GOFORE

Gofore White Paper: Creating value with digital twins

Contents

1 Motivation and scope

1.1 Executive summary

Digital twin initiatives often start as isolated pilots, driven by a single team, a single tool, or a single use case. They create value quickly—but without shared vocabulary, validation discipline, and architectural ownership, they rarely scale beyond their original context. This white paper provides a technically grounded, evidence-based approach for moving from “interesting pilots” to sustainable, decision-relevant capability.

The white paper is built around a simple idea: architecture should be chosen based on decision and validation needs—not platform features in isolation. Simulation, virtual commissioning (VC), and extended reality (XR) are introduced early to make validation loops explicit: scenarios, gates, artefacts, and measurable criteria for readiness. From there, platform archetypes are evaluated as architectural spines and complementary capabilities—helping organizations choose a dominant spine and avoid attempting to build everything everywhere at once.

For decision-makers, the white paper provides a practical framework for making trade-offs visible: lifecycle governance versus speed of validation, semantics versus integration cost, and long-term sustainability versus time-to-value. It also makes explicit the organizational and governance conditions required for these practices to be adopted in everyday decision-making—not only demonstrated in isolated pilots. For R&D and engineering, the white paper introduces concrete patterns such as VC-before-HIL gate thinking, scenario-driven regression, and the use of advanced methods (surrogates and Bayesian calibration) to improve speed and trust once architectural foundations are in place.

Key takeaway: Digital twins create durable value when architectures, validation practices, and governance evolve together—and when ownership, decision authority, and everyday ways of working support their use beyond isolated pilots. The final chapter summarizes recurring challenge clusters, outlines pragmatic implementation paths, and provides a roadmap for progressing from pilot initiatives to scalable, decision-relevant capability without turning the white paper into a project-specific solution description.

1.2 Purpose and audience

This white paper outlines how a **Digitalised Product Lifecycle** approach can combine virtual commissioning (VC), configurable simulation/digital twins, and uncertainty-aware analytics into practical, repeatable ways of working. The term “digitalised” is used deliberately to emphasize end-to-end working practices and traceability across lifecycle artefacts, not only the digitization of a product. The document is written for decision-makers and R&D practitioners who need a concise, technically sound view. In this white paper, Digitalised Product Lifecycle refers to a lifecycle-spanning way of working where requirements, configurations, validation artefacts (e.g., scenarios), and operational evidence remain traceable across tools and phases. The intent is to make decision-making repeatable rather than tool-dependent.

1.3 Document type and sourcing

This is a technical white paper with rigorous referencing. Peer-reviewed literature is prioritised; well-regarded industry white papers are used to complement areas where journals do not cover current implementation details. All claims are traceable to sources.

1.4 Positioning

The white paper is a compact playbook for concept-level work: it does not prescribe products or stacks, and it does not expose non-public architectures. The focus is on terms, patterns and evidence that can be carried into a formal statement of work if needed.

1.5 Scope (in scope)

This white paper covers the following aspects:

- Definitions and Standards: cross-domain terminology for DT/VC and uncertainty, plus a minimal governance baseline.
- Virtual Commissioning and XR: HIL (Hardware-in-the-Loop) -free storyboard (roles, artefacts, control points) that creates early value and connects to later phases.

- Platforms and Tooling: vendor-neutral approach-archetype comparison for how digital twins are managed and integrated across the lifecycle.
- Autonomy and AI: two compact, evidence-backed method exemplars (surrogate-based calibration for speed; Bayesian/KOH discrepancy reduction for fidelity) with KPIs (Key Performance Indicators, e.g., RMSE, MAPE, PICP).
- Challenge Map and Vision: condensed view of key risks (interoperability/semantics, fidelity vs. latency, governance/assurance) and near-term options.

1.6 Out-of-scope

This white paper omits the following:

- No system tailoring or implementation (no code, pipelines, integrations).
- No non-public architectures.
- No vendor selection or tool short-lists; no customer-specific effort/cost estimates.

1.7 Research problem

There is no single, concise reference that simultaneously:

- aligns lifecycle terminology and interfaces
- frames VC/XR so it produces value before HIL
- compares platform approaches without implying a stack, and
- demonstrates uncertainty-aware methods with credible KPIs.

This paper fills that gap at concept level.

1.8 Research questions

The white paper is guided by the following research questions. They are intentionally practical: each question maps to a chapter and to an explicit set of artefacts (criteria, scorecards, vignettes, and validation gates).

- **RQ1 — Operating model and vocabulary:** What minimal operating model and shared vocabulary enable consistent work across DT/VC, platforms and analytics?

- **RQ2 — VC/XR before HIL:** How should VC/XR be expressed (scenarios, roles, artefacts, governance) so that it produces early value and links cleanly to later phases?
- **RQ3 — Platforms and Tooling:** Which platform touchpoints (simulation runtime, control/orchestration, data/archives, planning, MLOps) matter at concept level, and how do approach archetypes compare against consistent criteria?
- **RQ4 — Autonomy and AI methods:** Which evidence-backed methods (e.g., surrogate calibration; Bayesian/KOH discrepancy reduction) best illustrate decision-grade modelling, and which KPIs should be used?
- **RQ5 — Outcomes and risks:** How do we connect findings to concrete outcomes for decision-making and R&D, and what is the realistic challenge-map with mitigations?

1.9 Motivation

This white paper is motivated by practical industrial initiatives where digital twin concepts are explored across the lifecycle. The white paper focuses on patterns, trade-offs, and decision criteria that generalize beyond a single implementation.

2 Concepts, definitions, and lifecycle framing

2.1 Purpose of this chapter

This chapter sets the **terminology,** and the **standards baseline** we will use throughout the white paper. It is not tied to any specific existing implementation, but rather re-uses widely adopted industrial standards where possible. The intent is not to reproduce the standards themselves but to provide a common frame that later chapters (VC/XR, platforms, methods, challenge-map) can reference.

2.2 Core terminology

We use the following concise definitions as working terminology throughout the white paper.

- **Digital Twin (DT):** manufacturing-oriented framework where a physical asset/process and its digital representation are linked for analysis, prediction and control; we align terminology with ISO 23247.

- **Digital Thread:** lifecycle-spanning flow of information and decisions (design → manufacturing → support) connecting requirements, models, tests and operational data; framed in NIST programmes.
- **Asset Administration Shell (AAS):** technology-agnostic metamodel for representing industrial assets and their submodels; official schemas provided by IDTA (JSON/RDF/XML).
- **Digital Factory (DF) framework:** IEC 62832-1 model elements, relationships and information flow for production systems across discrete/batch/continuous domains.

Terms are used deliberately in a non-normative way. Further terminology and its use in the white paper is situated in the glossary.

2.3 Standards we reference (purpose and how we use them)

The standards listed below form the non-vendor baseline for terminology, interoperability, and runtime patterns used in later chapters.

- **ISO 23247 — Digital twin framework for manufacturing**
  Part 1 (Overview and general principles): provides the overarching DT conceptual frame and vocabulary we follow. → We adopt its high-level structure to keep cross-chapter terminology consistent.
  Part 4 (Information exchange): points to how information exchange is conceptualised in DT contexts. → **Feeds chapter 4 Interoperability** criteria.
- **IEC 62832-1 — Digital Factory framework**
  Defines model elements, relationships and information flows for production systems. → Gives us factory-level context for “asset”, “relation” and “information flow” terms.
- **AAS (IDTA) — Specification, Part 1 (Metamodel)**
  The industrial **metamodel** for asset representations and submodels; normative JSON/RDF/XML schemas are openly available (v3.x). → Reference for **semantic interoperability** and an anchor for our “federated” platform pattern.
- **FMI — Functional Mock-up Interface (2.0 / 3.0.x)**
  Open standard for **model exchange** and **co-simulation** used across >270 tools; we use FMI as the canonical way to describe cross-tool model coupling in this white paper. → Supports **chapter 4** “Interop and co-sim” criteria.
- **OPC UA 62541-14:2026 — PubSub (2nd edition)**
  Publish–subscribe communication model complementing client–server; 2026 edition adds, among others, a **Quantity Model** and clarifies

ValuePrecision rules. → Our baseline for **runtime data distribution** and "Real-time"/"Interop" criteria in **chapter 4.**

- IEEE 1516-2025 — HLA (High Level Architecture, "HLA 4")
  Standard for distributed simulation (federations, RTI services, time management, FOMs). → A reference for co-simulation across distributed components in chapter 4.
- **NIST DT/Thread materials**
  We use NIST publications and programme pages for **neutral context** (DT standards landscape, Digital Thread framing). → Sources for background and roadmap context.

2.4 Interoperability building blocks (used later in the platform comparison)

These building blocks are referenced explicitly in chapter 4 scorecards and in the VC gate discussion in chapter 3.

- **Model exchange and co-simulation — FMI**
  We treat FMI as the default mechanism to talk about **black-box model exchange** (ME) and **co-simulation** (CS) between tools/domains; it underpins our *Interop and co-sim* criterion in **chapter 4**.
- **Runtime data and events — OPC UA PubSub**
  We use PubSub as the reference pattern for scalable **data/event distribution** between devices, edge and cloud; we call out the **2026** edition because it is current and relevant to quantity/unit handling and precision semantics.
- **Distributed simulation — HLA**
  For federated, time-managed simulation across multiple simulators, we reference **IEEE 1516-2025**; this is particularly relevant when platform archetypes must interoperate in a **federation** (see **chapter 4**).
- **SIL/HIL (terms used in chapter 3)**
  We use **SIL** (Software-in-the-Loop) for software-only execution/validation in a simulated plant and **HIL** (Hardware-in-the-Loop) for connecting real controllers/hardware to a real-time simulation before prototyping.

2.5 Concepts and framing

In industrial environments, information content and functional boundaries often resemble **ISA-95** style models. While this white paper does not prescribe ISA-95 compliance, the proposed lifecycle views and semantic contracts are compatible

with such established structures. This alignment helps reduce adoption friction in contexts where similar models are already familiar.

2.6 How to interpret references

This white paper prioritizes peer-reviewed literature and widely adopted industrial standards as primary reference points. Implementation-oriented sources are used selectively to reflect current practice where formal literature is limited.

References are used to clarify concepts and support decision-relevant reasoning, not to prescribe specific solutions, implementations, or vendor approaches.

2.7 Assumptions and limitations

The architectural guidance in this white paper is intentionally scoped. The discussion does not prescribe implementations, pipelines, or system-specific tailoring. Non-public reference architectures are explicitly excluded.

When deeper standard details are required, the white paper relies on open summaries (e.g. NIST, IDTA) or publicly available access routes, rather than reproducing full standard texts for this study.

References

Standards and neutral framing sources are listed in the references (ISO/IEC/IEEE/IDTA/NIST), aligned with this chapter's terminology.

3 Simulation/VC/XR as decision enablers

Executive takeaway

From a decision-maker's perspective, simulation, virtual commissioning (VC), and XR are primarily tools for reducing late-stage risk. They allow validation of assumptions before irreversible commitments are made, improving confidence in decisions rather than just improving engineering efficiency.

3.1 Purpose and scope

Virtual Commissioning (VC) and Extended Reality (XR) deliver most value early in a digital twin pipeline, where validating behaviour virtually is cheaper and safer than uncovering problems in HIL or field trials. The focus is on **risk reduction and iteration speed**: validate behaviour in virtual environments before committing to expensive or hazardous physical tests (e.g., HIL or field trials).

> Validation becomes a decision gate, not only a test activity.

We cover:

- A "VC before HIL" pipeline (roles, artefacts, checkpoints)
- Scenario-driven validation and regression testing
- Training simulators and XR as an extension of the same asset models and scenario library
- Practical metrics for readiness and value (coverage, lead time, defect escape rate)

This chapter focuses on **simulation, virtual commissioning, and XR as decision-enabling practices**, not as tools or platforms. The intent is to illustrate the types of validation, uncertainty reduction, and iteration loops that digital twin architectures must support across the lifecycle.

3.2 Virtual commissioning before HIL: Storyboard and gate

In a 'VC before HIL' pattern, scenario-driven virtual validation is the primary gate to hardware testing: problems are found and fixed virtually before any hardware is committed (Figure 3-1). The purpose of this pattern is to reduce late-stage risk by validating assumptions, configurations, and behaviour in a controlled and repeatable virtual environment before committing to hardware integration.

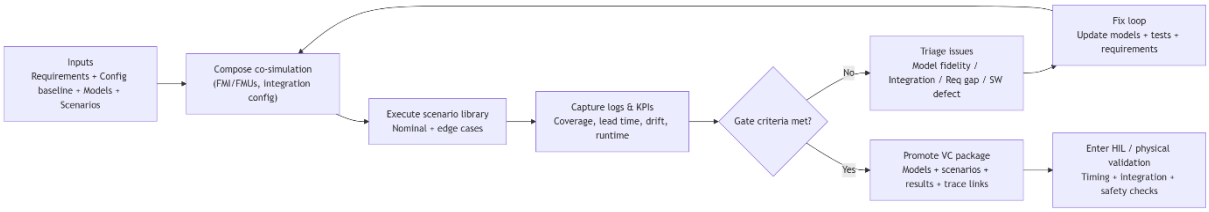

Figure 3-1: "VC before HIL: scenario-driven virtual validation as the gate to hardware testing."

The pipeline begins with a set of well-defined inputs, including requirements, a configuration baseline, system models (plant and controller), and scenario definitions. These inputs are composed into a co-simulation setup, typically using FMI-compatible model exchange, enabling deterministic and repeatable regression runs. Scenarios are then executed across nominal conditions as well as selected edge cases, with logs and key performance indicators captured to support analysis.

Validation results are evaluated against explicit gate criteria. If the criteria are not met, issues are triaged and classified, for example as model fidelity problems, integration issues, requirement gaps, or software defects. Corrective actions are applied by updating models, tests, or requirements, and the validation loop is repeated until the gate conditions are satisfied. Once the criteria are met, validated artefacts, including models, scenarios, test results, and trace links, are promoted as a readiness package for HIL or physical validation, where timing, integration, and safety aspects are assessed.

Roles and responsibilities in this pattern are clearly separated but tightly coupled. Control and automation teams are responsible for controller logic, timing constraints, and HIL readiness. Plant, physics, or environment specialists own model fidelity and parameterization. Test and QA functions maintain the scenario library, regression suites, and coverage tracking. Product owners or system engineers define acceptance criteria and decision gates that determine progression to the next validation stage.

The artefacts produced and consumed in this workflow include FMUs (Functional Mock-up Units) and co-simulation configurations, scenario libraries, test reports, log schemas, and KPI dashboards. Crucially, these artefacts are linked back to requirements and configuration baselines, enabling traceability of what changed and why a given validation outcome was accepted.

Gate criteria typically combine coverage, quality, and performance considerations. Examples include achieving sufficient coverage of must-have scenarios, absence of high-severity defects in recent regression runs, model drift remaining below an agreed threshold relative to reference data, and runtime performance staying within defined budgets for step time and determinism. While the exact thresholds vary by context, the key principle is that progression to HIL is governed by explicit, measurable conditions rather than informal judgement.

This pattern highlights virtual commissioning as a **decision gate**, not merely a testing activity. By making validation loops explicit and artefact-driven, it establishes concrete requirements for lifecycle traceability, semantic consistency, and runtime observability—requirements that directly inform the platform architecture choices discussed in the following chapter.

3.3 Scenario vignettes

The same validation principles hold across very different settings — a forestry machine programme, a tram driver simulator, and remote mining operations. They are not exhaustive case studies, but representative slices that highlight alternative entry points, validation approaches, and trade-offs. Together, they show how similar principles support decision-making across varying operational and technological settings.

They have been selected to reflect common industrial starting points and to illustrate different ways organizations typically approach value creation, validation, and scaling of digital twin capabilities.

3.3.1 Forestry machine simulator / operator assistance development

In a forestry machine programme, a simulator-based workflow replaces slow, costly field testing as the primary way to validate operator assistance functions (e.g., perception, guidance logic) before field deployment. The simulator provides repeatable “forest environment + machine + sensor” conditions, enabling systematic coverage of edge cases (e.g., visibility/weather variation) and rapid iteration cycles.

Why it matters
Field testing is costly and slow; simulator scenarios enable a measurable regression suite, and the same environment can later support operator training modules.

Decision context
Key decisions include control logic tuning, operator assistance behaviour, and training readiness under varying terrain, load, and environmental conditions. These decisions must be made before large-scale field deployment and often need to be revisited as configurations, software, or operational assumptions change.

Validation need
Validation requires repeatable scenario execution, comparison of outcomes across configurations, and the ability to regress previously validated behaviour after changes. Simulation must support both engineering validation and operator training, using the same or closely aligned models and scenarios to avoid divergence between “tested” and “trained” behaviour.

Architectural implication
This context highlights the need for architectures that treat simulation artefacts and scenarios as first-class citizens, support lifecycle traceability from configuration to validation results, and allow reuse of models and scenarios across engineering and training. Platform choices must enable iterative validation loops rather than one-off simulations.

3.3.2 Tram driver training simulator

A tram driver training simulator shows how closely training and validation overlap: both need the same deterministic replay, structured telemetry, and scenario variation across routes, traffic, and weather. Gamification patterns can be used to make learning objectives explicit and to provide structured feedback (e.g., goal indicators, postponed feedback, real-time instructive feedback) without compromising safety or realism.

Why it matters
Training scenarios overlap with validation scenarios: both require deterministic replay, structured telemetry and feedback, and scenario variation.

Decision context
Decisions include timetable robustness, control and signalling behaviour, operator training completeness, and system response to disturbances such as delays,

infrastructure faults, or unexpected traffic interactions. These decisions affect safety, service reliability, and public trust.

Validation need
Effective validation requires system-level simulation that can combine vehicle dynamics, infrastructure constraints, and operational rules. Scenarios must be repeatable and comparable over time, and validation results must remain traceable to specific system configurations and assumptions.

Architectural implication
This case emphasizes the importance of architectures that support system-level co-simulation, scenario management, and traceability across configuration changes. Simulation is not an isolated activity but a mechanism for validating operational readiness, which places concrete requirements on integration, data consistency, and lifecycle management.

3.3.3 Mining (remote operations with intermittent connectivity)

Mining makes the case for edge-first design: intermittent connectivity, constrained bandwidth, and delayed feedback force control to stay edge-local — and the same constraints recur in other remote, high-latency domains such as aerospace.

Why it matters
The same patterns apply to any remote, high-latency environment: delayed feedback, asynchronous operations, and strict safety boundaries.

Decision context
Mining operations must support safe and reliable decision-making under degraded or intermittent network conditions. Decisions include what actions are allowed locally at the edge versus centrally, how to maintain deterministic control for safety-critical functions, and how to handle non-safety-critical state using eventual consistency. Additional considerations include robust time-stamping and provenance for delayed telemetry ingestion, the ability to validate rare but high-impact scenarios (such as faults or hazard conditions), and clear governance rules defining who can change what when connectivity is limited or unavailable.

Validation need
Validation in this context requires explicit definition of acceptable state staleness for different signal classes, along with mechanisms to measure and enforce these limits. Systems must be able to demonstrate buffer durability, for example by quantifying the percentage of events successfully delivered after an outage. Safety envelope compliance must be verifiable under link loss, and incident response

times must be assessed even when telemetry arrives with delay. These validation needs extend beyond nominal operation and focus on degraded and exceptional conditions.

Architectural implications
This case highlights the importance of architectures where control and automation functions remain edge-local, while central systems consume summarized or delayed state. Runtime messaging patterns must support asynchronous delivery, buffering, replay, and correct ordering of events, rather than assuming low-latency streams. Explicit semantic contracts, such as asset and submodel identifiers, reduce integration brittleness when data arrives late or out of order. Simulation and VC/SIL pipelines remain valuable in this context, as the same scenarios can be used to validate both nominal operation and degraded-link behaviour. In addition, monitoring must include explicit "freshness" KPIs, such as staleness windows and gap detection, to make delayed or missing data visible and actionable.

3.4 Metrics and reporting

Without explicit metrics, validation effort in simulation, VC, and XR cannot be shown to reduce risk or improve decision quality. The following metrics are recommended to track progress and effectiveness across VC/XR programs, regardless of the underlying tools or platforms.

Coverage measures how comprehensively relevant scenarios have been executed compared to the required set. This includes both scenario coverage and, where applicable, functional coverage. High coverage indicates that validation effort aligns with the intended operational envelope rather than isolated test cases.

Lead time captures the time from defect detection to verified fix in regression. Short lead times are a strong indicator that simulation and VC are effectively integrated into development workflows and support rapid iteration rather than late-stage discovery.

Defect escape tracks defects discovered in HIL or field operation that should reasonably have been caught during VC. This metric directly reflects the quality and relevance of simulation-based validation and highlights gaps in scenarios or assumptions.

Scenario reuse indicates how often scenarios are reused across releases, variants, or related systems. High reuse suggests that scenarios are treated as durable assets rather than disposable test artefacts, supporting scalability and consistency.

Fidelity drift monitors deviations between simulation outputs and reference data under defined conditions. Tracking drift over time helps identify when models or assumptions require recalibration, especially as systems evolve.

Training KPIs (when applicable) include metrics such as completion rates, error patterns, and time-to-proficiency. When simulation environments are shared between engineering validation and training, these metrics help ensure that training reflects validated system behaviour rather than diverging representations.

Together, these metrics provide visibility into both the effectiveness and sustainability of simulation-based validation practices, enabling informed decisions about where to invest further effort or refinement.

These metrics are consistent with common VC and XR reporting practices discussed in the sources referenced at the end of this chapter.

3.5 Risks and mitigations

The risks that undermine simulation and VC are a small, recurring set — and most are architectural rather than tool-specific.

A common risk is the **trade-off between model fidelity and schedule pressure**. Over-engineering models can delay validation, while overly simplified models may provide false confidence. This risk can be mitigated by defining “good enough” fidelity upfront, aligning it with decision needs, and favouring reuse of FMI-compatible or otherwise standardized models where possible.

Another frequent risk arises during **handover across validation stages**, such as from SIL to HIL. Without clear interface contracts and acceptance criteria, assumptions validated in simulation may not hold in later stages. Enforcing explicit interface definitions, versioning, and acceptance checks helps maintain continuity across stages.

Scenario fragmentation is a related risk, where scenarios are created ad hoc for specific tests and then discarded. This leads to poor regression capability and weak traceability. Treating scenarios as governed artefacts, with identifiers, ownership, and reuse expectations, mitigates this risk.

There is also a risk of **misaligned validation objectives**, where simulation is optimized for engineering convenience rather than operational relevance. This can be mitigated by linking scenarios and KPIs directly to decision contexts and by reviewing validation scope against real operational risks.

Finally, **over-reliance on tools without architectural support** can limit scalability. Without lifecycle traceability, semantic consistency, and runtime observability, simulation remains a local optimization. Aligning validation practices with architectural foundations reduces this risk and prepares the ground for broader reuse.

Many of these risks are architectural rather than tool-specific, reinforcing the need to evaluate platforms based on how they support validation practices across the lifecycle.

The risks and mitigations listed here reflect recurring failure modes described across VC and XR practice-oriented literature and are summarized here at a non-normative level.

3.6 Connecting the examples to architecture choices

Simulation and virtual commissioning are not isolated activities, but mechanisms for managing uncertainty, validating assumptions, and enabling informed decisions before and during operation. These practices place concrete requirements on platform architectures such as lifecycle traceability, semantic consistency, runtime observability, and support for iterative validation loops. The following chapter evaluates platform archetypes based on how well they support these requirements across the lifecycle.

References

Virtual commissioning foundations and digital twin framing [REF: Siemens “Defining the Digital Twin”]; simulation and Digital Thread perspectives for lifecycle governance [REF: NIST Digital Thread]; XR and simulation use in manufacturing contexts [REF: MDPI XR review]; tooling interoperability and co-simulation exchange [REF: FMI].

Case-specific evidence includes simulator efficacy in operator training [REF: “Efficacy of Simulator Technology for Forwarder Operator Training”], simulation-supported assistance system development [REF: Saaranen 2023], gamification in training simulators [REF: Nieminen 2023], and mining and remote operations contexts highlighting latency and edge constraints [REF: VTT/FIIF NGMining; REF: Emerson “Edge Computing Unearths New Value for Mining and Metals Applications”].

Additional VC and XR editorials and special issue pages are listed in the supplementary references.

4 Platform archetypes and architecture choices

Executive takeaway

Choosing a dominant platform spine is not a technical optimization problem, but a governance and ownership decision. It defines how capabilities are structured, who makes decisions, and how digital twins can scale across the organization.

4.1 Purpose

This chapter compares platform archetypes not to rank them, but to give a decision framework for selecting a dominant architectural spine and complementary capabilities — using criteria anchored in ISO 23247/IEC 62832, AAS, FMI, OPC UA PubSub, and HLA.

4.1.1 Archetypes covered in this chapter

Five archetypes frame the rest of the chapter, so that architectural trade-offs are made explicit before any tool or vendor discussion begins:

- **Simulation-centric / VC-first**: validation and scenario-driven regression as the primary integration environment.
- **PLM-centric**: lifecycle governance and configuration management as the system-of-record spine.
- **Data Lake / AI-centric:** analytics and MLOps as the spine for fleet-scale learning and decision support.
- **Control / Automation-centric:** OT runtime integration and real-time constraints as the primary driver.
- **Federated platform-of-platforms:** standards-anchored integration across multiple existing platforms without forcing a single vendor spine.

> Decision-makers are choosing operating model as much as technology

While multiple starting points are possible in practice, this ordering also reflects a common evolution path: organizations often begin with simulation-centric approaches to achieve early validation and time-to-value, and only later adopt PLM-centric spines when lifecycle governance, traceability, and organizational scale become primary concerns that cannot be addressed by simulation-oriented approaches.

4.2 Evaluation criteria

The evaluation criteria below capture the real trade-offs between speed, control, scalability, and long-term ownership — derived from recurring decision and validation needs, not from tool-level features.

- **Coverage** (concept → engineering → production → operations → EoL)
  Measures how thoroughly the lifecycle of the ecosystem is handled in the approach.
- **Interop and co-sim** (standardised exchange, FMI)
  Considers the depth of interoperability and co-simulation conventions used in the approach.
- **Real-time** (runtime distribution, PubSub)
  Runtime / Real-time includes tolerance to **intermittent connectivity** and **high-latency** links (store-and-forward, eventual consistency, edge autonomy). This criteria highlights how well the approach covers such requirements.
- **Thread and trace** (linking requirements ↔ models ↔ tests ↔ ops)
  Focuses on how well the approach supports traceability of the outcome to the requirements stated, which by extension requires thread support.
- **Data and MLOps** (data fabric, lifecycle of models)
  Indicates how well the approach suits in scenarios where data is readily available and AI/ML has a well-stated role.
- **Time-to-value** (PoC → pilot → scale; service packaging)
  Implies how soon and with what effort the approach shows measurable value.

4.3 Quick selection guide

A quick decision tree clarifies the dominant decision driver before any platform spine is chosen; it is a first-pass articulation tool, with the detailed rationale in the scorecards from 4.5 onward (Figure 4-1).

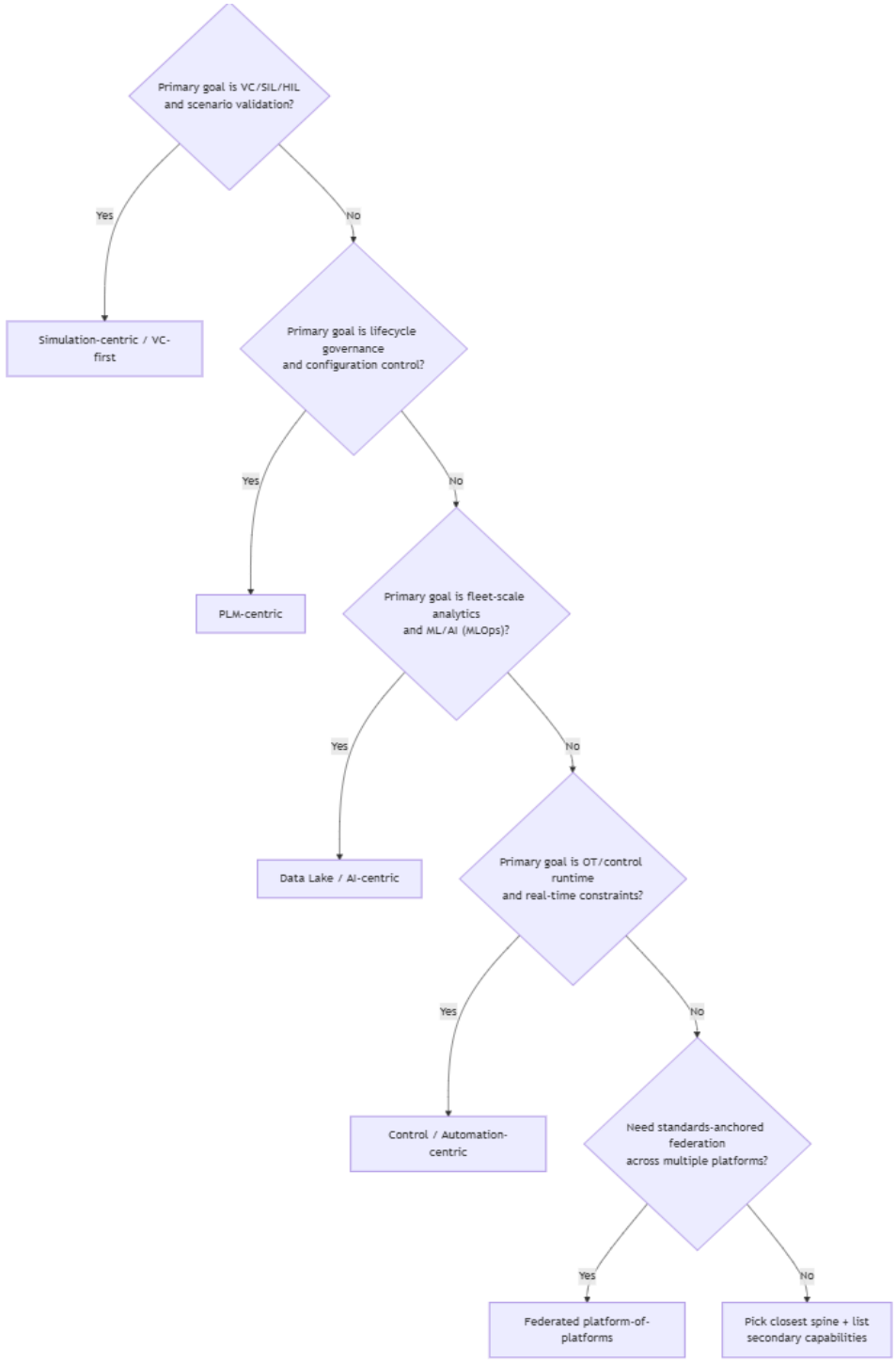

Figure 4-1: “Quick selection guide”

4.4 Summary comparison table

The summary table is a screening view, not a platform ranking: an H/M/L scale showing where each archetype is naturally strong (H), where integration is typically needed (M), and where significant mitigation is required (L).

Archetype	Coverage	Semantics and interop	Co-sim/ Model exchange	Runtime/ Real-time	Thread and trace	Data and MLOps	Time-to-value
Simulation-centric / VC-first	M	M	H	M	M	L	H
PLM-centric	H	M	M	L	H	M	M
Data Lake / AI-centric	M	M	L	M	M	H	M
Control /Automation-centric	M	L	M	H	M	L	H
Federated “platform-of-platforms”	H	H	H	H	H	M	L

Table 4-1. “Summary comparison table”

How to use this table: treat the H/M/L ratings as a screening view against the criteria stated above. The scorecards that follow provide the rationale behind each rating (fit, reasoning, risks, and mitigations) and should be used to validate the dominant spine and required secondary capabilities.

Each archetype should be read as a potential dominant spine complemented by secondary capabilities, rather than as a complete solution on its own.

4.5 Scorecard — Simulation-centric / VC-first

One-liner: VC-first makes the simulator the **primary integration and validation environment**; DT “truth” is proven in virtual tests before field deployment.

Scores (from 4.4): Coverage **M** | Semantics **M** | Co-sim **H** | Runtime **M** | Thread/Trace **M** | Data/MLOps **L** | Time-to-value **H**

Fit (when to choose):

- You need early confidence: SIL/HIL, test automation, regression suites, scenario libraries.
- The primary risk is expensive physical iteration; you want fast cycles in a virtual environment.

Criteria rationale:

- **Coverage (M):** strong for engineering/verification; weaker for operations governance unless linked to a thread spine.
- **Semantics and Interop (M):** interfaces can be standardized, but semantics often remain tool-specific without AAS-style explicit modeling.
- **Co-sim / Model exchange (H):** native fit for FMI/FMUs and multi-domain co-simulation.
- **Runtime / Real-time (M):** real-time constraints are addressed inside simulation; OT runtime integration usually needs an external layer (PubSub).
- **Thread and trace (M):** achievable if you connect test artefacts/scenarios back to a "thread" (NIST Digital Thread framing).
- **Data and MLOps (L):** not a natural home for feature stores/model registries; integrate externally if needed.
- **Time-to-value (H):** typically fast when the simulator is already central to development/testing.

Key building blocks:

- **Co-sim:** FMI standard with FMUs; supports tool composition.
- **Distributed sim (optional):** HLA concepts when multiple simulators must federate.
- **Semantics:** AAS for shared models across teams/tools if federation grows.

Risks (and mitigations):

- **Risk:** simulator becomes a silo (good tests but poor enterprise traceability). **Mitigation:** connect outputs to a thread/requirements repository and enforce scenario IDs/metadata.
- **Risk:** "reality gap" (model fidelity not sufficient for key decisions). **Mitigation:** explicit validation KPIs combined with selective high-fidelity models; document assumptions.

4.6 Scorecard — PLM-centric

One-liner: PLM is the **system-of-record spine** for lifecycle configuration and traceability; DT capabilities are composed around it.

Scores (from 4.4): Coverage **H** | Semantics **M** | Co-sim **M** | Runtime **L** | Thread/Trace **H** | Data/MLOps **M** | Time-to-value **M**

Fit (when to choose):

- Products/assets with long lifecycle and heavy change/configuration management requirements (as-designed/as-built/as-maintained).
- You need a single authoritative "thread" across engineering, manufacturing, service.

Criteria rationale (bullets):

- **Coverage (H):** strong lifecycle breadth by design (design → service), especially configuration and handovers.
- **Semantics and Interop (M):** semantics often need explicit standard mapping (e.g., AAS submodels) to be portable across tooling.
- **Co-sim / Model exchange (M):** feasible via integrations (e.g., FMI exchange), but not typically the native "spine".
- **Runtime / Real-time (L):** OT/real-time messaging is usually external to PLM; needs separate runtime data plane (e.g., PubSub).
- **Thread and trace (H):** core strength—requirements/config/models/test artefacts can be governed and linked.
- **Data and MLOps (M):** can be integrated with data platforms, but usually not optimized for analytics/MLOps as primary function.
- **Time-to-value (M):** good for organizations that already have PLM discipline; slower if process maturity is low.

Key building blocks (conceptual)

- **Semantics:** AAS Part 1 metamodel complemented with submodels for interoperability.
- **Runtime data plane:** OPC UA PubSub for events/telemetry distribution (separate layer).
- **Simulation bridge:** FMI for exchanging models/FMUs into VC environments.

Risks (and mitigations):

- **Risk:** becomes a "process-heavy" program with slow adoption.
  Mitigation: start with 1–2 high-value use cases; keep minimal governance initially.

- **Risk:** semantics remain implicit → poor cross-tool interoperability.
  Mitigation: standardize semantics explicitly (AAS) early for the critical asset classes.

4.7 Scorecard — Data Lake / AI-centric

One-liner: Data/AI-centric makes the **data plane and MLOps** the spine; DT is implemented as analytics-ready representations and models at scale.

Scores (from 4.4): Coverage **M** | Semantics **M** | Co-sim **L** | Runtime **M** | Thread/Trace **M** | Data/MLOps **H** | Time-to-value **M**

Fit (when to choose):

- Your dominant ROI is fleet-level monitoring, prediction, optimization, and ML-powered insights.
- You already have (or can build) data governance, pipelines, and model operations.

Criteria rationale:

- **Coverage (M):** strong for operations/analytics; weaker for engineering artifacts unless integrated with PLM/sim.
- **Semantics and Interop (M):** requires explicit semantic modeling (AAS/submodels or equivalent) to avoid “data swamp” effects.
- **Co-sim / Model exchange (L):** not the natural center for FMI/co-sim; typically consumes outputs rather than orchestrates simulation.
- **Runtime / Real-time (M):** can ingest streams/events; true OT real-time control is usually a separate layer.
- **Thread and trace (M):** achievable by linking data products and model versions to lifecycle IDs (Digital Thread framing).
- **Data and MLOps (H):** primary strength—feature lifecycle, monitoring, retraining, governance.
- **Time-to-value (M):** medium because data cleanup/contextualization can dominate early phases.

Key building blocks:

- **Thread anchor:** Digital Thread concept + lifecycle identifiers.
- **Semantics:** AAS to publish standardized asset context and reduce integration friction.

- **Runtime ingress (if needed):** PubSub/event distribution patterns feeding the lake.

Risks (and mitigations):

- **Risk:** data lake becomes "data swamp" (low trust, low reuse).
  Mitigation: start with narrow, high-value use cases; enforce metadata + ownership.
- **Risk:** poor linkage to engineering truth (models drift from "as-built").
  Mitigation: integrate identifiers/configuration snapshots from PLM or field configuration systems.

4.8 Scorecard — Control / Automation-centric

One-liner: Control-centric makes the **OT runtime** the spine; DT is used to improve reliability, control performance, and operational decisions under real-time constraints.

Scores (from 4.4): Coverage **M** | Semantics **L** | Co-sim **M** | Runtime **H** | Thread/Trace **M** | Data/MLOps **L** | Time-to-value **H**

Fit (when to choose):

- You need real-time operational behaviour: alarms/events, control loops, OT integration, edge-first patterns.
- DT is primarily about operational resilience, uptime, and safe automation.

Criteria rationale:

- **Coverage (M):** strong in operations; limited in design/service governance unless integrated.
- **Semantics and Interop (L):** OT stacks often use implicit semantics; standard semantic layer typically missing.
- **Co-sim / Model exchange (M):** can be integrated for testing/commissioning via FMI/HIL patterns.
- **Runtime / Real-time (H):** core capability; PubSub/event distribution fits well for scalable OT eventing.
- **Thread and trace (M):** possible if runtime changes, versions, and incidents are linked to lifecycle records.
- **Data and MLOps (L):** typically not optimized for ML lifecycle; external analytics platform needed.

- **Time-to-value (H):** fast when driven by operational pain points (downtime, safety, compliance).

Key building blocks:

- **Runtime messaging:** OPC UA PubSub (and related OT eventing).
- **Validation:** simulation-in-the-loop/HIL with FMI/FMUs where appropriate.
- **Optional semantics uplift:** AAS submodels for device/asset "contract" across teams.

Risks (and mitigations):

- **Risk:** local optimizations only (hard to scale cross-site).
  Mitigation: define a minimal canonical information model and event taxonomy early.
- **Risk:** limited visibility into engineering intent/configuration.
  Mitigation: connect to thread/PLM for configuration snapshots and trace.

4.9 Scorecard — Federated platform-of-platforms

One-liner: Federation is the **standards-anchored integration spine**: connect PLM, simulation, data, and OT without forcing a single vendor platform. This is typically chosen in relatively mature environments where considerable effort has already been taken in multiple viewpoints, and the requirements on the approach are more and more centered on the platform sustainability and general governance.

Scores (from 4.4): Coverage **H** | Semantics **H** | Co-sim **H** | Runtime **H** | Thread/Trace **H** | Data/MLOps **M** | Time-to-value **L**

Fit (when to choose):

- You already have multiple strong systems (PLM, sim, data lake, OT) and need to integrate them sustainably.
- Interoperability and incremental evolution are strategic requirements.

Criteria rationale:

- **Coverage (H):** federation can span lifecycle phases by connecting specialized systems rather than replacing them.
- **Semantics and Interop (H):** explicit semantics via AAS/submodels is the core enabler for cross-ecosystem interoperability.

- **Co-sim / Model exchange (H):** FMI provides tool-agnostic model exchange; HLA can complement for federated simulation.
- **Runtime / Real-time (H):** PubSub/event-driven patterns allow scalable distribution across edge/cloud boundaries.
- **Thread and trace (H):** Digital Thread framing and stable identifiers allow trace across systems of record.
- **Data and MLOps (M):** usually delegated to a dedicated data platform; federation standardizes interfaces/metadata.
- **Time-to-value (L):** integration design and governance take time; benefits compound after the first successful use case.

Key building blocks:

- **Semantics:** IDTA AAS metamodel/spec as shared semantic contract.
- **Co-sim:** FMI for multi-tool composition; optional HLA for distributed sim.
- **Runtime:** OPC UA PubSub for event/telemetry distribution across domains.
- **Governance:** Digital Thread roadmap/governance guidance (NIST) to prevent “integration sprawl”.

Risks (and mitigations):

- **Risk:** “integration sprawl”, i.e. uncontrolled growth of point-to-point integrations without shared semantic contracts or ownership, leading to fragile and hard-to-evolve architectures.
  Mitigation: define minimal contracts (AAS submodels, event taxonomy) and assign product owners per interface.
- **Risk:** slow first delivery.
  Mitigation: start with one end-to-end slice (one asset class and one use case) and scale by reuse.

References

Architecture criteria and scoring are anchored in the standards baseline listed in chapter 2 (ISO 23247, IEC 62832-1, IDTA AAS, FMI, OPC UA PubSub, and HLA). Additional implementation-oriented sources (e.g., vendor and consortium publications) are listed in supplementary references as context; they are not required to follow the scoring logic presented in this white paper.

5 AI, autonomy, and advanced methods

Executive takeaway

AI and advanced methods do not define direction; they amplify capability once direction is chosen. Used well, they increase decision speed or confidence; used poorly, they add complexity without changing outcomes.

5.1 Purpose and scope

Method choices decide two things at once: whether a digital twin is credible enough to act on, and whether it is fast enough for iterative engineering and operations. The emphasis is on *how to apply methods*, not on deep theoretical debates. From a decision-making perspective, these methods determine when results can be trusted and how quickly alternative options can be evaluated.

We focus on two complementary method families:

- **A) Surrogate-based calibration and acceleration** — improve iteration speed by approximating expensive simulations while preserving decision-relevant accuracy.
- **B) Bayesian calibration with model discrepancy** — improve trust by explicitly modeling uncertainty and acknowledging the mismatch between model and reality.

> Is the bottleneck speed or trust?

We also outline how these methods connect to the platform archetypes in chapter 4:

- **Simulation-centric / VC-first** benefits from surrogate acceleration for scenario regression and what-if exploration.
- **Data Lake / AI-centric** benefits from uncertainty-aware evaluation and drift monitoring for models deployed at scale.
- **Federated architectures** require explicit contracts for provenance and traceability of model versions, calibration data, and KPIs.

We have also illustrated the method families with diagrams in their respective subchapters. It should be noted that those are provided as conceptual diagrams intended to support discussion, not prescriptive implementation blueprints.

5.2 Surrogate-based calibration and acceleration (speed-focused)

Surrogates let simulation-based validation scale — fast enough for regression and what-if sweeps — without giving up decision-relevant accuracy. It addresses situations where high-fidelity models are valuable but impractical to run at the frequency required by iterative engineering or operational decision-making.

> Use surrogate models to accelerate exploration and regression

When to use

Use this pattern when:

- The high-fidelity model is **too slow** for regression testing, design space exploration, or real-time what-if analysis.
- You need fast iteration loops for **VC/SIL workflows** or for operational decision support.
- The decision requires **relative comparisons** (“which option is better?”) more often than absolute perfect prediction.

Core idea

Replace expensive simulation components (or entire simulations) with **surrogate models** that approximate outputs as a function of inputs. Calibrate the surrogate (and/or key simulation parameters) using trusted data (simulation runs, test rigs, or operational telemetry). Surrogates can be statistical or ML-based (e.g., Gaussian processes, tree ensembles, neural surrogates).

In practice, surrogate-based calibration follows a small number of recurring steps that can be adapted to different tools and domains.

Minimal workflow (recommended)

1. **Define decision outputs and constraints**
   Identify the 3–8 outputs that drive decisions (e.g., energy, cycle time, stability margin).
2. **Choose the fidelity baseline**
   Select the "ground truth" reference: high-fidelity sim runs, test bench, or validated datasets.
3. **Design experiments / sampling**
   Generate training points that cover the operational envelope (avoid overly narrow training).
4. **Fit surrogate + validate**
   Validate with hold-out cases and worst-case residual checks.
5. **Use surrogate for calibration and exploration**
   Apply for parameter tuning and what-if sweeps; integrate into VC regression loops.
6. **Monitor drift** (if used operationally)
   Track data drift and error drift; retrain or re-calibrate when thresholds are exceeded.

These steps should be treated as a flexible pattern rather than a rigid pipeline. The emphasis is on maintaining decision relevance and validation scope, not on enforcing a specific implementation sequence.

KPIs (Speed pattern)

- **Speedup factor:** baseline simulation runtime vs. surrogate runtime (target: 10×–100× where needed).
- **Fit error:** RMSE / MAPE on held-out cases; also max error on edge cases.
- **Coverage:** training set coverage over the operational domain; percentage of "out-of-domain" queries.
- **Stability:** error stability across releases (regression).

The figure below summarizes the surrogate-based calibration pattern as a compact loop. It highlights the minimum set of steps required to make surrogate use repeatable: training data selection, validation, and drift-aware operation when models are used beyond one-off exploration.

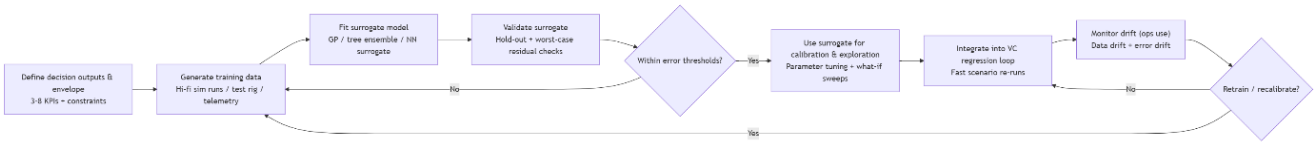

Figure 5-1: "Surrogate-based calibration loop."

Expensive simulations generate training data; a surrogate model is fitted and validated; the surrogate accelerates calibration and what-if exploration, then feeds back into the VC regression suite.

This pattern is less suitable when absolute accuracy and fully traceable uncertainty are required for high-risk decisions.

5.3 Bayesian calibration with model discrepancy (trust-focused)

Bayesian calibration buys trust rather than speed: it makes model uncertainty and the model–reality gap explicit, which matters most for safety-critical or high-cost decisions. It is particularly useful when decision-makers need to understand not only expected outcomes, but also the uncertainty and risk associated with them.

> Use Bayesian calibration when uncertainty must be explicit.

When to use

Use this pattern when:

- You must communicate **uncertainty** to decision-makers (e.g., safety, compliance, high-cost choices).
- Models are known to be **imperfect** (missing physics, simplifications, sensor biases).
- You need **probabilistic statements** rather than point estimates.

Core idea

A calibrated model should not pretend to be perfect. Bayesian calibration treats unknown parameters as uncertain and explicitly models the gap between

simulation and reality (**discrepancy**). The outcome is not just a "best fit," but a **distribution** of plausible predictions with credible intervals.

The workflow below illustrates the logical structure of Bayesian calibration, rather than a mandatory implementation sequence.

Minimal workflow (recommended)

1. **Define the calibration target**
   Select measurable outputs and define acceptable error/uncertainty thresholds.
2. **Specify uncertain parameters**
   Identify parameters to infer (and which ones to fix).
3. **Model discrepancy explicitly**
   Add a discrepancy term capturing systematic mismatch.
4. **Fit with Bayesian inference**
   Use MCMC/variational inference depending on complexity and time budget.
5. **Validate with predictive checks**
   Compare predictive intervals with held-out data; check calibration.
6. **Operationalize traceability**
   Record: dataset version, parameter posterior, discrepancy model, and decision KPIs.

In practice, the level of statistical rigor and computational effort should be proportional to the decision being supported, rather than to methodological completeness.

KPIs (Trust pattern)

- **Predictive accuracy:** RMSE/MAPE on held-out scenarios.
- **Uncertainty calibration:** coverage of prediction intervals (e.g., PICP), sharpness vs. reliability trade-off.
- **Robustness:** sensitivity of decisions under parameter/posterior uncertainty.
- **Traceability:** ability to reproduce results given model/data/version IDs.

The figure below visualizes Bayesian calibration as a trust-focused workflow. It emphasizes that the outcome is a predictive distribution (with uncertainty) and that validation is performed through posterior predictive checks before operationalizing the result.

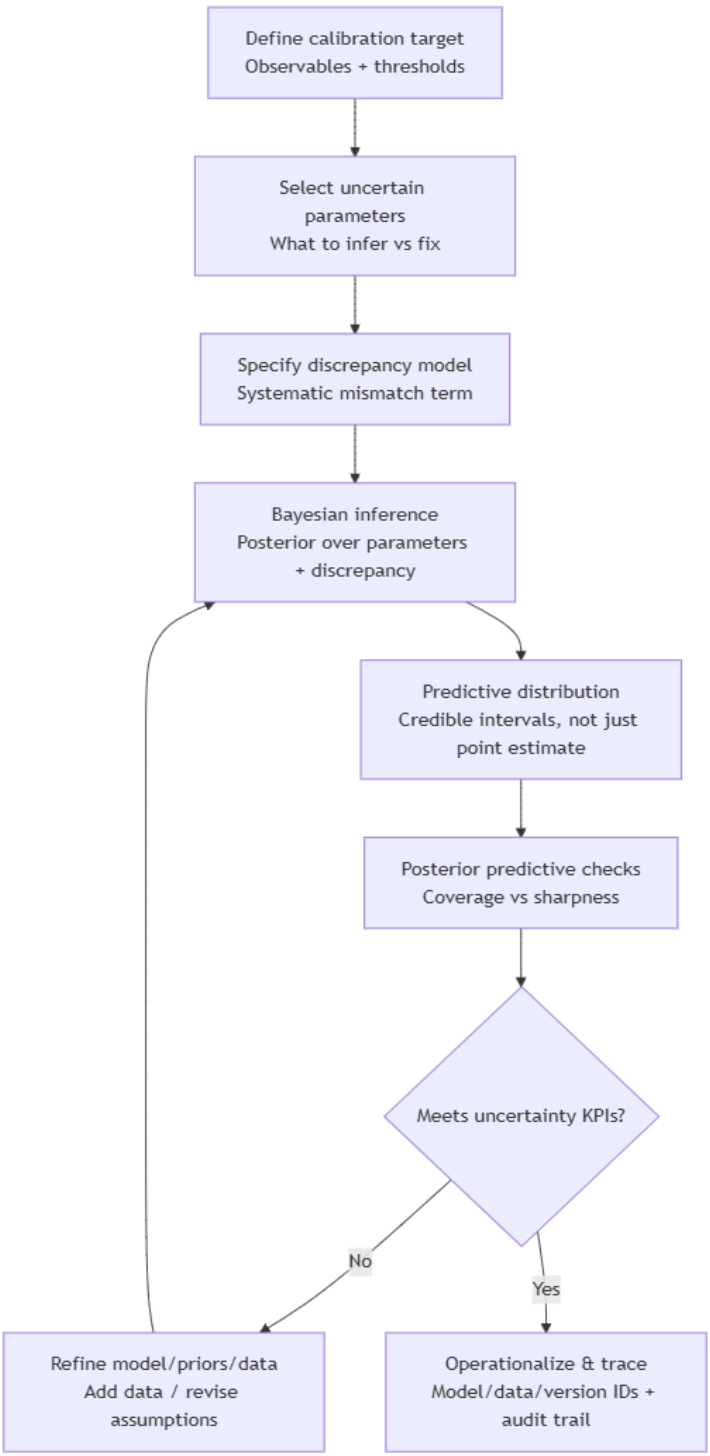

Figure 5-2: "Bayesian calibration with discrepancy."

Observations are compared to model predictions; uncertain parameters and discrepancy are inferred; outputs become predictive distributions with credible intervals used for decision gates.

This pattern may be excessive for low-impact decisions where iteration speed outweighs the need for fully characterized uncertainty.

5.4 Hybrid digital twins as a pragmatic bridge (physics and data)

Most industrial digital twins end up hybrid: a physics-based core for interpretability and extrapolation, with a data-driven correction for the effects physics misses. This approach reflects a pragmatic balance between interpretability, computational cost, and the ability to adapt to changing operational conditions.

A recurring practical pattern is to combine:

- **Physics-based core** (interpretable, extrapolates better across regimes) with
- **Data-driven correction/augmentation** (captures unmodeled effects, aging, and contextual behaviour).

This hybrid approach is often the most practical path when full-physics models are too expensive to run online, and purely data-driven models fail outside the training domain.

The figure below illustrates the hybrid twin concept at a system level. It clarifies how a physics-based core and a data-driven correction combine into a single decision-ready output, and how monitoring and drift handling close the loop over time.

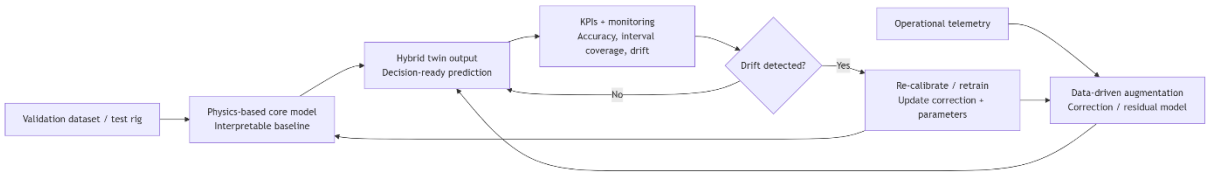

Figure 5-3: "Hybrid twin: physics core and data correction."

Physics-based model provides interpretable baseline; data-driven component corrects mismatch and supports operations under changing conditions.

5.5 How methods connect to the platform archetypes

From a decision-making perspective, methods only create value when they are anchored to architectural responsibilities and lifecycle ownership. The approaches described above are not standalone solutions; their impact depends on how well platform architectures support validation artefacts, traceability, and operational constraints.

- **Simulation-centric / VC-first:**
  Surrogates improve regression throughput and enable broader scenario sweeps under time constraints. Bayesian calibration is used selectively for high-impact decisions (gate criteria, safety-critical performance).
- **Data Lake / AI-centric:**
  Uncertainty-aware evaluation (interval metrics, coverage) and drift monitoring are key; surrogate models can act as scalable approximators for simulation-derived features.
- **Control / Automation-centric:**
  Real-time constraints favour reduced-order and surrogate models; uncertainty is used for safe envelopes and alarm thresholds.
- **Federated architecture:**
  Requires explicit contracts for model/data provenance: semantic identifiers, versioning, and reproducible calibration runs.

5.6 Risks and mitigations

The risks listed below are not arguments against advanced methods, but recurring failure modes observed when such methods are applied without sufficient architectural support or validation discipline.

Common risks

- **Overfitting / poor generalization:** surrogate performs well on training set but fails on edge cases.
- **Data quality and bias:** calibration data may be non-representative or systematically biased.
- **Model drift:** physical system evolves (aging, configuration changes) but models do not.
- **Hidden assumptions:** results are not reproducible due to undocumented versions and parameters.
- **Compute cost explosion:** Bayesian methods become too slow for practical timelines.

Mitigations

- Domain coverage checks and explicit out-of-domain detection.
- Hold-out validation and stress tests on edge regimes.
- Drift monitoring and re-calibration policy (threshold-based).
- Version everything: data, model, parameters, scenarios, KPIs.
- Use Bayesian methods selectively; prefer surrogate-assisted inference when needed.

References

Surrogate modeling and uncertainty quantification overviews [REF: Recent Advances in Surrogate Modeling Methods for UQ (MDPI, 2022)]; automated experimental design and Bayesian optimization patterns [REF: Bayesian optimization with adaptive surrogate models (2021)]; foundational calibration and model discrepancy framing [REF: Kennedy & O'Hagan, "Bayesian Calibration of Computer Models"]; pragmatic hybrid digital twin examples combining physics and data-driven approaches [REF: "Hybrid Digital Twins for Operations and Maintenance"].

Architectural and governance cross-references include semantic and provenance contracts [REF: IDTA AAS Part 1], model exchange and co-simulation [REF: FMI], runtime data distribution [REF: OPC UA IEC 62541-14], and Digital Thread governance perspectives [REF: NIST Digital Thread].

6 Challenges, constraints, and paths forward

Executive takeaway

Scaling digital twins is ultimately a leadership and operating model challenge. Without clear ownership, decision authority, and shared ways of working, even technically sound solutions remain local optimizations rather than organization-wide capabilities.

6.1 Purpose

Moving from single digital twin pilots to scalable, multi-stakeholder solutions runs into a recurring set of implementation challenges — mapped here into clusters and concrete implementation paths.

6.2 Challenge map

The clusters below are not exhaustive but recurring: they are the patterns that consistently decide whether a digital twin effort stays a local optimization or becomes a scalable, decision-relevant capability. These clusters emerge at the intersection of lifecycle ownership, validation practices, and operational deployment. They are selected because they consistently determine whether digital twin efforts remain local optimizations or evolve into scalable, multi-stakeholder and decision-relevant capabilities.

> Problems accumulate at the intersection of lifecycle, validation, and operations

Cluster A — Interoperability and semantics

- Multiple tools and teams require shared meaning and stable interfaces
- Risk: implicit semantics → fragile integrations
- Mitigation direction: explicit semantic contracts (AAS/submodels), interface versioning, minimal canonical identifiers

Cluster B — Fidelity vs. latency trade-offs

- Higher fidelity improves confidence but increases compute cost and slows iteration
- Risk: “reality gap” or “over-precision” in wrong areas
- Mitigation direction: define validation KPIs, tiered model fidelity, surrogate/hybrid approaches where justified

Cluster C — Governance, security and trust

- Who owns the twin, its data, and decision authority?
- Risk: uncontrolled changes, unverifiable decisions, unclear liability
- Mitigation direction: Digital thread governance (IDs, trace links, audit trails), security/trust guidance, controlled release gates

Cluster D — Scale and operationalization

- Pilots work; production fails due to ops and lifecycle costs
- Risk: data swamp / model drift / brittle pipelines
- Mitigation direction: productization mindset, MLOps/ops patterns, monitoring, clear operating model
- Intermittent connectivity and latency: asynchronous operations, delayed feedback loops, edge autonomy boundaries.

6.3 Implementation paths

Three recurring implementation paths describe how organizations actually navigate the trade-offs between speed, governance, and scalability — and they are organizational starting points, not technical architectures. The paths are not prescriptions, but typical ways organizations navigate trade-offs between speed,

governance, and long-term scalability. Importantly, these paths describe organizational starting points and decision patterns rather than technical architectures; their success depends less on the chosen tools than on how ownership, decision authority, and validation responsibilities are established.

Path A — "Single spine" (fast start)
Choose one dominant archetype (often PLM-centric or Data/AI-centric) and integrate minimal interfaces for the rest.

- **Pros:** faster time-to-value; simpler governance early
- **Cons:** risk of lock-in; later federation may require refactoring
- **Best when:** organization maturity is uneven; need a fast pilot to prove ROI

Path B — "Simulation-first to federation" (R&D-driven)
Start with VC-first for validation and scenario libraries; add thread/semantic contracts as the program scales.

- **Pros:** strong early engineering impact; measurable regression
- **Cons:** can become a silo without governance; needs intentional trace integration
- **Best when:** product risk is high; validation and commissioning dominate cost

Path C — "Standards-anchored federation" (strategic)
Design for multi-platform coexistence from the start using explicit semantic contracts and integration patterns.

- **Pros:** vendor neutrality; scalable across domains and sites; sustainable evolution
- **Cons:** slower first delivery; requires strong ownership and governance
- **Best when:** multiple strong systems already exist; long-term integration is strategic

While the implementation paths describe strategic starting points, they do not imply static end states. In practice, organizations often evolve from one path toward another as maturity, scope, and stakeholder involvement increase. The roadmap below provides a simplified view of this progression, focusing on capability building rather than organizational structure.

6.4 Roadmap

The roadmap shows how digital twin capability typically evolves as scope, stakeholders, and operational demands grow — a conceptual guide, not a prescriptive or strictly sequential plan. The purpose of the roadmap is to support planning and governance discussions by making progression and capability dependencies explicit.

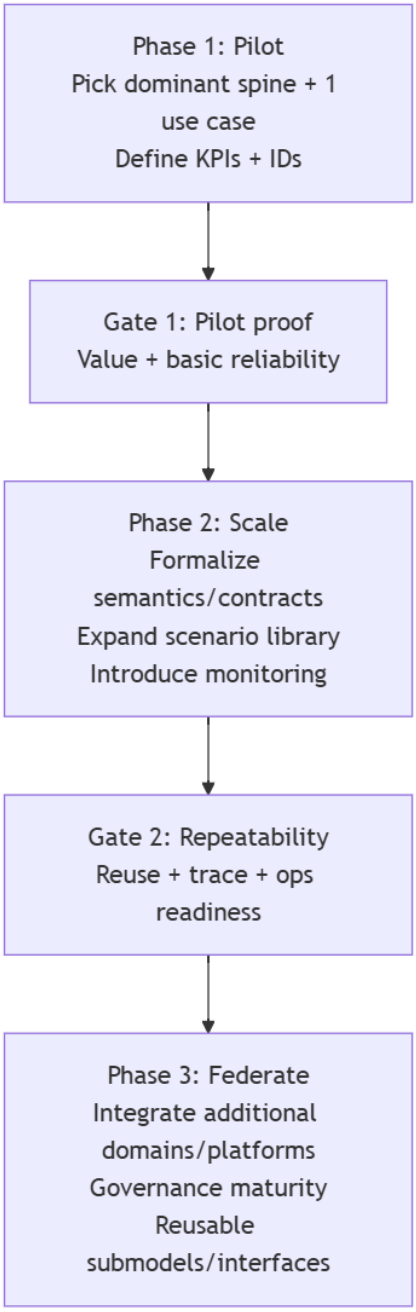

Figure 6-1: “From pilots to federated digital twin capability”

- **Phase 1 (Pilot):** pick dominant spine + one use case; define minimal KPIs and identifiers
- **Phase 2 (Scale):** formalize semantics/contracts; expand scenario library; introduce monitoring
- **Phase 3 (Federate):** integrate additional domains/platforms; governance maturity; reusable submodels and interfaces

The next section translates this roadmap into a concrete scenario, illustrating how the same principles can be applied in practice.

6.5 Applied example: aligning multiple digital twin pilots in one organisation

A common organisational pain point makes the roadmap concrete: several parallel 'twin pilots' that each succeed locally but cannot be combined into a coherent capability. Ultimately, the value of digital twin initiatives lies not in individual technologies, but in how well architectures, validation practices, and governance evolve together. By making trade-offs explicit and aligning implementation paths with organizational maturity, digital twins can progress from isolated pilots to sustainable, decision-relevant capabilities.

A common organisational pain point is the emergence of **multiple parallel "twin pilots"**. One team develops a simulation-based validation environment, another focuses on operational monitoring, a third builds analytics or AI models, and a fourth integrates OT signals for real-time visibility. Each pilot may succeed locally, yet the organisation struggles to combine them into a coherent capability. The result is duplicated work, incompatible assumptions, and brittle integrations that are expensive to evolve.

The underlying issue is rarely "missing technology." More often, pilots diverge because they lack a shared foundation in three areas: **(1) semantics and identifiers**, **(2) validation discipline and artefacts**, and **(3) ownership and governance of change**. When these are implicit, integration grows as point-to-point connections and ad hoc mappings—an "integration sprawl" effect where every new pilot adds complexity rather than reuse.

The framework in this white paper can be applied as a lightweight alignment process that avoids over-engineering while still enabling reuse. A practical way to proceed is to treat alignment as an end-to-end slice rather than an enterprise-wide redesign:

First, agree on a **dominant architectural spine** and list the supporting capabilities explicitly. The decision tree in chapter 4 exists for precisely this purpose: it makes the primary driver visible (lifecycle governance, VC validation, AI/data products, OT runtime, or federation) and forces secondary needs to be stated rather than assumed. The point is not to "pick one tool," but to clarify where authority and ownership reside.

Second, establish a **minimal semantic contract** for one or two asset classes. This does not require full standard compliance programs. It requires stable identifiers, a small set of required fields, and a shared understanding of what "the asset" and "its state" mean across teams. Without this, pilots cannot exchange meaning—only data.

Third, align pilots through **validation artefacts and gates**. Even operational twins benefit from scenario thinking: define what is considered "good behaviour," how changes are validated, and what evidence is required before something is promoted from experiment to reusable capability. Chapter 3's VC-before-HIL pattern is a useful mental model here: explicit inputs, scenario execution, measurable criteria, and controlled promotion.

Fourth, make runtime constraints explicit where pilots intersect operational reality. In practice this means agreeing signal classes and "freshness" expectations, as well as outage behaviour for remote or constrained environments. This prevents one pilot from assuming always-online low-latency streams while another silently relies on buffering and delayed delivery.

Scaling is a leadership problem

To keep this aligned process lightweight, the "minimum artefacts" can remain intentionally small. In many organisations, a single page per artefact is sufficient at this stage: one page describing the semantic contract, one list for the top scenarios and KPIs, one table for signal classes and staleness/outage rules, and a decision log capturing ownership and approval for changes. The benefit is not paperwork; the benefit is that the next pilot can reuse existing contracts rather than reinvent them.

What good looks like is measurable. Teams can share identifiers and semantics without bespoke mapping per integration. Changes can be validated against

agreed scenarios and criteria. And new pilots add capability by plugging into an existing spine and shared contracts, rather than creating yet another disconnected "twin" that the organisation cannot sustainably operate.

6.6 Applied perspective: people and change

Many digital twin initiatives stall for non-technical reasons. Repeatedly, the limiting factors are not missing tools or standards, but unclear ownership, fragmented incentives, and the difficulty of changing established ways of working. This perspective builds on the governance and trust challenges identified earlier in this chapter and makes explicit the people and change aspects that often determine whether technical foundations are actually adopted.

In practice, these issues surface when digital twins remain "someone's pilot" rather than becoming part of normal decision-making. Simulation and virtual commissioning may be perceived as engineering tools rather than organisational assets. Semantic contracts are seen as overhead instead of enablers. Validation artefacts exist, but decisions are still made informally or outside agreed gates. Over time, this erodes trust: teams stop relying on shared models and revert to local optimisations.

From the perspective of this white paper, these are governance questions expressed through people and roles. Who owns the digital twin as a product rather than a project? Who is accountable for approving changes to models, scenarios, and assumptions? Who decides when evidence is "good enough" to support a decision, and how is that evidence communicated across teams? Without explicit answers, even well-designed architectures struggle to scale beyond their original context.

Addressing the people dimension does not require a separate change programme detached from technical work. Instead, it typically involves a small number of concrete shifts that are tightly coupled to the technical foundations described earlier:

- establishing clear ownership for the digital twin capability (with decision authority, not only coordination responsibility),
- using a shared vocabulary and artefact model as part of everyday work, rather than as documentation produced after the fact,
- embedding validation gates (such as VC-before-HIL) into actual decision points, not treating them as optional quality checks,
- developing competence through scenarios and real cases, instead of tool-centric training detached from operational decisions,

- and making the link between digital twin outputs and business or safety decisions explicit and visible.

When these aspects are aligned, adoption tends to follow naturally. Teams understand why certain practices exist, trust the outputs they produce, and are willing to reuse shared assets instead of rebuilding their own. Without this alignment, even technically sound architectures and methods tend to remain local optimisations rather than organisation-wide capabilities.

References

Digital twin maturity models and capability framing [REF: Digital Twin Consortium publications]; Digital Thread governance and roadmap perspectives [REF: NIST Digital Thread; REF: NIST roadmap]; semantic and interoperability anchors for industrial digital twins [REF: IIC × Plattform I4.0 DT + AAS; REF: IDTA AAS].

Acknowledgements

The author wishes to thank the following people without the white paper would have been way harder to write:

The lead of the white paper project Atte Aunola, along with executive-level guidance Harri Laukkanen were instrumental in setting the expectations and direction for the work and to keep the standards high throughout the project.

The architects of the Gofore's DPL project, Juha Anttila and Juha Möttönen, offered their invaluable insights on architectural grounding for Digital Twin landscape and made things comprehensible to a fellow architect by speaking the language familiar to the author.

Timo Mustonen gave a swift but comprehensive and highly inspiring crash course for simulation landscape and challenges currently on industry's board, and that influenced highly on the simulation and commissioning sections on this paper.

Last but certainly not least, a huge shoutout for Noora Vuori on making the author feel at home with the project team, keeping the train on tracks and even providing copy editing services.

Glossary

Term	Abbreviation used in this white paper	Description
Artificial Intelligence / Autonomy	AI	Methods that automate or assist decision-making based on models and data. In this white paper, AI and autonomy are treated as capability amplifiers that depend on lifecycle governance, simulation artefacts, and runtime observability, rather than as standalone solutions.
Asset Administration Shell	AAS	A technology-agnostic semantic model for representing industrial assets and their submodels. In this white paper, AAS is used as an explicit semantic contract to support interoperability, traceability, and lifecycle consistency across tools and platforms. AAS is treated as a modelling and integration concept, not as a mandated product or platform.
Bayesian Calibration	—	Calibration approach that infers uncertain model parameters from data and yields uncertainty-aware predictions (often producing distributions, not single values).
Bayesian Optimization	BO	Data-efficient optimization method that selects new experiments/simulations using an acquisition strategy over a surrogate model.
Co-simulation	Co-sim	Coupling two or more simulators/models at runtime (often via FMI), exchanging states/signals during time-stepped execution.
Control / Automation-centric (archetype)	—	Platform spine centered on OT/control runtime (PLC/SCADA/DCS), focusing on real-time operation, events, and safe automation.
Data Lake / AI-centric (archetype)	—	Platform spine centered on data products and AI/ML (pipelines, governance, MLOps), optimized for analytics and fleet-scale learning.
Digital Factory framework	DF	A conceptual framework for assets, relationships, and information flows at factory level (cross-domain production context).

Term	Abbreviation used in this white paper	Description
Digital Thread	—	A lifecycle-spanning narrative that links requirements, configurations, models, tests, and operational evidence into a traceable decision history. The digital thread is not a single system, but a consistency and traceability concept spanning multiple tools.
Digital Twin	DT	A connected set of models and data representing an asset or system, used to support specific decisions (e.g. design, validation, operation, optimisation) across the lifecycle. In this white paper, a digital twin is treated as a purpose-driven construct, not as a monolithic system.
Distributed Simulation	—	Simulation approach where multiple simulators/components run as a federation (often with time management and shared object models).
Enterprise-Control System Integration (ISA-95)	ISA-95	An established industrial reference model describing information content and functional boundaries between enterprise and control systems. In this white paper, ISA-95 is referenced only as a compatibility and familiarity signal: many industrial information models and lifecycle concepts discussed align naturally with ISA-95-style structures. ISA-95 compliance is not required or prescribed.
Extended Reality	XR	Umbrella term (AR/VR/MR) used for human-in-the-loop design, training, and operational visualization contexts.
Federated "platform-of-platforms" (archetype)	—	An architectural approach where multiple existing platforms (e.g. PLM, simulation, data, OT) are integrated through explicit semantic and interface contracts, rather than replaced by a single system. Federation is treated as a means to manage heterogeneity, not as an objective on its own.
Feature Store	—	Managed repository for ML features (definition, reuse, governance, and serving) across model training and deployment.

Term	Abbreviation used in this white paper	Description
Functional Mock-up Interface	FMI	Open standard for model exchange and co-simulation; widely supported for tool-agnostic integration (FMI 2.0 / 3.0.x).
Functional Mock-up Unit	FMU	Packaged model artifact conforming to FMI (includes model + interface + metadata) used for exchange/co-simulation between tools.
Hardware-in-the-Loop	HIL	Connecting a real controller/hardware to a real-time simulator to validate integration and timing before field deployment.
High Level Architecture	HLA	Standard for distributed simulation (federations, RTI services, time management, and shared object models/FOMs).
Key Performance Indicator	KPI	A measurable indicator used to evaluate whether a validation gate, method, or implementation path meets its intended objective. In this white paper, KPIs are used to make progress and readiness explicit (e.g., scenario coverage, defect escape rate, model error metrics, drift indicators) rather than as generic reporting numbers.
Markov chain Monte Carlo	MCMC	A family of sampling methods for approximating posterior distributions in Bayesian inference.
Mean Absolute Percentage Error	MAPE	Error metric measuring average absolute percentage difference between predicted and observed values.
Model Discrepancy	—	Explicit representation of systematic mismatch between a model and reality; used to avoid overconfidence and improve trust in predictions.
Model Drift	—	Degradation of model validity over time due to changing system behaviour, environment, configuration, or data distributions.
MLOps	MLOps	Operational practices and tooling for deploying, monitoring, governing, and updating ML models and data pipelines.

Term	Abbreviation used in this white paper	Description
OPC UA Publish–Subscribe	OPC UA PubSub	Communication model for scalable data/event distribution; supports real-time-ish patterns for telemetry and eventing across domains.
Operational Technology	OT	Systems and technologies used to monitor and control physical industrial processes (e.g. PLCs, SCADA, DCS). In this white paper, OT is treated as the runtime execution and control domain, with distinct requirements for real-time behaviour, reliability, and safety. OT is explicitly distinguished from IT systems and is integrated with higher-level platforms through well-defined interfaces and eventing patterns, not replaced by them.
PLM-centric (archetype)	PLM	Platform spine centered on product lifecycle governance and configuration (as-designed/as-built/as-maintained) and traceability.
Prediction Interval Coverage Probability	PICP	Fraction of observations that fall within a model's predicted interval; used to evaluate uncertainty calibration.
Regression Suite	—	Automated set of tests/scenarios rerun on each change to detect regressions; key to scalable VC and model evolution.
Root Mean Squared Error	RMSE	Error metric measuring the square root of the mean squared difference between predicted and observed values.
Scenario Library	—	Curated set of reusable scenarios (nominal + edge cases) with metadata and expected outcomes; used for VC, testing, and training.
Software-in-the-Loop	SIL	Executing control software/algorithms in a simulated environment without physical hardware to accelerate early-stage validation.
Surrogate Model	—	Approximate model that emulates a more expensive simulation or system response to enable fast exploration, calibration, or optimization.

Term	Abbreviation used in this white paper	Description
Time-to-Value	—	Practical effort and lead time from initial proof-of-concept to a repeatable, scalable capability used in real decisions.
Virtual Commissioning	VC	Using simulation to validate logic/integration before physical commissioning; commonly covers SIL/HIL variants and scenario testing.

References

Core references

[1]

M. Redeker, J. N. Weskamp, B. Rössl, and F. Pethig, 'A Digital Twin Platform for Industrie 4.0', in *Data Spaces*, E. Curry, S. Scerri, and T. Tuikka, Eds, Cham: Springer International Publishing, 2022, pp. 173–200. doi: 10.1007/978-3-030-98636-0_9.

[2]

G. Shao, S. P. Frechette, and V. Srinivasan, 'An Analysis of the New ISO 23247 Series of Standards on Digital Twin Framework for Manufacturing', *NIST*, Jun. 2023, Accessed: Mar. 26, 2026. [Online]. Available: https://www.nist.gov/publications/analysis-new-iso-23247-series-standards-digital-twin-framework-manufacturing

[3]

'Asset Administration Shell Specification - Part 1: Metamodel Schema'.

[4]

M. C. Kennedy and A. O'Hagan, 'Bayesian Calibration of Computer Models', *Journal of the Royal Statistical Society Series B: Statistical Methodology*, vol. 63, no. 3, pp. 425–464, Sep. 2001, doi: 10.1111/1467-9868.00294.

[5]

B. Lei *et al.*, ‘Bayesian optimization with adaptive surrogate models for automated experimental design’, *npj Comput Mater*, vol. 7, no. 1, p. 194, Dec. 2021, doi: 10.1038/s41524-021-00662-x.

[6]

‘Digital Thread for Manufacturing’, *NIST*, Oct. 2021, Accessed: Mar. 17, 2026. [Online]. Available: https://www.nist.gov/programs-projects/digital-thread-manufacturing

[7]

B. Boss *et al.*, ‘Digital Twin and Asset Administration Shell Concepts and Application in the Industrial Internet and Industrie 4.0’. Accessed: Mar. 26, 2026. [Online]. Available: https://www.iiconsortium.org/pdf/Digital-Twin-and-Asset-Administration-Shell-Concepts-and-Application-Joint-Whitepaper.pdf

[8]

‘Digital Twin Consortium publications and white papers’, Digital Twin Consortium. Accessed: Mar. 31, 2026. [Online]. Available: https://www.digitaltwinconsortium.org/publications/

[9]

‘Digital twin reference model and standardization to realize a sustainable industry’. Accessed: Mar. 31, 2026. [Online]. Available: https://www.plattform-i40.de/IP/Redaktion/EN/Downloads/Publikation/202404_Digital_twin_sustainable_industry.html

[10]

‘Edge Computing Unearths New Value for Mining and Metals Applications’, Sep. 2020.

[11]

N. Saha, V. Gadow, and R. Harik, ‘Emerging Technologies in Augmented Reality (AR) and Virtual Reality (VR) for Manufacturing Applications: A Comprehensive Review’, *Journal of Manufacturing and Materials Processing*, vol. 9, no. 9, p. 297, Sep. 2025, doi: 10.3390/jmmp9090297.

[12]

N. Saha, V. Gadow, and R. Harik, ‘Emerging Technologies in Augmented Reality (AR) and Virtual Reality (VR) for Manufacturing Applications: A Comprehensive Review’, *Journal of Manufacturing and Materials Processing*, vol. 9, no. 9, p. 297, Sep. 2025, doi: 10.3390/jmmp9090297.

[13]

‘Functional Mock-up Interface’. Accessed: Mar. 26, 2026. [Online]. Available: https://fmi-standard.org/

[14]

T. Nieminen, ‘Gamification of a training simulator’, 2023.

[15]

R. Badrinarayanan and L. Ae, ‘Hybrid Digital Twins for Operations and Maintenance’.

[16]

‘IEC 62541-14:2026’. Accessed: Mar. 26, 2026. [Online]. Available: https://webstore.iec.ch/en/publication/82257

[17]

‘IEC 62832-1:2020’. Accessed: Mar. 26, 2026. [Online]. Available: https://webstore.iec.ch/en/publication/65858

[18]

‘IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA)–Framework and Rules’, *IEEE Std 1516-2025 (Revision of IEEE Std 1516-2010)*, pp. 1–40, May 2025, doi: 10.1109/IEEESTD.2025.11004567.

[19]

‘ISA-95 Standard: Enterprise-Control System Integration’, http://isa.org . Accessed: Apr. 14, 2026. [Online]. Available: https://www.isa.org/standards-and-publications/isa-standards/isa-95-standard

[20]

‘ISO 23247-1:2021(en), Automation systems and integration — Digital twin framework for manufacturing — Part 1: Overview and general principles’. Accessed: Mar. 16, 2026. [Online]. Available: https://www.iso.org/obp/ui/en/#iso:std:iso:23247:-1:ed-1:v1:en

[21]

‘ISO 23247-4:2021(en), Automation systems and integration — Digital twin framework for manufacturing — Part 4: Information exchange’. Accessed: Mar. 31, 2026. [Online]. Available: https://www.iso.org/obp/ui/en/#iso:std:iso:23247:-4:ed-1:v1:en

[22]

G. Shao, ‘Manufacturing Digital Twin Standards’, *NIST*, Sep. 2024, Accessed: Mar. 26, 2026. [Online]. Available: https://www.nist.gov/publications/manufacturing-digital-twin-standards

[23]

modelica/fmi-standard. (Mar. 25, 2026). C. Modelica Association. Accessed: Mar. 26, 2026. [Online]. Available: https://github.com/modelica/fmi-standard

[24]

E. Dyubanova, ‘Open Industry 4.0 Alliance’.

[25]

C. Wang, X. Qiang, M. Xu, and T. Wu, ‘Recent Advances in Surrogate Modeling Methods for Uncertainty Quantification and Propagation’, *Symmetry*, vol. 14, no. 6, p. 1219, Jun. 2022, doi: 10.3390/sym14061219.

[26]

‘Releases · admin-shell-io/aas-specs-metamodel’, GitHub. Accessed: Mar. 26, 2026. [Online]. Available: https://github.com/admin-shell-io/aas-specs-metamodel/releases

[27]

E. Holterman *et al.*, ‘Roadmap to strengthen the U.S. manufacturing supply chain via digital thread technology’, National Institute of Standards and Technology

(U.S.), Gaithersburg, MD, NIST GCR 24-057, Oct. 2024. doi: 10.6028/NIST.GCR.24-057.

[28]

J. Voas, 'Security and Trust Considerations for Digital Twin Technology', National Institute of Standards and Technology, Gaithersburg, MD, NIST IR 8356, 2025. doi: 10.6028/NIST.IR.8356.

[29]

Working Group Specifications, *Specification of the Asset Administration Shell Part 1: Metamodel (Version 3.1.2)*. doi: 10.62628/IDTA.01001-3-1-2.

[30]

E. Lee, H. Mun, H. Lim, and S. Park, 'The Efficacy of Simulator Technology for Forwarder Operator Training: A Preliminary Study in South Korea', *Forests*, vol. 16, no. 6, p. 882, May 2025, doi: 10.3390/f16060882.

[31]

T. Saaranen, 'Utilization of simulations for the development of operator assistant systems in forest machines', 2023.

Supplementary references

[1]

Y. Wang, Z. Su, S. Guo, M. Dai, T. H. Luan, and Y. Liu, 'A Survey on Digital Twins: Architecture, Enabling Technologies, Security and Privacy, and Future Prospects', Jan. 31, 2023. doi: 10.1109/JIOT.2023.3263909.

[2]

F. Tao, H. Zhang, and C. Zhang, 'Advancements and challenges of digital twins in industry', *Nat Comput Sci*, vol. 4, no. 3, pp. 169–177, Mar. 2024, doi: 10.1038/s43588-024-00603-w.

[3]

'Defining the digital twin'.

[4]

‘Definition of Digital Thread’, Digital Twin Consortium. Accessed: Mar. 17, 2026. [Online]. Available: https://www.digitaltwinconsortium.org/initiatives/the-definition-of-digital-thread/

[5]

‘Develop Industrial Facility Digital Twins Solutions With NVIDIA’, NVIDIA. Accessed: Mar. 17, 2026. [Online]. Available: https://www.nvidia.com/en-us/use-cases/industrial-facility-digital-twins/

[6]

‘Digital Twin IoT - AWS IoT TwinMaker - AWS’, Amazon Web Services, Inc. Accessed: Mar. 16, 2026. [Online]. Available: https://aws.amazon.com/iot-twinmaker/

[7]

Y. Sun *et al.*, ‘Digital Twin-Driven Reinforcement Learning for Obstacle Avoidance in Robot Manipulators: A Self-Improving Online Training Framework’, Mar. 19, 2024, *arXiv*: arXiv:2403.13090. doi: 10.48550/arXiv.2403.13090.

[8]

‘Digital Twins: Making the Vision Achievable | Ansys White Paper’. Accessed: Mar. 31, 2026. [Online]. Available: https://www.ansys.com/resource-center/white-paper/digital-twins-making-vision-achievable

[9]

A. Dayaratna, ‘Exploring the Intersection of NVIDIA Omniverse, Digital Twins, and the Industrial Metaverse Through Advanced 3D Modeling Technologies’.

[10]

D. Chiaro, P. Qi, A. Pescapè, and F. Piccialli, ‘Generative AI-Empowered Digital Twin: A Comprehensive Survey With Taxonomy’, *IEEE Transactions on Industrial Informatics*, vol. 21, no. 6, pp. 4287–4295, Jun. 2025, doi: 10.1109/TII.2025.3540473.

[11]

‘Guidance for Digital Twin Framework on AWS’, Amazon Web Services, Inc. Accessed: Mar. 31, 2026. [Online]. Available: https://aws.amazon.com/solutions/guidance/digital-twin-framework-on-aws/

[12]

‘Guidance for Industrial Digital Twin on AWS’, Amazon Web Services, Inc. Accessed: Mar. 31, 2026. [Online]. Available: https://aws.amazon.com/solutions/guidance/industrial-digital-twin-on-aws/

[13]

A. Frost and S. W. Paper, ‘How Bentley Systems’ iTwin Platform is enabling a digital twin ecosystem for the world’s infrastructure’, 2021.

[14]

‘IDTA – working together to promote the Digital Twin’, IDTA. Accessed: Mar. 31, 2026. [Online]. Available: https://industrialdigitaltwin.org/en/

[15]

J. McKenna, ‘Into the Omniverse: How Industrial AI and Digital Twins Accelerate Design, Engineering and Manufacturing Across Industries’, NVIDIA Blog. Accessed: Mar. 17, 2026. [Online]. Available: https://blogs.nvidia.com/blog/industrial-ai-digital-twins-omniverse/

[16]

baanders, ‘Quickstart - Get started with Azure Digital Twins Explorer - Azure Digital Twins’. Accessed: Mar. 16, 2026. [Online]. Available: https://learn.microsoft.com/en-us/azure/digital-twins/quickstart-azure-digital-twins-explorer

[17]

C.-H. Chu, W. Bernstein, Y. “WILL” Zhang, V. R. Krishnamurthy, and J. Ma, Eds, ‘Special Issue: Extended Reality in Design and Manufacturing’, *J. Comput. Inf. Sci. Eng*, vol. 24, no. 030301, Feb. 2024, doi: 10.1115/1.4064640.

[18]

C.-H. Chu, W. Bernstein, Y. “WILL” Zhang, V. R. Krishnamurthy, and J. Ma, Eds, ‘Special Issue: Extended Reality in Design and Manufacturing’, *J. Comput. Inf. Sci. Eng*, vol. 24, no. 030301, Feb. 2024, doi: 10.1115/1.4064640.

[19]

L. Schreiber, ‘Supercharging the industry transformation with the comprehensive Digital Twin’.

[20]

J.-F. Yao, Y. Yang, X.-C. Wang, and X.-P. Zhang, ‘Systematic review of digital twin technology and applications’, *Vis. Comput. Ind. Biomed. Art*, vol. 6, no. 1, p. 10, May 2023, doi: 10.1186/s42492-023-00137-4.

[21]

‘The promise of a digital twin strategy’, *Digital transformation*.

[22]

R. Ružarovský, R. Skýpala, J. Šido, M. Csekei, T. Horák, and P. Střelec, ‘Virtual Commissioning of Industrial Robot through Software-in-the-Loop Strategies for Cyber-Physical Systems’, 2024.

[23]

A. F. Nicolescu and C. Pupăză, ‘Virtual Commissioning of manufacturing systems. A review’, *Proceedings in Manufacturing Systems*, vol. Volume 19, no. Issue 3, pp. 91–96, 2024.

[24]

R. Raffaeli, P. Bilancia, M. Peruzzini, S. Pisu, G. Berselli, and M. Pellicciari, ‘Virtual Prototyping and Commissioning of Manufacturing Cycles in Robotic Cells’, in *Design Tools and Methods in Industrial Engineering III*, M. Carfagni, R. Furferi, P. Di Stefano, L. Governi, and F. Gherardini, Eds, Cham: Springer Nature Switzerland, 2024, pp. 391–398. doi: 10.1007/978-3-031-58094-9_43.

[25]

‘What is Amazon IoT TwinMaker? - Amazon IoT TwinMaker’. Accessed: Mar. 17, 2026. [Online]. Available: https://docs.amazonaws.cn/en_us/iot-twinmaker/latest/guide/what-is-twinmaker.html

[26]

baanders, ‘What is Azure Digital Twins? - Azure Digital Twins’. Accessed: Mar. 16, 2026. [Online]. Available: https://learn.microsoft.com/en-us/azure/digital-twins/overview

[27]

‘Why Companies Must Prioritize Digital Thread | PTC’. Accessed: Mar. 31, 2026. [Online]. Available: https://www.ptc.com/en/resources/corporate/whitepaper/why-companies-must-prioritize-digital-thread