Taylor & Francis
Taylor & Francis Group

International Journal of Production Research

ISSN: 0020-7543 (Print) 1366-588X (Online) Journal homepage: www.tandfonline.com/journals/tprs20

Purposeful decision-support with Digital Shadows: a model catalog architecture and its implementation for production scheduling in the injection molding domain

Patrick Sapel, Aymen Gannouni, Anas Abdelrazeq, Christian Hopmann & Robert H. Schmitt

To cite this article: Patrick Sapel, Aymen Gannouni, Anas Abdelrazeq, Christian Hopmann & Robert H. Schmitt (17 Oct 2025): Purposeful decision-support with Digital Shadows: a model catalog architecture and its implementation for production scheduling in the injection molding domain, International Journal of Production Research, DOI: 10.1080/00207543.2025.2573194

To link to this article: https://doi.org/10.1080/00207543.2025.2573194

Published online: 17 Oct 2025.

INTERNATIONAL JOURNAL OF PRODUCTION RESEARCH
https://doi.org/10.1080/00207543.2025.2573194

Taylor & Francis
Taylor & Francis Group

Purposeful decision-support with Digital Shadows: a model catalog architecture and its implementation for production scheduling in the injection molding domain

Patrick Sapel [a], Aymen Gannouni [b], Anas Abdelrazeq [b], Christian Hopmann [a] and Robert H. Schmitt [b]

[a]Institute for Plastics Processing, RWTH Aachen University, Aachen, Germany ; [b]Chair of Intelligence in Quality Sensing (WZL-IQS), RWTH Aachen University, Aachen, Germany

ABSTRACT

Industry 4.0 promises more efficient decision support systems by enabling the retrieval of data from various and heterogeneous sources and processing them in appropriate models. Consequently, Digital Shadows are introduced to automate the identification of models for specific issues and retrieve necessary input data automatically, streamlining decision-making processes. However, the absence of an architecture containing a model repository, information on model processing, required model input data, and their respective source systems poses challenges to implement this automated process. To address this, we propose a model catalog architecture as the cornerstone of automated model processing. Therefore, we ascertain the fundamental architectural requirements by delineating the essential interfaces, gates, and databases involved. These requirements are then translated into a modular and scalable architecture, facilitating the reuse of repetitive files, such as a translator acting as a middleware between different source systems. The architecture enables seamless integration of new models, provided they adhere to standardised definitions. Additionally, we transfer the architecture into a suitable framework. To validate our concept, we implement an exemplary model catalog that demonstrates production scheduling in an injection molding shop floor, completed with a web-based interface and integration of an enterprise resource planning system for real-time data access.

ARTICLE HISTORY
Received 15 June 2024
Accepted 26 September 2025

KEYWORDS
Model catalog; decision support system; industry 4.0; digital shadow; interoperability; standardisation

1. Introduction

In today's rapidly evolving industrial landscape, data has become elemental for driving informed decision-making and achieving operational excellence (Yin and Kaynak 2015). The transition from traditional manufacturing practices to data-driven approaches has been accelerated by advances in digital technologies and the advent of initiatives such as Industry 4.0. These industrial paradigms emphasise the integration of Cyber-Physical Production Systems (CPPS) or the Industrial Internet of Things (IIoT) that lead to smart factories capable of autonomous decision-making and optimisation (Kaur 2025; Raptis, Passarella, and Conti 2019). Here, the exponential growth in generated data, coupled with advancements in data processing and analytics, has transformed the role of data from a passive record to an active driver of strategic decisions. Modern decision support systems should process vast amounts of data to provide real-time insights and give data-driven recommendations, enabling industries to improve overall efficiency (Tao et al. 2018). Those recommendations are typically based on adequate models, which process the gathered data and transform them into knowledge (Cañas et al. 2021). Extensive model utilisation has become increasingly important, for example in model-based system engineering (MBSE), a methodology that uses models to support the entire lifecycle of a system, from conception and design to verification and validation, through to decommissioning. Therefore, models play a crucial role in decision support (Fernández Pérez and Hernández 2019). Today, data from different systems is typically gathered and manually pushed into models, which are also selected manually (Li 2025).

In an Industry 4.0 environment, this process should be transformed into an automated pull mechanism, wherein models designed to address predefined decision problems autonomously retrieve all necessary data from various source systems and directly initiate their processing.

CONTACT Patrick Sapel patrick.sapel@rwth-aachen.de Institute for Plastics Processing, RWTH Aachen University, Seffenter Weg 201, Aachen 52074, Germany

Implementing such a pull mechanism poses several significant challenges. Firstly, it necessitates data consistency and seamless integration across diverse software systems. Currently, many software systems operate as isolated data silos, often utilising proprietary data formats and lacking automated accessibility for other systems, particularly when sourced from different vendors. This leads to errors, data loss during transmission, and inefficiencies caused by the need for manual re-entry into different application software, increasing implementation efforts (Kamm, Jazdi, and Weyrich 2021). A first step toward addressing these issues is the use of central data dictionaries, which facilitate standardised data definitions, including metadata and unique identifiers (Panja, Chakraborty, and Kumar 2020). However, efficient frameworks and industry-ready solutions for managing data silos remain absent or insufficiently implemented. Secondly, models required for data processing must be semantically annotated to establish a direct connection to the necessary input data. While initial approaches for semantic model annotation exist, they often fail to incorporate widely accepted standards, thereby limiting interoperability. Thirdly, a comprehensive data architecture that integrates all essential components for a pull-based approach is currently lacking. In particular, a middleware solution is required to bridge the gap between models and proprietary software systems to overcome the challenges posed by different data silos. Additionally, for broad industrial adoption, this middleware must be aligned with established standards to ensure interoperability. Furthermore, legacy systems with outdated firmware or lack of digital interfaces add to the complexity. Economic challenges, such as scalability and flexibility, also hinder the gradual implementation of Industry 4.0 in existing production systems. Organisational and cultural challenges, including skill shortages and the need for competency development, further hamper progress (Bajic et al. 2021). In summary, current challenges hinder interoperability and the efficiency of business processes.

To enable interoperability across various industries, several frameworks have been developed. One such high-level framework is the European Interoperability Framework, which defines a model to ensure interoperability across legal, organisational, semantic, and technical layers (European Union 2017). This high-level framework acts as a foundation for more detailed frameworks and infrastructures.

One detailed infrastructure has been developed in the Cluster of Excellence 'Internet of Production' (IoP) at the RWTH Aachen University, which serves as the foundation for our research. The IoP ensures secure, real-time access to all relevant data, anytime and anywhere, making it a cornerstone of Industry 4.0. It facilitates the entire product life cycle and enables a fast and valid decision support by generating and utilising cross-domain knowledge (Brecher et al. 2024; Pause et al. 2019).

Enabling efficient decision support, the Digital Shadow was introduced. The Digital Shadow is a concept to automatically acquire the necessary data from various applications to efficiently solve a specific task, preventing errors caused by manual data handling (Becker et al. 2021). To achieve this, a middleware should manage and realise the connection to different, proprietary source systems, allowing the data gathering and forwarding to the right models (Pause et al. 2019; Schuh et al. 2017).

When applying the concept of Digital Shadows and adequate middleware to address challenges related to data silos, it becomes evident that while Digital Shadows have been conceptually modelled, their real-world implementation remains limited, necessitating further research. Thus, we introduce the concept of a semantic model catalog as a repository for diverse models, contributing to form the semantic layer of the European Interoperability Framework. The novelty of our research comprises a modular and scalable data architecture for a model catalog for the manufacturing domain, empowering a Digital Shadow to access requisite models and corresponding input data automatically. Hence, the research question we address in this contribution is as follows:

'How can we design a data architecture that is modular and scalable for a model catalog, enabling automatic acquisition of semantically defined models and their corresponding input data by Digital Shadows?'

Overall, the key contributions of this work are: (1) an extension of the Digital Shadow meta-model, (2) the development of a model signature for semantic model definition, (3) the design of a translator serving as middleware for various source systems, and (4) the establishment of a comprehensive data architecture that enables the automatic processing of models by Digital Shadows through the integration of the model signature and the translator. In addition, we provide a general framework to set up such a model catalog. The methodology and framework we present are independent of the underlying manufacturing process or the specific decision problem, so our contribution applies to various domains. To validate our findings and demonstrate the benefits, we implement a model catalog specifically for scheduling problems within an injection molding shopfloor.

The remainder of our contribution is as follows: We introduce the IoP as the theoretical foundation of our research with a special focus on Digital Shadows and its characteristics in Section 2. Then, we explore related

works in Section 3. Next, we present the Design Science Research Methodology (DSRM) as our methodical approach and outline our concrete work in Section 4. Afterwards, we detail the general process of using the model catalog as a sequence diagram and derive requirements for an architecture we subsequently built in Section 5 and introduce a general framework for implementing a model catalog. We demonstrate a real-world application in Section 6 by scheduling an injection molding shopfloor and discuss our results in Section 7. Finally, we conclude with future research directions in Section 8.

2. The model catalog in the context of digital shadows and the internet of production

The infrastructure of the Internet of Production consists horizontal cycles and vertical layers and is illustrated in Figure 1.

Horizontally, the infrastructure includes the life cycle of the product that spans from the development cycle over the production cycle till the user cycle. Each cycle requires distinct decisions, from optimal product designs to cost-efficient production processes. Usually, various software applications are available, such as Manufacturing Execution Systems (MES) or Plant Data Acquisition (PDA), each containing different raw data. Data (or updates) from one application is often required in another, e.g. the quantity of a finished job recorded via PDA must be directly visible in the MES. Vertically, the IoP infrastructure allows so called Smart Experts the purposeful use of data from application software or in a raw format, to tackle for example event-driven decisions, autonomous actions, or adaptive processes. Therefore, data has to become Smart Data that is enriched with semantics defining their context. Using contextualised Smart Data, a middleware can orchestrate data flows from proprietary source systems and provide appropriate models for their processing. In the IoP infrastructure, this middleware is called Middleware+. The '+' denotes its evolution from a simple component towards a semantically enriched one. In our contribution, we introduce the concept of a semantic model catalog as a repository for various models as a concrete realisation of a Middleware+ within the IoP infrastructure.

In the context of Industry 4.0, different concepts and technologies have evolved. In the IoP, we focus on the Digital Shadow as a data trace to assist Smart Experts. Considering publications dealing with novel digital concepts reveal that, beside Digital Shadows, most prominent concepts are Digital Models and Digital Twins, e.g. in van der Valk et al. (2022) or Sjarov et al. (2020). Digital Models are digital representations of entities in a simplified and pragmatic manner (Stachowiak 1973). They facilitate the integration with other digital systems and tools, enhancing adaptability across various applications and environments. In contrast, Digital Twins are comprehensive virtual replicas of assets that contain detailed asset information (Jones et al. 2020). However, for purposeful decision support, only a subset of Digital Twin

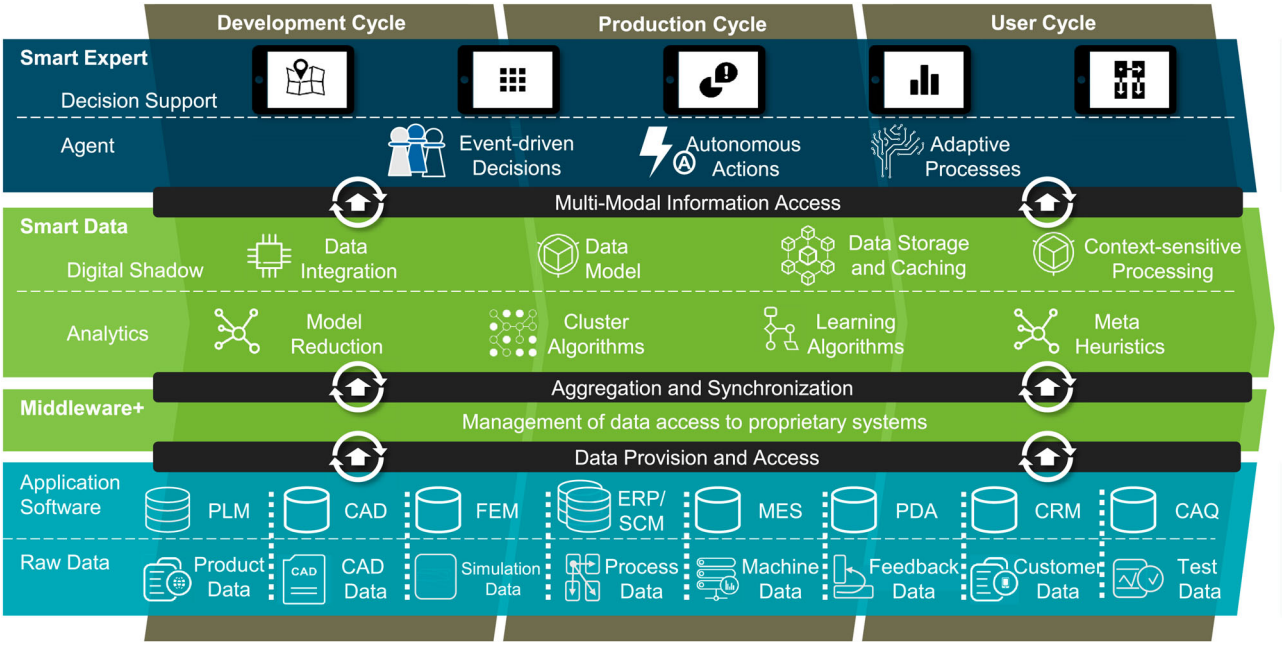


Figure 1. The infrastructure of the IoP demonstrating the need for data-based decision support, based on Schuh et al. (2017).

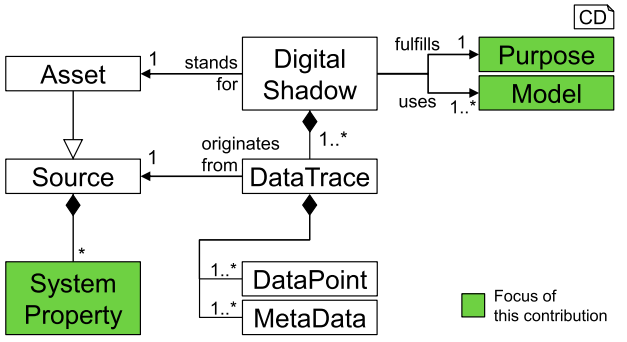


Figure 2. The DS meta-model, based on Becker et al. (2021).

data is typically required. Managing a fully detailed Digital Twin for a specific purpose can be impractical. To address this challenge, the Digital Shadow was introduced as a lightweight entity that consolidates only the data necessary for a specific decision support task from various source systems. Digital Shadows aggregate and structure production data to improve decision making and operational efficiency. Their primary function is to ensure that the right information is delivered to the right place at the right time, thereby optimising information flow across manufacturing systems (Bauernhansl, Hartleif, and Felix 2018; Braun et al. 2023). For these reasons, Digital Shadows are well-suited to addressing our specific requirements.

Considering this, we follow the definition of Bibow et al. (2020), that the 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, a Digital Shadow is the element that contains a result to provide purpose-driven decision support by using appropriate models processed with the right data. Figure 2 introduces the (reduced) meta-model of the Digital Shadow as UML class diagram.

The term 'Digital Shadow' stands for an `Asset`, functioning as a `Source` for data. Assets encompass various elements of interest, such as production machinery. `System Properties` define the characteristics of these sources, ideally in a unique and semantic manner to facilitate interoperable asset communication. These sources generate data, resulting in `DataPoints` and corresponding `MetaData`, including details such as data creation timestamps. Aggregating all data points forms a `DataTrace`. The trigger for data collection is determined by the `Purpose`, which may be set by either humans or software systems. Furthermore, the `Purpose` also plays a crucial role in selecting `Models` that are suitable for addressing the decision problem (Becker et al. 2021). Notably, the Digital Shadow is not restricted to specific model types (e.g. simulation or behaviour models); an overview of existing model types is provided in acatech – National Academy of Science and Engineering (2021).

A fundamental characteristic of Digital Shadows is that they do not directly influence the system. Rather, they provide decision support for subsequent processes (Kritzinger et al. 2018). While previous research defines the scope and key components of Digital Shadows, it lacks a concrete architecture that is essential for automatically identifying relevant data and models for decision support. In this contribution, we demonstrate how, based on a given `Purpose`, the Digital Shadow can identify appropriate models and their required input data to support decision-making. Our proposed solution enables the creation of versatile model catalogs that are independent of the underlying manufacturing processes.

3. Related work

In this section, we examine the current state of research regarding architectures and blueprints for managing distributed data silos and establishing semantic (reference) model catalogs.

Hirsch, Hoher, and Huber (2023) propose an architecture that exclusively focuses on the utilisation of OPC UA. To achieve this, they developed an OPC UA Query Model, which enables structured data retrieval from various sources by equipping them with OPC UA servers. However, in contrast to Digital Shadows, their work is limited exclusively to OPC UA.

Ameziane et al. (2022) developed the I4.0 Core Information Model, which comprises various submodels primarily derived from IEC 62264 (), such as models for equipment, products, and processes. These submodels are implemented as knowledge graphs, where extracted data is instantiated. A key distinction from Digital Shadows is that their solution does not connect directly to original systems, such as ERP systems, to retrieve live data.

Xing (2024) aims to address database heterogeneity from the perspective of in storage formats. The proposed solution employs predefined agents for multiple databases, transforming their content into XML format to ensure format interoperability. By defining an XML Document Type Definition (DTD), semantic interoperability could potentially be achieved. Hernández et al. (2023) propose a methodology based on data lakes. Data lakes represent a modern concept in which data is included in its raw form and processed as needed, whereas ERP systems store structured, validated, and processed data in real time for operational efficiency. However, while the authors acknowledge the necessity of populating the data lake and addressing queries with relevant data, they only

state that semantic considerations are required. However, in contrast to our approach, both contributions do not establish a methodology or a blueprint for realising a semantic data integration.

In the field of model catalogs, examples can be found in various domains, such as the energy sector (González and Uslar 2012), the design of machines, particularly mechanisms for motion transfer (i.a., VDI 2727 Part 1:1991-05 1991), business process models (Yan, Dijkman, and Grefen 2012), or business process analysis models (Pidun and Felden 2010). In addition, we compare the current state of research with the actual meta-model of the Digital Shadow to identify potential gaps.

Fettke and Loos (2002) provide a blueprint for a reference model catalog independent of the underlying domain. Their blueprint acts only as a structure for a repository and does not have a claim to realise data gathering or interconnecting models. In addition, Fettke and Loos (2003) suggest a classification methodology to establish domain-specific classification schemes for models. They highlight the importance of standardised model descriptions to make the models findable in the system. Their contributions serve as a cornerstone for the general requirements of annotating models and their management, which also should be considered in creating Digital Shadows, especially in including semantics in the form of metadata.

Zarate et al. (2022) built an abstract architecture of a Knowledge to Environment platform consisting of building blocks for utilising data and model catalogs, e.g. access control, import/export tools, and version control manager. However, challenges and details for the concrete implementation of the building blocks remain open. Another high-level reference architecture for leveraging model repositories for digital twins was developed by Lehner et al. (2020). Their architecture also consists of general building blocks, e.g. model repository or communication middleware, that allow the identification of models and their connection with underlying systems to receive data. The architecture is designed for software-independent communication of the introduced building blocks. Due to the high level of detail, concrete derivations regarding their implementation are not possible, but are valuable as first foundation for our architecture. Since both contributions centralise on the implementation of a running system, utilising models with semantics and metadata is not in their focus.

In contrast, the Model Identity Card (MIC) was introduced to semantically characterise general simulation models. The MIC contains relevant properties and metadata such as a description, the datatype, and the multiplicity (SystemX 2020). On this basis, a specification for Asset Administration Shells (AAS) was built, which enlarged the previous work. The AAS acts as one implementation of digital twins (IDTA 02005-1-0 2022). Reflecting the current meta-model of the Digital Shadow, it can be identified that a semantic annotation of models is currently not present.

Including semantics in their model catalog was made by Barcelos et al. (2022). They present a FAIR Model Catalog for Ontology-Driven Conceptual Modelling Research. This model catalog aims to pave the way for efficient ontology engineering, software design, and conceptual modelling, providing a central and standard-based repository. The schema of the model catalog is built on established vocabularies, e.g. Friend of a Friend (FOAF). They use a Uniform Resource Identifier (URI) to uniquely specify models and corresponding files. Although theoretically possible, they do not consider the automatic gathering of models. Furthermore, connections to underlying databases are not part of their architecture. Bayer, Ames, and Cleveland (2021) introduce a web-based Model Catalog architecture based on open-access software. Their architecture consists of a cloud storage for the models, a Model Repository, and a Model Viewer. After receiving a list including all models from the cloud storage, the user can select a suitable model in the model repository. Finally, the model viewer fetches the model file from the cloud storage and displays the result. Although the models are defined with metadata for fetching and filtering purposes, they disregard using FAIR data descriptions, which hinders the model catalogs' scalability. Tsay et al. (2022) created the AI Model Metadata eXtractor (AIMMX) library that mines model-specific metadata of AI models from software repositories in a simple and standardised manner. The extractor fetches model documentation, python files, model definitions, etc. from different sources. To ensure a standardised description of AI model metadata, they utilise a schema based on Amershi et al. (2019), which addresses high-level information such as the application domain. The authors finally implement a web-based model catalog for AI models. This model catalog serves as a repository; hence, data extraction from underlying systems and direct model execution are not required.

In the domain of MBSE, some approaches for annotating models were already present. Jacobs et al. (2022) introduce a model classification framework to support engineers in finding appropriate models efficiently. This framework consists of three axes, i.e. the system scope, the model purpose, and the modelling fidelity. The system scope specifies the general domain models required, e.g. electric machines or bearings. The model purpose includes all models for a particular task or area, e.g. productivity or lifetime estimation. Finally, the modelling fidelity determines the required accuracy of the model,

e.g. first estimations or exact solution calculation. Based on this, Spütz et al. (2022) present a classification of simulation models for the model-based design of plastic-metal hybrid joints in the form of a SysML meta-model. This meta-model classifies the domain models by introducing a standardised structure and specifying their integration and linkage to the underlying system model so that integrating new domain models can be performed more efficiently. Similar to the Digital Shadow meta-model from Becker et al. (2021), the authors specify the intended application of the models.

Husung et al. (2023) introduce the concept of model signatures that contain meta information of the models. Those signatures define the model's field of application by defining its purpose and fidelity. Therefore, their model signatures contain the required data flow and the model's encoding information. The authors highlight the importance of standardised meta information without providing detailed elaboration. Moreover, dealing with data acquisition from underlying production systems was not part of their contribution.

Concluding the current state of research, the general concept of model catalogs and their building blocks are already established. Also, it can be identified that the related work considers semantic annotation of their models and model catalogs respectively. However, their standardisation is only based on proprietary standards, which complicates interoperable behaviour. The Digital Shadow meta-model also does not dictate a standard to be used. Thus, approaches such as the model signature currently is missing in the Digital Shadow meta-model. Furthermore, an executable Digital Shadow that addresses the challenge of different data silos is not established yet. In our contribution, we build on this related work and extend it to a first proof of concept that is independent from the underlying domain.

4. Methodology

In addressing our research question, we adopted the Design Science Research Methodology (DSRM) proposed by Hevner et al. (2004) and expanded upon by Peffers et al. (2007). This methodology was chosen for its eligibility on practically implementing theoretical research through tangible artifacts. Figure 3 illustrates the DSRM process model, enhanced with our content. Next, we offer a detailed breakdown of each specific step.

*Identify Problem & Motivate:*In the first phase of the DSRM, the problem that needs to be addressed is identified and the motivation for our research is outlined. Our current research centres on the semantic foundations of Industry 4.0, specifically focussing on establishing an interoperable shopfloor. One element is decision support systems, where suitable models and corresponding input data are provided automatically with the help of Digital Shadows, depending on a decision-makers' objective. We investigated and found that such a decision support system is not currently present. Key challenges include the absence of domain-specific model repositories and inadequate interoperability between models and source systems providing input data, hinder efficient decision-support for users. The concept of Digital Shadows has not yet been translated from high-level perspective into a more detailed architecture, fostering the requirements for a practical implementation. A significant shortcoming exists in missing requirements and building blocks that

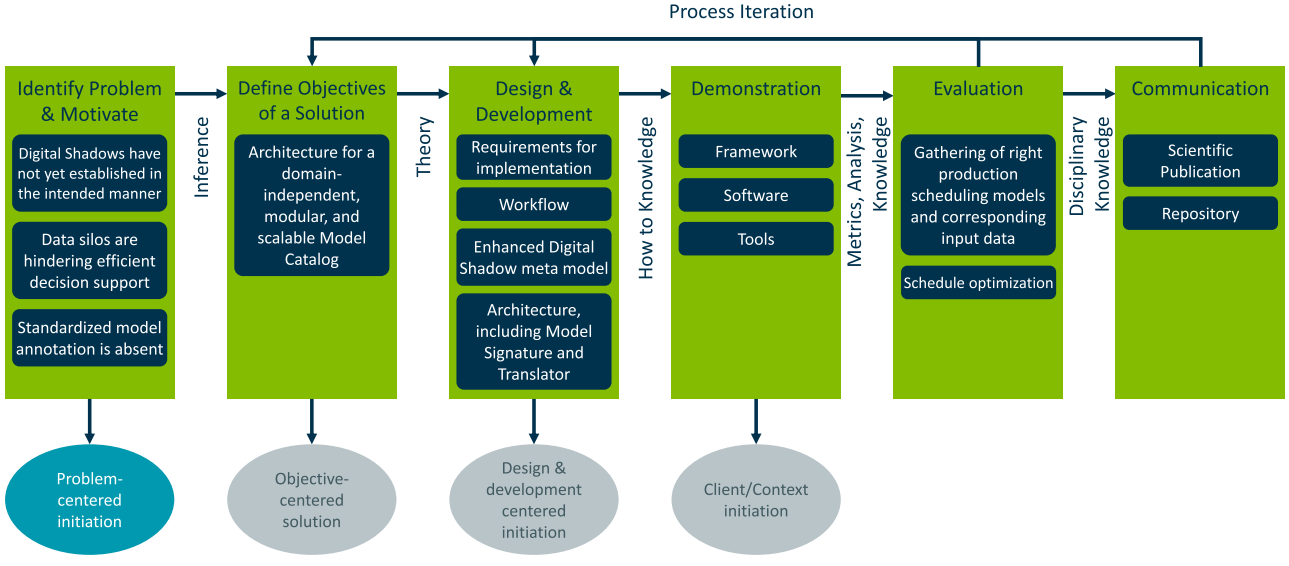


Figure 3. DSRM process model, based on Peffers et al. (2007), extended with our content.

finally establish the data flow from various source systems to the requesting models. These gaps hinder the effective use and integration of Digital Shadows into existing systems and processes.

*Define Objectives of a Solution:*In the second phase, specific objectives for a solution are defined to solve the identified problems. From a problem-centric perspective, our objective is to develop an architecture for a modular and scalable model catalog. This architecture is domain-independent and can be transferred and implemented into multiple use cases.

*Design & Development:*The third phase, Design & Development, involves the creation of artifacts. An artifact is a concrete construct designed to address a specific problem or need. It can take various forms, such as a model, method, framework, algorithm, etc., and serves as the core innovation developed during the research process. During this phase, we delineated the general workflow and engaged stakeholders crucial to the model catalog and derived essential requirements for its implementation. This artifact acts like a guideline of other users setting up a model catalog infrastructure for their applications. In addition, we have enhanced the Digital Shadow meta-model with missing but relevant classes. Subsequently, we built the model catalogs' data architecture that lists and depicts the functions in detail. The main parts of this architecture are the Model Signature, acting as a set of standardised metadata for models, and a Translator that allows a flexible connection to different source systems, also in a standardised manner. Finally, we propose a framework encompassing specific software and tools applicable to the model catalog implementation.

*Demonstration:*In the following Demonstration phase, the developed artifacts are applied in a real-world or simulated environment to show its utility in solving the problem. To validate our theoretical research, we created a 'lab of labs' demonstrator. This demonstrator allows us to test our approach through a practical use case: production planning within the Injection Molding (IM) domain.

*Evaluation:*Next, the Evaluation phase assesses the artifact's performance by comparing its outcomes to the defined objectives, identifying strengths, weaknesses, and opportunities for refinement. Here, we compared our expectations with the actual result. More specifically, we check if changes in the Model Signature, i.e. changing data points to be gathered and their source system(s), lead to the desired behaviour in the extraction flow.

*Communication:*Our research results in a collection of recommendations, architectural blueprints, and a publicly accessible GitHub[1] repository hosting the demonstrator code. These artifacts facilitate future implementations, enabling researchers and practitioners to build on our contribution.

5. Model catalog

This section presents the overall process of utilising the model catalog and the derived requirements. Therefore, we first we figure out the single stakeholders that are involved in the operative decision-making process and model the steps that have to be processed. In the following, we point out requirements for the implementation of the model catalog. After this, we present the extension of the Digital Shadow meta-model with missing classes that describes a model semantically and syntactically. Based on this, we derive a comprehensive model catalog architecture and subsequently build a framework for its possible implementation.

5.1. Model catalog sequence

A user's interaction with the model catalog involves several interfaces, gateways, and databases. Figure 4 presents a high-level view of the sequential process for accessing models from the catalog, fetching the required data, performing pre and postprocessing, and forwarding the result to the user. Subsequent sections delve deeper into individual steps, providing detailed insights and potential implementation strategies.

In the first step, users have to specify the domain of interest so a search engine can retrieve the correct model catalog from a database. This step is significant since different model catalogs contain different semantics and must be separated. If no model catalog of the domain is available, users receive a notification, leading to the termination of the process. However, this step becomes redundant if users can directly access the correct model catalog, such as when it is available as an encapsulated program. Upon obtaining the desired model catalog, users proceed to specify their purpose. This specification can manifest in various ways: predefined filters, e.g. drop-down selections or checkboxes, predefined queries tailored to specific purposes, or free-text search queries as in search engines. Based on the specified purpose and inputs, the search engine scans for models that meet the criteria. If no results match the search criteria, users receive a notification, terminating the process. Otherwise, the model catalog retrieves suitable models from the model database. Each model is distinguished by a meta file functioning similar to an identity card, the so-called 'model signature'. It contains metadata that describe the context and adds relevant information about the model. The metadata is used at the same time to help make the model more easily findable given a search query. The model catalog matches the user-specified search with the metadata of each model to retrieve the relevant ones.

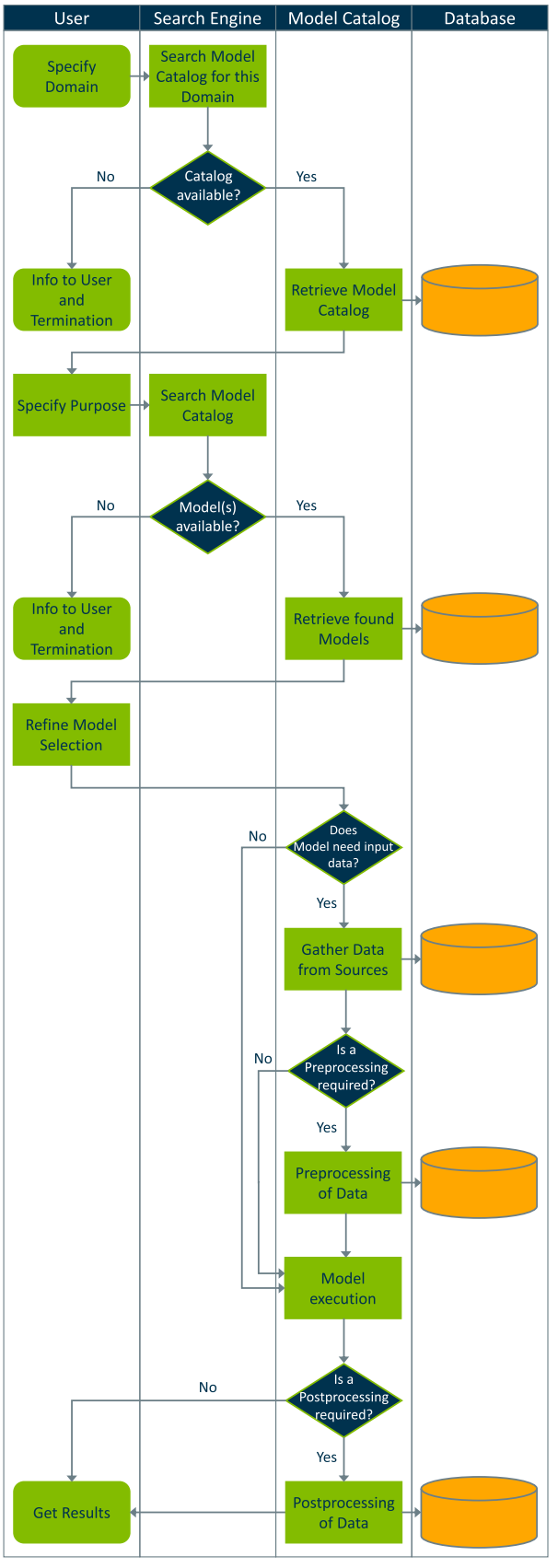


Figure 4. Model catalog sequence diagram.

Users can refine their model selection by considering only specific models if multiple options are suitable.

Subsequently, the model catalog collects required input data from the corresponding sources if the model needs data for computation, like mathematical optimisation algorithms. This process efficiently navigates multiple databases to gather essential input data. Depending on the diverse output formats from databases, preprocessing of collected data might be necessary. Ideally, preprocessing algorithms are modularised for reuse. Post model execution, potential postprocessing steps, such as reverting algorithm-generated job numbers to job names aligning with the originating ERP, may occur. Finally, the model catalog presents the results to users, who have the option to implement suggestions or decline the offered decision support.

5.2. Requirements for the model catalog

This subsection introduces the fundamental requirements derived from the general workflow illustrating the process of acquiring appropriate models for specific purposes within the model catalog. The listed elementary requirements (RQMT) are crucial considerations for formulating the framework and its subsequent implementation.

*RQMT 1: Digital availability of models.*To effectively execute models, they must exist in the virtual realm in a standardised data format, such as a Python or Java file. Unlike simulation or CAD models, often human expertise in decision-making is not formalised as algorithms, which are digitally available. We obtained this insight through a review of literature focussed on scheduling models tailored for the injection molding domain (Gannouni et al. 2024). Ensuring the digital availability of all models enables their execution and computational processing within a software system.

*RQMT 2: Standardised model definition.*The automatic gathering of suitable models in the model catalog aligns with the user's set goals. This automated process relies on a framework that defines characteristics unique to each model, serving as a crucial link between the user interface, the model catalog, and individual models. The international standard IEC 61360-1, which details data element types alongside a classification scheme, proves to be an ideal option for creating standardised model definitions (DIN EN 61360-1:2018-07 2018). This standard covers meta properties grouped into identification, semantic, administrative, and value properties, understandable by both humans and computer systems. By utilising predefined properties from IEC 61360-1, it ensures

a universal understanding and prevents the creation of varying meta properties, ensuring compatibility and scalability. Standardised definitions provide models with independence, separating them from the underlying software systems that provide input data, given that these systems follow standardised conventions (refer to RQMT 4).

*RQMT 3: Input data from established data sources.*For models reliant on input data, it is imperative to establish connectivity to accessible data sources that contain the required information as data points, e.g. from source systems such as ERP. Directly sourcing necessary input data from these systems offers several advantages: it ensures the use of actual information, reduces the risk of errors in copying and pasting, and expedites the data input process. A crucial criterion for these data sources is the presence of an application programming interface (API) facilitating straightforward connections to the data source.

*RQMT 4: Standardised input data.*Similar to RQMT 2, standardised definitions are crucial not only for models but also for input data, specifically in the form of unique identifiers. These identifiers act as a bridge between the model catalog and the source system, ensuring accurate data retrieval, as they are present for every term. Additionally, utilising standardised terminology enhances the universally applicable understanding within a particular domain and ensures a consistent nomenclature. Moreover, they enable interoperable behaviour between models and source systems, allowing the automatic gathering of necessary input data. The use of identifiers eliminates the need for manual configuration to link data points from source systems to models. To enable this automatic data transfer, a configured middleware is required (refer to RQMT 5).

*RQMT 5: Established middleware as system connector.*Enabling the automatic import of accurate data into selected models necessitates the presence of a middleware, a fundamental component for automating data gathering for model execution. When all elements, such as the user interface, model catalog, models, middleware, and data, are described using common and standardised metadata, seamless communication and efficient data transfer between these elements become achievable. An inherent challenge arises from the existence of multiple data sources, each with its distinct API, resulting in varied connectivity options. Consequently, an individual middleware is required for each data source to facilitate their connection. However, due to a standardised definition of data points, the middleware can be initially defined and then reused with minimal configuration adjustments, like modifying user login credentials. Vendors could offer such middleware and corresponding updates for commercial software systems such as ERP. Once the middleware is established, specifying the precise path to location of the data points for users becomes unnecessary.

*RQMT 6: Consideration of pre- and postprocessing.*To effectively utilise the selected models with gathered input data, ensuring a harmonisation between the input data format and the format utilised within the model may be necessary. One notable instance involves aligning dates, which might exist in various formats (e.g. DD.MM.YYYY or MM/DD/YY). Similar scenarios pertain to scheduling models. These models often calculate start dates and job durations relative to a reference point (e.g. the start date of job A is 0, the start date of job B is 18, and so forth). This necessitates preprocessing where data is transformed into relative values, and postprocessing after model execution to convert resulting relative values back into date formats for use by production schedulers. Similar to predefined middleware that facilitates asset connections, modules can be established for standardised, repetitive preprocessing and postprocessing tasks.

*RQMT 7: Dealing with models as black boxes.*Users, focussed solely on receiving decision support for specific problems, do not necessitate insight into the precise algorithms within a model. Therefore, both the model catalog and the content within these models can operate as black boxes for users, given their sole focus on specifying the purpose requiring decision support. The development and definition of models might lead to the emergence of new business areas centred around standard models (e.g. lot sizes) or 'model-as-a-service.' Additionally, a Digital Shadow itself does not require insights regarding the efficiency of a model, e.g. its runtime, as it primarily focuses on the accuracy and relevance of the decision support provided.

*RQMT 8: Ability for adjusting model selection.*Once the purpose is specified, the Digital Shadow gathers all relevant models from the model catalog that match the selection criteria. Users should have the ability to refine their model selection by deselecting individual models. An example of this scenario could be enhanced knowledge that users have about a special situation that is not considered within a model and thus leads to its exclusion. Thus, the model catalog should allow users to refine the selected models.

*RQMT 9: Leaving ultimate decision-making to users.*In certain domains and for specific decision-making requirements, there might be multiple optimal solutions available for a given problem, such as in production scheduling scenarios. In these cases, the Digital Shadow must display all feasible optimal solutions, enabling users to select the most suitable option for implementation. It is important to note that the Digital Shadow only serves as decision support and does not directly impact the

actual system. Implementation decisions are made exclusively by users or through the execution within a software system by users.

*RQMT 10: Handling dynamic and heterogeneous manufacturing setups.*The Digital Shadow must provide robust mechanisms for real-time data synchronisation and latency management to maintain accuracy and efficiency in fast-changing production setups, while also ensuring seamless integration of heterogeneous data sources and an adaptive response to unexpected changes in the manufacturing environment. This means that the architecture is not tailored to one specific domain but can address different ones.

In essence, establishing an automated process for model and data gathering across diverse sources using Digital Shadows relies fundamentally on standardised definitions for individual components. The IEC 61360-1 standard offers properties for delineating the characteristics, semantics, and identification of models and data points in the metadata form. This standardised definition ensures the scalability and independence of the model catalog from a single use case within a single company. Finally, only users affect the real system, as the Digital Shadow is only present for giving decision support.

5.3. Model catalog data architecture

Before we introduce the comprehensive model catalog data architecture, we first provide an update of the Digital Shadow meta-model.

5.3.1. Extension of the digital shadow meta-model

The current version of the Digital Shadow meta-model permits the definition of a `Source`'s properties through `System Properties`. However, existing models are not defined semantically, resulting in the inability of Digital Shadows to automatically identify suitable models based on a specified `Purpose`. To address this limitation, we extend the meta-model to incorporate this capability. Figure 5 presents the updated meta-model.

To ensure that a Digital Shadow selects appropriate models for a given `Purpose`, two model specifications must be defined: semantic annotation and syntactic annotation. The semantic annotation is implemented through the `ModelSignature`. This component enables the Digital Shadow to determine what models are suitable for fulfilling a specified `Purpose`, which data are required as input for model processing, and where this data are located. Additionally, the system responsible for executing the model requires information on how the model can be processed, such as the programming language or execution environment. This information is specified within the `ModelDependencies`. The repository that stores and manages the models is referred to as the `ModelCatalog`. The next sections will present the possible structure and content of the `ModelSignature` and `ModelDependencies`, along with their integration into a comprehensive data architecture for a model catalog.

5.3.2. Data architecture

After pointing out the general requirements and their general solution approach in combination with the expanded meta-model, we translate them to a data architecture independent of underlying software and programs. Figure 6 illustrates the data architecture.

We built the architecture by establishing discrete building blocks, each responsible for a specific, encapsulated function. One aim is to achieve a highly modular and scalable architecture, minimising programmers' configuration efforts. Therefore, we organise the functions into different containers and identify three types of functions:

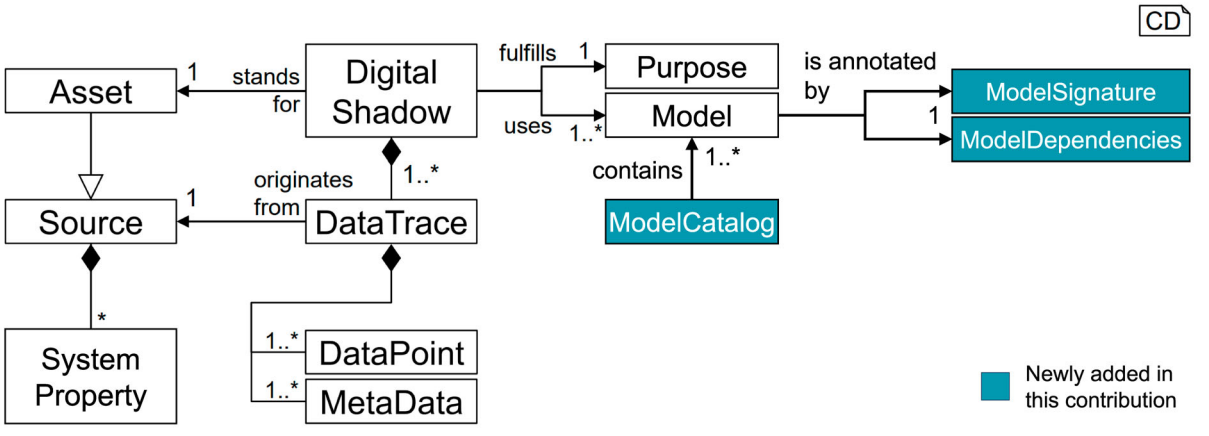


Figure 5. Updated digital shadow meta-model, extended with classes for semantic model annotation.

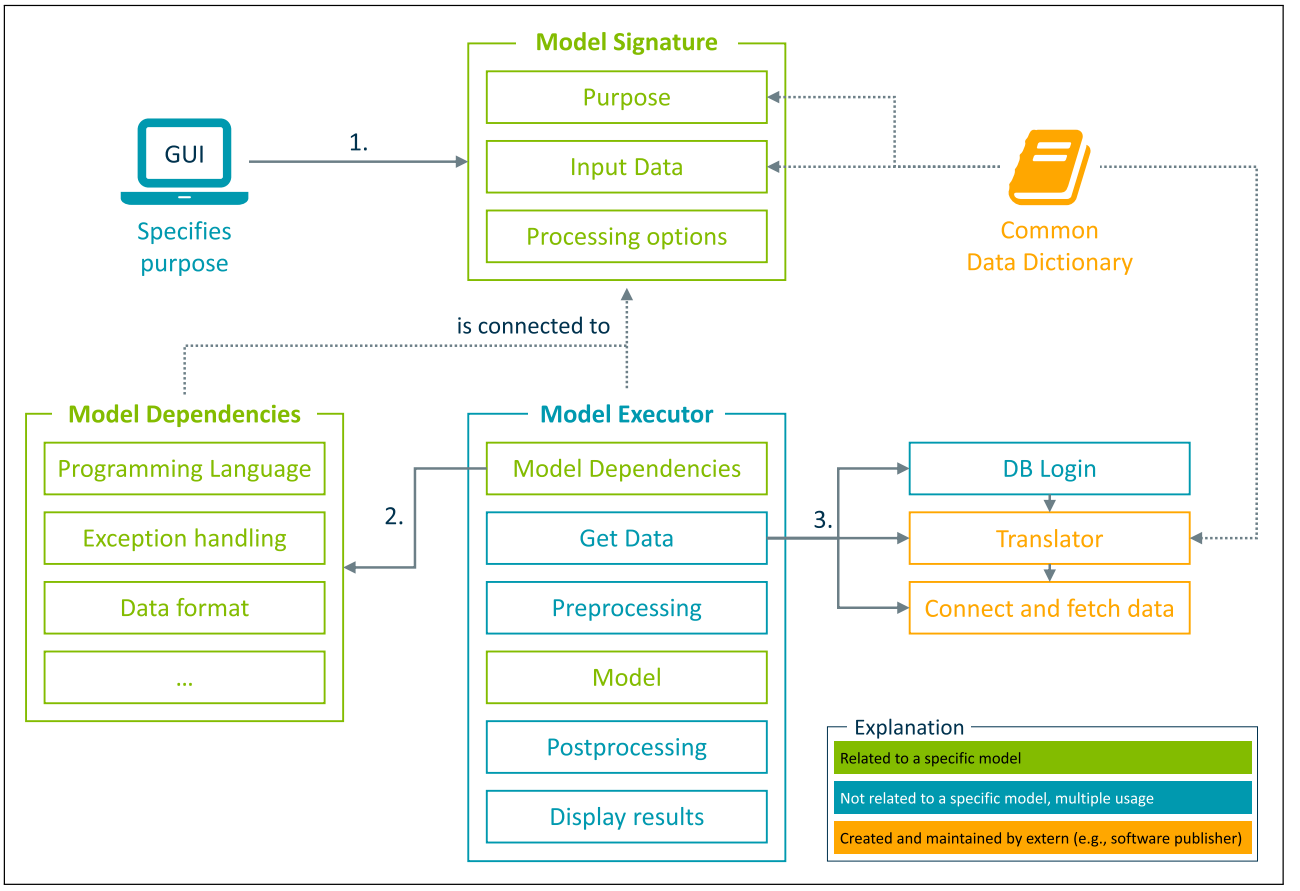


Figure 6. High-level data architecture of the model catalog.

(1) Functions related to a specific model, e.g. its input data.
(2) Functions unrelated to a specific model, e.g. user credentials for a database.
(3) Functions created and maintained by external organisations, e.g. a database connector.

In addition to the `Model`, which encompasses the algorithm, three distinct containers facilitate the accurate processing of the model. The `Model Signature` (according to Husung et al. 2023) encapsulates all semantic data relevant to the model, linking it directly to a specific model. Within this container, the `Purpose` classifies the application domain of the model, the `Input Data` defines the necessary data for computation and its sources, and `Processing options` determine whether preprocessing of input data or postprocessing of the algorithm's outcome is required, referencing the corresponding file. A single model may possess various `Model Signatures` owing to differing databases or the necessity for distinct preprocessing and postprocessing methods depending on the database's output format.

The `Model Signature`, particularly the `Purpose`, is the trigger for aggregating suitable models. Initially, an operator specifies the purpose (c.f. Figure 6, 1.), denoting the required decision support, using a Graphical User Interface (GUI). We assume that an adequate GUI is previously developed by programmers. Upon user selection, a search engine scans the `Model Signature` database within all model catalogs, comparing the user's input with defined `Purpose` values to satisfy the search purpose. To ensure a scalable model catalog, standardising the GUI's input forms and `Purpose` values is crucial. Ideally, these values are centrally registered within a common data dictionary, providing unique identifiers and ensuring universal comprehension across the community. Consequently, when new models aligning with this standardised definition are added to the model catalog, the search engine can seamlessly retrieve them without requiring code adjustments.

Following the identification of suitable models by the search engine, the verification of technical capabilities for model processing is necessary (2.). This verification is handled by the `Model Dependencies` container,

encompassing all technical details required for a model's processing within a system. It describes the model's `Programming language`, `Data format`, or `Exception handling`. Since these characteristics are intrinsic to a model, only one `Model Dependencies` container exists per model, invariably linked to a specific model.

The `Model Executor` serves as the pivotal element executing all required functions. Upon validating the technical capabilities for model processing, the `Get Data` function (3.) within this container undertakes the necessary steps for data retrieval. Initially, it gathers login data for the database (DB) specified in the `Model Signature` through `DB Login`. This function, housing general database endpoints and user credentials, can be reused for other models accessing the same database. As `DB Login` contains sensitive user data, its handling requires special attention to data security, which is beyond the scope of our contribution. Subsequently, the `Get Data` function requests the necessary data as outlined in the `Input Data` within the `Model Signature` container. Similar to standardising the `Purpose` for defining a model's application area, standardisation of `Input Data` within a shared data dictionary is crucial for interoperability.

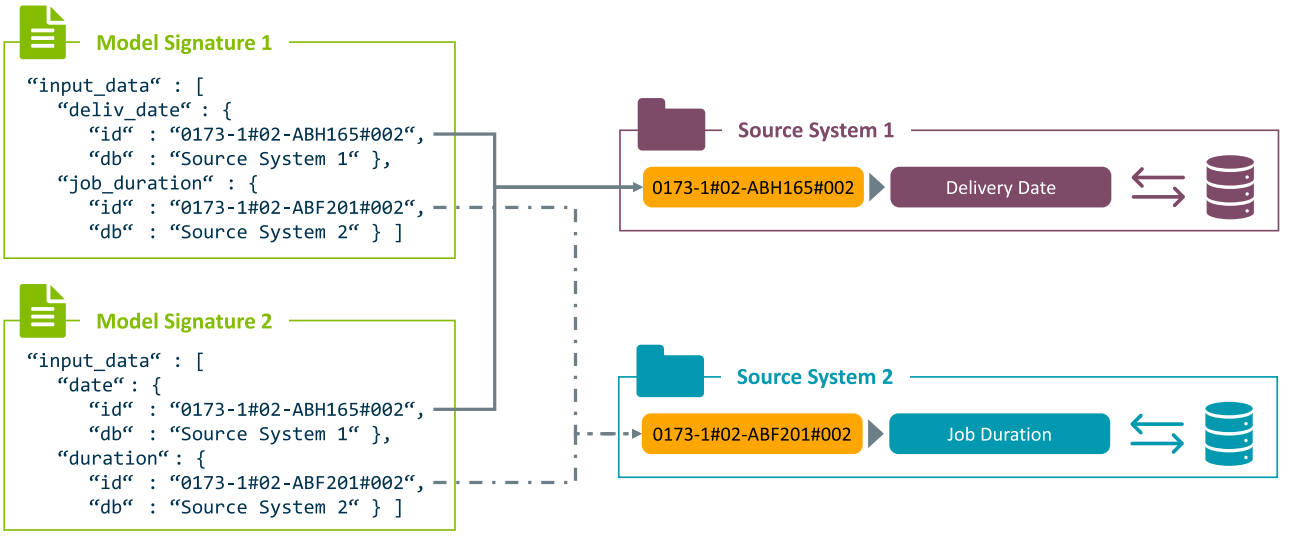
A `Translator`, acting as a middleware, is one of the key elements for ensuring the interoperability, modularity, and scalability across different systems. Since the `Translator` is a central element in achieving interoperability, we highlight the translation process in Figure 7.

The `Translator` should be supplied by the vendor of a software system and takes the form of a basic translation table. It contains, on one side, the centrally registered format of the data point in the form of a unique identifier, and on the other side, it provides its corresponding name in the proprietary source system. The first key function of the `Translator` is to convert these identifiers into the format of the source systems, independent of the designations of the data points that were made in the model signatures (cf. Figure 7(a)). For example, the data point 'Delivery Date' refers to the date on which the product was received by the customer. In Model Signature 1, this data point is called 'deliv_date', while in Model Signature 2, the data point with the same meaning is labelled as 'date'. The `Translator` standardises these varying designations, making them independent of the underlying source system. Conversely, the specific naming conventions within different source systems are irrelevant; the same attribute may be termed 'Delivery Date' in one system and 'Shipping Date' in another. To address these inconsistencies, common data dictionaries are established, housing properties in a standardised manner, and augmented with metadata. These dictionaries consolidate different names of a property under a single identifier. For instance, in ECLASS,[2] a notable common data dictionary, the property 'Delivery Date' might be assigned an identifier such as 0173-1#02-ABH165#002. Utilising these identifiers eliminates the need to use the specific name given by the source system.

The second key function is to integrate multiple source systems (cf. Figure 7(b)). As the required model input data may be stored across different systems, such as delivery dates and job durations, it is essential to automatically address the correct source systems. This can be achieved by assigning a unique identifier to each source system, for instance, 'Source System 1' and 'Source System 2'. For modularity and scalability, new source systems can be integrated as illustrated in Figure 7(b), where a new 'Source System 3' replaces the previous ones. In this scenario, the model catalog architecture only requires updating the source system's 'id' in the model signature file.

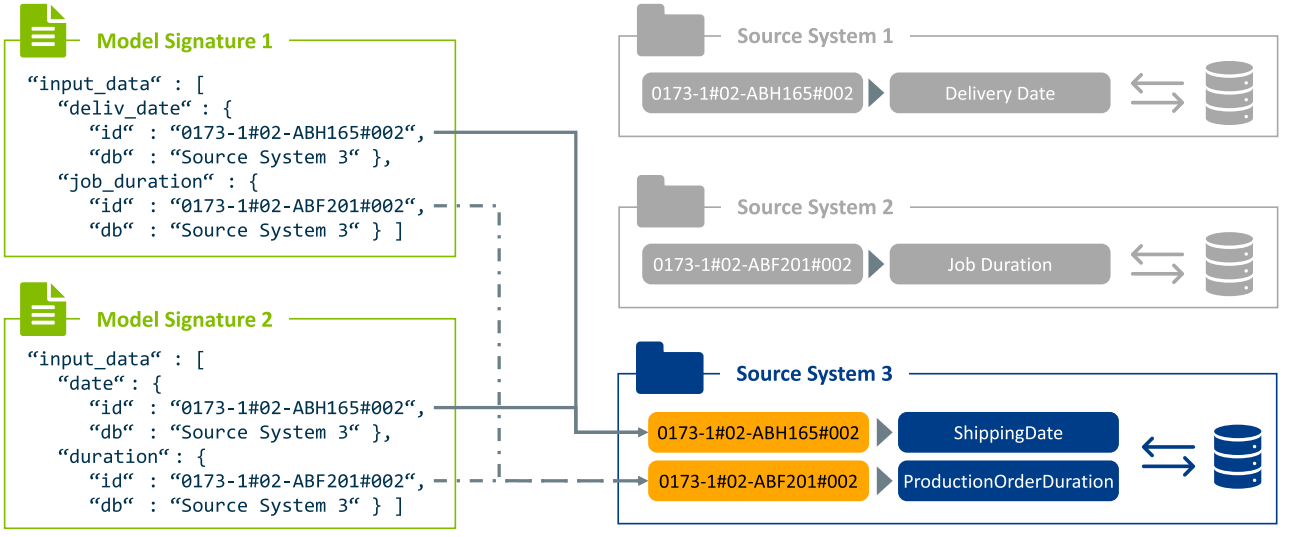
Subsequently, the `Connect and fetch data` container working as the API of the software system establishes the database connection based on `DB Login` and retrieves the required input data. The application of a `Translator` and a `Connect and fetch data` module within a specific software system abstracts users from the necessity of comprehending the underlying database structure, allowing them to treat the source systems as black boxes. In an Industry 4.0 context, these modules are typically developed and maintained by software system vendors. The sole responsibility of users is specifying the correct identifier associated with a data point from a common data dictionary and identifying the source system. This implies that utilising the `Translator` in conjunction with a `Connect and fetch data` container, provided by vendors, promotes a modular and scalable configuration. This is because the connection to underlying source systems can be established with minimal effort by the user.

Based on the `Processing options` within `Model Signatures`, the executor runs necessary `Preprocessing` before engaging the model itself. Finally, after potential `Postprocessing`, the `Display results` function furnishes decision support for operators to either implement or disregard the outcomes. With the help of the architecture, Digital Shadows now rely on a fundamental framework for overcoming data silos and process models regarding a predefined purpose automatically. However, since Digital Shadows never affect the system, a function of our architecture that implements the results automatically is absent. In our architecture, the execution steps for each model are uniform. As the logic for defining required input data or underlying databases is externalised, the `Model`



(a) Conversion of identifiers into the source systems' format

Figure Alt Text: The standardized IRDI properties and the designation of their storage location within the model catalog facilitate the translation process from the standardized format to proprietary formats for Source System 1 and Source System 2.



(b) Integration of a new source system

Figure 7. Key functions of the translator. (a) Conversion of identifiers into the source systems' format and (b) Integration of a new source system.

`Executor` is not bound to a specific model and can be arbitrarily reused. In Section 6, we illustrate this architecture with a concrete example.

5.4. Model catalog framework

In this section, we propose a framework tailored for implementing the model catalog and its seamless integration with Digital Shadows, as depicted in Figure 8.

To facilitate user interaction, we adopted Angular[3] to create the GUI. Leveraging Nebular's pre-built GUI library, we tailored a template that suits our needs. For the backend infrastructure, we harnessed Flask,[4] a Python-based web framework. Efficient model search and retrieval of necessary data were orchestrated through elasticsearch.[5]

All essential file categories, encompassing models, meta files associated with the models, scripts for pre- and postprocessing, middleware, and configuration files, were housed within Minio,[6] an object store. Minio's structure, enabling storage within designated 'buckets' for each file category, aligns well with our setup.

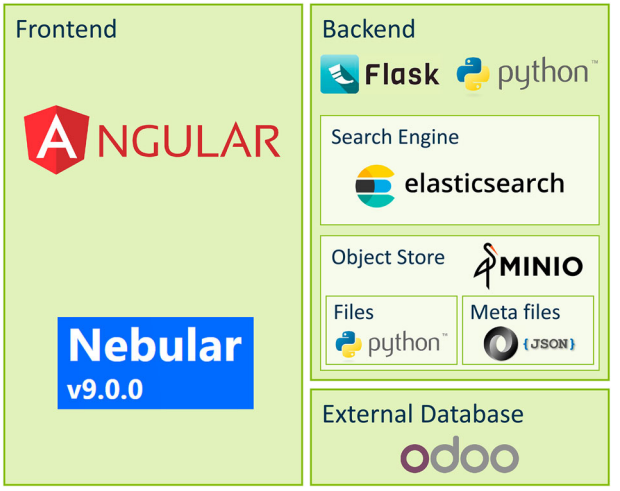


Figure 8. Model catalog framework we used for the implementation.

For a realistic demonstration mirroring real-world applications, we integrated odoo,[7] a widely recognised ERP system within the industry. Beyond its comprehensive transactional capabilities for managing shopfloor activities, odoo furnishes a user-friendly API for seamless data exchange and offers extensive customisation options.

A key advantage of our proposed framework is its scalability. All components have been containerised using Docker, allowing for seamless deployment on cloud infrastructures, whether on-premises or in enterprise environments. This containerised architecture ensures that the framework can dynamically scale based on demand, efficiently handling varying volumes of data, models, and processing requirements. By leveraging container orchestration solutions such as Kubernetes, the system can automatically adjust resources, ensuring optimal performance and resilience.

Our implemented conceptual architecture, providing a comprehensive model catalog setup, is accessible on GitHub,[8] offering insights into the implementation details and enabling further exploration.

As already highlighted in Section 2, the proposed architecture and the derived framework focus on the model, its purpose, and system properties. Therefore, it allows the seamless integration into Digital Shadows through the represented meta-model.

6. Real-world application: production scheduling of an injection molding shopfloor

Decision support from a Digital Shadow using the model catalog's architecture and framework we introduced in the previous sections is independent of the underlying manufacturing process or the domain. Each manufacturing or decision process is individually characterised by varied models that can be easily integrated with their model signatures, which eases their findability and accessibility.

In this section, we instantiated the proposed architecture to show which dependencies and requirements regarding a standardised description are necessary and where to apply them. Therefore, we present a real-world application in the form of a single-objective production scheduling problem within the injection molding domain to demonstrate how a decision support system is provided with the required data for processing decision support. Here, we use a model catalog that we built in a previous contribution, which proposes a blueprint for scheduling models (Gannouni et al. 2024). More complex scenarios, such as multi-objective scheduling or other scenarios, e.g. data gathering to provide them for other applications, can be addressed similarly by integrating models and their corresponding model signatures, so it is applicable to various applications and industries.

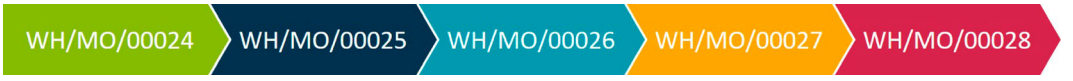
6.1. Use case overview

In this scenario, a production scheduler strives to arrange the given jobs optimally. Depending on the production scheduler's purpose, various optimal sequences of jobs can be identified. For instance, prioritising the minimal makespan for completing all jobs might yield a different sequence compared to prioritising the reduction of deviations in job completion times from customer due dates. In addition, the shopfloor configuration and constraints play a significant role in production scheduling since they influence the possible number of schedules. For an unambiguous definition of a production scheduling problem that considers different machine environments, constraints, and objectives, Graham et al. (1979) introduced a three-field notation in the form of $\alpha \mid \beta \mid \gamma$. Here,

- α: stands for the machine environment, e.g. single machine or flow shop,
- β: characterises the job's constraints, e.g. sequence-dependent setup times
- γ: defines the objective function, e.g. minimising the makespan.

In this specific scenario, the objective is to determine the optimal job schedule for a single-machine environment. The primary goal is to minimise the maximum makespan while considering sequence-dependent setup times, denoted by the previously introduced Graham notation $(1 \mid s_{i,j} \mid C_{max})$. The graphical representation in

Figure 9 visually depicts the entire process. Figure 9(a) displays the initial job order as present in the ERP system. Meanwhile, Figure 9(b) shows the job duration, and Figure 9(c) illustrates the setup times. Here, we assume a given pre-established setup matrix valid for our use case (typically, in practice, the analysis involves a comparison among different products rather than focussing on the specific job names[9] originating from the ERP system. For clarity and better comprehension, we have utilised the job names in this setup matrix). Finally, Figure 9(d) displays



(a) The initial schedule

Figure Alt Text: A sequence diagram illustrating five jobs arranged in their initial order.

Job name [-]	Duration [time units]
WH/MO/00024	360
WH/MO/00025	240
WH/MO/00026	180
WH/MO/00027	210
WH/MO/00028	1000

(b) Duration of the jobs

Figure Alt Text: A list of job names from the demonstrator database, along with their respective durations in time units.

from \ to	WH/MO/00024	WH/MO/00025	WH/MO/00026	WH/MO/00027	WH/MO/00028
WH/MO/00024	-	130	145	140	120
WH/MO/00025	45	-	55	50	55
WH/MO/00026	60	45	-	30	35
WH/MO/00027	55	45	55	-	60
WH/MO/00028	130	125	135	125	-

(c) Matrix of setup times for the products

Figure Alt Text: A setup matrix detailing the time required for transitioning from one job to another.

(d) The optimized schedule

Figure 9. Use case data for scheduling an injection molding shopfloor. (a) The initial schedule. (b) Duration of the jobs. (c) Matrix of setup times for the products and (d) The optimised schedule.

the optimal schedule after model execution. The scheduling optimisation corresponds to solving the Traveling Salesman Problem.

The initial schedule processing, as depicted in Figure 9(a), results in an overall makespan of 2,265 time units, with non-value-adding setup activities consuming 275 time units. Examination of the setup matrix presented in Figure 9(c) reveals that the setup duration from product A to product B may differ from the reverse transition. This discrepancy is a common occurrence in setup processes of injection molding machines, caused by temperature adjustments and raw plastic material changes during cleaning procedures (Burghoff 1983; Hüttner 1979). The optimised schedule (cf. Figure 9(d)), prioritises minimising the maximum makespan, effectively reducing setup times for the specified jobs. This refined schedule reflects a makespan of 2230 time units, marking a 13% reduction of the overall setup time compared to the initial schedule.

6.2. Instantiation of the architecture

In order to illustrate the practical application of our architecture (cf. Figure 10), we proceed by instantiating its functions. We display the necessary files for implementation and present their exemplary content using pseudo-code, employing the use case previously introduced. The outcome of this instantiation for the real-world application scenario introduced before is represented in Figure 10.

The GUI provides the options for selecting the purpose in the form of the Graham notation elements. However, the values of these `purpose_properties` are not yet centrally registered in a common data definition. Therefore, in our prior work, we standardised the Graham notation elements and assigned them unique identifiers (Gannouni et al. 2024). For property identification, we use the Internationalised Resource Identifier (IRI) since it ensures unique and simple identification (Fensel, Kerrigan, and Zaremba 2008). As our research originates from the Cluster of Excellence 'Internet of Production' at the RWTH Aachen University, we determine *https://iop.rwth-aachen.de/PPC/1/1/propertyName* as a unique identifier according to RFC 3987 (Berners-Lee 2005; VDI 2193 Part 1:2020-04 2020). Here, PPC stands for Production Planning and Control, while the numbers indicate the version and revision of the property.

After the user has specified the shopfloor conditions on the GUI, the search engine searches the `purpose_properties` (c.f. Figure 10, line 1) of all model signatures. In our use case, three `purpose_properties`

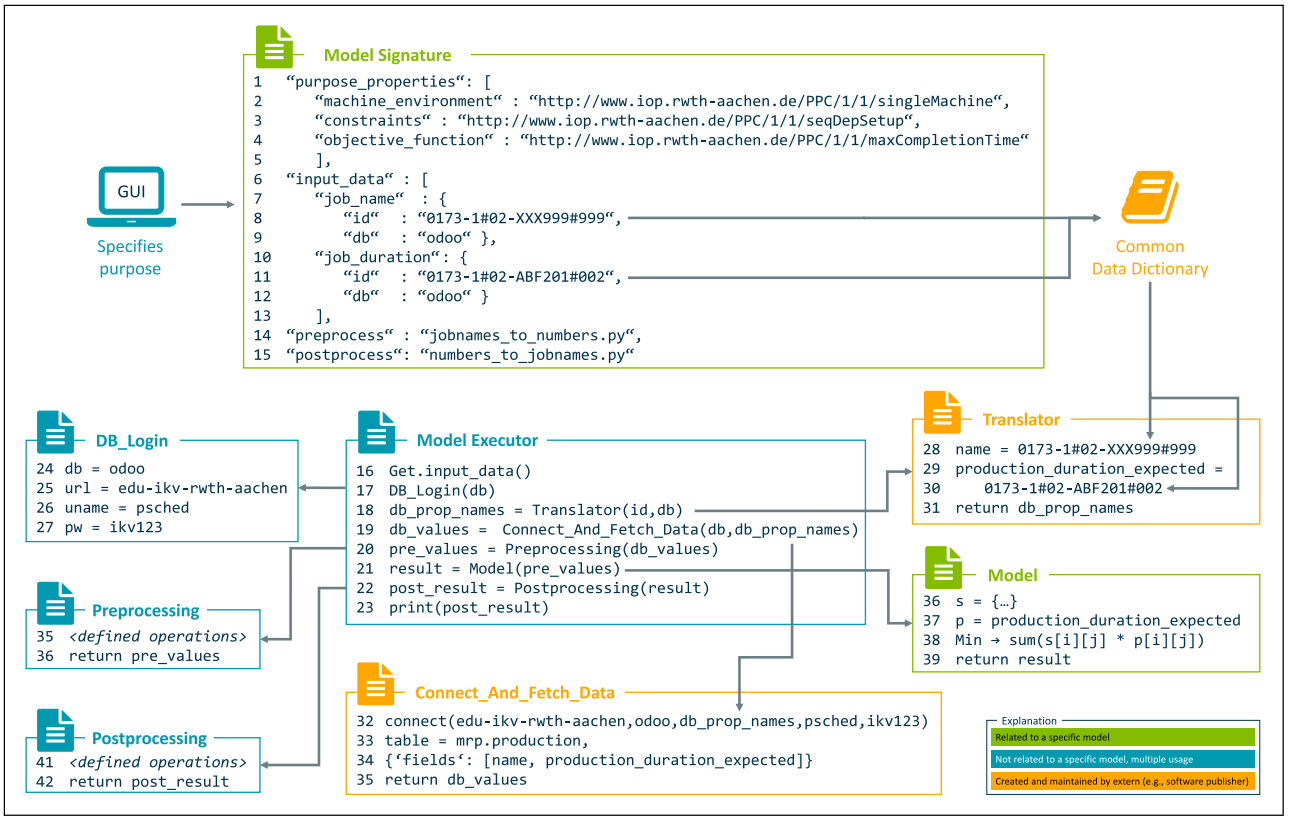


Figure 10. Instantiated data architecture using pseudo-code.

(2–4) are relevant that representing the shopfloor configuration. The search engine retrieves all models matching the input selection based on the purpose. Now, the model executor receives the information (16) regarding the input data (6), i.e. the `job_name` and `job_duration`. The designation of these variables (7 and 10) is freely selectable since they do not affect further processes. We recommend using the same designation as in the model to ensure a uniform appearance that reduces misunderstandings. In contrast, the values in (8–9) and (11–12) must not be freely chosen. Here, (8) and (11) contain the identifier from a common data dictionary, which are the basis for an interoperable model catalog, using by the translator. Additionally, odoo was specified as a source for the corresponding data points. Using this information, in the next step (17), the login credentials for the odoo database are fetched.

Since odoo is not equipped with standardised property names and identifiers, a translator is required to translate the IRI of the required input data into the odoo format. Here, the importance of standardised properties can be identified as the identifier in (8) and (28) as well as (11) and (30) matches so that the translation process can proceed without human intervention. In (31), the translator returns the database property names, so fetching the required data automatically is possible in the next step. Besides the odoo property designation, the previously collected information of the odoo login credentials is used to establish the connection to odoo and fetch the right data. In our use case, pre- and postprocessing of the data is required. The corresponding files are also specified in the model signature (14) and (15). During preprocessing, the fetched job names from odoo are transformed into integer numbers. The reason is that the models' optimisation algorithm in (38) interprets the jobs as continuous integer numbers, starting with one. Since the odoo job names follow the pattern WH/MO/*number*, the algorithm cannot handle the odoo format. We directly integrate the setup matrix (37) into the production scheduling model for our demonstrator. However, a separation could be advantageous to maximise the modularity of this implementation. In return, after the model has calculated the optimal schedule, the postprocessor has to translate the job designation into integers back to the original odoo format so a human can rearrange the jobs after the result is published (23).

Summarised, we demonstrate that with the help of standardised properties, a modular model catalog that serves as decision-support for operators can be implemented. In addition, the architecture is designed for low setup and maintenance effort, so the barriers to integrating the model catalog are low. In the following subsection, we show the real implementation of this demonstrator.

6.3. Demonstration

When accessing the web interface, users first have select the purpose, the Digital Shadow has to fulfill. As our demonstrator focuses on production scheduling, decision makers select the shopfloor conditions following Graham's notation elements. Referring to the use case description in Section 6.1, the input parameters for this demonstration are Single machine (1), Sequence dependent setup times ($s_{i,j}$), and Makespan (C_{max}), as shown in the initial screen in Figure 11.

After selecting shopfloor conditions, the search engine retrieves models aligning with the corresponding purpose that is defined in the model signatures (Figure 12).

Simultaneously, a connection to the underlying data base (in our case, odoo) is established, offering real-time visualisation of jobs currently active in the system. The real-time data can be seen in the right area. This stage enables the production scheduler to manually refine the identified models in the left area). By accessing the 'Details' button, the content of the corresponding model signature file is revealed in the middle area. For our demonstrator, we use odoo 15.0+e (Enterprise Edition). Additionally, data such as job duration, quantity, and status are readily accessible. Executing the model triggers the algorithm to compute the optimal job order based on the provided ERP data (Figure 13).

The pertinent ERP system information reappears in the left box. In the right box, the production scheduler receives decision-support in the form of an optimal job order, which is concomitant to the expected result pointed out in Figure 9(d). As the Digital Shadow operates independently of the real system, the production scheduler is responsible for executing job rearrangements or rejecting the proposed order. Additionally, the production scheduler gains meta information, including the considered jobs from the ERP, the model name, the solution status, and the objective.

Successful execution of the algorithm mandates preprocessing of input data. As the algorithm processes integer values representing individual jobs rather than job names, converting job names to integers is necessary. Conversely, the production scheduler necessitates job names to identify the relevant jobs in the ERP, thus requiring conversion of integer values back to their original job names. The tools for pre- and postprocessing are automatically gathered and executed, requiring no intervention from the production planner. For those interested in the algorithm's insights, a comprehensive execution log is available. In our use case, the execution log documents the operations conducted by the optimisation models and aids in the development of solutions and the debugging in the case of inconsistencies.

Figure 11. The initial screen of the GUI where the production scheduler can specify the purpose according to the Graham notation.

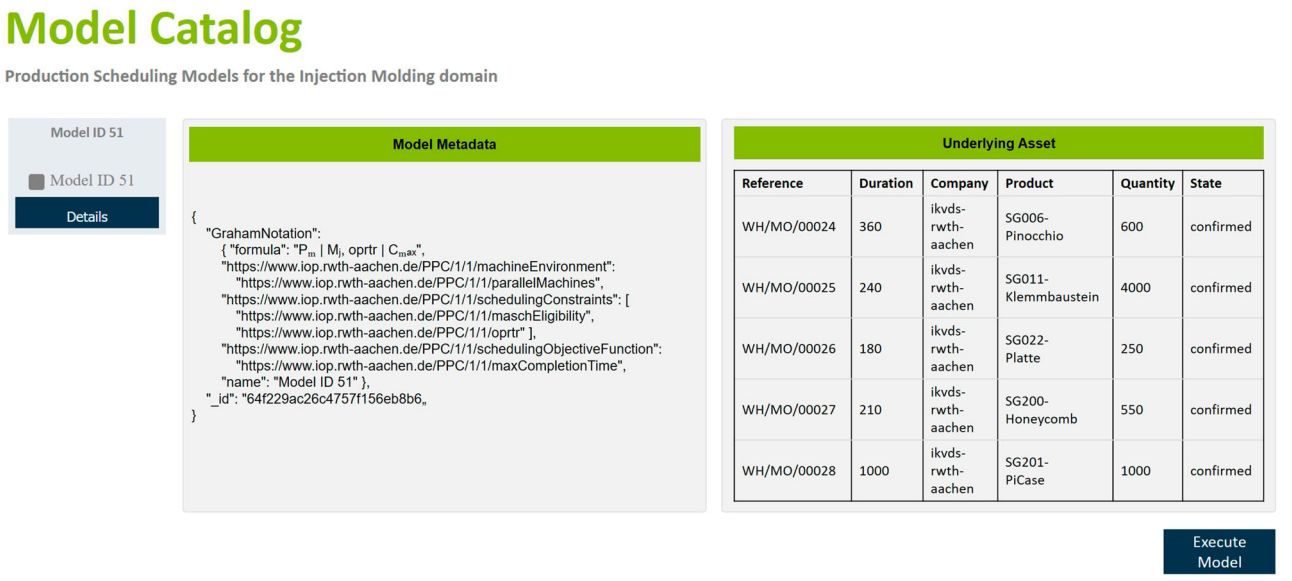


Figure 12. Details of the selected model and live data of the jobs from the ERP.

In comparison to other approaches, our architecture accelerates decision support for the given scheduling problem. Users are no longer required to identify an appropriate model concerning their scheduling objectives, log in to the ERP system, search for the necessary input data from each production order within the odoo system, or manually input these values into a model interface. Our architecture offers substantial time savings and reduces the occurrence of errors and media breaks, depending on the number of production orders and the user's familiarity with the odoo ERP system.

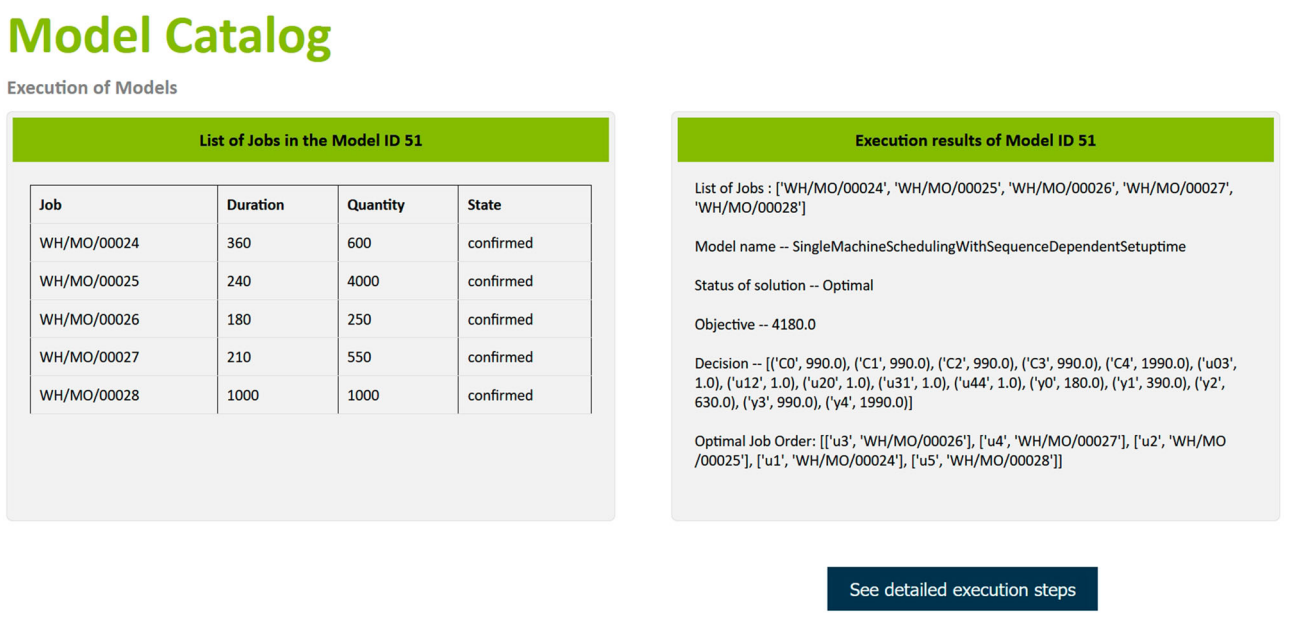


Figure 13. Result in the form of an optimal schedule.

7. Discussion

Successfully running our designed decision-support system requires several key prerequisites, which we discuss below. Additionally, we provide solution approaches for the single points we discuss.

Standardised domain description required: The scalability and modularity of our decision support architecture rely on a unified, standardised domain description. Establishing this requires a common data dictionary to ensure FAIR (Findable, Accessible, Interoperable, Reusable) data properties (Wilkinson et al. 2016). A crucial aspect involves incorporating unique identifiers, such as the International Registration Data Identifier (IRDI) or IRI, within the models' meta files to ensure the retrieval of relevant data points. Moreover, software vendors, e.g. ERP providers, also should equip their systems with standardised properties or at least have to provide and maintain middleware in the form of translators that map between IRDI or IRI notation and vendor-specific formats. Here, the presence of comprehensive Common Data Dictionaries (CDD) is pivotal for automatic and interoperable data extraction from subsystems such as ERP, ensuring data integrity through standardised definitions. Regarding a standardised domain description, we anticipate that the content of CDD will expand soon since multiple initiatives currently support their enhancement in multiple ongoing initiatives. For our domain, we have already specified the domain properties, whereby some of them are currently under standardisation as IDTA specification and will be available at ECLASS, allowing reuse across industries from injection molding to PPC.

Standardised model syntax: The models adhere to a particular syntax necessary for accurate processing. Within this contribution, we present an initial executable syntax. Nevertheless, regardless of whether this proposal is accepted or another is chosen, achieving consensus on a definitive structure is essential for modularity. We believe that if influential companies in the market align on a single syntax, it is likely to gain widespread adoption, potentially evolving into an industry standard or guideline.

Consideration of semantics: When a Digital Shadow gathers data from different systems, semantic consistency is crucial. Harmonizing units, such as weights or lengths, is critical to prevent inaccuracies and wrong results. To overcome this challenge, a preprocessor can validate that the input data or digital twins, which comprise metadata such as the Asset Administration Shell (AAS), are applicable. Using the AAS is concomitant with standardised metadata regarding IEC 61360-1, which also specifies the unit of a measure. Furthermore, the AAS ensures fair data principles since the contained properties have originated from a CDD (Marcon et al. 2018).

Creation of pre- and postprocessing files: Pre- and postprocessing are critical for ensuring that algorithms and production schedulers correctly interpret input and output data. The number of necessary files depends on the specific preprocessing and postprocessing tasks. Ideally, these files should be reusable across multiple instances. For example, in our demonstrator, transforming job names into numerical values serves as a reusable preprocessing step. As Industry 4.0 evolves, we anticipate increased reliance on model-driven decision making,

creating opportunities for new business models, such as model-creation-as-a-service. Platforms like the odoo app store[10] illustrate this trend, offering third-party applications and highlighting the potential for a growing marketplace centred on model-based tools and services.

Presence of model files: The models and algorithms that should be executed have to be present as a file, which is compatible with the model catalog architecture. In our previous research, we provided a foundational basis for the understanding of standardised scheduling models. Again, providing models taking into account standardised notation elements can also be a business model that emerged through Industry 4.0 (Mukhopadhyay, Pingali, and Satyam 2022).

Finding a suitable model classification: The model catalog we introduced is tailored exclusively to production scheduling, primarily emphasizing the Graham notation within its GUI. In our scenario, defining the Digital Shadow's purpose for gathering the right models and data was straightforward since scheduling models can be characterised commonly and explicitly using the Graham notation. However, since different domains contain different purposes, finding a suitable model classification can be challenging. To address this, experts from different companies can participate in workshops to discuss and establish a common model classification.

Computing complex problems: Due to numerous possible solutions within job scheduling, this task is highly complex and requires long computational times, especially when aiming to solve a given problem optimally. Our demonstrator, centred on data architecture, presents a simplified example by addressing a scenario encompassing five jobs and a single machine on a single stage, resulting in $5! = 120$ possible schedules. However, complexity increases significantly with more machines and intricate shopfloor configurations, such as open shops where job operations lack determinism. Finding an optimal solution becomes significantly challenging within reasonable computational time. In such cases, the computational feasibility of attaining an optimal solution may become impractical. Hence, heuristics can be used as an initially preprocessing step to quickly generate satisfactory solutions before trying to solve the problem optimally. This strategic use of heuristic methods aims to navigate the computational challenges inherent in highly complex job scheduling scenarios, offering viable solutions within reasonable computational constraints (Ruiz 2019).

Consideration of security aspects: In our architectural framework, we have given negligible attention to security aspects. Nevertheless, maintaining a secure environment that restricts access only to authorised users necessitates the management of user credentials for the underlying software systems. This ensures that only authenticated and authorised users can interact with the system, fortifying the integrity and security of the software infrastructure. One prominent protocol for authorisation is OAuth, allowing third-party applications to access user resources without exposing user credentials (Hardt 2012).

Different programming languages: The majority of files we have generated to construct the architecture are Python scripts. Given Python's widespread adoption and compatibility with the odoo database, this choice was strategic. However, working with multiple programming languages introduces challenges, such as differences in data parsing and processing. We addressed this challenge by introducing Model Dependencies, but as more programming languages or tools are involved, more dependencies need to be managed within the system architecture.

Technology neutral: Since different frameworks for data processing in different domains exist, it is beneficial when developed concepts and solutions are independent from them. Although we demonstrate our concept in the context of PPC in the injection molding domain, it can be applied as a possible implementation for a semantic middleware in other frameworks.

Limitations of the demonstrator: The demonstrator primarily aims to showcase the benefits of utilising standardised data elements. As such, it is not a comprehensive software solution and lacks key industrial functionalities. For instance, in our demonstrator, the data from all current manufacturing orders in the ERP system is retrieved without filtering capabilities. Since implementing such functionality is an engineering task rather than a research question, we did not include it in our scope of study.

8. Conclusion and outlook

The rise of Industry 4.0 is concomitant with increased data and model-driven decision-support. In the era of Industry 4.0, Digital Shadows are established to automate the retrieval of models for specific decision-making scenarios along with the corresponding input data, a task previously undertaken manually. However, current publications only focus on modelling the Digital Shadow, describe its benefits, challenges, or general behaviour, or present use cases on a high level without deep technical details. Thus, an implementation approach that reveals missing classes and the design of specific building blocks for their processing was absent, leading to the fact that no domain-independent Digital Shadow currently exists that fulfills the goal.

To close this gap, we first have illustrated the model catalog's interfaces, gates, and databases through a sequence diagram. Leveraging this diagram, we have

identified the necessary requirements for an inclusive model catalog. Then we enhanced the current Digital Shadow meta-model from our prior publication (Becker et al. 2021) with classes for annotating models. Based on this, we developed a comprehensive data architecture and an associated implementation framework. Notably, these requisites, architecture, and framework are independent of specific manufacturing processes, aiding developers in constructing a model catalog tailored to their domain. The architecture's strengths lies in its modularity and scalability, permitting the definition of subtasks (such as database login or translation to proprietary formats) just once for reuse across various models. Furthermore, the architecture realises the seamless addition of new models via standardised property definitions without necessitating the rework of the entire structure. In our architecture, we leverage standard identifiers recommended in IEC 61360-1 and utilise ECLASS as a common data dictionary.

We have instantiated a web-based demonstrative model catalog for scheduling purposes to validate our data architecture. Therefore, we utilise a predefined model catalog for production planning and control tasks within the injection molding domain, which we have established before (Gannouni et al. 2024). This model catalog empowers production schedulers to select shop floor specifications, encompassing machine environments, constraints, and objective functions, retrieving models that align with these criteria. Since this model catalog is based on standardised model annotation, as proposed in this contribution, we have demonstrated that our architecture can seamlessly process such predefined models. Consequently, the architecture ensures modularity, as the integration of new models poses no challenges. Additionally, real-time data from odoo, an underlying ERP system, is utilised for model computation in our demonstrator. Our findings indicate that production schedulers benefit from automatic model and data retrieval, resulting in accelerated computation and decreased errors (e.g. from typographical mistakes). The incorporation of ERP data fosters the accuracy of computational results. The methodology for implementing the model catalog, along with the demonstrative architecture and framework for a scheduling model catalog, serves as a blueprint for developing other model catalogs with various purposes. Consequently, our findings are not confined to a specific domain but can be applied across different manufacturing sectors and applications.

Further work is the transfer of the architecture into specific technologies. Here, the Asset Administration Shell AAS serves as a potential technology since it includes relevant properties of an asset in a standardised manner and currently undergo international standardisation. Based on that, the architecture can be a building block for the establishment of data ecosystems. Data ecosystems are networks of data sources and technologies that interact to manage and utilise data to be used in services across enterprise borders (Hrustek, Mekovec, and Alexopolus 2023). Another important area of further work is to integrate security mechanisms into the architecture, ensuring users' access and execution align strictly with their authorisation levels. Also, it is essential to address how the architecture manages latency and data synchronisation issues. Specifically, the mechanisms ensuring data accuracy during model execution in rapidly changing production environments need to be thoroughly investigated.

Notes

1. https://github.com/psapel/DS-ModelCatalog-Implementation.
2. https://eclass.eu/en.
3. https://angular.io/.
4. https://flask.palletsprojects.com/en/3.0.x/.
5. https://www.elastic.co/.
6. https://min.io/.
7. https://www.odoo.com/.
8. https://github.com/psapel/DS-ModelCatalog-Implementation.
9. Pattern of odoo jobs: WH = Warehouse, MO = Manufacturing Order, incremental job number.
10. https://apps.odoo.com/apps.

Disclosure statement

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

Funding

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

CRediT authorship contribution statement

Patrick Sapel: Investigation, Conceptualisation, Methodology, Validation, Writing - Original Draft, Visualisation, Data Curation, Software. **Aymen Gannouni:** Methodology, Validation, Writing – Original Draft, Visualisation, Data Curation, Software. **Anas Abdelrazeq:** Supervision, Writing – Review & Editing. **Christian Hopmann:** Resources, Supervision, Writing – Review & Editing. **Robert H. Schmitt:** Resources, Supervision, Writing – Review & Editing.

Notes on contributors

Patrick Sapel has been working as a research assistant at the Institute for Plastics Processing at RWTH Aachen University since May 2019. He is employed in the Department of Injection Molding and leads the workgroup Production Planning and Automation. He studied part-time business administration and engineering at the University of Applied Sciences in Hagen, Germany (Bachelor). During his part-time study, he worked in a medium-sized company in the field of master data and manufacturing preparation. In his master's program at the University Duisburg-Essen, he studied Logistics Engineering. His research focuses on Industry 4.0, particularly the fundamentals and implementation of interoperable injection molding facilities using digital twins, which he conducts within the Cluster of Excellence Internet of Production at the RWTH Aachen University.

Dr.-Ing. Aymen Gannouni has been working as a research assistant at the Chair of Intelligence in Quality Sensing as part of the Laboratory for Machine Tools and Production Engineering (WZL) at RWTH Aachen University since January 2020. Since 2023, he leads the research group 'Production Enhancing Technologies'. Mr Gannouni is since May 2025 a doctor of engineering, whose work focussed on tackling the joint optimisation of production scheduling and multi-robot task allocation in flexibly assembly systems. Before that, he studied computer science with a minor in business administration at RWTH Aachen University. Mr Gannouni's work in the Excellence Cluster Internet of Production at the RWTH Aachen University sits at the interface between reinforcement learning, data science, operations research, digital shadows and Industry 4.0.

Dr.-Ing. Anas Abdelrazeq has been working as a researcher at the chair of Intelligence in Quality Sensing since April 2015. Since 2021, he is fulfilling the position of head of department 'Data Intelligence'. In the summer of 2020, Anas gained his doctoral degree (Dr.-Ing.) in the field of 3D Vision (Augmented Reality) and Cloud Computing. Before that, Anas finished his master's degree at RWTH Aachen University from the international program of 'Software Systems Engineering' focussing on applied computer sciences. Before that, Anas gained his bachelor's in 'Computer Systems Engineering' at Birzeit University. During his studies and work at the chair leading different national and European-wide projects, he gained knowledge and experience in different fields especially in data analytics, data management, machine learning as well as industrial metaverse and mixed reality applications.

Prof. Dr.-Ing. Christian Hopmann has been a Professor of the Chair of Plastics Processing at the Faculty of Mechanical Engineering at RWTH Aachen University since April 2011. He also is the Head of the IKV – Institute for Plastics Processing in Industry and Craft at RWTH Aachen University and Managing Director of the IKV's Association of Sponsors. Following his studies in Mechanical Engineering with particular focus on Plastics Processing, he received his doctoral degree (Dr.-Ing.) with a thesis on Ceramic Injection Moulding. He was research fellow at the IKV from 1996 to 1999 and head of its department of Part Design/Materials Technology in 2000. From 2001 to 2004 he was Chief Engineer and Senior Vice Director of the institute. In 2005 Hopmann started his industrial career at RKW AG Rheinische Kunststoffwerke (today: RKW SE), being head of the Quality Management at RKW's site in Petersaurach, South Germany. From 2006 to end of 2009 he was the Head of Extrusion. From January 2010 to April 2011, he was Managing Director of RKW Sweden AB in Helsingborg/Sweden. Since October 2017 he is Visiting Professor of Beijing University of Chemical Technology.

Prof. Dr.-Ing. Robert H. Schmitt has been Professor and Chair of Intelligence in Quality Sensing at the Laboratory for Machine Tools and Production Engineering (WZL) of RWTH Aachen University since 2004. He is also Director of the WZL and serves on the Board of Directors of the Fraunhofer Institute for Production Technology IPT. After studying electrical engineering, he began his career as Chief Engineer at WZL before joining the quality management executive staff of a leading commercial vehicle manufacturer in 1997, where he held responsibility for production across multiple sites and final assembly of vehicles up to 18 tons. His research focuses on sensor-based data processing, intelligent metrology, and their integration into modern quality management systems to enable digital, sustainable, and resilient production. In addition to his academic roles, he has been President of the German Society for Quality (DGQ) since 2021.

Data availability

The dataset generated during the current study is available in the GitHub repository, https://github.com/psapel/DS-Model Catalog-Implementation.

ORCID

Patrick Sapel https://orcid.org/0000-0002-2838-3141
Aymen Gannouni https://orcid.org/0000-0002-7102-5307
Anas Abdelrazeq https://orcid.org/0000-0002-8450-2889
Christian Hopmann https://orcid.org/0000-0003-4343-1508
Robert H. Schmitt https://orcid.org/0000-0002-0011-5962

References

acatech – National Academy of Science and Engineering. 2021. "Modellierungs- und Simulationsbedarfe der intelligenten Fabrik." München. https://www.plattform-i40.de/IP/Redaktion/DE/Downloads/Publikation/FB_Modellierung.pdf?_blob=publicationFile&v=1

Amershi, Saleema, Andrew Begel, Christian Bird, Robert DeLine, Harald Gall, Ece Kamar, Nachiappan Nagappan, Besmira Nushi, and Thomas Zimmermann. 2019. "Software Engineering for Machine Learning: A Case Study." In *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice, Piscataway, NJ*, 291–300. IEEE.

Ameziane, S., I. Grangel-Gonzalez, P. Janik, M. Keller, F. Loesch, F. Massari, D. Molin, and N. Wilhelm. 2022. "Industry 4.0 Core Information Model: A Publication by the Open Manufacturing Platform." https://raw.githubusercontent.com/OpenManufacturingPlatform/openmanufacturingplatform.github.io/master/docs/sds/OMP-SDS-Whitepaper_I4.0_Core_Information_Model.pdf.

Bajic, Bojana, Aleksandar Rikalovic, Nikola Suzic, and Vincenzo Piuri. 2021. "Industry 4.0 Implementation Challenges and Opportunities: A Managerial Perspective." *IEEE Systems Journal* 15 (1): 546–559. https://doi.org/10.1109/JSYST.2020.3023041.

Barcelos, Pedro Paulo F., Tiago Prince Sales, Mattia Fumagalli, Claudenir M. Fonseca, Isadora Valle Sousa, Elena Romanenko, Joshua Kritz, and Giancarlo Guizzardi. 2022. "A FAIR Model Catalog for Ontology-Driven Conceptual Modeling Research." In *Conceptual Modeling*, edited by Jolita Ralyté, Sharma Chakravarthy, Mukesh Mohania, Manfred A. Jeusfeld, and Kamalakar Karlapalem, Vol. 13607 of *Lecture Notes in Computer Science*, 3–17. Cham: Springer International Publishing.

Bauernhansl, Thomas, Silke Hartleif, and Thomas Felix. 2018. "The Digital Shadow of Production – A Concept for the Effective and Efficient Information Supply in Dynamic Industrial Environments." *Procedia CIRP* 72:69–74. 51st CIRP Conference on Manufacturing Systems, https://doi.org/10.1016/j.procir.2018.03.188.

Bayer, Tylor, Daniel P. Ames, and Theodore G. Cleveland. 2021. "Design and Development of a Web-Based EPANET Model Catalogue and Execution Environment." *Annals of GIS* 27 (3): 247–260. https://doi.org/10.1080/19475683.2021.1936171.

Becker, Fabian, Pascal Bibow, Manuela Dalibor, Aymen Gannouni, Viviane Hahn, Christian Hopmann, Matthias Jarke, et al. 2021. "A Conceptual Model for Digital Shadows in Industry and Its Application." In *Conceptual Modeling, ER 2021*, edited by Aditya Ghose, Jennifer Horkoff, Vitor E. Silva Souza, Jeffrey Parsons, and Joerg Evermann, October, 271–281. Springer.

Berners-Lee, T. 2005. *Uniform Resource Identifier (URI): Generic Syntax*. RFC 3986. RFC Editor. https://datatracker.ietf.org/doc/html/rfc3986.

Bibow, Pascal, Manuela Dalibor, Christian Hopmann, Ben Mainz, Bernhard Rumpe, David Schmalzing, Mauritius Schmitz, and Andreas Wortmann. 2020. "Model-Driven Development of a Digital Twin for Injection Molding." In *Advanced Information Systems Engineering*, edited by Schahram Dustdar, Eric Yu, Camille Salinesi, Dominique Rieu, and Vik Pant, Vol. 12127 of *Lecture Notes in Computer Science*, 85–100. Cham: Springer International Publishing.

Braun, Stefan, Manuela Dalibor, Nico Jansen, Matthias Jarke, István Koren, Christoph Quix, Bernhard Rumpe, Manuel Wimmer, and Andreas Wortmann. 2023. "Engineering Digital Twins and Digital Shadows as Key Enablers for Industry 4.0." In *Digital Transformation*, edited by Birgit Vogel-Heuser and Manuel Wimmer, 3–31. Berlin, Heidelberg: Springer Berlin Heidelberg.

Brecher, Christian, Melanie Padberg, Matthias Jarke, Wil van der Aalst, and Günther Schuh. 2024. "The Internet of Production: Interdisciplinary Visions and Concepts for the Production of Tomorrow." In *Internet of Production*, edited by Christian Brecher, Günther Schuh, Wil van der Aalst, Matthias Jarke, Frank T. Piller, and Melanie Padberg, Interdisciplinary Excellence Accelerator Series, 3–14. Cham: Springer International Publishing.

Burghoff, Günther. 1983. "Rüstzeitreduzierung in Spritzgiessbetrieben." PhD Thesis, Institute for Plastics Processing at RWTH Aachen University, Aachen.

Cañas, Héctor, Josefa Mula, Manuel Díaz-Madroñero, and Francisco Campuzano-Bolarín. 2021. "Implementing Industry 4.0 Principles." *Computers & Industrial Engineering* 158:107379. https://doi.org/10.1016/j.cie.2021.107379.

DIN EN 61360-1:2018-07. 2018. *DIN EN 61360-1:2018-07: Standard Data Element Types with Associated Classification Scheme – Part 1: Definitions – Principles and Methods*. Standard. Berlin: Beuth Verlag GmbH.

European Union. 2017. "New European Interoperability Framework – Promoting Seamless Services and Data Flows for European Public Administrations." https://ec.europa.eu/isa2/sites/default/files/eif_brochure_final.pdf.

Fensel, Dieter, Mick Kerrigan, and Michal Zaremba eds. 2008. *Implementing Semantic Web Services: The SESA Framework*. Berlin, Heidelberg: Springer Berlin Heidelberg. http://nbn-resolving.org/urn:nbn:de:bsz:31-epflicht-1485400.

Fernández Pérez, José Luis, and Carlos Hernández. 2019. *Practical Model-Based Systems Engineering*. Boston and London: Artech House. https://ebookcentral.proquest.com/lib/kxp/detail.action?docID=5848444.

Fettke, Peter, and Peter Loos. 2002. "Der Referenzmodellkatalog als Instrument des Wissensmanagements: Methodik und Anwendung." In *Wissensmanagement mit Referenzmodellen*, edited by Jörg Becker and Ralf Knackstedt, 3–24. Heidelberg: Physica-Verlag HD.

Fettke, Peter, and Peter Loos. 2003. "Classification of Reference Models: A Methodology and Its Application." *Information Systems and e-Business Management* 1 (1): 35–53. https://doi.org/10.1007/BF02683509.

Gannouni, Aymen, Patrick Sapel, Anas Abdelrazeq, Christian Hopmann, and Robert H. Schmitt. 2024. "A Blueprint Description of Production Scheduling Models Using Asset Administration Shells." *Production & Manufacturing Research* 12 (1): 1–23. https://doi.org/10.1080/21693277.2024.2324339.

González, José M., and Mathias Uslar. 2012. "An Ontology-Based Method to Construct a Reference Model Catalogue for the Energy Sector." In *Semantic Technologies for Business and Information Systems Engineering*, edited by Stefan Smolnik, Frank Teuteberg, and Oliver Thomas, 16–39. IGI Global.

Graham, R. L., E. L. Lawler, J. K. Lenstra, and A. H. G. Rinnooy Kan. 1979. "Optimization and Approximation in Deterministic Sequencing and Scheduling: A Survey." In *Discrete Optimization II, Proceedings of the Advanced Research Institute on Discrete Optimization and Systems Applications of the Systems Science Panel of NATO and of the Discrete Optimization Symposium Co-sponsored by IBM Canada and SIAM Banff, Aha. and Vancouver*, Vol. 5 of *Annals of Discrete Mathematics*, 287–326. Elsevier.

Hardt, Dick. 2012. *The OAuth 2.0 Authorization Framework*. RFC 6749. RFC Editor. https://datatracker.ietf.org/doc/html/rfc3986.

Hernández, José L., Susana Martín, Vangelis Marinakis, and Ignacio de Miguel. 2023. "From Silos to Open, Federated and Enriched Data Lakes for Smart Building Data Management." In *2023 IEEE International Workshop on Metrology for Living Environment (MetroLivEnv)*, 29–33.

Hevner, Alan R., Salvatore T. March, Jinsoo Park, and Sudha Ram. 2004. "Design Science in Information Systems Research." *MIS Quarterly* 28 (1): 75–105. https://doi.org/10.2307/25148625.

Hirsch, Eduard, Simon Hoher, and Stefan Huber. 2023. "An OPC UA-based Industrial Big Data Architecture." In *2023 IEEE 21st International Conference on Industrial Informatics (INDIN)* 1–7. https://api.semanticscholar.org/CorpusID:259063679.

Hrustek, Larisa, Renata Mekovec, and Charalampos Alexopolus. 2023. "Concept for an Open Data Ecosystem to Build a Powerful Data Environment." In *Information Systems*, edited by Maria Papadaki, Paulo Rupino da Cunha, Marinos Themistocleous, and Klitos Christodoulou, 251–263. Cham: Springer Nature Switzerland.

Husung, Stephan, Detlef Gerhard, Georg Jacobs, Julia Kowalski, Bernhard Rumpe, Klaus Zeman, and Thilo Zerwas. 2023. "Model Signatures for Design and Usage of Simulation-Capable Model Networks in MBSE." In *Product Lifecycle Management. PLM in Transition Times: The Place of Humans and Transformative Technologies*, edited by Frédéric Noël, Felix Nyffenegger, Louis Rivest, and Abdelaziz Bouras, Vol. 667 of *IFIP Advances in Information and Communication Technology*, 155–164. Cham: Springer Nature Switzerland.

Hüttner, Hanns Jürgen. 1979. "Beitrag zur Maschinenbelegungsplanung für die einstufige Fertigung in Spritzgießbetrieben." PhD Thesis, Institute for Plastics Processing at RWTH Aachen University, Aachen.

IDTA 02005-1-0. 2022. "Provision of Simulation Models." https://industrialdigitaltwin.org/wp-content/uploads/2023/01/IDTA-02005-1-0_Submodel_ProvisionOfSimulationModels.pdf.

Jacobs, Georg, Christian Konrad, Joerg Berroth, Thilo Zerwas, Gregor Höpfner, and Kathrin Spütz. 2022. "Function-Oriented Model-Based Product Development." In *Design Methodology for Future Products*, edited by Dieter Krause and Emil Heyden, 243–263. Cham: Springer International Publishing.

Jones, David, Chris Snider, Aydin Nassehi, Jason Yon, and Ben Hicks. 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.

Kamm, Simon, Nasser Jazdi, and Michael Weyrich. 2021. "Knowledge Discovery in Heterogeneous and Unstructured Data of Industry 4.0 Systems: Challenges and Approaches." *Procedia CIRP* 104:975–980. https://doi.org/10.1016/j.procir.2021.11.164.

Kaur, Navroop. 2025. "Intelligent Manufacturing in Industry 4.0." In *Intelligent Manufacturing*, edited by Navroop Kaur, Gurpreet S. Dhillon, Sita Rani, and Ahmed A. Elngar, 5–25. Boca Raton: CRC Press.

Kritzinger, Werner, Matthias Karner, Georg Traar, Jan Henjes, and Wilfried Sihn. 2018. "Digital Twin in Manufacturing: A Categorical Literature Review and Classification." *IFAC-PapersOnLine*51 (11): 1016–1022. 16th Symp. on Information Control Problems in Manufacturing INCOM'18, https://doi.org/10.1016/j.ifacol.2018.08.474.

Lehner, Daniel, Sabine Wolny, Alexandra Mazak-Huemer, and Manuel Wimmer. 2020. "Towards a Reference Architecture for Leveraging Model Repositories for Digital Twins." In *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, Vol. 1, 1077–1080.

Li, Xiaofeng. 2025. "Data Acquisition and Management in Industrial Businesses Fundamentals and Methodologies." *Journal of Enterprise and Business Intelligence* 5 (1): 30–39. https://doi.org/10.53759/5181/JEBI202505004.

Marcon, Petr, Christian Diedrich, Frantiek Zezulka, Tizian Schröder, Alexander Belyaev, Jakub Arm, Tomas Benesl, Zdenek Bradác, and Ivo Veselý. 2018. "The Asset Administration Shell of Operator in the Platform of Industry 4.0." In *2018 18th International Conference on Mechatronics – Mechatronika (ME)*, 1–5. https://api.semanticscholar.org/CorpusID:59235706.

Mukhopadhyay, Sandip, Srinivas Pingali, and Amitabh Satyam. 2022. *Selling IT: The Science of Selling, Buying and Deal-Making*. London: Routledge. https://www.taylorfrancis.com/books/9781003155270.

Panja, Madhurima, Tanujit Chakraborty, and Uttam Kumar. 2020. "Data Dictionary." In *Encyclopedia of Mathematical Geosciences*, edited by B. S. Daya Sagar, Qiuming Cheng, Jennifer McKinley, and Frits Agterberg, Springer eBook Collection, 1–5. Cham: Springer International Publishing and Imprint Springer.

Pause, Daniel, Philipp Brauner, Marco Faber, Markus Fischer, Philipp Hunnekes, Henning Petruck, Alexander Mertens, et al. 2019. "Task-Specific Decision Support Systems in Multi-Level Production Systems based on the digital shadow." In *2019 IEEE International Conference on Industrial Cyber Physical Systems (ICPS 2019), Piscataway, NJ*, 603–608. IEEE.

Peffers, Ken, Tuure Tuunanen, Marcus A. Rothenberger, and Samir Chatterjee. 2007. "A Design Science Research Methodology for Information Systems Research." *Journal of Management Information Systems*24 (3): 45–77. https://doi.org/10.2753/MIS0742-1222240302.

Pidun, Tim, and Carsten Felden. 2010. "A Reference Model Catalog of Models for Business Process Analysis." In *16th Americas Conference on Information Systems, Red Hook, NY* . Curran. https://aisel.aisnet.org/cgi/viewcontent.cgi?article=1167&context=amcis2010.

Raptis, Theofanis P., Andrea Passarella, and Marco Conti. 2019. "Data Management in Industry 4.0: State of the Art and Open Challenges." *IEEE Access*7:97052–97093. https://doi.org/10.1109/ACCESS.2019.2929296.

Ruiz, Rubén. 2019. "Scheduling Heuristics." In *Handbook of Heuristics*, edited by Rafael Martí, Pardalos Panos, and

Mauricio G. C. Resende, Springer eBook Collection, 1–24. Cham: Springer.

Schuh, Günther, Christian Brecher, Fritz Klocke, and Robert Schmitt, eds. 2017. *Engineering valley - Internet of production auf dem RWTH Aachen Campus: Festschrift für Univ.-Prof. em. Dr.-Ing. Dipl.-Wirt. Ing. Dr. h.c. mult. Walter Eversheim*. 1st ed. Aachen: Apprimus Verlag.

Sjarov, Martin, Tobias Lechler, Jonathan Fuchs, Matthias Brossog, Andreas Selmaier, Florian Faltus, Toni Donhauser, and Jörg Franke. 2020. "The Digital Twin Concept in Industry – A Review and Systematization." In *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, Vol. 1, 1789–1796.

Spütz, Kathrin, Julius Berges, Georg Jacobs, Joerg Berroth, and Christian Konrad. 2022. "Classification of Simulation Models for the Model-based Design of Plastic-Metal Hybrid Joints." *Procedia CIRP* 109:37–42. https://doi.org/10.1016/j.procir.2022.05.211.

Stachowiak, Herbert. 1973. *Allgemeine Modelltheorie*. Wien and New York: Springer.

SystemX, I. R. T. 2020. *Model Identity Card (MIC): Towards a Standardization of the Specification and Description of Simulation Models*. Technical Report. https://mic.irt-systemx.fr.

Tao, Fei, Qinglin Qi, Ang Liu, and Andrew Kusiak. 2018. "Data-Driven Smart Manufacturing." *Journal of Manufacturing Systems* 48:157–169. https://doi.org/10.1016/j.jmsy.2018.01.006.

Tsay, Jason, Alan Braz, Martin Hirzel, Avraham Shinnar, and Todd Mummert. 2022. "Extracting Enhanced Artificial Intelligence Model Metadata from Software Repositories." *Empirical Software Engineering* 27 (7): 1–37. https://doi.org/10.1007/s10664-022-10206-6.

van der Valk, Hendrik, Hendrik Haße, Frederik Möller, and Boris Otto. 2022. "Archetypes of Digital Twins." *Business & Information Systems Engineering* 64 (3): 375–391. https://doi.org/10.1007/s12599-021-00727-7.

VDI 2193 Part 1:2020-04. 2020. *Language for I4.0 Components – Structure of Messages*. Guideline. Berlin: Beuth Verlag GmbH.

VDI 2727 Part 1:1991-05. 1991. *Catalogues for Machine Design; Mechanisms for Motiontransfer; Fundamentals*. Guideline. Berlin: Beuth Verlag GmbH.

Wilkinson, Mark D., Michel Dumontier, I. Jsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, et al. 2016. "The FAIR Guiding Principles for Scientific Data Management and Stewardship." *Scientific Data* 3:160018. https://doi.org/10.1038/sdata.2016.18.

Xing, Lei. 2024. "Heterogeneous Data Integration and Transformation Scheme Based on XML and Agent." In *Intelligent Computing Technology and Automation*, edited by Zhixiang Hou, Advances in Transdisciplinary Engineering. IOS Press.

Yan, Zhiqiang, Remco Dijkman, and Paul Grefen. 2012. "Business Process Model Repositories – Framework and Survey." *Information and Software Technology* 54 (4): 380–395. https://doi.org/10.1016/j.infsof.2011.11.005.

Yin, Shen, and Okyay Kaynak. 2015. "Big Data for Modern Industry: Challenges and Trends [Point of View]." *Proceedings of the IEEE* 103 (2): 143–146. https://doi.org/10.1109/JPROC.2015.2388958.

Zarate, Gorka, Raul Minon, Josu Diaz-de Arcaya, and Ana I. Torre-Bastida. 2022. "K2E: Building MLOps Environments for Governing Data and Models Catalogues While Tracking Versions." In *2022 IEEE 19th International Conference on Software Architecture Companion (ICSA-C)*, 206–209. IEEE.