to using the whole data in the Data Lake. The challenge of interoperability and traceability of data traces in the data layers is addressed by [19], introducing the concept of FACTDAG. Closely related to interoperability and traceability is the desired reusability of Digital Shadows. Reusability is particularly powerful when the groundwork of agreed-upon ontologies has been established beforehand. Using Digital Shadows, not only reusability of the data is to be considered, but also the reusability of the component extracting this data, the Shadow Caster. As Digital Shadows are digital artifacts describing real objects, they serve a specific purpose regarding this real object. Beyond this, they can be used as input for other systems or for analysis exceeding the extent of the primary use of the Digital Shadow. This is typically achieved by conforming to best practices of development, e.g., modularization and reuse. The difficulty to know which data is relevant and the data selection process that shall reduce the used data for computability, are addressed by on the one hand collaborating with application domain experts—for Industry 4.0 mostly mechanical engineers—and on the other hand with the approach to systematically use artificial intelligence methods to first find structure and insights in the data and second to automatize the process of building Digital Shadows, yielding a combination of the two. The commitment of whether data is relevant usually has to be made during the data gathering and design phase. However, based on the Data Lake infrastructure, we obtain the possibility to gather all data and only with the engineering of the Digital Shadow commit to a specific data model as input. This leads us to the importance of Artificial Intelligence in Digital Shadows.

5.3 Artificial Intelligence in Digital Shadows

In order to understand how data that is about to be incorporated into the Digital Shadow has to be prepared and handled, we first must have a closer look at the composition of the *Digital Shadow System* (cf. Fig. 2). In general, the Digital Shadow System consists of two main components—a passive part, the *Shadow Data*, and an active part, the *Shadow Caster* that may be executed and manipulates the passive part. The first component Shadow Data is typically located in a Data Lake. The second component, Shadow Caster, can embrace advanced techniques, especially from the domain of artificial intelligence, transcending

traditional transformations and thus incorporating intelligence. From a data-flow perspective, intelligence is applied earlier than before. This especially entails that Digital Shadows themselves are already incorporating domain knowledge and intelligence, establishing a new smart data layer.

Regarding the employment of AI in Digital Shadows, we distinguish between two approaches: Digital Shadows repeatedly using AI methods to generate the Shadows' data on the one hand and Digital Shadows using AI initially and only once to determine the structure of the Digital Shadow. The first approach employs continuous learning and employment of AI, whereas the second employs one-time learning of the structure. An example of the first case would be to have some parameterized machine learning component in the Digital Shadow, e.g., a neural network, where the parameters, i.e., in the case of the neural network the weights of the edges, are continuously recalculated. An example of the second approach would be to determine the inputs for the Digital Shadow, e.g., the input layer of a neural network. Naturally, the two approaches are not exclusive and can be combined. The integration of intelligence into the Digital Shadow raises the question at which point a Digital Shadow ends, and the application using the Digital Shadow begins: depending on the complexity of the Digital Shadow, intelligence that previously was contained in the application layer, may now be shifted into the Digital Shadow instead. This can have the advantage that this artifact can be used by several applications. In this case, however, they have to share the same ontology to be able to use the same Digital Shadow, as they would otherwise use the same data with different interpretations. Finding the right degree of transferring logic into the Digital Shadow, and therefore deciding how much logic should remain in the application layer and how much should instead be located in the Digital Shadow, poses an interesting research question of its own.

6 The Future of Digital Twins and Digital Shadows

This chapter has covered the concepts of Digital Shadows and Digital Twins, with a particular focus on engineering methods. Digital Shadows consist of data traces of an observed CPPS. A Digital Twin operates on them to enable novel analyses, monitoring, and optimization. They both offer extensive support for production companies throughout the development, production, and usage life cycles, while harnessing the interconnectivity of manufacturing devices with processes and domain knowledge. This lifts the Internet of Things to an Internet of Production, particularly considering the aspects of Industry 4.0. Integrated applications on top then allow to specifically plan, simulate, inspect, and control processes. This is an essential requirement to manage the ever-increasing complexity of modern production processes.

Depending on the levels of sophistication and integration, these concepts elevate the operability of CPPS. Low-level realizations, e.g. for real-time analysis, allow deployment directly on the CPPS. In contrast, a variety of solutions rely on cloud-based systems, as these

can dynamically provide necessary storage and computation capacities, often required for complex operations. Following the platform as a service (PaaS) paradigm, virtual platforms can be provided, which offer dedicated services for engineering individual Digital Twin applications. In turn, this fosters general interchangeability and extension of these frameworks. After all, complete and ready-made systems may be provided, which companies can configure and directly put into operation, similar to the software as a service (SaaS) approach. The conceptual separation of the Digital Shadow particularly offers enormous application potential for sustainability in Industry 4.0. The archival storage of raw data in Data Lakes ensures its long-term usability. This is an important factor in commissioning production systems; the infrastructure must be able to sustain for years or even decades. For instance, machine learning algorithms may operate on years of acquired data to enable more precise decision support. As a challenge, interoperability between various providers needs to be ensured; the potential withdrawal of an operator may not cause disruptions. Similarly, switching providers should not infer severe migration costs.

Developing corresponding frameworks offers exceptional possibilities for both research and industry in the near future. The data collections can be used to perform sustainable analyses on raw data collected over many years, to provide insights into CPPSs and automatically optimize them by integrating domain expertise. It will also enable new data-driven business model opportunities. It is particularly in cross-company collaboration that the automatic exchange of data opens up entirely new perspectives. First and foremost, challenges regarding data sovereignty must be solved [33]. Ultimately, however, a data-driven worldwide integration of CPPS leads to the emergence of synergy effects. While the advantages for machine manufacturers through distributed data collection are most obvious here, it also offers new possibilities for the production industry in general, e.g. to mutually benefit from the experience of other companies, for instance in terms of concrete machine parameters. Similar effects can also be anticipated through the exchange of engineering models. Therefore, Digital Shadows and Digital Twins will continue to be the key enablers of Industry 4.0, offering exciting new avenues regarding sustainability and data sovereignty.

Acknowledgements Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy—EXC-2023 Internet of Production—390621612

References

1. Mojeeb Al-Rhman Al-Khiaty and Moataz Ahmed. UML class diagrams: Similarity aspects and matching. *Lecture Notes on Software Engineering*, 4(1):41, 2016.
2. Gordon Blair, Nelly Bencomo, and Robert B France. Models@run.time. *Computer*, 42(10):22–27, 2009.
3. Pascal Bibow, Manuela Dalibor, Christian Hopmann, Ben Mainz, Bernhard Rumpe, David Schmalzing, Mauritius Schmitz, and Andreas Wortmann. Model-Driven Development of a Digital Twin for Injection Molding. In Schahram Dustdar, Eric Yu, Camille Salinesi, Dominique Rieu,

- and Vik Pant, editors, *International Conference on Advanced Information Systems Engineering (CAiSE'20)*, volume 12127 of *Lecture Notes in Computer Science*, pages 85–100. Springer International Publishing, June 2020.
4. Nelly Bencomo, Sebastian Götz, and Hui Song. Models@run.time: a guided tour of the state of the art and research challenges. *Software & Systems Modeling*, 18(5):3049–3082, 2019.
 5. Erik Burger, Jörg Henss, Martin Küster, Steffen Kruse, and Lucia Happe. View-based model-driven software development with modeljoin. *Software & Systems Modeling*, 15(2):473–496, 2016.
 6. Bundesministerium für Bildung und Forschung. Zukunftsprojekt Industrie 4.0. <https://www.bmbf.de/de/zukunftsprojekt-industrie-4-0-848.html>. Accessed: 2020-05-29.
 7. High Value Manufacturing Catapult. <https://hvm.catapult.org.uk/>. Accessed: 2018-06-05.
 8. Benoit Combemale, Robert France, Jean-Marc Jézéquel, Bernhard Rumpe, James Steel, and Didier Vojtisek. *Engineering Modeling Languages: Turning Domain Knowledge into Tools*. Chapman & Hall/CRC Innovations in Software Engineering and Software Development Series, November 2016.
 9. María Victoria Cengarle, Hans Grönniger, and Bernhard Rumpe. Variability within Modeling Language Definitions. In *Conference on Model Driven Engineering Languages and Systems (MODELS'09)*, LNCS 5795, pages 670–684. Springer, 2009.
 10. Benoit Combemale, Cécile Hardebolle, Christophe Jacquet, Frédéric Boulanger, and Benoit Baudry. Bridging the chasm between executable metamodeling and models of computation. 09 2012.
 11. James B Dabney and Thomas L Harman. *Mastering simulink*. Pearson, 2004.
 12. Manuela Dalibor, Nico Jansen, Judith Michael, Bernhard Rumpe, and Andreas Wortmann. Towards Sustainable Systems Engineering-Integrating Tools via Component and Connector Architectures. In Georg Jacobs and Jonas Marheineke, editors, *Antriebstechnisches Kolloquium 2019: Tagungsband zur Konferenz*, pages 121–133. Books on Demand, February 2019.
 13. Manuela Dalibor, Judith Michael, Bernhard Rumpe, Simon Varga, and Andreas Wortmann. Towards a Model-Driven Architecture for Interactive Digital Twin Cockpits. In Gillian Dobbie, Ulrich Frank, Gerti Kappel, Stephen W. Liddle, and Heinrich C. Mayr, editors, *Conceptual Modeling*, pages 377–387. Springer International Publishing, October 2020.
 14. Sanford Friedenthal, Alan Moore, and Rick Steiner. *A practical guide to SysML: the systems modeling language*. Morgan Kaufmann, 2014.
 15. Robert France and Bernhard Rumpe. Model-driven Development of Complex Software: A Research Roadmap. *Future of Software Engineering (FOSE '07)*, pages 37–54, May 2007.
 16. Peter Fritzon. *Principles of object-oriented modeling and simulation with Modelica 2.1*. John Wiley & Sons, 2010.
 17. A. Filieri, G. Tamburrelli, and C. Ghezzi. Supporting self-adaptation via quantitative verification and sensitivity analysis at run time. *IEEE Transactions on Software Engineering*, 42(1):75–99, 2016.
 18. Hans Grönniger, Jochen Hartmann, Holger Krahn, Stefan Kriebel, and Bernhard Rumpe. View-based Modeling of Function Nets. In *Object-oriented Modelling of Embedded Real-Time Systems Workshop (OMER4'07)*, 2007.
 19. L. Gleim, J. Pennekamp, M. Liebenberg, M. Buchsbaum, P. Niemietz, S. Knape, A. Epple, S. Storms, D. Trauth, T. Bergs, C. Brecher, S. Decker, G. Lakemeyer, and K. Wehrle. Factdag: Formalizing data interoperability in an internet of production. *IEEE Internet of Things Journal*, 7(4):3243–3253, 2020.
 20. Sandra Geisler, Christoph Quix, Sven Weber, and Matthias Jarke. Ontology-based data quality management for data streams. *ACM Journal of Data and Information Quality*, 7(4):18:1–18:34, 2016.

21. Georges GE Gielen and Rob A Rutenbar. Computer-aided design of analog and mixed-signal integrated circuits. *Proceedings of the IEEE*, 88(12):1825–1854, 2000.
22. Mikell Groover and EWJR Zimmers. *CAD/CAM: computer-aided design and manufacturing*. Pearson Education, 1983.
23. Matthew Hause et al. The SysML Modelling Language. In *Fifteenth European Systems Engineering Conference*, volume 9, pages 1–12, 2006.
24. Christian Heinzemann, Steffen Becker, and Andreas Volk. Transactional execution of hierarchical reconfigurations in cyber-physical systems. *Software and Systems Modeling*, 18:157–189, 02 2017.
25. Rihan Hai, Sandra Geisler, and Christoph Quix. Constance: An Intelligent Data Lake System. In Fatma Özcan, Georgia Koutrika, and Sam Madden, editors, *Proceedings of the 2016 International Conference on Management of Data - SIGMOD '16*, pages 2097–2100, New York, New York, USA, 2016. ACM Press.
26. Richard Hopkins and Kevin Jenkins. *Eating the IT elephant: Moving from greenfield development to brownfield*. Addison-Wesley Professional, 2008.
27. Arne Haber, Markus Look, Pedram Mir Seyed Nazari, Antonio Navarro Perez, Bernhard Rumpe, Steven Völkel, and Andreas Wortmann. Integration of Heterogeneous Modeling Languages via Extensible and Composable Language Components. In *Model-Driven Engineering and Software Development Conference (MODELSWARD'15)*, pages 19–31. SciTePress, 2015.
28. Katrin Hölldobler, Judith Michael, Jan Oliver Ringert, Bernhard Rumpe, and Andreas Wortmann. Innovations in Model-based Software And Systems Engineering. *The Journal of Object Technology*, 18(1):1–60, July 2019.
29. David Harel and Bernhard Rumpe. Meaningful Modeling: What’s the Semantics of „Semantics“? *IEEE Computer*, 37(10):64–72, 2004.
30. Katrin Hölldobler and Bernhard Rumpe. *MontiCore 5 Language Workbench Edition 2017*. Aachener Informatik-Berichte, Software Engineering, Band 32. Shaker Verlag, December 2017.
31. Katrin Hölldobler, Bernhard Rumpe, and Andreas Wortmann. Software Language Engineering in the Large: Towards Composing and Deriving Languages. *Computer Languages, Systems & Structures*, 54:386–405, 2018.
32. The Industrial Value Chain Initiative. <https://iv-i.org/wp/en/about-us/whatsivi/>. Accessed: 2018-06-04.
33. Matthias Jarke. Data Sovereignty and the Internet of Production. In *Advanced Information Systems Engineering*, Lecture Notes in Computer Science, Cham, Switzerland, 2020. Springer International Publishing.
34. Matthias Jarke, Maurizio Lenzerini, Yannis Vassiliou, and Panos Vassiliadis. *Fundamentals of Data Warehouses*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003.
35. Matthias Jarke and Christoph Quix. *On Warehouses, Lakes, and Spaces: The Changing Role of Conceptual Modeling for Data Integration*, pages 231–245. Springer International Publishing, Cham, 2017.
36. Matthias Jarke, Günther Schuh, Christian Brecher, Matthias Brockmann, and Jan-Philipp Prote. Digital shadows in the internet of production. *ERCIM News*, 2018(115), 2018.
37. Hejun Jiao, Jing Zhang, Jun Huai Li, and Jinfa Shi. Research on cloud manufacturing service discovery based on latent semantic preference about owl-s. *International Journal of Computer Integrated Manufacturing*, 30(4-5):433–441, 2017.
38. Anja Klein and Wolfgang Lehner. Representing data quality in sensor data streaming environments. *J. Data and Information Quality*, 1(2):10:1–10:28, 2009.
39. Anneke Kleppe. *Software Language Engineering: Creating Domain-Specific Languages using Metamodels*. Pearson Education, 2008.

40. Sathish AP Kumar, R Madhumathi, Pethuru Raj Chelliah, Lei Tao, and Shangguang Wang. A novel digital twin-centric approach for driver intention prediction and traffic congestion avoidance. *Journal of Reliable Intelligent Environments*, 4(4):199–209, 2018.
41. Johannes Kößler, Kristin Paetzold, et al. Support of the System Integration with Automatically Generated Behaviour Models. In *DS 80-11 Proceedings of the 20th International Conference on Engineering Design (ICED 15) Vol 11: Human Behaviour in Design, Design Education; Milan, Italy, 27-30.07. 15*, pages 021–030, 2015.
42. Martin Liebenberg and Matthias Jarke. Information Systems Engineering with Digital Shadows: Concept and Case Studies. In *Advanced Information Systems Engineering, Lecture Notes in Computer Science*, Cham, Switzerland, 2020. Springer International Publishing.
43. WD Li, Wen Feng Lu, Jerry YH Fuh, and YS Wong. Collaborative computer-aided design-research and development status. *Computer-aided design*, 37(9):931–940, 2005.
44. Markus Look, Antonio Navarro Pérez, Jan Oliver Ringert, Bernhard Rumpe, and Andreas Wortmann. Black-box Integration of Heterogeneous Modeling Languages for Cyber-Physical Systems. In *Globalization of Modeling Languages Workshop (GEMOC'13)*, volume 1102 of *CEUR Workshop Proceedings*, 2013.
45. Severin Lemaignan, Ali Siadat, J-Y Dantan, and Anatoli Semenenko. MASON: A proposal for an ontology of manufacturing domain. In *IEEE Workshop on Distributed Intelligent Systems: Collective Intelligence and Its Applications (DIS'06)*, pages 195–200. IEEE, 2006.
46. Matthias Loskyll, Jochen Schlick, Stefan Hodek, Lisa Ollinger, Tobias Gerber, and Bogdan Pirvu. Semantic service discovery and orchestration for manufacturing processes. In *ETFA2011*, pages 1–8. IEEE, 2011.
47. Pattie Maes. Concepts and experiments in computational reflection. In *Conference Proceedings on Object-Oriented Programming Systems, Languages and Applications, OOPSLA '87*, page 147-155, New York, NY, USA, 1987. Association for Computing Machinery.
48. merics - Mercator Institute for China Studies. Made in China 2025. https://www.merics.org/sites/default/files/2017-09/MPOC_No.2_MadeinChina2025.pdf. Accessed: 2018-06-06.
49. Pierre-Alain Muller, Franck Fleurey, and Jean-Marc Jézéquel. Weaving executability into object-oriented meta-languages. In *International Conference on Model Driven Engineering Languages and Systems*, pages 264–278. Springer, 2005.
50. Ivano Malavolta, Patricia Lago, Henry Muccini, Patrizio Pelliccione, and Antony Tang. What Industry Needs from Architectural Languages: A Survey. *IEEE Transactions on Software Engineering*, 39(6):869–891, 2013.
51. Azad M. Madni, Carla C. Madni, and Scott D. Lucero. Leveraging Digital Twin Technology in Model-Based Systems Engineering. *Systems*, 7(1):7, 2019.
52. Michael F. Molnar. The U.S. Advanced Manufacturing Initiative. https://www.nist.gov/system/files/documents/2017/04/28/Molnar_091211.pdf, 2017. Accessed: 2018-06-06.
53. R. Meyes, H. Tercan, T. Thiele, A. Krämer, J. Heinisch, M. Liebenberg, G. Hirt, Ch. Hopmann, G. Lakemeyer, T. Meisen, and S. Jeschke. Interdisciplinary data driven production process analysis for the internet of production. *Procedia Manufacturing*, 26:1065 – 1076, 2018. 46th SME North American Manufacturing Research Conference, NAMRC 46, Texas, USA.
54. Arne Nordmann, Nico Hochgeschwender, and Sebastian Wrede. A survey on domain-specific languages in robotics. In *International Conference on Simulation, Modeling, and Programming for Autonomous Robots*, pages 195–206. Springer, 2014.
55. Petru Nicolaescu, Mario Rosenstengel, Michael Derntl, Ralf Klamma, and Matthias Jarke. View-Based Near Real-Time Collaborative Modeling for Information Systems Engineering. In *International Conference on Advanced Information Systems Engineering*, pages 3–17. Springer, 2016.
56. Boris Otto and Matthias Jarke. Designing a multi-sided data platform: findings from the international data spaces case. *Electron. Mark.*, 29(4):561–580, 2019.

57. Eldad Palachi, Chaim Cohen, and Sakairi Takashi. Simulation of cyber physical models using SysML and numerical solvers. In *2013 IEEE International Systems Conference (SysCon)*, pages 671–675. IEEE, 2013.
58. Jan Pennekamp, René Glebke, Martin Henze, Tobias Meisen, Christoph Quix, Rihan Hai, Lars Gleim, Philipp Niemietz, Maximilian Markus Rudack, Simon Knappe, Alexander Epple, Daniel Trauth, Uwe Vroomen, Thomas Bergs, Christian Brecher, Andreas Bührig-Polaczek, Matthias Jarke, and Klaus Wehrle. Towards an Infrastructure Enabling the Internet of Production. In *2019 IEEE International Conference on Industrial Cyber Physical Systems (ICPS 2019) : Howards Plaza Hotel Taipei, Taiwan, 06-09 May, 2019 / publisher: IEEE*, pages 31–37, Piscataway, USA, May 2019. 2nd IEEE International Conference on Industrial Cyber-Physical Systems, Taipei (Taiwan), 6 May 2019 - 9 May 2019, IEEE.
59. André Pomp, Johannes Lipp, and Tobias Meisen. You are missing a concept! enhancing ontology-based data access with evolving ontologies. In *13th IEEE International Conference on Semantic Computing, ICSC 2019, Newport Beach, CA, USA, January 30 - February 1, 2019*, pages 98–105. IEEE, 2019.
60. Christoph Quix and Rihan Hai. Data Lake. In Sherif Sakr and Albert Zomaya, editors, *Encyclopedia of Big Data Technologies*, pages 1–8. Springer International Publishing, Cham, 2018.
61. Ana Luísa Ramos, José Vasconcelos Ferreira, and Jaume Barceló. Model-Based Systems Engineering: An Emerging Approach for Modern Systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(1):101–111, 2011.
62. K Ramesh, R Girish Ganesan, and K Mahalakshmi. Approximation and optimization of discrete systems using order reduction technique. *Energy Procedia*, 117:761–768, 2017.
63. Bernhard Rumpe, Christoph Schulze, Michael von Wenckstern, Jan Oliver Ringert, and Peter Manhart. Behavioral Compatibility of Simulink Models for Product Line Maintenance and Evolution. In *Software Product Line Conference (SPLC'15)*, pages 141–150. ACM, 2015.
64. Bernhard Rumpe. *Modeling with UML: Language, Concepts, Methods*. Springer International, July 2016.
65. Bernhard Rumpe. *Agile Modeling with UML: Code Generation, Testing, Refactoring*. Springer International, May 2017.
66. Dave Steinberg, Frank Budinsky, Ed Merks, and Marcelo Paternostro. *EMF: eclipse modeling framework*. Pearson Education, 2008.
67. Duansen Shanguan, Liping Chen, and Jianwan Ding. A Hierarchical Digital Twin Model Framework for Dynamic Cyber-Physical System Design. In Unknown, editor, *Proceedings of the 5th International Conference on Mechatronics and Robotics Engineering - ICMRE'19*, pages 123–129, New York, New York, USA, 2019. ACM Press.
68. Raquel Sanchis, Óscar García-Perales, Francisco Fraile, and Raul Poler. Low-Code as Enabler of Digital Transformation in Manufacturing Industry. *Applied Sciences*, 10(1):12, 2020.
69. Adalberto Sampaio Junior, Fabio Costa, and Peter Clarke. A model-driven approach to develop and manage cyber-physical systems. volume 1079, 09 2013.
70. Sumit Singh, Essam Shehab, Nigel Higgins, Kevin Fowler, Tetsuo Tomiyama, and Chris Fowler. Challenges of digital twin in high value manufacturing. In *SAE Technical Paper*. SAE International, 10 2018.
71. Wilhelmus HA Schilders, Henk A Van der Vorst, and Joost Rommes. *Model order reduction: theory, research aspects and applications*, volume 13. Springer, 2008.
72. Thomas Uhlemann, Christian Lehmann, and Rolf Steinhilper. The digital twin: Realizing the cyber-physical production system for industry 4.0. *Procedia CIRP*, 61:335–340, 12 2017.
73. Rick F. van der Lans. The Fusion of Distributed Data Lakes Developing Modern Data Lakes A Technical Whitepaper. [https://itlligenze.conthub.io/uploads/5/241/files/6430_Whitepaper%20TIBCO%20BigData%20V2%20\(1\)_0.pdf](https://itlligenze.conthub.io/uploads/5/241/files/6430_Whitepaper%20TIBCO%20BigData%20V2%20(1)_0.pdf), 2019. Accessed: 2020-05-13.

-
74. Andreas Wortmann, Olivier Barais, Benoit Combemale, and Manuel Wimmer. Modeling Languages in Industry 4.0: an Extended Systematic Mapping Study. *Software and Systems Modeling*, 19(1):67–94, January 2020.
 75. Tim Weikiens. *Systems engineering with SysML/UML: modeling, analysis, design*. Elsevier, 2011.