A Framework for the Integration of Hybrid Models in Digital Twin Architectures for PHM

Jill Mercedes Linneweber [1], Andreas Maximilian Schultz [2], Laura Müller [2], Osarenren Kennedy Aimiyekagbon [1], Walter Sextro [1], and Iryna Mozgova [2]

[1] *University Paderborn, Faculty of Mechanical Engineering, Chair of Dynamics and Mechatronics (LDM), Paderborn, 33098, Germany*
jill.mercedes.linneweber@uni-paderborn.de

[2] *University Paderborn, Faculty of Mechanical Engineering, Chair of Data Management in Mechanical Engineering (DMB), Paderborn, 33098, Germany*

ABSTRACT

Hybrid model structures combine physics-based and machine-learning (ML) models to leverage complementary strengths in prognostics and health management (PHM). While both hybrid modeling and digital twin (DT) architectures are widely studied, their structural interaction is rarely addressed systematically. In practice, hybrid models are often developed application-specifically, and their structural integration into DT architectures remains weakly formalized.

This paper analyzes hybrid model structures focusing on their architectural composition and establishes a requirement-driven configuration framework based on core PHM constraints. Four hybrid coupling strategies are classified according to their structural integration principles and positioned within a design space defined by structural dominance and integration depth. Based on this configuration framework, architectural integration requirements for DT environments are derived and operationalized in a project-specific DT implementation.

The study contributes (1) a structured classification of hybrid coupling strategies, (2) a requirement-driven configuration framework for hybrid model structures in PHM contexts, and (3) a structured integration workflow for embedding hybrid models into DT architectures. The results provide a consistent foundation for managing hybrid model structures within scalable DT environments.

1. INTRODUCTION

Condition monitoring and predictive maintenance have become essential approaches for increasing efficiency, reliability, and cost effectiveness in modern transportation systems, particularly in the rail sector where asset availability and operational safety are critical. Prognostics and Health Management (PHM) aims to support these objectives through continuous system monitoring, health state estimation, and Remaining Useful Life (RUL) prediction.

While basic condition monitoring may rely on sensor-based state observation, diagnostic and prognostic tasks require explicit system representations capable of capturing system dynamics, degradation behavior, and uncertainty under evolving operating conditions. Consequently, model-based approaches form the analytical core of PHM systems.

Physics-based models enable structured incorporation of domain knowledge and ensure physically consistent behavior. However, they may be limited by incomplete knowledge of degradation mechanisms, parameter uncertainties, and increasing model complexity. Data-driven approaches, including machine learning (ML), learn system behavior directly from operational data and can capture previously unmodeled effects. Yet, their extrapolation capability and physical interpretability may be limited. Hybrid model structures combine physics-based and ML-based models in order to leverage complementary strengths.

In PHM environments, models do not operate in isolation. They must continuously process incoming measurement data, maintain internal states, and provide outputs for decision support. Digital twin (DT) architectures provide a structured environment that links physical assets, data streams, and analytical models within a unified system. Within such architectures, the configuration of hybrid model structures becomes closely coupled to architectural and lifecycle considerations.

Although both hybrid modeling and DT concepts are widely

discussed, their interaction is rarely addressed in a systematic manner. Existing contributions typically focus either on specific hybrid modeling techniques or on DT architectures without explicitly linking structural model configuration to architectural integration principles. As a result, hybrid models are often developed in an application-specific manner, limiting transferability and scalability.

This paper addresses this gap by systematically analyzing hybrid model structures in the context of PHM and deriving requirement-driven configuration principles. Building on this structural differentiation, the study establishes architectural integration requirements and proposes a structured workflow for embedding hybrid model structures within DT architectures.

The contributions of this work are threefold:

- a structured classification of hybrid coupling strategies based on their structural integration principles (Section 2),
- a requirement-driven conceptual framework for configuring hybrid model structures in PHM contexts (Section 3), and
- a structured integration workflow for embedding hybrid model structures into DT architectures (Section 4)

By clarifying the relationship between hybrid model structure and DT architecture, this work provides a consistent foundation for developing scalable and transferable PHM solutions.

2. STATE OF THE ART AND LITERATURE CONTEXT

This section provides an overview of the foundational concepts relevant to this study and positions the proposed work within the existing literature. In recent years, significant research efforts have focused on advancing fault diagnosis and prognostics in PHM contexts (Zachariades & Xavier, 2025), (Gherghina, Bizon, Iana, & Vasilică, 2025), (S. Chen, Bekar, Bokrantz, & Skoogh, 2025). At the same time, DT have become increasingly important due to their potential to improve reliability, efficiency, and system intelligence in industrial applications (Tao, Qi, Wang, & Nee, 2019), (Jacob & Santhosh Kumar, 2025), as well as research data (Dierend, Altun, Mozgova, & Lachmayer, 2022).

A growing number of studies are focusing on the integration of physical models and machine learning (ML) techniques into digital twin environments (Gupta & Kundu, 2024) (Chowdhury et al., 2025). These approaches are often referred to as AI-driven DTs. In this context, ML-based models are used to support predictive maintenance, real-time quality control, and adaptive lifecycle management (S. Chen, Turanoglu Bekar, Bokrantz, & Skoogh, 2025), (Najafzadeh & Yeganeh, 2025). Hybrid digital twins have also been proposed, combining physics-based representations with data-driven methods to improve predictive accuracy and adaptability (Şahin, Wolff, von Danwitz, & Popp, 2024), (Mayr, Gross, Krenn, Kunze, & Zehetner, 2024), (Kareem, Domingo, & Hur, 2025).

However, there is only a limited systematic structural link between the configuration of hybrid models and the design of DT architecture. Many studies present hybrid models at the application level without explicitly addressing their structural configuration or architectural embedding in DT. On the contrary, DT architectures are often described independently of the structural characteristics of the models they contain. As a result, the relationship between hybrid model structure and DT architecture is often discussed implicitly. To establish a unified basis for the subsequent analysis, the following subsections first summarize the role of digital twins in PHM and then provide an overview of hybrid modeling approaches in this context. This provides the conceptual basis for identifying structural gaps and motivates the development of a requirements-oriented configuration and integration framework.

2.1. Digital twins in PHM

The term *Digital Twin* (DT) requires clarification, as there exist several definitions in literature that share common principles but differ in terms of their scope of application and implementation details. This paper adopts the DT definition of the Wissenschaftliche Gesellschaft für Produktentwicklung WiGeP e.V. (Scientific Society for product Development) as described in (Stark, Anderl, Thoben, & Wartzack, 2020).

According to this definition, a DT represents a specific product instance or product-service instance and includes both its states and behaviors. The DT can be structurally divided into two main elements: the *digital master* and one or more *digital shadow*(s).

The digital master embodies the generalized digital representation of the product. It stores digital product information and models, including CAD models, physics-based simulation models, and hybrid model structures. The digital master typically exists prior to the creation of a physical instance and provides the structural model basis throughout the lifecycle (Stark et al., 2020).

For each physical model, a corresponding digital shadow represents the specific operational instance. The digital shadow stores measurement data and operational information generated during runtime. Within PHM contexts, this includes sensor data, diagnostic indicators, and prognostic results such as degradation states or RUL estimates.

The digital shadow is characterized by continuous data acquisition and frequent updates. In contrast, changes within the digital master, such as model updates or structural modifications, occur less frequently but require systematic ver-

sion tracking and documentation. This distinction between dynamic operational data (digital shadow) and comparatively stable model structures (digital master) is particularly relevant for managing hybrid model structures within the DT architecture.

The division in digital master and digital shadow provides a structural basis for managing hybrid model structures, particularly with respect to parameter persistence, modular execution, and lifecycle coordination.

2.2. Hybrid Modeling in PHM

PHM applications require model structures that remain reliable under evolving operating conditions, limited failure data, and partially understood degradation mechanisms. In many industrial contexts, neither purely physics-based nor purely data-driven modeling approaches are sufficient to address these challenges in isolation. Physics-based models provide structural interpretability and physical consistency, while ML-based models offer flexibility and adaptive learning capabilities.

The following subsections introduce the concept of hybrid modeling and outline the structural perspective adopted in this work.

2.2.1. Hybrid Modeling

Hybrid model structures combine a physics-based model with an ML-based model within a unified system representation. Physics-based models, often referred to as white-box models, are derived from first principles and represent system behavior using physical laws and mathematical formulations such as differential equations. They ensure structural consistency and interpretability but may be computationally intensive and limited in representing unknown or partially understood effects.

ML-based models, frequently described as black-box models, approximate system behavior directly from data. They do not require explicit physical knowledge but depend on sufficient and representative training data. While flexible and adaptable, they may lack physical consistency and robustness outside the observed operating domain.

Based on the distinction between white-box and black-box, hybrid models are often referred to as gray-box models. However, the term *hybrid* contains a broad range of structural configurations. Although hybrid modeling is widely applied in engineering contexts, a consistent structural classification based on integration principles remains limited (Von Stosch, Oliveira, Peres, & Feyo De Azevedo, 2014), (Wohlleben, Bender, Peitz, & Sextro, 2022), (Von Rueden, Mayer, Sifa, Bauckhage, & Garcke, 2020), (Von Krannichfeldt, Orehounig, & Fink, 2025).

In this work, hybrid model structures are characterized according to the structural role of the ML-based model relative to the physics-based model. This perspective enables a systematic differentiation of hybrid coupling strategies independent of specific ML algorithms. Within a DT architecture, hybrid modeling can thus be interpreted as the structured integration of explicit system knowledge and learned representations. Depending on the functional role of the physics-based and ML-based models within the overall model structure, distinct integration principles emerge.

The structural differentiation of hybrid model structures is formalized in the following subsection through the definition of distinct hybrid coupling strategies.

2.2.2. Hybrid coupling strategies

A DT architecture cannot be defined independently of the structural properties of the models it integrates. Since hybrid model structures differ fundamentally in how physics-based and ML-based models interact, a structured classification is necessary to derive consistent architectural integration principles.

Hybrid modeling represent a broad range of architectural concepts. Rather than distinguishing approaches by algorithmic implementation, the following classification is derived from structurally distinct integration mechanisms between physics-based and ML-based models. These structural differences generate distinct architectural requirements within a DT environment.

The classification focuses on how the respective model components are structurally organized and how they must be integrated, stored, and managed within a digital twin architecture. It therefore emphasizes integration-relevant model characteristics rather than methodological differences in model training or algorithm design.

Four representative coupling strategies are identified:

a) Residual coupling
b) Parallel model fusion
c) ML-assisted calibration
d) Physics-constrained Machine Learning

These strategies represent structurally distinct integration principles. The four coupling strategies can be divided into two structural levels, which differ in terms of which models remain active during subsequent operation. At the first level, the two-model approach, the modeling process results in two independent models, jointly computing the system response. At the second level, the one-model approach, one of the two sources of knowledge is integrated into the model development but is no longer explicitly represented in the final model.

Practical implementations may combine elements of multiple strategies and therefore occupy intermediate configurations.

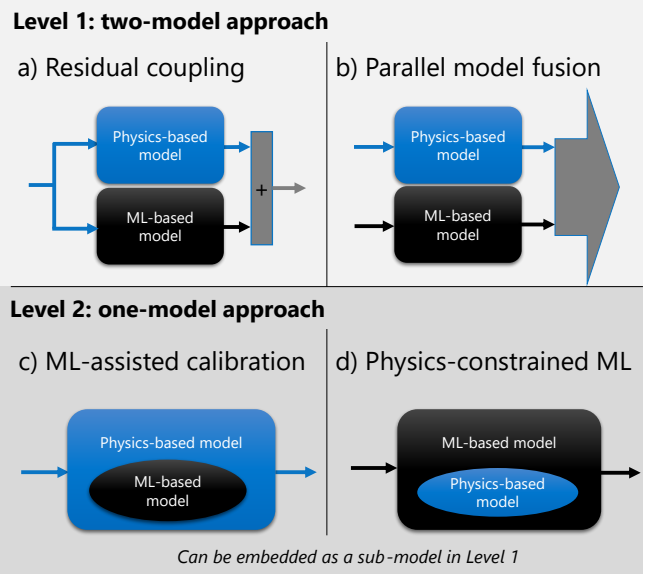

Figure 1. Structurally representative coupling strategies for hybrid modeling

The terminology used here is structural, equivalent concepts may appear in literature under different naming conventions. Further details and representative examples corresponding to each integration principle are discussed in the respective subsections below.

Figure 1 illustrates the physics-based and ML-based models and their structural interaction for each coupling strategy.

Residual coupling

Residual coupling describes hybrid configurations in which a physics-based model provides the primary prediction, while an ML-based model captures the residual between this prediction and the observed system behavior. Although both models are represented as separate model blocks, the ML-based model is functionally linked to the physics-based output and acts as an output-level correction.

$$y(\underline{x}) = f_{\text{physic}}(\underline{x}) + f_{\text{ML}}(\underline{x}) \tag{1}$$

In Equation (1), f_{physic} represents the physics-based model and f_{ML} represents the ML-based model. Both models receive the same system input vector $\underline{x}$. To enable comparison between the coupling strategies, the system input is denoted by $\underline{x}$ for all models. The system input $\underline{x}$ represents the set of relevant input variables describing the operating condition and state of the system. Depending on the application domain, these variables may include mechanical quantities such as displacement, velocity, or acceleration, but also electrical, chemical, thermal, or fluid-dynamic variables such as voltage, current, concentration, temperature, pressure, or flow rate.

The ML-based model approximates the residual between the physics-based prediction and the observed system response. This deviation is incorporated into its training process, such that f_{ML} is trained conditionally on f_{physic}, introducing a directional dependency that is absent in parallel model fusion. A correction is provided by the ML-based model without modifying the internal structure of the physics-based model.

Schematically as shown in Figure 1 a), two model blocks share the same input. The overall system output is generated by the additive combination of the two submodel outputs. To ensure comparability across all coupling strategies, the system output is denoted by y. This output may represent, the resulting force within a system.

This principle corresponds to additive discrepancy modeling approaches in the literature including trajectory-based discrepancy modeling (Wohlleben et al., 2024), Gaussian Process discrepancy models (Gardner, Rogers, Lord, & Barthorpe, 2021) and neural network residual correction (Rofatto et al., 2026).

Architectural Relevance: Since the ML-based model operates as an external correction module, this structure supports modular execution and independent updating of the ML model within a DT architecture. It enables separable deployment and flexible retraining mechanisms.

Parallel model fusion

Parallel model fusion describes configurations in which physics-based and ML-based models operate independently, and their outputs are combined through a fusion mechanism.

The two models may process different input representations and are structurally independent. The fusion mechanism determines the relative influence of each model.

$$y(\underline{x}) = w_1(\underline{x}) f_{\text{physic}}(\underline{x}_{\text{physic}}) + w_2(\underline{x}) f_{\text{ML}}(\underline{x}_{\text{ML}}) \tag{2}$$

In Equation (2), both models may receive the same input vector $\underline{x}$. The physics-based model uses a subset of the components of $\underline{x}$, denoted by $\underline{x}_{\text{physic}} \subseteq \underline{x}$, while the ML-based model uses $\underline{x}_{\text{ML}} \subseteq \underline{x}$. In this case, $\underline{x}$ is projected onto the respective components required by each model. Thus, both models can process identical inputs, but this is not mandatory. The fusion weights $w_1(\underline{x})$ and $w_2(\underline{x})$ determine the relative contribution of each model and may themselves depend on the input.

Schematically, as shown in Figure 1 b), two independent model blocks process their respective inputs. Their outputs are combined at the output level without internal coupling between the models.

This principle reflects concepts known from ensemble learning and multi-model combination frameworks , (Zhou, 2012), (Dietterich, 2000). Representative examples include weighted

ensemble models (Rajkolhe, Bhagwat, & Deshmukh, 2025) and hybrid physics data-driven model based-fusion approaches (Gao, Zhu, Wu, Lu, & Zhang, 2024).

The main structural difference to residual coupling lies in the independence of the models, residual coupling establishes a corrective dependency between models, whereas parallel fusion maintains structural independence and combines outputs externally.

Architectural Relevance: As neither model depends on the output of the other, both models can in principle be trained, updated, and replaced independently. This structural decoupling is architecturally relevant, as it allows each model to evolve separately within a DT without requiring coordinated retraining of the combined system. Independent model execution requires orchestration and output synchronization within the DT architecture. The DT must provide coordination logic for combining, validating, and managing multiple model outputs.

ML-assisted calibration

ML-assisted calibration refers to configurations in which the ML-based model estimates unknown or time-varying parameters of the physics-based model. The structural equations of the physics-based model remain unchanged.

$$y(\underline{x}) = f_{\text{physic}}(\underline{x}, \underline{\theta}_{\text{ML}}) \tag{3}$$

Here, the ML-based model influences the internal parameterization of the model rather than its output structure.

In Equation (3), $\underline{\theta}_{\text{ML}}$ denotes parameters determined by ML-based model. The ML-based model influences the internal parametrization of the model rather than modifying its output structure. Figure 1 c) illustrates this configuration schematically, the physics-based model forms the primary structural model, while the ML-based model supplements it by updating internal parameters.

Representative example include neural-network-based parameter estimation approaches (Lenzi, Bessac, Rudi, & Stein, 2023).

This strategy maintains physical interpretability and is particularly suitable when structural knowledge is available but parameter identification is challenging.

Architectural Relevance: Because model behavior depends on updated parameter states, persistent storage of calibrated parameters and coordinated version management are required within the DT architecture. Updates affect the internal model state rather than an external module.

Physics-constrained ML

Physics-constrained ML is closely related to the concept of Physics-Informed Machine Learning (PIML), which is widely discussed in the literature (Karniadakis et al., 2021). PIML typically refers to configurations in which an ML-based model forms the primary predictive structure and physical knowledge is incorporated into the training process, for example through constraint terms or regularization within the loss function.

In contrast, the term Physics-constrained ML is used here in a broader structural sense. While it includes the classical PIML formulation, it also encompasses configurations in which physical knowledge influences multiple stages of the model development process, including model design, parameterization, and training. Thus, physics-constrained ML extends the PIML concept beyond loss-level regularization toward a more comprehensive integration of physical constraints within the overall model structure.

$$y(\underline{x}) = f_{\text{ML}}(\underline{x}, \underline{\theta}_{\text{physic}}) \tag{4}$$

In this configuration, physics does not appear as a standalone simulation model but is incorporated into the model through physically informed constraints. In Equation (4), $\underline{\theta}_{\text{physic}}$ denotes the set of physically informed constraints incorporated into the model. Depending on the configuration, $\underline{\theta}_{\text{physic}}$ may influence the model architecture, the input representation, the parameterization, or the training objective. The ML-based model remains the primary structural representation of the system, while physical knowledge is embedded in a way that may extend beyond the loss formulation to other stages of model development.

Figure 1 d) illustrates this configuration schematically. The ML-based model forms the central predictive block, while physical knowledge is incorporated via constraints, represented by an auxiliary block.

Representative examples include Physics-Informed Neural Networks (PINNs) (Raissi, Perdikaris, & Karniadakis, 2019) and Physics-Constrained Neural Networks (Patel, Bhartiya, & Gudi, 2022).

Architectural Relevance: Because physical consistency is structurally embedded within the ML-based model formulation, adaptation typically requires coordinated retraining and potential redeployment of the model. This increases lifecycle coordination demands within the DT architecture.

The four coupling strategies reflect structurally distinct integration principles corresponding to established concepts in discrepancy modeling, gray-box parameter estimation, ensemble learning, and physics-informed machine learning literature. These two levels (see Figure 1) are not independent of one another: the one-model-approach describes how an individual sub-model can be constructed. ML-assisted calibration (Figure 1 c) and Physics-constrained ML (Figure 1 d) can

therefore serve as methods for building the constituent models within the two-model-approach. A physical model used within a residual coupling may itself have been parametrized through ML-assisted calibration. Equally, the ML model within a parallel fusion may have been produced through physics-constrained training. The two levels thus describe different abstraction layers of the modeling process and not competing alternatives.

The structural differences between these hybrid model structures directly influence their architectural embedding within a DT environment. However, structural differentiation alone does not provide guidance for selecting an appropriate configuration in a given PHM context. Therefore, a requirement-driven configuration framework is necessary to translate functional PHM requirements into suitable hybrid model structures.

3. REQUIREMENT-DRIVEN CONFIGURATION OF HYBRID MODELS

The previous section established that hybrid model structures differ fundamentally in their structural configuration and architectural implications. However, the structural classification alone does not provide guidance on when and under which conditions a specific hybrid model structure should be selected.

In PHM applications, model configuration cannot be determined independently of functional requirements. Prediction objectives, data characteristics, and operational constraints impose structural demands on the underlying model structure. Therefore, a systematic linkage between PHM requirements and hybrid model configuration is necessary.

While the classification of coupling strategies enables structural differentiation, it does not yet provide configurational orientation. To translate functional PHM requirements into consistent model configurations, a structured design space is introduced. Without such a configuration framework, hybrid modeling would remain a collection of isolated strategies rather than a systematically configurable modeling paradigm.

This section derives structural design implications for hybrid models from core PHM requirements. Rather than prescribing a single preferred coupling strategy. The objective is to define configuration principles that enable consistent positioning of hybrid model structures within this design space.

In PHM contexts where hybrid models are deployed within DT architectures, their structural configuration must also remain compatible with system-level integration constraints.

3.1. PHM Design Requirements

PHM comprises a broad range of application scenarios. Nevertheless, independent of the specific scenario, certain requirements consistently characterize PHM modeling task (C. Chen, Fu, Zheng, Tao, & Liu, 2023), (Krespach, Blum, Pottmann, Rehfeldt, & Klein, 2025), (Ma et al., 2025), (Ifeanyi & Coble, 2025). In this work, three core requirements are considered:

- Data integration and consistency
- Prediction capability
- Scalability and adaptability

These requirements define functional expectations toward both the underlying model structure and its embedding within the DT architecture.

Data integration and consistency: PHM systems continuously generate condition monitoring data during runtime that must be incorporated into the modeling framework. Models must therefore process heterogeneous sensor inputs, account for systematic deviations between predicted and observed behavior, and integrate new operational information in a consistent manner. This requirement implies the necessity of a flexible mechanisms for integrating ML-based models within the overall hybrid model structure, while maintaining consistency with the underlying system representation.

Prediction capability: PHM requires the prediction of future health states and, in particular, the estimation of the RUL. This implies extrapolation beyond previously observed condition monitoring data. Consequently, structural model consistency and physically plausible behavior become critical. To support Prediction capability, a hybrid model structure must enable extrapolation into degraded states, maintain long-term stability, and preserve physically consistent system behavior. These aspects highlight the relevance of structurally grounded representations that are not solely dependent on observed data distributions.

Scalability and adaptability: PHM models must be transferable across similar assets and adaptable to evolving system conditions such as degradation or operational drift. In addition, maintainability of the model structure over its lifecycle is required. This leads to structural implications regarding modular separability of models, and the ability to update or adapt parts of the model without complete redevelopment. These aspects directly influence how hybrid model structures can be embedded, updated and managed within a DT architecture.

Transition to structural implications

The functional PHM requirements described above do not directly determine a specific hybrid coupling strategy. However, they impose structural demands on how physics-based and ML-based models must interact within the hybrid model structure. To translate these functional requirements into configurable model characteristics, their structural implications must be made explicit. The following section formalizes these

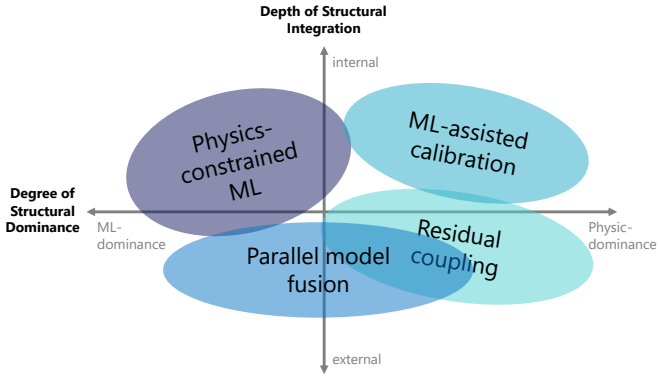

Figure 2. Hybrid Architecture Design Space Based on Structural Dominance and Integration Depth.

implications and derives structural properties that enable systematic positioning of hybrid model structures.

3.2. Design Implications for Hybrid Modeling

The PHM requirements outlined in Section 3.1 do not directly prescribe a specific hybrid coupling strategy. However, they define structural constraints on how physics-based and ML-based models must be configured within a hybrid model structure.

To formalize these implications, hybrid models can be characterized by four structural properties:

- structural dominance of the representation,
- integration depth of ML-based models,
- modular separability of model parts, and
- adaptability over the lifecycle.

These structural properties are independent of specific machine learning algorithms. They describe how the physics-based and ML-based models are organized and how they behave under operational constraints. Not all structural properties equally determine the configurational positioning of a hybrid model structure. From a system-architectural perspective, two properties primarily define the configuration space of hybrid models:

- the degree of structural dominance, and
- the depth of structural integration.

Structural dominance reflects the extend to which system behavior is determined by physics-based modeling principles versus ML-based approximation. Integration depth describes the level within the model structure at which both models interact, ranging from external output-level correction to internal integration within parameters or training objectives.

Modularity and adaptability remain structurally relevant but primarily affect lifecycle management and architectural embedding within a DT environment rather than the primary configurational positioning.

Together, structural dominance and integration depth define a conceptual design space in which hybrid model structures can be positioned. Unlike the structural classification presented in Section 2.2, which differentiates integration principles, this design space provides configurational orientation. It enables functional PHM requirements to be mapped onto structurally consistent model configurations.

The previously defined hybrid coupling strategies represent characteristic regions within this design space. Their positioning reflects structural tendencies rather than strict boundaries. Residual coupling and ML-assisted calibration are located toward the physics-dominant side, while differing in integration depth. Physics-constrained ML is positioned toward the ML-dominant side with high integration depth, and parallel model fusion occupies an intermediate region characterized by external combination of independent models. This distribution in the design space is shown graphically in Figure 2.

Depending on the relative weighting of PHM requirements, different regions of the design space become more appropriate. For instance, high extrapolation demands favor stronger structural grounding, whereas scenarios with high data heterogeneity may necessitate deeper integration of ML-based models.

This configuration perspective establishes the structural basis for deriving architectural integration requirements within a DT environment, which are discussed in the following section.

4. DIGITAL TWIN ARCHITECTURE AND MODEL INTEGRATION

Hybrid model structures are not inherently dependent on DT environments and may also be deployed as standalone models. However, in operational PHM scenarios characterized by continuous data acquisition, multi-asset management, and lifecycle coordination, a DT architecture provides the structural framework required for systematic model integration and management.

The structural configuration of the hybrid model, as defined in the previous section, directly influences the requirements imposed on the DT architecture. While the hybrid model structure determines how physics-based and ML-based models interact at the model level, the DT provides the architectural environment in which this structure is embedded, executed, and managed. Consequently, integration extends beyond model implementation. It affects data management, execution orchestration, state persistence, and lifecycle coordination within the DT architecture.

4.1. Architectural Requirements

The structural properties of hybrid model configurations, structural dominance, integration depth, modular separability, and adaptability, impose specific architectural requirements on the DT. These requirements arise from the interaction between model structure and operational PHM constraints.

Based on the PHM requirements discussed earlier, the following architectural requirements must be provided at the system level:

- persistent storage of measurement data, model parameters and degradation states,
- modular execution of physics-based and ML-based models,
- event-based triggering of model execution upon arrival of new measurement data,
- traceable update mechanisms for adaptive models, and
- consistent integration of model outputs into higher-level decision-support structures

These aspects are not solely algorithmic considerations, but define structural constraints that must be reflected in the design of the DT architecture.

Depending on the selected hybrid model structure, different integration patterns may be required. Some configurations enable largely independent execution of physics-based and ML-based models, while others involve tighter internal coupling. For instance, model structures characterized by strong structural dominance of the physics-based model may require persistent storage and versioning of calibrated parameters. In contrast, configurations with deep integration of ML-based models may necessitate coordinated update and retraining mechanisms within the DT. Accordingly, modularity implies support for separable deployment and structured version management of models. Adaptability requires traceable and orchestrated updating processes, potentially including event-based retraining triggered by new measurement data. Such coordinated control becomes particularly relevant in online PHM environments, where timely and consistent model updates are essential

4.2. Implementation

The following implementation illustrates how the previously derived architectural requirements are operationalized in a project-specific DT architecture.

This architecture provides the structural and technical foundation for executing hybrid model structures in a rail-focused PHM scenario. In particular, it supports persistent data storage, modular model deployment, event-based execution, and coordinated lifecycle management as derived in Section 4.1.

The DT acts as a data repository and an orchestration layer. It organizes asset-related information and coordinates the interaction between physics-based and ML-based models.

Following the adopted definition, the DT is divided into a digital master and one or more digital shadows. The digital master stores generalized model structures and digital product information that apply across asset instances. This includes physics-based and ML-based models, versioned and managed as structured elements.

The digital master stores the physics- and data-driven models that hold in general for all instances of a asset. For each real world entity, a digital shadow captures runtime-specific information such as measurement data, diagnostic indicators, and prognostic results. This separation enables persistent parameter storage and instance-level data management, as required by hybrid model configurations involving parameter learning or adaptive models.

The technical implementation of the DT is built up on the concepts of the open-source Asset Administration Shell (AAS) which is a framework for the handling of industrial data and information. It was developed by a German initiative to support the evolution of Industry 4.0 and is maintained by the Industrial Digital Twin Association (IDTA)[1] (Boss et al., 2020).

The IDTA focuses on the standardization of AAS assets to provide common ground for a development of future Industry 4.0 applications. The implementation used in this project is based on the Eclipse BaSyx project[2]. Modular service deployment enables separable execution of physics-based and ML-based models where structurally appropriate. Persistent storage models ensure traceable version management of model parameters and degradation states. A message broker mechanism supports event-based triggering of model execution upon arrival of new measurement data, facilitating coordinated lifecycle management.

The project consists of different modular building blocks that can be linked individually to provide different services for the users. The project provides a configurator to create a Docker[3] virtualization environment which allows a structured way of creating the different servers for the different services. The docker-compose-file defines which building blocks of the BaSyx project should be instantiated and can configure their ports and behaviors. The BaSyx implementation provides an application programming interface (API) that allows other programs to connect to its different building blocks/models and send or receive data from them.

The BaSyx implementation further provides a building block that allows access to the information stored via a Graphical User Interface to the AAS environment. Another useful

[1] https://www.industrialdigitaltwin.org/en/
[2] https://www.basyx.org/
[3] https://www.docker.com/

model is the persistent storage building-block which stores the information permanently in a database.

Per asset, a AAS is created as a DT which consists of the following submodels: a digital master and one or multiple digital shadows.

The digital master is used to store the common product information and Computer Aided Engineering (CAE) models used in the design and development. Further, the digital master comprises physics-based and ML-based models as *Submodel Elements* of a submodel, which can, for example, be files. Different versions of the physics- and data-driven models can be differentiated by using a naming convention that is both, human and machine understandable. In addition to files, properties or references can also be *Submodel Elements* of a submodel. If necessary, parameters can be added as properties (key-value-pairs) to the submodels to provide necessary information for the model execution.

For each instance of the product, a separate submodel in the AAS is created as a digital shadow to capture all related data for this individual entity. The API allows other software to transfer data into the corresponding digital shadow, e.g. measurement data from a sensor system. These stored measurement data is required for model execution.

The BaSyx implementation also provides a message broker as a building block that can help to automate workflows. The message broker can provide different topics (e.g. digital shadows) that can be subscribed by a client. According to the Publish-Subscribe-Model, a client (e.g. a program that runs the RUL calculations) can subscribe to a topic (e.g. measurement data of a digital shadow) and will receive an information once new data is available. This automatism helps to trigger the calculations once new data is available and to make use of the capabilities of the digital twin.

In Figure 3 a screenshot is shown from the current implementation of the BaSyx framework with our application example. For the train, a AAS is created (left column) that consists of a digital master and a digital shadow (middle column). If a submodel element from the middle column is selected, its content is shown in the right column.

The implementation demonstrates how hybrid model structures can be embedded in a DT architecture without prescribing a universal reference architecture.

4.3. Data Flow and Hybrid Model Execution

The overall data flow within the implemented DT architecture is illustrated in Figure 4. Measurement features of the physical system are acquired by the sensing infrastructure and, if required, pre-processed or transformed into a standardized format. These data are subsequently transferred via the API into the digital shadow of the corresponding asset instance.

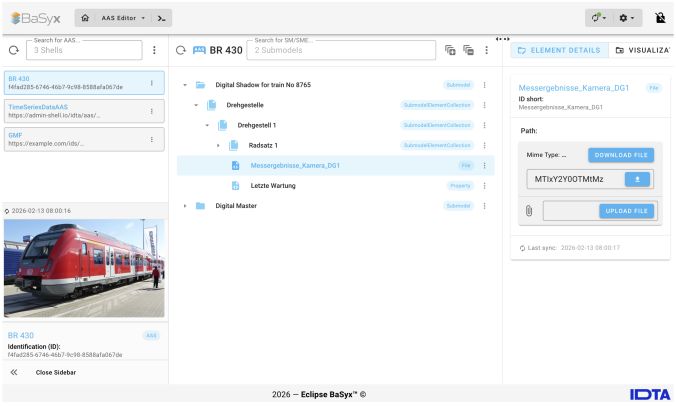

Figure 3. Screenshot of the implementation of the DT in the BaSyx AAS-Framework

Within the DT architecture, the AAS server coordinates the interaction between the digital master and the digital shadows. The digital shadow provides the runtime-specific data context, while the digital master maintains the model structures and associated product and configuration information.

When new measurement data arrive in the digital shadow, an event-based trigger notifies an message broker. The broker initiates the execution of the hybrid model structure to calculate the RUL and provides access to the newly stored data within the AAS environment.

Depending on the selected hybrid coupling strategy, model execution may involve output-level residual correction, internal parameter adaptation, or constraint-based inference in physics-informed configurations.

The hybrid model structure processes the incoming data and computes degradation indicators and an updated RUL estimate. The resulting outputs are written back to the AAS and stored in the corresponding digital shadow. These results are then made available to analytics and higher-level decision-support models. If required, additional automated processes can be triggered through the message broker.

The summarized execution process is structured as follows

1. New measurement data enters digital shadow
2. Event-based trigger initiates hybrid model execution
3. The model processes the data according to its structural configuration
4. Degradation indicators or RUL values are returned to the AAS server.
5. Results are stored and made accessible to analytics and decision-support layers.
6. If required, update mechanisms may be initiated

While the execution flow remains structurally consistent, the specific architectural implications differ depending on the cho-

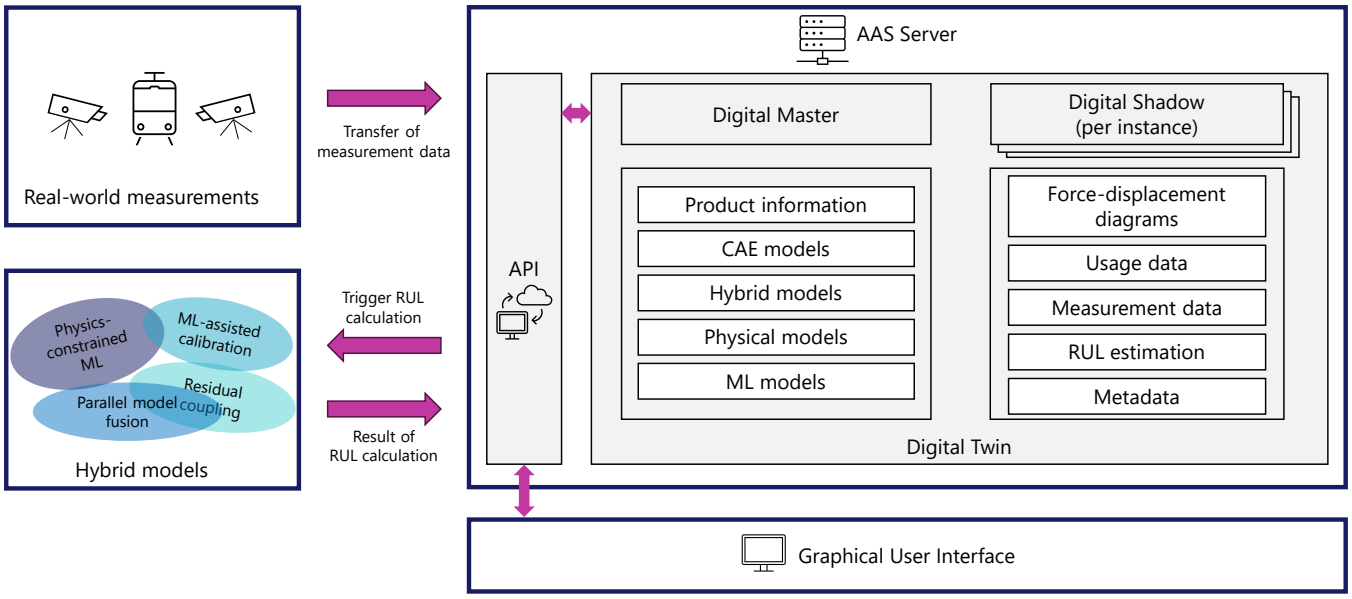

Figure 4. Digital Twin Architecture

sen hybrid model structure. These differences are analyzed in the following subsection.

4.4. Architectural Implications of Hybrid Configurations

While Section 4.1 derived general architectural requirements, the following discussion differentiates how these requirements manifest depending on the selected hybrid coupling strategy.

Different hybrid model structures impose different constraints on data persistence, execution orchestration, and update coordination within the DT architecture.

Residual coupling enables independent updating of the ML-based model, supporting modular retraining and localized deployment. Parameter learning requires persistent storage and versioning of calibrated parameter states, as internal model behavior depends directly on parameter updates. Parallel model fusion necessitates coordination mechanisms for aggregating and validating outputs from independently executed models. Physics-informed ML may require coordinated retraining and redeployment, as updates typically affect the complete model structure due to its integration depth. Beyond structural execution patterns, lifecycle management becomes a central integration aspect. Hybrid model structures require clearly defined update strategies, traceable version histories, and mechanisms to distinguish between full model redeployment and partial model updates. In addition, the interaction between different system or models must be considered to ensure consistency across the DT environment. Sensor data used for model updates must therefore be systematically preprocessed and validated to maintain structural consistency.

In this project context, the DT architecture provides the operational environment for managing hybrid model structures, including data persistence, execution control, and lifecycle coordination. The added value emerges from the structured integration of model configuration and architectural orchestration rather than from the DT in isolation.

Based on the derived integration principles, a structured development workflow can be formulated:

1. Define PHM objective
2. Configure hybrid model structure
3. Derive architectural integration requirements
4. Implement model embedding within DT services
5. Establish update and lifecycle mechanisms
6. Deploy and monitor system behavior

The presented architecture represents a project-specific realization of these integration requirements. While implemented within a rail-focused PHM context, the structural linkage between hybrid model configuration and DT architecture design indicates the potential for a more systematic derivation of DT frameworks tailored to hybrid model lifecycle management. This perspective is further discussed in the concluding section.

5. DISCUSSION

This paper presents a structured approach for integrating hybrid model structures into DT architectures in PHM contexts, illustrated using a rail-sector scenario. The main relevance of the approach lies in making the structural dependencies between hybrid model configuration and DT architecture ex-

plicit. This is important because different hybrid model structures impose different requirements on persistence, execution orchestration, update mechanisms, and lifecycle coordination. Therefore, hybrid model structures should not be treated merely as isolated modeling elements, but as components whose structural properties need to be considered when designing or adapting a DT architecture for operational PHM use.

The proposed integration workflow provides methodological guidance for aligning hybrid model configuration with DT architectural requirements in maintenance-oriented PHM systems. Rather than prescribing a single implementation pattern, the approach supports consistent reasoning about the dependencies between model structure and system architecture.

This work focuses on conceptual development and structural systematization. It does not provide a comprehensive empirical evaluation, quantitative benchmarking, or performance comparison across different hybrid configurations. Consequently, the proposed design space and workflow should be understood as conceptual guidance rather than as a validated decision model.

In the presented implementation, the DT architecture follows the definition proposed by WiGeP. Given the variety of DT definitions in the literature, alternative conceptualizations may also be valid. The architectural realization in this work therefore represents a project-specific operationalization of the derived integration requirements rather than a universal DT reference framework. Nevertheless, the underlying linkage between hybrid model structure and DT architectural requirements is intended to be transferable to comparable PHM contexts.

6. CONCLUSION AND OUTLOOK

This paper systematically analyzed hybrid model structures and examined their integration within DT architectures in PHM contexts. Based on a structural classification of hybrid coupling strategies, a requirement-driven configuration framework was developed to enable consistent positioning of hybrid model structures. The resulting architectural implications were then translated into a structured integration workflow for embedding, executing, and coordinating hybrid models within a DT environment.

By explicitly linking hybrid model configuration with DT architectural requirements, this study contributes to a more integrated understanding of how PHM systems can be designed and operated in a structured and scalable manner. Rather than treating hybrid modeling and DT architectures as independent concepts, the proposed approach establishes a coherent relationship between model structure and system architecture.

The long-term objective is to move from the presented integration workflow toward a more comprehensive methodology for operating hybrid model structures within DT-based PHM systems.

As a next step, future work will address the automation of the developed integration workflow and the systematic extension of the DT architecture to additional models in order to represent larger system scopes.

Beyond these implementation-oriented extensions, further research is required to investigate how DT architectures can be systematically derived from hybrid model lifecycle requirements. The structural dependency identified in this study between hybrid model configuration and DT architecture design suggests the potential for developing DT frameworks explicitly tailored to hybrid modeling environments. This bidirectional relationship between hybrid model structure and digital twin architecture constitutes a promising direction for future research.

ACKNOWLEDGMENT

The project is funded by the Federal Ministry of Transport on the basis of a resolution passed by the German Bundestag, the German Federal Ministry of Transport (BMV) and the German Centre for Future Mobility (DZM) , enableATO project, project number 19DZ23002.

REFERENCES

Boss, B., Malakuti, S., Lin, S.-W., Usländer, T., Clauer, E., Hoffmeister, M., ... Flubacher, B. (2020). Digital twin and asset administration shell concepts and application in the industrial internet and industrie 4.0. *Plattform Industrie*, *4*, 13–14.

Chen, C., Fu, H., Zheng, Y., Tao, F., & Liu, Y. (2023, December). The advance of digital twin for predictive maintenance: The role and function of machine learning. *Journal of Manufacturing Systems*, *71*, 581–594. doi: 10.1016/j.jmsy.2023.10.010

Chen, S., Bekar, E. T., Bokrantz, J., & Skoogh, A. (2025). Ai-enhanced digital twins in maintenance: Systematic review, industrial challenges, and bridging research–practice gaps. *Journal of Manufacturing Systems*, *82*, 678–699.

Chen, S., Turanoglu Bekar, E., Bokrantz, J., & Skoogh, A. (2025). AI-enhanced digital twins in maintenance: Systematic review, industrial challenges, and bridging research–practice gaps. *Journal of Manufacturing Systems*, *82*, 678–699. doi: 10.1016/j.jmsy.2025.07.006

Chowdhury, S., Ghosh, A., Acharya, S., Pal, P., Roy, S., & Lahiri, S. (2025). Transforming chemical process engineering: The role of AI and machine learning in revolutionizing process systems. *Canadian Journal of Chemical Engineering*. doi: 10.1002/cjce.70032

Dierend, H., Altun, O., Mozgova, I., & Lachmayer, R. (2022,

September). Management of research field data within the concept of digital twin. In *Advances in system-integrated intelligence* (p. 205–214). Springer International Publishing. doi: 10.1007/978-3-031-16281-7_20

Dietterich, T. G. (2000). Ensemble Methods in Machine Learning. In G. Goos, J. Hartmanis, & J. Van Leeuwen (Eds.), *Multiple Classifier Systems* (Vol. 1857, pp. 1–15). Berlin, Heidelberg: Springer Berlin Heidelberg. doi: 10.1007/3-540-45014-9_1

Gao, T., Zhu, H., Wu, J., Lu, Z., & Zhang, S. (2024, May). Hybrid physics data-driven model-based fusion framework for machining tool wear prediction. *The International Journal of Advanced Manufacturing Technology*, *132*(3), 1481–1496. doi: 10.1007/s00170-024-13365-6

Gardner, P., Rogers, T., Lord, C., & Barthorpe, R. (2021, May). Learning model discrepancy: A Gaussian process and sampling-based approach. *Mechanical Systems and Signal Processing*, *152*, 107381. doi: 10.1016/j.ymssp.2020.107381

Gherghina, I.-S., Bizon, N., Iana, G.-V., & Vasilică, B.-V. (2025). Recent Advances in Fault Detection and Analysis of Synchronous Motors: A Review. *Machines*, *13*(9). doi: 10.3390/machines13090815

Gupta, H., & Kundu, P. (2024, June). Digital Twin Development for Feed Drive Systems Condition Monitoring and Maintenance Planning. *PHM Society European Conference*, *8*(1), 4. doi: 10.36001/phme.2024.v8i1.3952

Ifeanyi, A., & Coble, J. (2025, June). Enhancing System-Level Prognostics with Structural Information: A Graph-Based Approach. In *2025 IEEE International Conference on Prognostics and Health Management (ICPHM)* (pp. 1–8). doi: 10.1109/ICPHM65385.2025.11061821

Jacob, B., & Santhosh Kumar, G. (2025). Towards Artifacts-Based Behavior Modeling of Cyber-Physical Systems for Building Digital Twins. In *Proceedings of the Annual International Conference on Control, Automation and Robotics, ICCAR* (pp. 574–579). doi: 10.1109/ICCAR64901.2025.11073032

Kareem, A., Domingo, D., & Hur, J.-W. (2025). A Hybrid IoT-Based Framework for Real-Time Parameter Estimation and Predictive Maintenance in Hydraulic Excavators. *IEEE Internet of Things Journal*. doi: 10.1109/JIOT.2025.3605642

Karniadakis, G. E., Kevrekidis, I. G., Lu, L., Perdikaris, P., Wang, S., & Yang, L. (2021, May). Physics-informed machine learning. *Nature Reviews Physics*, *3*(6), 422–440. doi: 10.1038/s42254-021-00314-5

Krespach, V., Blum, N., Pottmann, M., Rehfeldt, S., & Klein, H. (2025, March). Improving extrapolation capabilities of a data-driven prediction model for control of an air separation unit. *Computers & Chemical Engineering*, *194*, 108953. doi: 10.1016/j.compchemeng.2024.108953

Lenzi, A., Bessac, J., Rudi, J., & Stein, M. L. (2023, September). Neural networks for parameter estimation in intractable models. *Computational Statistics & Data Analysis*, *185*, 107762. doi: 10.1016/j.csda.2023.107762

Ma, J., Luo, C., Li, K., Bai, M., Dong, S., & Yin, L. (2025, October). Leveraging hybrid knowledge-based and data-driven modeling techniques for complex contact/impact phenomena. *Mechanism and Machine Theory*, *214*, 106122. doi: 10.1016/j.mechmachtheory.2025.106122

Mayr, S., Gross, T., Krenn, S., Kunze, W., & Zehetner, C. (2024, January). Digital Twin-based Predictive Maintenance for Sheet Metal Bending. *Procedia Computer Science*, *232*, 504–512. doi: 10.1016/j.procs.2024.01.050

Najafzadeh, M., & Yeganeh, A. (2025). Ai-driven digital twins in industrialized offsite construction: A systematic review. *Buildings*, *15*(17), 2997.

Patel, R. S., Bhartiya, S., & Gudi, R. D. (2022, January). Physics Constrained Learning in Neural Network based Modeling. *IFAC-PapersOnLine*, *55*(7), 79–85. doi: 10.1016/j.ifacol.2022.07.425

Raissi, M., Perdikaris, P., & Karniadakis, G. E. (2019, February). Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, *378*, 686–707. doi: 10.1016/j.jcp.2018.10.045

Rajkolhe, R. V., Bhagwat, D. S. S., & Deshmukh, D. P. V. (2025, June). Dynamic weighted ensemble model for predictive optimization in green sand casting: Advancing industry 4.0 manufacturing. *MethodsX*, *14*, 103393. doi: 10.1016/j.mex.2025.103393

Rofatto, V. F., Almeida, L. F. R. D., Matsuoka, M. T., Klein, I., Veronez, M. R., & Da Silveira, L. G. (2026, January). Residual-based neural network for unmodeled distortions in 2D coordinate transformation. *Geodesy and Geodynamics*, *17*(1), 104–119. doi: 10.1016/j.geog.2025.06.005

Şahin, T., Wolff, D., von Danwitz, M., & Popp, A. (2024). Towards a Hybrid Digital Twin: Fusing Sensor Information and Physics in Surrogate Modeling of a Reinforced Concrete Beam.. doi: 10.1109/SDF63218.2024.10773885

Stark, R., Anderl, R., Thoben, K.-D., & Wartzack, S. (2020). Wigep-positionspapier:„digitaler zwilling “. *Zeitschrift für wirtschaftlichen Fabrikbetrieb*, *115*(s1), 47–50.

Tao, F., Qi, Q., Wang, L., & Nee, A. (2019). Digital twins and cyber–physical systems toward smart manufacturing and industry 4.0: Correlation and comparison. *Engineering*, *5*(4), 653–661.

Von Krannichfeldt, L., Orehounig, K., & Fink, O. (2025, August). Combining physics-based and data-driven modeling for building energy systems. *Applied Energy*, *391*, 125853. doi: 10.1016/j.apenergy.2025.125853

Von Rueden, L., Mayer, S., Sifa, R., Bauckhage, C., & Garcke, J. (2020). Combining Machine Learning and Simulation to a Hybrid Modelling Approach: Current and Future Directions. In M. R. Berthold, A. Feelders, & G. Krempl (Eds.), *Advances in Intelligent Data Analysis XVIII* (Vol. 12080, pp. 548–560). Cham: Springer International Publishing. doi: 10.1007/978-3-030-44584-$3_4$3

Von Stosch, M., Oliveira, R., Peres, J., & Feyo De Azevedo, S. (2014, January). Hybrid semi-parametric modeling in process systems engineering: Past, present and future. *Computers & Chemical Engineering*, *60*, 86–101. doi: 10.1016/j.compchemeng.2013.08.008

Wohlleben, M., Bender, A., Peitz, S., & Sextro, W. (2022). Development of a Hybrid Modeling Methodology for Oscillating Systems with Friction. In G. Nicosia et al. (Eds.), *Machine Learning, Optimization, and Data Science* (Vol. 13164, pp. 101–115). Cham: Springer International Publishing. doi: 10.1007/978-3-030-95470-3_8

Wohlleben, M., Röder, B., Ebel, H., Muth, L., Sextro, W., & Eberhard, P. (2024, August). Hybrid modeling of multibody systems: Comparison of two discrepancy models for trajectory prediction. *PAMM*, *24*(2), e202400027. doi: 10.1002/pamm.202400027

Zachariades, C., & Xavier, V. (2025). A review of artificial intelligence techniques in fault diagnosis of electric machines. *Sensors (Basel, Switzerland)*, *25*(16), 5128.

Zhou, Z.-H. (2012). *Ensemble Methods: Foundations and Algorithms*. New York: Chapman and Hall/CRC. doi: 10.1201/b12207