Digital Twin Platforms – Deep Research Report

> Multi-perspective, corpus-grounded research. Method adapted from hadufer/claude-storm (MIT) / Stanford OVAL STORM (Shao et al., NAACL 2024). Depth: standard - Perspectives: the Platform Builder, the Reference-Architecture Scholar, the Skeptic, the Adoption & Business-Model Analyst, and the Lifecycle/DevOps Engineer, plus a basic fact writer - Citekeys: see References - Date: 2026-09-21

Summary

This report covers what a digital twin platform is, the commercial, open-source, and academic platform landscape, reference architectures and standards and their measured gap with practice, what builders must add on top of a chosen platform, platform lifecycle management, and platform economics, adoption, and trust – all as documented in this project's synced corpus. It does not cover per-platform runtime benchmarking, a domain-specific digital twin literature review beyond what bears on platform choice, twin retirement/decommissioning, quantified integration cost in person-hours, or realized DTaaS pricing; none of these is answered in the corpus this report searched.

A digital twin platform is not a digital twin: a platform is the reusable substrate of services, applications, and subsystems used to implement and operate twin instances, while a twin is one synchronized virtual counterpart of one physical asset [1], [2]. The ground under both terms is unsettled – one review collected 46 definitions of the twin alone [3] – and the field's own shared measuring stick, Kritzinger et al.'s model/digital-shadow/digital-twin integration ladder [4], is strict enough that a paper self-titled as a digital twin case study rates its own worked examples as shadow-level or below under its own rubric [5]. "Digital twin platform" itself spans at least three different things in this literature: vendor cloud services (Azure Digital Twins/DTDL, AWS IoT TwinMaker), open-source infrastructure components that require assembly (Eclipse Ditto, Eclipse BaSyx/AAS), and academic composites that wire the first two together (OpenTwins, KTWIN, DTaaS) [6]–[8].

No platform examined here ships a complete twin: OpenTwins names Eclipse Ditto as its central piece and still requires hand-assembled storage, a custom extended API, and Kafka as message-broker glue [9], [10], and documentation maturity gates candidate selection before any technical comparison happens [8], [11]. Reference architectures have multiplied – ISO 23247, the Digital Twin Consortium's Platform Stack, makeTwin, and RAMI 4.0/AAS all remain live, non-converged candidates [12]–[15] – and two independent empirical studies, using different corpora and methods, both find implemented practice diverging sharply from them: a triangulated 140-paper/33-respondent/four-interview study of ISO 23247 conformance [16], and a 22-study survey where only one paper (4.5%) referenced any reference architecture at all [17]. Even inside a single, comparatively mature standard, no open-source AAS implementation is spec-complete [18], and the standard itself has not converged on one structural form [19].

Across the platform lifecycle, the pattern repeats: Build/Dev tooling for digital twins is concretely worked out, while Operate/Monitor – drift management, schema evolution without teardown, safe runtime reconfiguration – is consistently deferred, described as still requiring "a lot of manual effort," or left as an explicitly named future work item by the very papers proposing it [20]–[23]. An interview study across 10 organizations independently

confirms that standard CI/CD practice does not transfer cleanly to cyber-physical settings [24], and an industry maturity assessment corroborates that current practice sits at only "medium maturity" [25]. Economics remain similarly unsettled: three market-size forecasts in the corpus cannot describe the same market by construction [26]–[28], SME adoption lags large firms on cost and ROI-horizon grounds [29], [30], and the field's own institutions – independently, in near-identical language – confess that publicity currently outweighs the evidence base of success [31], [32].

Synthesis Briefing

Executive summary: Digital twin platforms are best understood as metamodels plus plumbing: each platform fixes a description language and a functional slice, and the working twin emerges only from builder-supplied glue between slices. The literature's reference architectures and its implemented platforms have drifted apart – measurably so, in two independent studies – and several perspectives interviewed for this report converge on a related missing artifact: standardized, audited, cross-platform evidence of performance, integration cost, and business value that would let the field's central claims be tested rather than asserted.

Key findings (ranked by reliability):

1. **Implemented practice diverges sharply from the reference architectures the field keeps publishing.** Reliability 9/10 – the only triangulated result in the corpus (140-paper SLR + 33-respondent survey + interviews), independently corroborated by a second, differently-scoped 22-study survey. Supported by the Scholar and Lifecycle perspectives [16], [17].
2. **No platform examined in this corpus ships a complete twin; builders supply the glue.** Two evidence types of different strength support this, scored separately rather than bundled: the glue-code claim itself rests on one academic platform, OpenTwins, which requires hand-assembled storage, a custom API, and Kafka as broker glue on top of its "central piece" – reliability 6/10, since it is a single platform's case rather than a corpus-wide count, and no commercial or academic platform besides OpenTwins is checked against this criterion here. The documentation-maturity-as-selection-gate finding is better-evidenced – reliability 8/10 – resting on a systematic 14-candidate open-source survey and corroborated independently by a 19-platform commercial comparison. Supported by the Platform Builder and Basic-fact perspectives [8]–[11].
3. **The operational phase is the least mature stage of the platform lifecycle.** Reliability 8/10. Every DevOps-for-DT proposal in the corpus works out Build/Dev concretely while deferring or explicitly declining to detail Operate/Monitor; an independent 10-organization interview study confirms CI/CD does not transfer cleanly to CPS; an industry maturity assessment corroborates "medium maturity" at best. Supported by the Lifecycle/DevOps perspective [20], [21], [24], [25].
4. **Terminological inflation is documented in at least one first-party instance, not just asserted from outside.** Reliability 5/10. A paper self-titled as a digital twin case study rates its own worked examples as shadow-level or below under its own rubric. Supported by the Skeptic perspective; this is one documented case rather than a corpus-wide count, and the field's continued, active re-mapping of its own vocabulary [33] is equally consistent with definitional rigor as with inflation, so the

field-wide version of this claim is deliberately not asserted [4], [5].

5. **Platform economics remain unproven and market-size forecasts are internally inconsistent.** Reliability 6/10. Three market forecasts in the corpus cannot describe the same market by construction; standards conformance is credited with ROI in named single cases (Sandvik, 15% tool-life increase) rather than a replicated body of evidence; SME adoption lags on cost and ROI-horizon grounds. Supported by the Adoption Analyst perspective [12], [26]–[29].

Hidden connection: The Builder's unmeasured glue-code cost and the Scholar's finding that reference-architecture conformance is barely measured in practice are not the same gap – one is an integration-effort question, the other a standards-adoption question – but both share a missing prerequisite: no agreed metrics exist to compare what one platform costs a builder against what another does. The DTaaS survey names candidate metrics (twinning rate, twin fidelity, age-of-twin) as still absent from common use, citing a separate DT-tools survey that proposed synchronization latency as one candidate metric among several [34]. The Lifecycle Engineer's unspecified Operate phase is a related but distinct problem – a capability gap (safe runtime reconfiguration, drift correction) that would remain unsolved even if cross-platform comparison metrics existed. Two honesty notes: the Skeptic perspective guarantees an evidence-deficit theme by design, so this partial convergence is partly a property of the lenses chosen; and the connection drawn here is narrower than "all roads lead to one missing benchmark" – it covers the Builder and Scholar findings, not a unification of every perspective's gap.

Actionable insight (for a digital twin platform builder): The highest-leverage open contribution is a reproducible cross-platform benchmark – one exemplar twin, implemented on several platforms, with standardized performance metrics [34] *and* accounted integration/glue-code effort, extending [35]'s one quantified effort-reduction figure (55%/25% for a model-driven metamodel layer) from a single platform to a cross-platform comparison. A companion study testing whether ISO 23247's never-implemented functional entities (plug-and-play, peer interface, data assurance) reduce integration seams when actually built would connect the benchmark directly to the standards-vs-practice debate [16].

Frontier question: Can any platform demonstrate, with measured evidence rather than architecture diagrams, that it reduces the lifetime cost of building, evolving, and safely reconfiguring a digital twin at runtime – closing the gap between the concretely worked-out Build/Dev tooling this corpus documents and the Operate/Monitor phase every source that touches it describes as unsolved?

What a Digital Twin Platform Is

A digital twin instance and a digital twin platform are not the same object, and the corpus treats collapsing the two as the first mistake a builder can make. A platform is the reusable substrate of infrastructure, services, and subsystems that lets a twin's model, data, and services be assembled rather than built from scratch for each project [2]. The DTC's Platform Stack Architectural Framework makes this distinction explicit by layering a security/trustworthiness foundation, an IT/OT infrastructure layer, a virtual-representation layer, a service layer, and an application layer, and it positions itself as an integrative overlay across RAMI 4.0, the IIC Connectivity Framework, FIWARE, the Asset Administration Shell, and the commercial IBM/Microsoft/AWS stacks rather than a rival standard to any of

them [13]. Read together, these two sources agree that "platform" names an infrastructural layer distinct from any single twin running on it, even though they approach the distinction from opposite ends – one as a minimal functional definition, the other as a layered reference framework meant to subsume competing ones.

What a "digital twin" itself denotes, however, is not settled underneath that infrastructural distinction. A decade-spanning review collected 46 published definitions of the term and argued that their proliferation "threatens to dilute the concept" [3]. The term traces to Michael Grieves in 2002, originally proposed under the name "Conceptual Ideal for Product Life Cycle Management," and divergent application fields have since generated their own interpretations rather than converging on his [36]. The field's own community-building efforts corroborate that this is an open problem rather than a settled one awaiting popularization: establishing even a shared vocabulary was a stated objective of the EDT.Community's seminars, evidence that practitioners themselves treat definitional consensus as unfinished work [37].

Faced with that proliferation, the corpus repeatedly falls back on one shared measuring stick: Kritzinger et al.'s integration ladder, which classifies systems by the direction and automation of data flow between physical and virtual objects. A digital model exchanges data only manually in both directions; a digital shadow adds automated one-way flow from physical to virtual; only automated bidirectional flow qualifies as a digital twin [4]. The same review found frequent misuse of these terms in the literature it surveyed, with many self-described twins in fact being models or shadows under its own criterion [4]. That the ladder functions as the field's common substrate despite competing reference vocabularies is corroborated independently: one cross-mapping exercise explicitly aligns Kritzinger et al.'s terms against Tao et al.'s five-dimension model, ISO 23247's vocabulary (observable manufacturing element, digital representation, data collection entity), and IDTA/AAS terms (asset, submodel, data element), treating the integration ladder as the shared layer underneath four otherwise divergent standards [33].

The ladder is not merely a taxonomy academics apply to others' work from the outside – it is strict enough that papers applying it to their own systems can fail their own rubric. A case study titled "A Case Study for Enhancing Maintenance of Cyber-Physical Systems With Digital Twin" rated its own worked examples' integration dimension as L1 (digital shadow, one-way automated flow) in three of its four failure-example rows, and as L0 (offline twin, no automated flow at all) in the fourth [5]. None of the paper's own named case studies reached bidirectional "digital twin" status by its own stated criterion – a first-party admission rather than an outside critique, and a concrete instance of the terminological inflation the definitional-proliferation literature describes in the abstract [3], [5].

Beneath the definitional and taxonomic questions sits a more concrete one for anyone choosing a platform: each platform's description language imposes its own conceptual schema, or metamodel, on how a twin may be represented. A comparison of modeling capabilities across platforms – including Azure's DTDL – finds that they mostly rely on well-known object-oriented modeling concepts, sharing a common core, but each also extends that core with its own domain-specific constructs, posing the question – its own title – of whether these amount to "old wine in new bottles" [38]. Even the shared core does not eliminate switching cost: the same study's portability experiment demonstrates a working transformation from one platform's modeling language to another's via a UML pivot model, but identifies concrete, quantifiable information loss in the process (version numbers

and quantities among the details that do not survive the transformation) [38]. Choosing a digital twin platform therefore means choosing a metamodel that shares structure with its competitors but does not port losslessly to them.

The Platform Landscape

The corpus uses "digital twin platform" for at least three genuinely different things: vendor-branded commercial cloud services sold as a subscription product, open-source infrastructure-level components that require assembly before they do anything twin-specific, and academic composites that wire the first two together with cloud-native tooling into a fuller stack [6]–[8]. A head-to-head comparison that puts Azure Digital Twins, Eclipse Ditto, and OpenTwins in the same table risks scoring a finished product against a toolkit and a research prototype as if they occupied one market; the sections below keep the three apart.

Commercial cloud platforms

Azure Digital Twins and AWS IoT TwinMaker recur across the corpus as the two platforms treated as primary references for commercial offerings [6], [39]. Azure Digital Twins is consistently described as a Platform-as-a-Service that ships its own modeling language, DTDL [6], [40] – two independent sources converge on this framing, which matters because it is the DTDL metamodel, not Azure's compute layer, that KTWIN later identifies as the constraint it builds beyond. Bosch IoT Suite differs from both by offering an explicit on-premise deployment option with no third-party access, alongside its cloud-connected mode, for manufacturing use [41]; where Azure and AWS are described as hosted services, Bosch's suite is documented as giving manufacturers a choice the other two do not.

Cost and lock-in surface as the two recurring weaknesses of the commercial tier, and here the sources corroborate each other on the mechanism even though they frame it differently. Zech et al. detail the Azure case concretely: using Azure Digital Twins in practice means drawing on "a battery of Azure services, e.g., Azure IoT, Azure Functions, Azure Twin Explorer, Logic App," each incurring its own additional cost, and note that AECO-sector commercial platforms generally combine proprietary modeling languages, closed tools, and immediate vendor lock-in [42]. Parle et al. independently corroborate the cost side at a coarser grain, stating that high upfront implementation costs are a significant barrier specifically for small and medium-sized enterprises [30]. The same source runs a SWOT-style comparison of Azure Digital Twins against Siemens Xcelerator: Azure's weaknesses are a lack of local-server hosting and a training/learning-curve burden, while Xcelerator's chief weakness is not technical but a lower market awareness among platform buyers specifically in the digital-twin space [30]. Khoshkenar et al.'s independent 19-platform comparison was itself built "from provider websites, white papers, press releases and user documentation" rather than hands-on testing [11] – a methodological admission that public documentation, not trial deployment, is the primary signal available to a buyer evaluating any of these platforms before committing.

The open-source ecosystem

Gil et al.'s systematic survey, which identified 14 open-source digital twin frameworks and tools – including Eclipse Ditto, Eclipse BaSyx, PYI40AAS, SAP I4.0 AAS, iTwin.js, CPS-Twinning, Twined, and INTO-CPS – via Google Search, GitHub, and Google Scholar

through July 2022, is the corpus's load-bearing account of this tier [8]. Its selection method is itself a finding worth stating plainly: candidates were dropped before any technical comparison if "the maturity of the documentation available is not enough to be used" [8]. Documentation maturity functions as a hard gate on the open-source side of the landscape, in a way that has no analog for the commercial platforms, whose documentation is instead the primary evaluation signal itself, per Khoshkenar et al. above [11]. Eclipse BaSyx, one of the Gil et al. survey's 14 candidates, is examined in more depth by a dedicated implementation survey: a reusable test framework built against the Asset Administration Shell (AAS) specification's model elements and API calls was run across four open-source AAS implementations, including BaSyx [18]. All four supported at least a minimal required feature set, but none implemented the full specification – described as "the first attempt at a comprehensive comparison of AAS implementations" [18]. This is the open-source ecosystem's characteristic shape in the corpus: individually useful infrastructure-level components, none of which is a complete platform on its own, that a builder must assemble and, per this finding, cannot assume are spec-complete even when they claim conformance.

Academic and research platforms

Academic platforms in the corpus are described as compositions of the commercial and open-source layers rather than as ground-up alternatives to either. OpenTwins is framed as an open-source framework for "next-gen compositional digital twins," combining asset monitoring, AI/ML integration for streaming prediction, and 3D visualization into a single container-based platform [7]. KTWIN takes the same compositional approach from a different starting problem: it treats Ditto, Azure Digital Twins, and AWS IoT TwinMaker as "the primary references" and argues that DTDL-style languages "lack … ways to describe technical application aspects that compose the deployment … such as the number of allocated CPU and memory … and the application auto-scaling settings" [6]. KTWIN's response is deployment portability – flexibility to run on any Kubernetes cluster, on-premise, edge, or cloud – aimed explicitly at reducing the vendor lock-in that Zech et al. document as an Azure weakness [6], [42]. DTaaS (the Talasila/INTO-CPS platform, a proper noun distinct from the generic Digital-Twin-as-a-Service category discussed below) is documented as "an open-source development intended to provide all organisations an affordable entry point into engineering their own DTs" [43], and Gil et al.'s separate 2024 source independently confirms that a public flex-cell digital twin example built on this platform is hosted on the INTO-CPS Association's GitHub [44]. Read together, OpenTwins, KTWIN, and DTaaS corroborate the section's opening finding rather than complicating it: each names specific commercial and open-source components it is built from, which is the defining trait separating this tier from either of the other two [6], [7], [43].

Standardization across all three tiers remains recent rather than settled. Gil et al. note that "the recent ISO-23247 standard and IEC-63278-1 are pioneer standards in DT technology" [44], language that implies no agreed conformance baseline existed against which platform claims – commercial, open-source, or academic – could be checked until very recently. That absence is consistent with the AAS implementation survey's finding that even a comparatively mature specification has no fully conformant open-source implementation [18], and it qualifies every comparison in this section: SWOT tables, feature matrices, and taxonomy surveys alike are being drawn against a standards baseline that only recently began to exist.

Reference Architectures and the Standards They Lean On

ISO 23247's four-domain structure

ISO 23247 decomposes a manufacturing digital twin into four interconnected domains – Observable Manufacturing, Device Communication, Digital Twin, and User – linked by a Cross-system Entity that supplies shared functions such as data translation, data assurance, and security support [12]. NIST's own elaboration of the standard works this structure through three concrete use-case scenarios, implementable on its Smart Manufacturing Systems Test Bed, that walk a real shop floor through the framework end to end [45]. Both NIST-authored sources present the standard as a deliberate adoption aid rather than a purely descriptive taxonomy: [45] frames ISO/DIS 23247's common terminology and guidelines as addressing implementation challenges "especially among small and medium-sized enterprises," and [12] ties conformance to measurable outcomes, citing a Sandvik cutting-tool-life digital twin that "helped increase tool life for milling operations by 15%" alongside an airframe-weight and fastener-length optimization example.

The Digital Twin Consortium's Platform Stack

The Digital Twin Consortium's (DTC) Platform Stack Architectural Framework layers a security-and-trustworthiness foundation, an IT/OT infrastructure layer, a virtual-representation layer, a service layer, and an application layer, and it explicitly cross-maps itself onto RAMI 4.0, the Industrial Internet Consortium's Connectivity Framework, FIWARE, the Asset Administration Shell, and commercial stacks from IBM, Microsoft, and AWS – positioning itself as an integrative overlay rather than a competing standard [13], [46]. That self-positioning is put to work rather than left theoretical: the DISCS project uses the Platform Stack together with the DTC's capability "Periodic Table" to architect a system-of-systems composition of digital twins for smart cities, applying the framework's layered decomposition to a domain well outside its manufacturing origins [47]. [13] also supplies the definitional distinction this report relies on throughout – a digital twin platform as the reusable substrate an instance runs on, not the instance itself.

makeTwin and the wider field of competing candidates

A fourth architecture, makeTwin, derives its structure from Tao's five-dimension digital twin model and is explicitly motivated by the claim that existing digital twin platforms on the market carry significant functional limitations [14]. Secondary treatments corroborate this lineage without adding a competing account: [48] situates makeTwin as a software-platform extension of the five-dimension model, and [49] places it within the broader reference-architecture landscape while separately noting that existing digital twin platform evaluation methods "are mostly based on subjective opinions, simple decision-making methods, and general-purpose criteria" – an observation that applies as much to choosing among ISO 23247, the DTC stack, makeTwin, and AAS as to evaluating any single platform. A broader survey by [50] covers ISO 23247, the DTC reference architecture, and IDTA/AAS together as the field's competing standardization efforts, treating them as parallel entries in a landscape rather than stages of a single lineage.

The clearest evidence that these architectures function as parallel, non-converging candidates is a methodological artifact rather than a stated conclusion: [15] cross-walks its own model-driven-engineering architecture against all four major reference frameworks – the

DTC reference architecture, RAMI 4.0, AAS, and ISO 23247 – in a single comparison table. That a contributor felt it necessary to check a new architecture against four existing ones, rather than extend or subsume one of them, is itself evidence the community has not settled on a converged stack. One paper works the opposite direction, mapping vocabulary rather than architecture: [33] cross-maps Kritzinger et al.'s integration-ladder terms against Tao's five-dimension model, ISO 23247's terms (observable manufacturing element, digital representation, data collection entity), and IDTA/AAS terms (asset, submodel, data element), suggesting the integration ladder functions as a shared substrate underneath four otherwise divergent reference vocabularies even where the architectures themselves do not converge.

Two independent studies find practice diverges from the standards

The best-evidenced finding in this section is that implemented practice diverges sharply from the reference architectures on paper, and two independent empirical studies reach that conclusion by different routes. [16] combines a 140-paper systematic literature review yielding 29 documented digital twin architectures, a 33-respondent survey, and four expert interviews, and reports detailed misalignment between current digital twin architectures and the ISO 23247 reference architecture: 81% of survey respondents rated Continuous Deployment as architecturally unimportant, a judgment the authors concur with in arguing it does not belong in the reference architecture, while other practiced capabilities such as data storage and versioning go uncaptured by the standard's functional entities, and some standardized entities – plug-and-play, peer interface, data assurance – show 0% implementation among the studied systems. [17] reaches a comparable conclusion from an entirely different corpus and method: a systematic survey of 22 primary studies on digital-twin-enabled AI simulation found "the complete lack of digital twin standards, architectural blueprints, and reporting guidelines" in the sampled literature, with only one study (4.5%) referencing RAMI 4.0 at all, and even that one sole case failed to connect RAMI 4.0 to the Asset Administration Shell, RAMI's own standardized component for digital twinning. The two studies disagree in scope – one measures architectural feature adoption against ISO 23247 specifically, the other measures bare mention of any reference architecture across AI-simulation papers – but they agree on the underlying pattern: reference architectures exist in the literature largely as design targets that implementations do not track.

Even AAS, the more mature standard, has not converged on one form

The divergence is not confined to comparisons between standards; it recurs inside a single, comparatively mature one. [18] built a reusable test framework against the Asset Administration Shell (AAS) specification's model elements and API calls and ran it on four open-source AAS implementations, including Eclipse BaSyx; all four support at least a minimal required feature set, but none implements the full specification, and the authors describe their work as "the first attempt at a comprehensive comparison of AAS implementations." [19] documents a related but distinct gap: it identifies three structurally different "types" of AAS in circulation and evaluates each separately against digital-twin engineering requirements, rather than treating "the AAS" as a single, converged standard with one realized form. Read together, these two sources show the same pattern found at the level of ISO 23247 versus practice recurring one level down, inside AAS itself – no implementation is standard-complete, and the standard has not settled on a single structural realization for implementations to converge toward. The AAS specification's own scope, defined in [51], is

used here only to ground what "standard-complete" would mean, not as an independent finding about adoption.

Translation between metamodels as a symptom of non-convergence

That reference architectures and their metamodels have not converged is corroborated by the existence of dedicated cross-translation research: [52] implements and tests a DTDL-to-AAS transformation, and [53] proposes a mapping from DTDL into OPC UA. Neither transformation is trivial engineering; both exist as standalone research contributions precisely because no shared metamodel lets a twin described in one platform's language be read directly by another. Their existence is evidence, not merely a workaround – if the reference architectures surveyed above had converged on a common conceptual schema, translation layers between platform description languages would be unnecessary rather than publishable.

Building on a Platform in Practice

What a platform ships versus what a builder must add

OpenTwins illustrates the gap between a platform's stated core and the infrastructure required to run it. Its architecture names Eclipse Ditto as its "central piece," but Ditto itself does not ship storage for its own state, historical data retention, or simulation support [9]. Builders adopting OpenTwins must supply MongoDB for Ditto's internal storage, InfluxDB for historical data Ditto does not keep, and a custom "Ditto Extended API" written to cover functionality Ditto lacks outright [9]. A lightweight variant of the same architecture adds Apache Kafka as message broker not because Ditto calls for one, but because the assembled system needs one to move data between components [9]. That same Kafka layer, via Kafka-ML, becomes the seam connecting Ditto's digital twin state to machine-learning and simulation services – glue code the builder assembles rather than a capability the platform provides [10]. Read together, the two OpenTwins sources describe a single pattern: a platform can present one named "central" component while the working system is a stack of separately sourced storage, API, and messaging layers built around it [9], [10].

Documentation maturity as a selection filter

Before any platform's technical merits are weighed, its documentation must clear a bar. A systematic survey of open-source digital twin frameworks excluded GitHub repositories outright where "the maturity of the documentation available is not enough to be used," filtering candidates before evaluating what they could do [8]. The same survey, searching Google Search, GitHub, and Google Scholar through July 2022, still identified 14 open-source frameworks that cleared that bar, including Eclipse Ditto, Eclipse BaSyx, PYI40AAS, SAP I4.0 AAS, iTwin.js, CPS-Twinning, Twined, and INTO-CPS [8]. A separate, larger industry comparison of 19 platforms shows the same constraint operating at commercial scale: it was built "from provider websites, white papers, press releases and user documentation" rather than hands-on testing [11]. Neither source treats this as incidental – one excludes candidates for it, the other concedes it as the only information actually available before committing to a platform – and together they indicate that public documentation, not trial deployment, is the primary evidence base a builder or reviewer has to work from [8], [11].

Metamodel and modeling-language effort

Choosing a platform means choosing its metamodel, and that choice carries a cost this corpus quantifies only once. Adopting a model-driven metamodel layer, AML4DT, over AutomationML reduced system setup effort by 55% and system evolution effort by 25% compared with a traditional, hand-coded digital twin setup [35]. This single figure gives numeric weight to a pattern otherwise described only qualitatively: a comparison of modeling capabilities across platforms, including Azure's DTDL, finds that each platform reinvents overlapping but incompatible modeling constructs rather than converging on shared ones – "old wine in new bottles" – so that a builder's language choice is also a lock-in decision [38]. The 55%/25% figure and the "old wine in new bottles" finding describe the same underlying cost from opposite directions: one measures what a better metamodel layer saves, the other explains why platforms keep needing one in the first place [35], [38].

That incompatibility across modeling languages is non-trivial to resolve is evidenced by the existence of dedicated translation research rather than off-the-shelf tooling. A DTDL-to-AAS transformation has been implemented and tested as a standalone research contribution [52], and a separate mapping from DTDL into OPC UA has been proposed, also as a research contribution rather than a shipped feature of any platform [53]. Both papers exist because translating between platform-specific modeling languages was not otherwise solved; their presence in the corpus, alongside the incompatibility finding in [38], corroborates that metamodel choice is a switching cost a builder pays once and then carries [38], [52], [53].

Vendor lock-in

Lock-in is named explicitly, not inferred, for specific commercial platforms. For Azure, "one cannot avoid using a battery of Azure services, e.g., Azure IoT, Azure Functions, Azure Twin Explorer, Logic App ... all incurring additional costs" [42]. The same source describes AECO (architecture, engineering, construction, and operations) commercial platforms generally as coming with proprietary modeling languages and formats, closed tools and infrastructures, and immediate vendor lock-in [42]. KTWIN's stated motivation for building on Kubernetes rather than a vendor platform makes the same cost concrete from the builder's side: deployment to any Kubernetes cluster – on-premises, edge, or cloud – is presented as reducing the chance of vendor lock-in relative to Azure Digital Twins, described in the same source as a Platform-as-a-Service offering built around its own DTDL modeling language [6]. The naming of Azure specifically in both a DTaaS-comparison source and a Kubernetes-alternative source, for different underlying reasons – accumulating service costs in one case, deployment topology in the other – indicates lock-in is treated as a property with more than one cause [6], [42].

Deployment topology: cloud-only versus edge-to-cloud

Deployment topology is the dimension where platform choice most visibly trades flexibility for lock-in. A comparison table rates Azure and AWS "NO/Cloud-only" for edge-to-cloud mobility, while Eclipse Ditto and WLDT score only "MEDIUM/LOW" despite a microservices-oriented architecture, because "there is not a platform for deployment and orchestration across the edge-to-cloud continuum" [54]. The same table rates Ditto and WLDT "NO" for variable-load resilience, since native scaling depends entirely on whatever orchestration layer sits underneath them rather than on anything the platform itself provides

[54]. KTWIN was built as a direct response to that gap: DTDL-style modeling languages, in the source's own framing, "lack … ways to describe technical application aspects that compose the deployment … such as the number of allocated CPU and memory … and the application auto-scaling settings," and KTWIN's Kubernetes foundation is offered as a fix, deployable to on-premises, edge, or cloud clusters and positioned explicitly against Ditto, Azure Digital Twins, and AWS TwinMaker as "the primary references" [6]. The two sources describe the same shortfall from either side of it: the comparison table measures cloud-only and orchestration-dependent platforms falling short on mobility and scaling, and the KTWIN paper explains why a Kubernetes-native architecture was built specifically to close that shortfall [6], [54].

Lifecycle: DevOps, Evolution, and Operations

DevOps proposals: Build/Dev worked out, Operate/Monitor deferred

TwinOps is the corpus's foundational proposal for DevOps applied to cyber-physical systems, unifying model-based engineering, digital twins, and DevOps into what its authors term "ModDevOps" [20], [55]. The approach reuses model-based artifacts to drive GitLab-based CI/CD pipelines that build simulation testbenches, validation platforms, and digital twins from a common source [20]. The Build and Dev sides of this pipeline are worked out concretely; the Ops side is not. Hugues et al. state directly that linking the pipeline configuration to a model that in turn configures deployment targets remains manual and is left as future work [20], a gap the 2022 elaboration corroborates without closing [55].

The same asymmetry recurs in the built-asset digital twin literature, which extends DevOps beyond TwinOps's model-based-engineering setting. JuNo-OPS frames continuous CI/CD, automated testing, and monitoring as necessary "to ensure that DTs remain accurate reflections of their physical counterparts" over time [56], and a companion paper generalizes the framework for the systematic engineering and evolution of built-asset twins [57]. A third paper in this cluster goes furthest in naming the gap explicitly: it presents a full DevOps loop for built-asset digital twins but states outright that it will not discuss "the technical aspects related to the Operate and Monitor phases, including their automation and tool support," citing space [21]. That omission is itself evidence for the throughline of this section – even a paper whose stated purpose is to propose DevOps for digital twins treats the Operate side as the part it cannot yet fully specify.

GitOps offers a partial answer to that gap by mapping a declarative, version-controlled, CI/CD-driven model of cluster state onto digital twin runtime operations such as connect and synchronize, combining a static Digital Twin Definition Language with GitOps practice [58]. Its validation, however, is a proof-of-concept on a Fischertechnik Training Factory 4.0 testbed [58] – a small, controlled demonstrator rather than a production deployment, which limits how far its Operate-side claims generalize beyond the case it was tested on.

CI/CD does not transfer cleanly to CPS

The strongest empirical evidence in this corpus that CI/CD does not transfer cleanly from software to cyber-physical systems is an interview study spanning 10 organizations across 8 domains, which identifies concrete structural barriers rather than generic difficulty: combining continuous and periodic builds while balancing hardware-in-the-loop (HiL) and

simulator use, deploying software to physical targets, behavioral divergence between simulators and HiL rigs, and the organizational need to combine hardware and software expertise within a single team [24]. These are not restatements of the TwinOps or JuNo-OPS gap in different words; they are independent, named obstacles that explain why the Operate side of DevOps-for-DT proposals stays underspecified – the practices that make Build/Dev tractable in ordinary software CI/CD do not have a settled CPS analogue.

One paper in the corpus works through a concrete CPS-adapted CI/CD pipeline rather than only naming the barriers. A microservice-based testing framework spans Software-in-the-Loop, Hardware-in-the-Loop, and Operation test levels, and was applied to a real elevator case study, where it reduced testing time and cost [59]. Read alongside Zampetti et al.'s barrier taxonomy, this example shows the barriers are addressable in a specific, narrow case, but the corpus offers no evidence that the same three-tier pipeline design generalizes across the domains the interview study surveyed [24], [59].

Drift, evolution, and reconfiguration: consistently named, consistently unsolved

Model decay after deployment is named explicitly as an open problem rather than a solved engineering concern. Digital twin predictions "can easily become obsolete when the physical world is highly dynamic and evolves in real-time, and when the DT knowledge drifts, decays and/or is not updated," and detecting that drift is nontrivial when sensors fail or update cadence is coarse [60]. Schema and model evolution without teardown sits in the same state: proposing a fluent API for schema-, instance-, and data-level evolution at runtime, one paper describes current practice as involving "a lot of manual effort" and risking data loss "if not done correctly," framing its own contribution as filling a gap rather than replacing an established method [22]. Both papers treat the problem as open at the point of writing; neither describes a deployed, teardown-free evolution pipeline in production use.

Safe runtime reconfiguration of the physical twin is the least developed capability among those the corpus discusses. Fitzgerald et al. identify autonomous reconfiguration at runtime as the capability that distinguishes a digital twin from a digital shadow, and devote a chapter to challenges that "need to be tackled in order to improve the utilisation of DT technology for autonomously reconfiguring systems at runtime," presenting these as unsolved research questions rather than engineering practice [23]. Frasheri et al. independently list "how to effect change in the physical twin through reconfiguration" as an open service capability alongside fault injection and predictive maintenance [61], and Kamburjan et al. motivate a two-layered MAPE-K self-adaptive architecture on the grounds that "the transitions between the lifecycle stages of physical assets may be hard or even impossible to predict accurately by the digital twin at runtime," introducing "stage management" as a new architectural concept to manage that unpredictability [62]. Across these three sources, no paper in the corpus describes a safety or rollback mechanism that would let a reconfiguration be undone if it produces an unsafe physical state – the absence is consistent across independent proposals, not a gap in a single paper's scope.

Corroborating maturity assessments

Two independent assessments, arrived at through different methods, converge on the same conclusion: current digital twin practice sits below the level its DevOps and reconfiguration literature aspires to. An industry-facing maturity assessment places current practice at

only “medium maturity level, characterised by high-fidelity virtual models, limited two-way communication capabilities and a moderate degree of automation,” and notes that standardization efforts such as ISO/IEC 30186’s maturity-assessment guidance are “still in their early stages” [25]. A systematic survey of 22 primary studies on digital-twin-enabled AI simulation reaches a parallel conclusion from the research-literature side: it finds “the complete lack of digital twin standards, architectural blueprints, and reporting guidelines” among its sampled studies, with only one study (4.5%) referencing RAMI 4.0 at all, and that reference applied inconsistently [17]. Together with the DevOps and CI/CD evidence above, these two independent measurements – one industry-practice-facing, one literature-facing – corroborate the same throughline from a different angle: the Build/Dev side of DT engineering is comparatively concrete and standardized, while the Operate/Monitor side, and the standards that would govern it, remain early-stage across the corpus [17], [25].

Economics, Adoption, and Trust

DTaaS and marketplace business models

Here “DTaaS” is used, as throughout this section, for the generic Digital-Twin-as-a-Service category, not the Talasila/INTO-CPS platform of the same name discussed in “The Platform Landscape.” Architectural framing for consumption-billed digital twin delivery is well developed, but the corpus offers little on realized pricing or unit economics. [63] supplies a Digital-Twin-as-a-Service (DTaaS) Architecture Reference Model for Industry 4.0, establishing how a twin’s data, model, and service layers can be decomposed for cloud delivery, while [34] extends this into a 2026 component-based taxonomy spanning data, model, service, and communication core components alongside infrastructure and security supportive components. Neither source reports subscription mechanics, per-instance cost, or margin data; both are architectural and performance-metric-focused [34], [63]. This gap matters for a platform-building reader: the corpus can justify *how* to structure a DTaaS offering but not *what it should cost* or *how it should be priced*.

Marketplaces are the corpus’s clearest evidence that DTaaS is moving from architecture toward adoption mechanism, though still at an early stage. The EU Change2Twin project built an explicit Digital Twin Marketplace with model-based knowledge representation and discovery features aimed at SME uptake, and frames its target end state as “a connected ecosystem of marketplaces” rather than a single monolithic platform [64]. Change2Twin’s companion tools catalogue reinforces the SME orientation directly: its deliverable is framed around making digital-twin-enabling technology “understandable and relevant” to SMEs specifically, which implies that mainstream tooling and marketplace design otherwise assumes larger-firm resourcing [65].

The market-size-forecast contradiction

The corpus contains three market-size forecasts for digital twins that cannot describe the same market. [26] cites a figure of $35bn by 2025. [27] reports growth from $3.2bn in 2020 to $48.27bn by 2026 at a 58% compound annual growth rate (CAGR). [28] cites Fortune Business Insights figures of $8.6bn in 2022 growing to $137.7bn by 2030 at 42.6% CAGR [26]–[28]. These differ in base year, target year, and implied growth rate in ways that are mutually inconsistent by construction: a market reported at $35bn for 2025 cannot also be built from a 2020-to-2026 trajectory that puts 2025 well below $48bn, nor reconciled

with a 2022 base of $8.6bn growing to $137.7bn by 2030 at a materially different rate. No paper in the corpus that cites one of these figures cross-checks it against another; each is reproduced as a single unexamined data point supporting the paper's own motivation section [26]–[28]. The pattern is more consistent with unaudited, propagating citation of vendor or analyst market-research figures than with a settled estimate of digital twin market size, and a reader relying on the corpus for investment sizing should treat every such figure as provenance-unverified rather than converging evidence.

The SME-versus-large-firm adoption asymmetry

Direct evidence in the corpus locates the central adoption barrier in firm size and risk tolerance rather than in technology readiness. [29] (duplicated in the corpus under the citekey `obaidat_hubcap_2022` for the same HUBCAP paper) frames the problem explicitly: some domains, aerospace among them, tolerate model-based-design return-on-investment (ROI) horizons "that can take decades," while SMEs "require considerable flexibility" and cannot absorb that horizon [29], [66]. This is independently corroborated rather than restated by a second, unrelated source: [30] states that the high upfront costs of implementing digital twin platforms can be a significant barrier, "especially for small and medium-sized enterprises" [30]. The two sources agree on the asymmetry's direction – large firms with long investment horizons can absorb digital twin adoption costs that SMEs cannot – but neither quantifies the cost gap or proposes a threshold at which the barrier resolves, leaving the asymmetry established qualitatively rather than measured. The same cost concern surfaces in platform-selection detail: [30]'s SWOT-style comparison of Azure Digital Twins against Siemens Xcelerator finds Azure weakened by the absence of a local-server hosting option and by a training and learning-curve barrier, while Siemens Xcelerator's chief weakness is lower market awareness specifically within the digital-twin-platform space – neither platform is presented as SME-optimized by design [30].

Standards conformance as an adoption and ROI lever

Where the corpus offers a rare case for conformance to a specific standard as an ROI mechanism rather than a compliance cost, it centers on ISO 23247. [12] organizes ISO 23247's structure into four interconnected domains – Observable Manufacturing, Device Communication, Digital Twin, and User – linked by a Cross-system Entity that supplies common functionality such as data translation, data assurance, and security support, and then ties that structure to two concrete outcomes: a Sandvik cutting-tool-life digital twin that "helped increase tool life for milling operations by 15%," and an airframe-weight and fastener-length optimization case [12]. [45] corroborates ISO 23247's practical orientation from a different angle, elaborating three use-case scenarios implementable on NIST's Smart Manufacturing Systems Test Bed and framing the standard as addressing implementation challenges "especially among small and medium-sized enterprises" by supplying common terminology and guidelines [12], [45]. The two sources agree that ISO 23247 is being positioned deliberately as an SME-accessibility mechanism as well as a technical interoperability spec, but only [12] in this corpus attaches a named, quantified outcome (the 15% tool-life figure) to that conformance, so the standards-to-ROI link rests on a single documented case rather than a body of replicated results.

Trust as an engineered rather than automatic property

The corpus treats trust in digital twin systems as something that must be actively designed, not a byproduct of governance or standards compliance. [67] proposes "trust vectors" as an architectural mechanism for certifying dynamically composed digital-twin-based systems, arguing that static "proven-in-use" certification – adequate for a fixed system – breaks down once twins are composed ad hoc at runtime [67]. This is consistent with a field-level assessment in [68], which finds that despite existing national digital twin policies, including a US framework and Australia's National Digital Twin Roadmap, there remains "a lack of standards" and calls explicitly for building "collective trust and reliability" [68]. Both sources locate the trust gap after governance already exists – [68] at the policy level, [67] at the composition-architecture level – indicating that trust is a property the corpus treats as still unresolved even where national or standards-body oversight is in place, rather than one that governance automatically confers [67], [68].

A methodological caveat applies to much of the platform-comparison evidence underlying this section. [11]'s 19-platform industry comparison was built "from provider websites, white papers, press releases and user documentation" rather than from hands-on testing, which implicitly concedes that public-facing documentation is the primary signal available to a prospective adopter before committing to a platform. The same source's social-network analysis identifies manufacturing, automotive, energy, and construction as the leading adopter industries, notes a gap in adoption among Asian user organizations, and reports that providers are "moving beyond isolated proofs of concept into scaling adoption across industries" [11].

Open Problems and the Evidence Gap

The confessed evidence deficit

Two institutionally independent sources state, in near-identical substance, that the field's publicity has outrun its evidence base. The US National Academies' committee on foundational research gaps for digital twins concludes that "the sentiment expressed across multiple committee information-gathering sessions is that the publicity around digital twins and digital twin solutions currently outweighs the evidence base of success," and separately, in its Conclusion 2-5, states that "it is important to separate the aspirational from the actual to strengthen the credibility of the research in digital twins" [31], [69]. A Nature Computational Science editorial, produced independently of the Academies process, reaches the same conclusion in almost the same words: digital twins "hold immense promise in accelerating scientific discovery, but the publicity currently outweighs the evidence base of success" [32]. That two bodies with no evident coordination converge on the identical claim – publicity outpacing evidence – is itself evidence that the gap is a recognized feature of the field rather than an outside critic's opinion. The Academies committee extends the point beyond publicity to governance, naming open questions on privacy, data ownership, and who bears responsibility for data accuracy, and warning explicitly of "the dangers of scaling digital twins without actionable standards for appropriate use" [31]. The corpus does not resolve these questions; it only records that the field's own institutions have posed them.

Terminological inflation and ad hoc practice

The gap between claim and substance also appears inside individual papers, not only in institutional retrospectives. A case study self-titled "A Case Study for Enhancing Maintenance of Cyber-Physical Systems With Digital Twin" applies Kritzinger et al.'s own integration-ladder rubric [4] to its worked examples and finds that three of its four named failure cases score L1 – digital shadow, one-way automated flow – and the fourth scores L0, an offline digital model; none reaches the bidirectional automation that would justify the paper's own title [5]. This is a first-party admission, not an external critique: the authors' rubric, applied to the authors' examples, does not support the authors' label. That single result sits inside a much larger body of literature that keeps returning to the model/shadow/twin distinction rather than treating it as settled – sources spanning conceptual framing, hierarchical classification, and domain-specific applications all restate or probe the same taxonomy [70]–[75]. The recurrence of the debate across independent sources is consistent with [5]'s case-level finding, but it is not, on its own, further evidence of inflation: a field returning repeatedly to refine a shared taxonomy is equally consistent with taking definitional rigor seriously as with a field whose usage has drifted, and none of these background sources individually claims inflation is occurring. The stronger, corpus-wide version of the claim is not asserted here – only that [5] documents one paper's rubric failing to support its own title, and the definitional debate remains genuinely unsettled around it.

Practice is named as poorly calibrated as well as loosely labeled. A textbook chapter states that most past and current digital twin development efforts "have relied on ad-hoc approaches, where most of the efforts start with building models without properly framing the problem," a pattern the chapter links to poor model calibration from lack of appropriate data and to weak interoperability [76]. The corpus does not carry a second source making the same practice-level claim, so it is reported here as one textbook's assessment rather than a corroborated finding.

Security as an open research problem, not a deployed defense

The corpus treats digital twin security as a forward-looking research agenda rather than a catalog of solved problems. Two independent surveys converge on the same underlying weakness in the DT-to-physical-system link: synchronization "lacks a secure and trusted environment, which reduces the performance and security of the IIoT" [77], [78]. A separate line of work goes further, arguing that digital twins do not merely inherit the security posture of the systems they mirror but actively enlarge the attack surface, "especially in industrial contexts involving real-time" operation [79]. Two dedicated security surveys corroborate that this is unresolved rather than catalogued: one enumerates threats across cloud, fog, and edge infrastructures as a live research area [80], and a further chapter frames digital twin security generally as an open, forward-looking problem rather than a closed one [81]. None of these four sources describes a deployed, validated defense that closes the synchronization gap or reverses the attack-surface expansion; the agreement among them is about what remains unsolved, not about how it has been solved.

Standardization is "pioneer," not settled

Where the literature does point to concrete standards, it describes them as nascent. [44] characterizes "the recent ISO-23247 standard and IEC-63278-1" as "pioneer standards in DT

technology" – language that concedes these are early entrants rather than an established conformance baseline against which a platform's "digital twin" claim could be checked. This framing is consistent with, and helps explain, the terminological looseness documented above: without a settled standard to test against, self-labeling as a digital twin case study [5] or a digital twin platform is left to the authors' own judgment.

The absence of a settled baseline extends to evaluation itself. A platform-evaluation study observes that existing methods for comparing digital twin platforms "are mostly based on subjective opinions, simple decision-making methods, and general-purpose criteria," and proposes a fuzzy, Best-Worst-Method-based framework as a partial remedy while situating its own target platform, makeTwin, within the broader reference-architecture landscape [49]. Read alongside the pioneer status of ISO-23247 and IEC-63278-1 [44], the corpus indicates that the field currently lacks not only agreed criteria for what counts as a digital twin, but also agreed criteria for judging platform quality once a system clears that bar – a second, compounding layer of the same evidence gap.

Contradiction Map

- **Conflicts:**
  - *Market size:* [26] ($35bn by 2025), [27] ($3.2bn in 2020 growing to $48.27bn by 2026 at 58% CAGR), and [28] ($8.6bn in 2022 growing to $137.7bn by 2030 at 42.6% CAGR) cannot describe the same market by construction – differing base years, target years, and implied CAGRs – and none of the papers that cite one of these figures cross-checks it against another. Best read as evidence of unaudited, propagating market-forecast citations rather than a reconciled valuation.
  - *Standards value, asserted vs. measured:* [45] frames ISO/DIS 23247 as addressing implementation challenges "especially among small and medium-sized enterprises," and [12] reports a concrete 15% tool-life increase from an ISO-23247-aligned deployment – against [16]'s triangulated finding that several standardized ISO 23247 functional entities (plug-and-play, peer interface, data assurance) show 0% implementation in practice, and [17]'s independent finding that only 1 of 22 sampled AI-simulation studies referenced any reference architecture at all. The ROI evidence and the non-adoption evidence are not about the same deployments, but they sit in tension: the standard is credited with adoption-enabling and ROI outcomes in specific named cases while two independent literature-wide audits find the broader field mostly ignoring reference architectures altogether.
  - *What counts as a "digital twin" platform/instance:* the model/digital-shadow/digital-twin integration ladder [4] is used across the corpus as the shared measuring stick, yet [5] – a paper self-titled as a digital twin case study – rates its own worked examples as digital-shadow-level (L1) in three of four cases and offline-twin-level (L0) in the fourth, a first-party admission that a paper's own self-description as "digital twin" does not survive its own rubric.
  - *Vendor lock-in vs. the convenience that drives adoption:* [42] names explicit, costed vendor lock-in in Azure's service battery and general AECO commercial-platform practice, and [54]'s comparison table rates Azure/AWS "cloud-only" for edge mobility – yet [6] positions the same Azure Digital Twins and AWS IoT TwinMaker as "the primary references" a research platform builds beyond rather than avoids, and [11] finds providers "moving beyond isolated proofs of concept into

scaling adoption across industries." Lock-in is named as a real cost in the same corpus that shows these platforms as the adoption benchmark.

- **Strongest / weakest evidence:** Strongest, by independent corroboration – [16]'s triangulated standard-vs-practice study and [17]'s independent 22-study corroboration; [18]'s tested four-implementation AAS comparison. Precise but unreplicated – [35]'s quantified 55%/25% effort-reduction figure is the corpus's only such number and should not be read as equally reliable to the triangulated findings above; it is a strong number from a single experiment, not a corroborated one. Weakest – the three mutually inconsistent market-size forecasts [26]–[28]; [12]'s named ROI figures, which are vendor-case anecdotes rather than a controlled study.
- **Resolving question:** What would the same digital twin, built on two or more platforms with standardized, audited metrics (twinning rate, twin fidelity, age-of-twin [34]) and accounted integration/glue-code effort, actually reveal about the size of the standards-vs-practice gap, and whether lock-in-avoiding platforms like KTWIN close it at a measurable cost or benefit relative to the vendor platforms builders keep citing as references?
- **Universal agreement:** every source that evaluates it agrees documentation quality gates platform selection before features matter [8], [11]; the operational/lifecycle phase (drift, safe runtime reconfiguration, schema evolution without teardown) is the least mature part of the stack across every source that touches it [22], [23], [25]; and the field's own institutions independently confess that publicity outweighs the evidence base of success [31], [32]. A narrower claim, not a corpus-wide one: within the specific systems this report examines, no platform ships a complete twin out of the box – OpenTwins by direct evidence, the open-source frameworks [8] evaluates by its own completeness criterion – but commercial and academic platforms beyond those specific cases are not tested against this criterion here [8], [9].
- **Blind spot:** no source in this corpus quantifies integration/glue-code cost in person-hours; no longitudinal (multi-year) case study of a digital twin platform's operational phase was found; no source describes a safety/rollback mechanism for runtime reconfiguration of the physical twin; no realized DTaaS pricing/unit-economics data was found [34], [63]; and [82]'s process-industry enablers/barriers findings could not be extracted from the retrieval snippets across two interview rounds.

Peer-Review & Reliability Scorecard

Table 1 gives the reconciled result of a four-role independent panel (no reviewer saw another's critique).

Table 1: Reconciled panel confidence in each headline finding, with the reason for each score.

Finding	Confidence (1-10)	Why
Reference architectures diverge from implemented practice	9	Triangulated (140-paper SLR + 33-respondent survey + interviews), independently corroborated by a differently-scoped 22-study survey; unchallenged by any reviewer

Finding	Confidence (1-10)	Why
Documentation maturity gates platform selection	8	Systematic 14-candidate open-source survey, corroborated independently by a 19-platform commercial comparison
The operational/lifecycle phase is comparatively underdeveloped	7	Rescoped after review: strongest where independently measured (a 10-organization interview study, an industry maturity assessment), weaker where it rested only on DevOps-proposal papers declining to cover Operate/Monitor within their own stated scope
No platform examined in this corpus ships a complete twin (glue-code finding)	6	Rescoped after review: rests on one academic platform (OpenTwins) rather than a corpus-wide count; commercial and academic platforms beyond that case are not tested against this criterion here
Terminological inflation is documented in one first-party instance	5	Rescoped after review: one case study's own rubric failing its own title, not a corpus-wide measurement; the field's active re-mapping of its own vocabulary is equally consistent with definitional rigor
Platform economics unproven / market forecasts inconsistent	6	Three forecasts differ by construction (base year, target year, CAGR); standards-to-ROI link rests on one named case, not a replicated body of evidence

- **Panel verdicts:** domain-accuracy: needs revision (revised) – methodology-rigor: needs revision (revised) – clarity-completeness: needs revision (revised) – devils-advocate: needs revision (revised). Every high-severity concern, and every medium-or-higher concern raised independently by two or more reviewers, was addressed before presenting; none was silently dropped.
- **Concerns addressed** (met the concession threshold): a quotation attributed to the National Academies committee report was found to splice two separate sentences into one continuous "verbatim" quote and was corrected to quote each sentence accurately (domain-accuracy, high); the Synthesis Briefing's Finding 2 bundled a single-platform case (OpenTwins) with a systematically-surveyed finding (documentation-as-gate) under one reliability score and was split into two separately scored claims (methodology-rigor, high); the Contradiction Map's "Universal agreement" bullet extrapolated a no-complete-twin finding from OpenTwins and an open-source-framework survey to the entire platform landscape and was rescoped to name what it actually covers (methodology-rigor, high); the "Hidden connection" paragraph forced three structurally different gaps (integration effort, standards-adoption measurement, and runtime-capability maturity) into one unified absence, motivating the report's own

recommended benchmark in a way that read as manufactured rather than discovered, and was narrowed to the two findings it actually connects (devils-advocate, high). Raised independently by two or more reviewers and also addressed: the reliability-scoring apparatus understated the evidentiary gap between triangulated multi-source findings and single-case or single-experiment ones (methodology-rigor and devils-advocate both, on Findings 2, 3, and the "Strongest evidence" list) – Finding 3's score was lowered and its two evidence tiers named explicitly, the "Strongest evidence" list now separates [35]'s unreplicated figure from the triangulated studies, and Finding 4's headline and score were both revised to match its own hedge. Also fixed, below the mandatory threshold but cheap and correct against the source: [38]'s finding was overstated across three locations (the source finds substantial shared structure with a partial, quantified portability gap, not blanket incompatibility) and was corrected to match the source; a metric ("synchronization latency") was misattributed to [34]'s own contribution when the source actually cites it as another survey's metric in its related-work discussion, and was corrected in all three locations; the [17] "applied inconsistently" paraphrase was replaced with the source's actual, more specific finding (failure to connect RAMI 4.0 to the Asset Administration Shell); a missing does-not-cover scope statement was added to the Summary; and the DTaaS proper-noun/generic-category collision was given a cross-reference at its second point of use in the Economics section.

- **Concerns logged, not addressed** (below threshold): acronym first-use expansion (DTDL, AAS, and others) is deferred to the Phase 7(f) prose-standards check, which is the tool built for it; the "digital twin platform" definition's wording differs slightly between the Summary and the "What a Digital Twin Platform Is" section without contradicting itself (clarity-completeness, low); the `## References` heading was left without an explicit placeholder comment, unlike other pending sections in this draft (clarity-completeness, low) – both cosmetic and left for a future copy-edit pass rather than acted on here.
- **Weakest link:** the market-size-forecast contradiction and the standards-to-ROI claim (confidence 6): the former is unresolvable from this corpus by design (no source audits the forecasts against each other); the latter rests on one named vendor case ([12]'s Sandvik figure). What would verify it: an audited, methodology-disclosed market-sizing study, and a second, independent ROI case tied to ISO 23247 conformance.
- **Bias check:** [8] is the most-cited single source (11 citations across four sections), load-bearing for the platform taxonomy, the open-source landscape, and half of the no-complete-twin finding; its own dataset is drawn from searches run through July 2022, over four years old relative to this report's date, and its documentation-as-gate finding is only partially, not fully, corroborated by [11] (a related but distinct claim – documentation as the only available signal, not as an exclusion criterion). The Skeptic perspective's lens guarantees an evidence-deficit theme by construction, a bias the "Hidden connection" paragraph already discloses.
- **Missing perspective:** a security-engineering-specific perspective (distinct from the Skeptic's broader evidence-deficit framing) might have pushed further on the corpus's security findings, which this report treats as a single open-problems subsection rather than a dedicated angle.
- **Overall grade:** B+ – the corpus's strongest finding (standards-vs- practice divergence) is well-triangulated and the report's honesty discipline (single-source flags, hedged claims) is above average for this genre; what needed fixing was internal calibration

between how confidently different findings are stated and how much evidence actually backs each one, now corrected above.

References

[1] "Glossary of Digital Twins by Digital Twin Consortium," *Digital Twin Consortium*. Accessed: May 27, 2024. [Online]. Available: https://www.digitaltwinconsortium.org/glossary/glossary/

[2] M. Callisto De Donato, F. Corradini, F. Fornari, B. Re, and M. Romagnoli, *Enabling 3D Simulation in ThingsBoard: A First Step Towards A Digital Twin Platform*. 2024. doi: 10.1007/978-3-031-54712-6_20.

[3] E. VanDerHorn and S. Mahadevan, "Digital Twin: Generalization, characterization and implementation," *Decision Support Systems*, vol. 145, p. 113524, Jun. 2021, doi: 10.1016/j.dss.2021.113524.

[4] W. Kritzinger, M. Karner, G. Traar, J. Henjes, and W. Sihn, "Digital Twin in manufacturing: A categorical literature review and classification," *Ifac-PapersOnline*, vol. 51, no. 11, pp. 1016–1022, 2018, Accessed: Dec. 11, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405896318316021

[5] J. Liu, J. Vatn, and S. Yin, "A Case Study for Enhancing Maintenance of Cyber-Physical Systems With Digital Twin," *IEEE Transactions on Industrial Cyber-Physical Systems*, vol. 2, pp. 597–605, 2024, doi: 10.1109/TICPS.2024.3485038.

[6] A. G. Wermann and J. A. Wickboldt, "KTWIN: A Serverless Kubernetes-based Digital Twin Platform." arXiv, Aug. 2024. doi: 10.48550/arXiv.2408.01635.

[7] J. Robles, C. Martín, and M. Díaz, "OpenTwins: An open-source framework for the development of next-gen compositional digital twins," *Computers in Industry*, vol. 152, p. 104007, Aug. 2023, doi: 10.1016/j.compind.2023.104007.

[8] S. Gil, P. H. Mikkelsen, C. Gomes, and P. G. Larsen, "Survey on open-source digital twin frameworks–A case study approach," *Software: Practice and Experience*, vol. 54, no. 6, pp. 929–960, Jun. 2024, doi: 10.1002/spe.3305.

[9] S. Infante, J. Robles, C. Martín, B. Rubio, and M. Díaz, "Distributed digital twins on the open-source OpenTwins framework," *Advanced Engineering Informatics*, vol. 64, p. 102970, Mar. 2025, doi: 10.1016/j.aei.2024.102970.

[10] S. Infante *et al.*, "Integrating FMI and ML/AI models on the open-source digital twin framework OpenTwins," *Software Practice and Experience*, Mar. 2024, doi: 10.1002/spe.3322.

[11] A. Khoshkenar and H. Nassereddine, *Exploring Digital Twin platforms across industries*. 2024. doi: 10.22260/ISARC2024/0119.

[12] G. Shao, S. Frechette, and V. Srinivasan, "An Analysis of the New ISO 23247 Series of Standards on Digital Twin Framework for Manufacturing," American Society of Mechanical Engineers Digital Collection, Sep. 2023. doi: 10.1115/MSEC2023-101127.

[13] L. Heaton, "Platform Stack Architectural Framework: An Introductory Guide."

[14] F. Tao *et al.*, "makeTwin: A reference architecture for digital twin software platform," *Chinese Journal of Aeronautics*, vol. 37, no. 1, pp. 1–18, Jan. 2024, doi: 10.1016/j.cja.2023.05.002.

[15] M. Heithoff, N. Jansen, J. Michael, F. Rademacher, and B. Rumpe, "Model-Based Engineering of Multi-Purpose Digital Twins in Manufacturing," in *Digital Twin: Fundamentals and Applications*, S. Sabri, K. Alexandridis, and N. Lee, Eds., Cham: Springer Nature Switzerland, 2024, pp. 89–126. doi: 10.1007/978-3-031-67778-6_5.

[16] E. Ferko, A. Bucaioni, P. Pelliccione, and M. Behnam, "Standardisation in Digital Twin Architectures in Manufacturing," in *2023 IEEE 20th International Conference on Software Architecture (ICSA)*, Mar. 2023, pp. 70–81. doi: 10.1109/ICSA56044.2023.00015.

[17] X. Liu and I. David, "AI simulation by digital twins: Systematic survey, reference framework, and mapping to a standardized architecture," *Software and Systems Modeling*, Aug. 2025, doi: 10.1007/s10270-025-01306-0.

[18] M. Jacoby *et al.*, "Open-Source Implementations of the Reactive Asset Administration Shell: A Survey," *Sensors*, vol. 23, p. 5229, May 2023, doi: 10.3390/s23115229.

[19] J. Zhang, C. Ellwein, M. Heithoff, J. Michael, and A. Wortmann, "Digital twin and the asset administration shell," 2024, Accessed: Apr. 09, 2025. [Online]. Available: https://awortmann.github.io/downloads/paper/Digital_twin_and_the_asset_administration_shell.pdf

[20] J. Hugues, A. Hristosov, J. J. Hudak, and J. Yankel, "TwinOps - DevOps meets model-based engineering and digital twins for the engineering of CPS," in *Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings*, in MODELS '20. New York, NY, USA: Association for Computing Machinery, Oct. 2020, pp. 1–5. doi: 10.1145/3417990.3421446.

[21] F. Bordeleau, A. Motamedi, and É. Poirier, "A DevOps Approach for the Systematic Development and Evolution of Built Assets Digital Twins," Accessed: Nov. 05, 2024. [Online]. Available: https://itc.scix.net/pdfs/w78-2024-paper_114.pdf

[22] D. Lehner, A. Garmendia, and M. Wimmer, "Towards Flexible Evolution of Digital Twins with Fluent APIs," in *2021 26th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, Sep. 2021, pp. 1–4. doi: 10.1109/ETFA45728.2021.9613222.

[23] J. Fitzgerald, C. Gomes, and P. G. Larsen, Eds., *The Engineering of Digital Twins*. Cham: Springer International Publishing, 2024. doi: 10.1007/978-3-031-66719-0.

[24] F. Zampetti, D. Tamburri, S. Panichella, A. Panichella, G. Canfora, and M. Di Penta, "Continuous Integration and Delivery Practices for Cyber-Physical Systems: An Interview-Based Study," *ACM Trans. Softw. Eng. Methodol.*, vol. 32, no. 3, pp. 73:1–73:44, Apr. 2023, doi: 10.1145/3571854.

[25] N. Anwer, R. Stark, F. Tao, and J. Erkoyuncu, "Developing and leveraging digital twins in engineering design," *CIRP Annals*, vol. 2025, Jun. 2025, doi: 10.1016/j.cirp.2025.05.002.

[26] M. Picone, M. Mamei, and F. Zambonelli, "A Flexible and Modular Architecture for Edge Digital Twin: Implementation and Evaluation," *ACM Transactions on Internet of Things*, vol. 4, no. 1, pp. 8:1–8:32, Feb. 2023, doi: 10.1145/3573206.

[27] S. Mihai *et al.*, "Digital Twins: A Survey on Enabling Technologies, Challenges, Trends and Future Prospects," *IEEE Communications Surveys & Tutorials*, vol. 24, no. 4, pp. 2255–2291, 2022, doi: 10.1109/COMST.2022.3208773.

[28] G. Lugaresi and H. Vangheluwe, *From Digital Twins to Twinning Systems*. 2025.

[29] P. G. Larsen *et al.*, "HUBCAP: A Novel Collaborative Approach to Model-Based Design of Cyber-Physical Systems," in *Simulation and Modeling Methodologies, Technologies and Applications*, M. S. Obaidat, T. Oren, and F. D. Rango, Eds., Cham: Springer International Publishing, 2022, pp. 90–110. doi: 10.1007/978-3-030-84811-8_5.

[30] D. Parle, G. Sharma, N. Anand, N. Padgaonkar, D. Stoddart, and D. Malley, "A Comparative Analysis for Harnessing Digital Twin Platforms for Net-Zero Manufacturing," *IEEE Access*, vol. PP, Aug. 2024, doi: 10.1109/ACCESS.2024.3447475.

[31] *Foundational Research Gaps and Future Directions for Digital Twins*. Washington, D.C.: National Academies Press, 2024. doi: 10.17226/26894.

[32] "The rise of digital twins," *Nature*. Mar. 2024. Accessed: May 27, 2024. [Online]. Available: https://www.nature.com/collections/feicjiideh

[33] J. Pfeiffer *et al.*, "Towards a Unifying Reference Model for Digital Twins of Cyber-Physical Systems." arXiv, Jul. 2025. doi: 10.48550/arXiv.2507.04871.

[34] K. Duran *et al.*, "Toward Digital Twin-as-a-Service (DTaaS) Platforms: A Survey on Architecture, Design Requirements, and Performance Metrics," *IEEE Communications Surveys & Tutorials*, vol. 28, pp. 1845–1878, 2026, doi: 10.1109/COMST.2025.3635582.

[35] D. Lehner, S. Sint, M. Vierhauser, W. Narzt, and M. Wimmer, "AML4DT: A Model-Driven Framework for Developing and Maintaining Digital Twins with AutomationML," in *2021 26th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, Sep. 2021, pp. 1–8. doi: 10.1109/ETFA45728.2021.9613376.

[36] R. Richstein and K.-U. Schröder, "Characterizing the Digital Twin in Structural Mechanics," *Designs*, vol. 8, no. 1, p. 8, Feb. 2024, doi: 10.3390/designs8010008.

[37] L. Cleophas *et al.*, "A community-sourced view on engineering digital twins: A report from the EDT.Community," in *Proceedings of the 25th International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings*, in MODELS '22. New York, NY, USA: Association for Computing Machinery, Nov. 2022, pp. 481–485. doi: 10.1145/3550356.3561549.

[38] J. Pfeiffer, D. Lehner, A. Wortmann, and M. Wimmer, "Modeling capabilities of digital twin platforms-old wine in new bottles?" *J. Object Technol.*, vol. 21, no. 3, pp. 3–1, 2022, Accessed: Sep. 04, 2024. [Online]. Available: https://raw.githubusercontent.com/awortmann/awortmann.github.io/master/downloads/paper/Modeling_Capabilities_of_Digital_Twin_Platforms_-_Old_Wine_in_New_Bottles.pdf

[39] A. Barbone, S. Burattini, M. Martinelli, M. Picone, A. Ricci, and A. Virdis, "Digital Twin Continuum: A Key Enabler for Pervasive Cyber-Physical Environments," in *2024 33rd International Conference on Computer Communications and Networks (ICCCN)*, Jul. 2024, pp. 1–9. doi: 10.1109/ICCCN61486.2024.10637565.

[40] D. Lehner, J. Pfeiffer, S. Klikovits, A. Wortmann, and M. Wimmer, "A Method for Template-based Architecture Modeling and its Application to Digital Twins," Accessed: Sep. 04, 2024. [Online]. Available: https://www.jot.fm/issues/issue_2024_03/article8.pdf

[41] A. Rasheed, O. San, and T. Kvamsdal, "Digital Twin: Values, Challenges and Enablers From a Modeling Perspective," *IEEE Access*, vol. 8, pp. 21980–22012, 2020, doi: 10.1109/ACCESS.2020.2970143.

[42] P. Zech, C. Nardin, S. Ristov, M. Flora, and R. Breu, "Digital-Twins-as-a-Service in Construction Engineering," in *2024 IEEE 20th International Conference on Automation Science and Engineering (CASE)*, Aug. 2024, pp. 3004–3010. doi: 10.1109/CASE59546.2024.10711409.

[43] P. Talasila, P. H. Mikkelsen, S. Gil, and P. G. Larsen, "Realising Digital Twins," in *The Engineering of Digital Twins*, J. Fitzgerald, C. Gomes, and P. G. Larsen, Eds., Cham: Springer International Publishing, 2024, pp. 225–256. doi: 10.1007/978-3-031-66719-0_11.

[44] S. Gil, B. Oakes, C. Gomes, M. Frasheri, and P. Larsen, "Toward a systematic reporting framework for Digital Twins: A cooperative robotics case study," *SIMULATION*, Aug. 2024, doi: 10.1177/00375497241261406.

[45] G. Shao, "Use Case Scenarios for Digital Twin Implementation Based on ISO 23247," *NIST*, May 2021, Accessed: Mar. 06, 2024. [Online]. Available: https://www.nist.gov/publications/use-case-scenarios-digital-twin-implementation-based-iso-23247

[46] kayla, "Digital Twin Platform Stack Architectural Framework by Digital Twin Consortium." Nov. 2023. Accessed: May 27, 2024. [Online]. Available: https://www.digitaltwinconsortium.org/2023/11/understanding-dtcs-digital-twin-platform-stack-architectural-framework/

[47] D. McKee and D. Dokter, "DISCS: An Approach for Accelerating the Development of Digital Twins for Smart Cities," in *Digital Twin: Fundamentals and Applications*, S. Sabri, K. Alexandridis, and N. Lee, Eds., Cham: Springer Nature Switzerland, 2024, pp. 31–58. doi: 10.1007/978-3-031-67778-6_3.

[48] N. Li *et al.*, "Six-dimensional digital twin modeling and software platform design for complex industrial systems," *Journal of Intelligent Manufacturing*, Jan. 2025, doi: 10.1007/s10845-025-02567-8.

[49] Z. Tang, D. Zhuang, and J. Zhang, "Evaluation framework for domain-specific digital twin platforms," *Scientific Reports*, vol. 15, no. 1, p. 10544, Mar. 2025, doi: 10.1038/s41598-024-82154-8.

[50] A. Hakiri, A. Gokhale, S. B. Yahia, and N. Mellouli, "A comprehensive survey on digital twin for future networks and emerging Internet of Things industry," *Computer Networks*, vol. 244, p. 110350, May 2024, doi: 10.1016/j.comnet.2024.110350.

[51] "Asset Administration Shell Part - 1."

[52] C. Schmidt, F. Volz, L. Stojanovic, and G. Sutschet, "Increasing Interoperability between Digital Twin Standards and Specifications: Transformation of DTDL to AAS," *Sensors*, vol. 23, p. 7742, Sep. 2023, doi: 10.3390/s23187742.

[53] S. Cavalieri and S. Gambadoro, "Proposal of Mapping Digital Twins Definition Language to Open Platform Communications Unified Architecture," *Sensors*, vol. 23, p. 2349, Feb. 2023, doi: 10.3390/s23042349.

[54] P. Bellavista, N. Bicocchi, M. Fogli, C. Giannelli, M. Mamei, and M. Picone, "An Entanglement-Aware Middleware for Digital Twins," *ACM Trans. Internet Things*, Oct. 2024, doi: 10.1145/3699520.

[55] J. Hugues, J. Yankel, J. Hudak, and A. Hristozov, "Twinops: Digital twins meets devops," *CARNEGIE-MELLON UNIV PITTSBURGH PA, Tech. Rep.*, 2022, Accessed: Jan. 08, 2025. [Online]. Available: https://apps.dtic.mil/sti/trecms/pdf/AD1168471.pdf

[56] S. Aissat, J. Beaulieu, F. Bordeleau, J. Gascon-Samson, E. A. Poirier, and A. Motamedi, "JuNo-OPS: A DevOps Framework for the Engineering of Digital Twins for Built Assets," in *Proceedings of the ACM/IEEE 27th International Conference on Model Driven Engineering Languages and Systems*, in MODELS Companion '24. New York, NY, USA: Association for Computing Machinery, Oct. 2024, pp. 496–506. doi: 10.1145/3652620.3688266.

[57] S. Aissat, J. Beaulieu, É. Poirier, A. Motamedi, J. Gascon-Samson, and F. Bordeleau, "A devops framework for the systematic engineering and evolution of digital twins for built assets," *Software and Systems Modeling*, Oct. 2025, doi: 10.1007/s10270-025-01330-0.

[58] G. Beaumont, A. Beugnard, S. Martínez, C. Urtado, and S. Vauttier, "Towards Automating the Life Cycle Management of Digital Twins," in *ER2025-44th International Conference on Conceptual Modeling*, 2025. Accessed: Oct. 22, 2025. [Online]. Available: https://hal.science/hal-05225783/

[59] I. Aldalur, A. Arrieta, A. Agirre, G. Sagardui, and M. Arratibel, "A microservice-based framework for multi-level testing of cyber-physical systems," *Software Quality Journal*, vol. 32, no. 1, pp. 193–223, Mar. 2024, doi: 10.1007/s11219-023-09639-z.

[60] N. Zhang, R. Bahsoon, N. Tziritas, and G. Theodoropoulos, "Knowledge Equivalence in Digital Twins of Intelligent Systems," *ACM Transactions on Modeling and Computer Simulation*, vol. 34, no. 1, pp. 1–37, Jan. 2024, doi: 10.1145/3635306.

[61] M. Frasheri, T. Böttjer, P. G. Larsen, L. Esterle, and C. Gomes, "Advanced Digital Twin Services," in *The Engineering of Digital Twins*, J. Fitzgerald, C. Gomes, and P. G. Larsen, Eds., Cham: Springer International Publishing, 2024, pp. 209–222. doi: 10.1007/978-3-031-66719-0_10.

[62] E. Kamburjan, N. Bencomo, S. L. Tapia Tarifa, and E. B. Johnsen, "Declarative Lifecycle Management in Digital Twins," in *Proceedings of the ACM/IEEE 27th International Conference on Model Driven Engineering Languages and Systems*, Linz Austria: ACM, Sep. 2024, pp. 353–363. doi: 10.1145/3652620.3688248.

[63] S. Aheleroff, X. Xu, R. Y. Zhong, and Y. Lu, "Digital Twin as a Service (DTaaS) in Industry 4.0: An Architecture Reference Model," *Advanced Engineering Informatics*, vol. 47, p. 101225, Jan. 2021, doi: 10.1016/j.aei.2020.101225.

[64] "Change2Twin marketplace design." Change2Twin project. Available: https://www.change2twin.eu/about/deliverables/

[65] "Tools and libraries catalogue." Change2Twin project.

[66] P. G. Larsen *et al.*, "HUBCAP: A Novel Collaborative Approach to Model-Based Design of Cyber-Physical Systems," in *Simulation and Modeling Methodologies, Technologies and Applications*, vol. 306, M. S. Obaidat, T. Oren, and F. D. Rango, Eds., Cham: Springer International Publishing, 2022, pp. 90–110. doi: 10.1007/978-3-030-84811-8_5.

[67] "Assuring Trustworthiness in Dynamic Systems Using Digital Twins and Trust Vectors." Digital Twin Consortium, May 2024. Available: https://www.digitaltwinconsortium.org/assuring-trustworthiness-in-dynamic-systems-using-digital-twins-and-trust-vectors-form/

[68] S. Sabri, K. Alexandridis, and N. Lee, Eds., *Digital Twin: Fundamentals and Applications*. Cham: Springer Nature Switzerland, 2024. doi: 10.1007/978-3-031-67778-6.

[69] "Foundational Research Gaps and Future Directions for Digital Twins \textbar National Academies." Accessed: Apr. 23, 2024. [Online]. Available: https://www.nationalacademies.org/our-work/foundational-research-gaps-and-future-directions-for-digital-twins

[70] S. Sabri, K. Alexandridis, and N. Lee, "Introduction to Digital Twins," in *Digital Twin: Fundamentals and Applications*, S. Sabri, K. Alexandridis, and N. Lee, Eds., Cham: Springer Nature Switzerland, 2024, pp. 1–12. doi: 10.1007/978-3-031-67778-6_1.

[71] C. Ellwein *et al.*, "Rethinking Asset Administration Shell Communication Types: A Systematic Mapping Study and Portfolio-Based Classification," *Production Engineering*, vol. 20, Dec. 2025, doi: 10.1007/s11740-025-01378-3.

[72] T. L. Leirmo, "Digital Twins for Industry 5.0: Unlocking the Human Potential," *Procedia CIRP*, vol. 130, pp. 761–766, Jan. 2024, doi: 10.1016/j.procir.2024.10.161.

[73] M. Heithoff, M. Trinh, J. Michael, B. Rumpe, and C. Brecher, "A Digital Shadow for Accurate Robot Motion Control: Integrating Data with Friction Models," Grand Rapids, USA, Jul. 2025. doi: 10.5283/epub.77667.

[74] T. Bergs, S. Gierlings, T. Auerbach, A. Klink, D. Schraknepper, and T. Augspurger, "The concept of digital twin and digital shadow in manufacturing," *Procedia CIRP*, vol. 101, pp. 81–84, 2021, Accessed: Jan. 21, 2025. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2212827121006612

[75] M. Martinelli, J. Zhang, A.-K. Splettstoßer, M. Picone, M. Lippi, and A. Wortmann, "Hierarchical Digital Twin Ecosystem for Industrial Manufacturing Scenarios," in *2024 50th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, Aug. 2024, pp. 56–63. doi: 10.1109/SEAA64295.2024.00018.

[76] O. J. Pinon Fischer, S. Sabri, and Y. Chen, "Fundamentals of Digital Twins, Modeling Approaches, and Governance," in *Digital Twin: Fundamentals and Applications*, S. Sabri, K. Alexandridis, and N. Lee, Eds., Cham: Springer Nature Switzerland, 2024, pp. 13–29. doi: 10.1007/978-3-031-67778-6_2.

[77] H. Xu, J. Wu, Q. Pan, X. Guan, and M. Guizani, "A Survey on Digital Twin for Industrial Internet of Things: Applications, Technologies and Tools," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 4, pp. 2569–2598, 2023, doi: 10.1109/COMST.2023.3297395.

[78] "A Survey on Digital Twin for Industrial Internet of Things: Applications, Technologies and Tools \textbar IEEE Journals & Magazine \textbar IEEE Xplore." Accessed: Sep. 21, 2024. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10188847

[79] M. Picone, E. Bouquillon, M. Pietri, and M. Mamei, "Assessing the Impact of Cybersecurity Attacks on Digital Twin Metrics: An Experimental Study," in *2026 IEEE 23rd Consumer Communications & Networking Conference (CCNC)*, IEEE, 2026, pp. 1–4. Accessed: Mar. 27, 2026. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/11366535/

[80] C. Alcaraz and J. Lopez, "Digital Twin: A Comprehensive Survey of Security Threats," *IEEE Communications Surveys & Tutorials*, vol. 24, no. 3, pp. 1475–1503, 2022, doi: 10.1109/COMST.2022.3171465.

[81] T. Kulik, Z. Kazemi, and P. G. Larsen, "Security and Privacy-related Issues in a Digital Twin Context," in *The Engineering of Digital Twins*, J. Fitzgerald, C. Gomes, and P. G. Larsen, Eds., Cham: Springer International Publishing, 2024, pp. 313–344. doi: 10.1007/978-3-031-66719-0_13.

[82] M. Perno, L. Hvam, and A. Haug, “Implementation of digital twins in the process industry: A systematic literature review of enablers and barriers,” *Computers in Industry*, vol. 134, p. 103558, Jan. 2022, doi: 10.1016/j.compind.2021.103558.