2026 NDIA MICHIGAN CHAPTER
GROUND VEHICLE SYSTEMS ENGINEERING
AND TECHNOLOGY SYMPOSIUM
DIGITAL ENGINEERING/SYSTEMS ENGINEERING (DE/SE) TECHNICAL SESSION
AUGUST 11-13, 2026 - NOVI, MICHIGAN

AN AI-READY DECISION DIGITAL THREAD SCHEMA FOR DIGITAL ENGINEERING

Ratna Babu Chinnam, PhD[1], Alper Murat, PhD[1], Satyendra Rana, PhD[1], Stephen H. Rapp, PhD[2], Joseph G. O'Bruba, PhD[2], Michael McGregor Jr.[3], James E. Bechtel[4], Laura W. Costa[5]

[1]Agile Systems LLC & DAOSoft Corp, Rochester, MI
[2] USA T2COM, DEVCOM –GVSC, Warren, MI
[3] PAE Fires, Warren, MI,
[4] PAE Maneuver Ground, PEO GCS, Warren, MI
[5]USA Maneuver FCD, Warren, MI
{amurat,rchinnam,srana}@daosoft.ai, {stephen.h.rapp, joseph.g.obruba, michael.mcgregor5, james.e.bechtel2, laura.w.costa}.civ@army.mil

ABSTRACT

Digital engineering (DE) and model-based systems engineering (MBSE) improve traceability for requirements, architecture, and verification, but concept decisions—the governance events that turn evolving evidence into binding commitments—are poorly captured. Rationale, assumptions, alternatives, model baselines, and approval conditions are scattered across slides and minutes, limiting auditability, reproducibility, and automation. We propose a Decision Digital Thread (DDT): a typed graph schema that makes decisions governable by linking framing and scope, structured (including set-based) alternatives, uncertainty and risk, immutable evaluation-run provenance with reviewed evidence, and commitment events with machine-actionable conditions, authorized actions, and outcome feedback. DDT serves as the decision system of record and a contract between platform modules and enterprise policy while referencing MBSE/PLM/simulation artifacts via stable identifiers and configuration context. Policy-driven readiness gates block lifecycle transitions when evaluator coverage, evidence review, or bias checks are incomplete. An electric pickup range-extension case demonstrates auditable gates, evidence lineage, and safe AI-agent authority boundaries.

Citation: R.B. Chinnam, A. Murat, S. Rana, S. Rapp, J. O'Bruba, M. McGregor, J. Bechtel, and L. Costa, "An AI-Ready Decision Digital Thread Schema for Digital Engineering," In *Proceedings of the Ground Vehicle Systems Engineering and Technology Symposium* (GVSETS), NDIA, Novi, MI, Aug. 11-13, 2026.

1. INTRODUCTION

Vehicle concept development increasingly operates under shrinking product cycles, volatile supply chains, rapid electrification advances, and heightened regulatory scrutiny. Defense and automotive organizations have therefore invested heavily in digital engineering (DE) and model-based systems engineering (MBSE) to connect authoritative lifecycle artifacts and improve cross-stakeholder communication [1]–[4]. Yet many "digital threads" remain incomplete at a critical control point: they do not explicitly represent the decisions that transform evolving evidence into binding commitments that allocate budget, fix interfaces, and constrain downstream work. Decision rationale, assumptions, alternative sets, model baselines, and approval conditions are typically captured in narrative briefings and meeting notes, leaving no machine-interpretable record of what was considered, what was accepted, and under which uncertainties and conditions—so traceability reaches "what we built" and "what we verified," but not reliably "why we built it" or "why alternatives were rejected" [2], [4]. Programs then struggle to answer audit-grade questions months later (e.g., which dataset justified a recommendation; whether required bias checks were completed before a gate review).

We distinguish a decision artifact from an analysis model or a bundle of supporting documents. A decision artifact is the minimal machine-interpretable record that makes a commitment governable: it encodes the decision question and scope, alternatives, objectives and constraints, acknowledged uncertainties, reviewed evidence, approvals, and binding conditions. In this sense, the decision artifact functions as a control-plane for the digital thread: the Decision Digital Thread (DDT) is authoritative for decision governance state and provenance, while MBSE/PLM remain authoritative for product definition and verification artifacts.

Agentic AI increases the urgency of a governed decision representation. The term "agentic" refers to AI systems endowed with autonomy to perceive their environment, make decisions toward specified objectives, and execute tasks without human intervention beyond high-level guidance. Agents can discover evidence, propose alternatives, execute evaluations, and draft rationale, but they can also introduce accountability failures if evidence is misattributed, assumptions drift without detection, or commitments are implied without authority. A DE-ready decision system therefore requires explicit authority boundaries, tamper-evident provenance, and policy-enforced lifecycle transitions so automation accelerates work without weakening accountability.

Recent work underscores the importance of decision support and decision management in DE while revealing remaining gaps. Gautier et al. [5] propose an ontology-based digital thread for analysis and decision support, emphasizing interoperability across toolchains. Singh and Willcox [6] formulate decision-making under uncertainty for a digital-thread-enabled design process in which thread state evolves with decisions and evidence. Parnell et al. [7] introduce a Decision Analysis Data Model (DADM) for decision management in an MBSE environment, standardizing decision data elements and process. Gery [8] synthesizes digital thread benefits and challenges and notes persistent industry immaturity and unmet needs in change management and tradeoff decision support. Collectively, these contributions strengthen the ecosystem for digital-thread-enabled decision support; however, they do not fully resolve the enterprise "decision artifact problem": a single, auditable, reusable, machine-readable decision record that binds (i) framing and scope, (ii) objective hierarchies and measures, (iii) alternatives and constraints, (iv) explicit uncertainty and risk, (v) reproducible evaluation runs and evidence provenance, (vi) governance checks to reduce cognitive and organizational biases, and (vii) commitments with approval conditions, downstream actions, and observed outcomes. This governance-centered structure is increasingly essential as programs rely on AI-agentic decision intelligence. Murat et al. [9] advocate decision intelligence and analytics platforms to strengthen decision-making in product development, and Murat et al. [10] propose an AI-agentic framework leveraging large language models to enhance product planning and concept development. An agent can call a simulator or summarize a report, but it cannot reliably manage a high-stakes decision program unless the decision itself is represented as a coherent, governed object.

This paper provides the decision-data infrastructure required to realize AI-agentic decision intelligence in a DE ecosystem. We contribute: (1) enterprise-grade technical requirements for decision representation driven by operational governance needs; (2) a decision-centric digital thread schema implemented as a typed graph (metamodel + registries) that structurally separates Decision, Evaluation Run, Evidence, Commitment, Action, and Outcome, enabling post-hoc auditing of decision quality independent of outcomes; (3) operational governance mechanics—immutable run provenance and policy-based readiness gating—that convert governance from narrative guidance into enforceable constraints; and (4) an electric pickup range-extension concept decision demonstrating auditable gate readiness, reproducible evidence lineage, and conditional commitments that remain binding constraints downstream.

The remainder of the paper is organized as follows: Section 2 reviews related work, Section 3 derives requirements, Section 4 defines the schema, Section 5 describes governance mechanics, Section 6 presents the case study, and Section 7 concludes.

2. BACKGROUND AND RELATED WORK

The U.S. Department of Defense (DoD) Digital Engineering Strategy defines digital engineering as an "integrated digital approach" that improves engineering activities—including communication, design, analysis, and decision-making—across stakeholders and lifecycle phases [1]. Digital engineering (DE) is repeatedly framed not as a collection of tools, but as an ecosystem of trustworthy digital threads connecting authoritative sources of truth to program execution and governance [11]. This framing is important for product planning and concept development – in particular, as electrification and other technologies evolve rapidly and program cycles compress, the ability to *trace* and *justify* commitments becomes an essential engineering capability, not merely a documentation practice.

2.1. Traditional Decision Support and Analysis

Decision support systems (DSS) and decision analysis (DA) provide normative foundations for structuring choices under uncertainty [12]. Graphical representations such as influence diagrams offer powerful formalisms for reasoning about probabilistic dependencies and preferences [13]. However, these methods primarily focus on the analysis of an isolated decision event. They often assume relatively well-structured decision types with stable objectives, bounded alternatives, and controlled datasets, and they do not specify how an enterprise should preserve an auditable decision record across time. In concept development, programs need more than "a model that outputs a recommendation"; they need a durable, machine-interpretable record of framing, evolving evidence versions, reproducible evaluation runs, approval conditions, and downstream actions and outcomes. Put differently, DA explains how to reason, but it does not define how that reasoning is governed, reproduced, and traced within a complex DE ecosystem.

2.2. MBSE and SysML-Centric Approaches

SysML v2 and MBSE repositories are essential sources of authoritative technical evidence, but they are not designed to serve as governance workflow engines for enterprise decisions. Concept decisions draw on heterogeneous evidence beyond the system model—market studies, supplier quotes, test results, cost models, risk assessments, and policy constraints—and they require multi-stakeholder approvals with enforceable conditions and stop rules. Forcing all decision governance semantics into SysML commonly produces brittle, non-standard extensions that undermine interoperability. We therefore treat MBSE artifacts as referenced evidence (with explicit configuration context) and represent the decision record as a distinct, tool-agnostic governance contract that can link to SysML v2 through stable identifiers and APIs without being embedded inside the model.

2.3. Emerging Digital Thread Ontologies and Data Models

Recent DE research directly addresses some of the gaps outlined earlier. Gautier et al. [5] propose an ontology-based digital thread specifically for analysis and decision support activities, emphasizing semantic interoperability across toolchains. Singh and Willcox [7] present a principled formulation for decision-making under uncertainty where the digital thread's state evolves with new evidence and decisions. Parnell et al. [7] have introduced a Decision Analysis Data Model (DADM) to standardize decision data elements within DE environments.

Complementary standards for decision logic, assurance argumentation, and provenance address important sub-problems but do not, by themselves, produce an end-to-end decision lifecycle record that binds alternatives, uncertainty context, immutable evaluation provenance, readiness gating, commitment conditions, and downstream actions/outcomes across heterogeneous engineering tools. These standards and the remaining gap are positioned next.

2.4. Standards for Decision Logic and Provenance

Several established standards address important sub-problems but do not provide an end-to-end, auditable decision artifact for high-stakes engineering programs. Decision Model and Notation (DMN) standardizes decision logic and decision tables and is valuable when decision policy is primarily rule-based and stable [16]. However, engineering concept decisions require additional structure: evolving alternatives, quantitative uncertainty representations, reproducible evaluation runs, and approvals with enforceable conditions linked to downstream actions. Goal Structuring Notation (GSN) provides strong support for structuring assurance arguments (claims, strategies, evidence) and can complement governance by representing justification logic [17]. However, GSN is not a lifecycle decision record: it does not natively represent alternative sets, evaluation execution provenance, state transitions, or readiness gating. For provenance, W3C PROV provides a generic, widely adopted vocabulary for entities, activities, and agents [18]. It is highly relevant because enterprise governance repeatedly asks "who produced what evidence, by what process, using which inputs." Yet PROV is intentionally generic; it does not define domain semantics for Decision, Commitment, Evaluation Run immutability, or bias-check gating. Therefore, the contribution of this work is not to replace these standards, but to provide a decision-centric contract that can reference DMN artifacts as evaluators, represent GSN artifacts as evidence/justification, and remain mappable to PROV while adding the domain-specific primitives and lifecycle governance required for DE decision accountability.

In summary, existing DE decision-support threads and decision data models improve interoperability and standardize decision elements, but they typically do not make governance mechanically enforceable. The unresolved enterprise gap is an auditable decision artifact that (i) binds framing, objectives, alternatives, uncertainty, and evidence to a specific commitment event, (ii) records immutable execution provenance for evaluations independent of tool-specific versioning, and (iii) enforces lifecycle readiness through policy-checkable gates and role-constrained approvals. This paper addresses that gap by treating the decision record as a system-of-record contract rather than as documentation attached to other artifacts.

Overall, these prior works are significant steps toward standardizing decision data structures. However, a gap remains in operationalizing these structures as active *governance mechanisms*. An enterprise schema must do more than define fields for "alternatives" and "criteria." It must act as an operational contract that enforces process rigor—for example, systematically blocking a gate transition if required bias checks are missing, or ensuring that every piece of evidence is traceable to an immutable run record independent of the originating tool's versioning system.

2.5. The Army's Need

The decision cycle for engineering and re-engineering has compressed from years down to 2 months in current warfare. These conflicts have created an existential threat to military Product Design and Development (PDD) regardless of whether products are commercially, defense contractor, or government source. Non-traditional, streamlined, and greater commercial sourcing may help, but the primary issue is that the Observe, Orient, Decide, Act (OODA) loop of PDD must now operate in weeks as opposed to months or years.

The use of AI/DE will allow the defense PDD OODA loop to improve, but to also undercut the threat's PDD OODA loop. The resultant time and cost savings coupled with quicker to deployment of over-matching capabilities will dramatically support U.S. and Allied capabilities to deter and avoid war, as well as end it quickly.

3. REQUIREMENTS FOR ENTERPRISE-GRADE DECISION REPRESENTATION

The design of a decision representation for high-stakes engineering programs must be driven by operational governance requirements, not just analytical needs. Based on DE best practices [8], decision science principles regarding bias [19], and the practical challenges of managing complex vehicle programs, we derive the following key technical requirements:

R1. Decision as a Governable Object: The representation must explicitly encode the decision's framing (need, scope, objectives, alternatives, constraints, uncertainty) as a queryable, identifiable object.

R2. Structural Separation of Decision, Action, and Outcome: Structural Separation of Decision, Action, and Outcome: A rigorous decision process can still lead to a poor outcome due to exogenous uncertainty.

The schema must separate (i) the decision record (framing and analysis state) from the binding approval event (Commitment), and (ii) the Commitment from implementation work (Actions) and realized results (Outcomes), enabling accurate post-hoc auditing of decision quality independent of results.

R3. Provenance and Reproducibility: High-stakes decisions depend on rapidly evolving models. Pointing to a current SysML model URL is insufficient if that model changes. Every evaluation informing a decision must be traceable to an *immutable run record* that binds the specific evaluator version, input parameter set, and resulting output artifacts, enabling exact reproduction years later [1], [8], [20].

R4. Uncertainty & Risk as Structured Elements: Uncertainty representations must support distributions, scenario sets, and intervals, and risk elements must connect to mitigations, monitors, and acceptance criteria [21], [22].

R5. Bias Governance: Engineering decisions are vulnerable to cognitive biases under pressure [19], [23]. The schema must support machine-readable policies that enforce specific checks (e.g., premortems [24]) as required evidence artifacts before a decision can progress through gates.

R6. Hierarchical & Delegation: Programs routinely contain subsystems and components with delegated decision authority. The representation must support multi-level product hierarchies and decision interactions across levels (vertical partition/synthesis and horizontal coupling) and the "flow items" exchanged between decisions (e.g., shared parameters, constraints, risks), consistent with decision interaction ontology findings [25].

R7. Support for Set-Based and Concurrent Development: Modern vehicle programs often employ set-based design practices to manage uncertainty [26], [27]. The schema must represent evolving option sets, conditional commitments, and explicit fallback triggers, rather than assuming a simplistic single-point selection process [28].

R8. Cross-program Reuse & Lineage: Shared subsystems and technologies create dependencies across programs. The representation must allow decisions, risks, and evidence to be reused and referenced across graphs with stable identifiers and version lineage.

R9. Integration with Engineering Ecosystems: The schema must function as an integration layer. It must integrate with MBSE, PLM, and simulation tools through adapters to capture authoritative references to identities and run artifacts within those source systems [14], [15]. Integration must also preserve analysis semantics—parameter roles, domains, and validity regions—so that evaluation workflows and trade studies can be checked for consistency rather than treated as opaque scripts [5].

R10. Readiness Enforcement: The platform should be able to block gate transitions unless required decision artifacts exist (stakeholders, objectives, alternatives, evaluation runs, approvals, bias checks). Structural validity is necessary but not sufficient; semantic readiness must be enforceable.

R11. Security and Access Policy Inheritance: Enterprise deployment requires access control and handling policy integration for decision artifacts and evidence, including inheritance and exception semantics so sensitive evidence can be governed consistently without fragmenting the decision record.

R12. Knowledge Management: Decision records should support outcome feedback and capture lessons learned, enabling future programs to reuse proven metric definitions, evaluators, and decision templates.

R13. Ingestion Provenance for Legacy Decision Artifacts: Because decision rationale often originates in slide decks, minutes, and reports, the schema must represent whether content was manually authored, imported, or NLP/agent-assisted, and must preserve durable links back to the originating evidence artifacts to prevent "retyped history" and to maintain auditability [29].

R14. Federated Configuration & Link Validity Semantics: For enterprise digital threads spanning multiple authoritative sources, the representation should capture baseline/stream/changeset context and explicit link validity states (valid/suspect/invalid) to manage drift and maintain trust as referenced artifacts change [8].

R15. Role-Constrained Automation and Agent Accountability: The representation must explicitly record actor type (human, service, AI agent) for created/modified artifacts and allow enterprise policy to restrict which object kinds and lifecycle transitions each actor may perform. In particular, automation may propose alternatives, execute evaluation runs, and attach evidence, but commitments and lifecycle transitions must remain role-governed and must leave an auditable policy-check trace.

Table 1 maps requirements R1–R15 to the concrete schema primitives and representative fields that satisfy them and previews the operational contract defined in Sections 4–5. These requirements imply that a decision representation must be more than a passive record schema. It must encode enforceable invariants (e.g., append-only evaluation provenance; explicit lifecycle state and transition authority; role-bound review status) and must support programmatic validation of readiness for gate transitions. Accordingly, the schema in Sections 4–5 is presented not only as a metamodel but as an operational contract: it defines what must exist, what must be immutable, what must be reviewed, and what may change only through governed transitions.

4. THE DECISION DIGITAL THREAD SCHEMA

We represent a decision program as a typed, directed graph because high-stakes engineering decisions are interdependent rather than isolated. Concept decisions constrain one another through precedence relationships, reuse shared evidence and evaluation runs, propagate common risks (e.g., supplier or regulatory uncertainty), and compete for shared resources and schedules. A graph-based representation makes these dependencies explicit and computable, supporting traceability and lifecycle continuity objectives associated with digital engineering [1], [8]. It also supports AI-enabled decision workflows by providing a structured representation that can be traversed for dependency analysis, consistency checks, and governance enforcement.

The graph abstraction offers two practical benefits. First, it supports multi-scale decision structures, in which program-level decisions decompose into subsystem and component decisions and are subsequently consolidated at formal gate commitments. Second, it provides a unified substrate for decision-centric lifecycle reasoning: the same representation links concept decisions to the evidence used to justify them, the actions authorized for implementation or risk mitigation, and the outcomes observed during execution. This continuity enables post hoc reconstruction of decision rationale and strengthens accountability.

Within the schema, node types represent governable entities (e.g., Decision, Evaluation Run, Evidence, Commitment, Action, Outcome), and edge types encode explicit semantics (e.g., Decision→requires→Evaluation Run; Evaluation Run→produces→Evidence; Commitment→decides→Decision; Commitment→authorizes→Action). This separation supports policy-based validation of required node/edge patterns prior to lifecycle transitions and enables audit queries that reconstruct the evidence basis for a commitment.

4.1. Schema Semantics and Invariants

The Decision Digital Thread is intended to be audit-grade; therefore, key semantics are defined as invariants rather than as conventions. (i) Evaluation Run records are immutable: once sealed, their evaluator identity/version, inputs, referenced artifacts, and integrity hashes cannot be altered, only superseded by a new run. (ii) Evidence objects carry explicit trust state (e.g., draft, reviewed, rejected) and the reviewer role(s) required for readiness; readiness policies may depend on Evidence state, not merely Evidence existence. (iii) A Commitment is the only artifact that can move a decision from analysis to a binding committed state; it must record approver identity/role and must preserve any approval conditions and stop rules as machine-actionable constraints.

Table 1. Mapping of enterprise requirements to representative schema constructs.

Requirement	Representative schema constructs (examples)
R1: Decision as first-class object	`Need` → `Decision` linkage; `Decision.valueModel`; `Decision.choiceModel`; `Decision.outcomeModel.decisionCommitment`; typed edges `addresses`, `decides`
R2: Decision/action/outcome separation	Node kinds: `Decision`, `Action`, `Outcome`; edges `implemented_by`, `realizes`, `results_in`; post-hoc `Outcome` feedback loops
R3: Provenance & reproducibility	`Evidence` (`uri`, `hash`, `reviewStatus`); `EvaluatorDefinition` + `ModelDefinition`; immutable `EvaluationRun` (evaluator version, inputs, referenced artifacts, outputs); PROV-aligned export fields
R4: Uncertainty & risk	`UncertaintyDefinition` (scenario_set / distribution / interval); `RiskDefinition` linked to objectives/uncertainties; mitigation `Action` links; risk appetite/acceptance criteria
R5: Bias governance	`BiasModel` + `BiasAssessment` + `BiasCheck`; `DecisionClass.requiredBiasChecks`; planner auto-generated bias-check `Action` nodes; readiness gating
R6: Hierarchy & delegation	`Entity` hierarchy via `parentEntityId`; decision scope bindings; `decomposes_into` edges for decision decomposition
R7: Set-based + concurrent	`ChoiceModel.setBasedDesignPolicy`; alternative readiness attributes (e.g., TRL/MRL); explicit fallback relationships; conditional commitment conditions
R8: Reuse + lineage	Definition registry with stable IDs: `Metric`, `Objective`, `Risk`, `Uncertainty`, `Evaluator`, `Product`, `Model`; `Revision` and `supersedes` edges; optional cross-graph references
R9: Toolchain integration	`EvaluatorDefinition` adapters (lookup/code/RPC; MBSE adapter); `ModelDefinition` identity/version; `EvaluationRun` artifacts referencing PLM/CAE/MBSE results
R10: Readiness enforcement	`DecisionClass.readinessCriteria`; semantic validator policies; state-transition guards requiring evidence + eval runs + checks + approvals
R11: Security/classification	classification on graph/nodes/evidence; `metadata.accessPolicyRef` and `handlingPolicyRef`; inheritance rules
R12: Knowledge & learning loops	`Outcome` capture; predicted vs realized comparisons; promotion of reusable definitions to library; "lessons learned" `Evidence` linked to decisions
R13: Ingestion provenance	`IngestionProvenance` (authored/imported/NLP/agent-assisted); `sourceArtifactRef` (durable URI + hash + repository identifiers); `derived_from` / `extracted_from` links to originating artifacts; ingestion/extraction event record (tool/version, actor, timestamp)
R14: Federation & link validity	`ExternalReference` (URI + baseline/stream/changeset context); `linkValidityState` (valid/suspect/invalid) + resolution timestamp; drift/consistency checks tied to referenced artifacts; policy use of link validity in readiness gating
R15: Role-constrained automation & agent accountability	`actorType` recorded on create/modify (human/service/agent); `DecisionClass.permissionPolicy` for artifact kinds + lifecycle transitions; `PolicyCheckRecord` + `StateTransition` audit event (actor, policy version, pass/fail, blockers); `Commitment` creation and lifecycle advancement restricted to authorized roles

(iv) Decision lifecycle transitions are governed events: transitions must be accompanied by an explicit policy-check record, and authority to transition is role-scoped. These invariants ensure that "what was known when" can be reconstructed independently of tool-specific versioning and independent of later narrative reinterpretation.

4.2. Core Schema Primitives

The schema is organized around a minimal set of typed primitives designed to satisfy the requirements in Section 3; the resulting metamodel is summarized in Figure 1.

- **Need:** Captures the initiating trigger and scope boundary for a decision program (e.g., a capability gap or requirement deviation). It binds the decision program to measurable success criteria and establishes the organizational context for governance.
- **Decision:** Serves as the primary unit of governance. It captures decision framing, including references to objective definitions, the alternatives (or option sets) under consideration, relevant constraints, uncertainty representations, and linked risks. Decision readiness is evaluated across lifecycle states. Appendix A provides an illustrative Decision node that encodes scope boundaries and references objective definitions via registry identifiers.
- **Evaluation Run:** Represents the execution provenance that connects decision framing to technical evidence. It is an immutable record of a specific assessment event, such as a CAE simulation sweep, a SysML parametric evaluation, or a structured expert assessment. The Evaluation Run captures input bindings, model and evaluator versions, scenario parameters, and references to output artifacts. This construct is central to provenance and reproducibility (R3). Appendix B provides an illustrative immutable Evaluation Run record (inputs, evaluator version, output hashes, sealing hash) to make the provenance fields concrete.
- **Evidence:** Anchors evaluations and decisions in authoritative sources (e.g., datasets, test reports, reviewed documents, model artifacts). Evidence objects carry review status, provenance pointers (aligned with W3C PROV [18]), and integrity hashes.

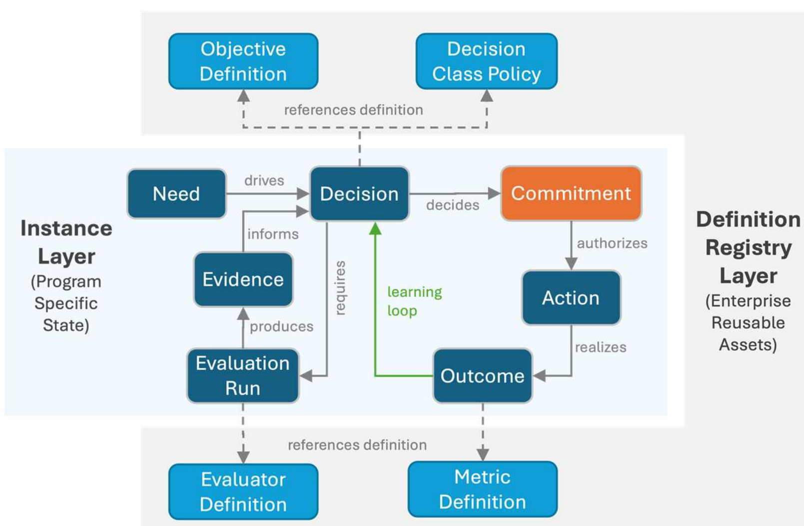

Figure 1: Decision Digital Thread Schema Metamodel.

- **Commitment:** Records the formal approval event that transitions a decision to the COMMITTED lifecycle state (i.e., a binding program decision). It specifies the selected alternative, approver identity and role, approval date, and any conditions or stop rules. This primitive separates analytical recommendation from binding program agreement (R2,R10). Appendix C provides an illustrative DecisionCommitment record that captures approver role/identity and machine-actionable conditions and stop rules.
- **Action:** Represents executable work items driven by the decision thread, including implementation tasks, risk mitigations, and required governance checks (e.g., debiasing activities). Actions link decisions to execution and enable traceability of follow-through (R2).
- **Outcome:** Captures realized results and observed effects, enabling comparison between expected and realized performance and supporting continuous improvement (e.g., model calibration and lesson capture) (R2, R12).

DDT is compatible with W3C PROV while adding domain semantics required for decision governance. In a PROV mapping, Evaluation Runs correspond to prov:Activity; Evidence and referenced artifacts correspond to prov:Entity; and human/services/agents correspond to prov:Agent. Edges such as EvaluationRun→produces→Evidence and EvaluationRun→uses→Evidence align with prov:wasGeneratedBy and prov:used, while decision governance introduces domain-specific primitives—Decision, Commitment, Action, Outcome, and readiness gates—that PROV intentionally leaves unspecified. This approach preserves exportability to generic provenance tooling without sacrificing enforceable decision lifecycle semantics.

4.3. Enterprise Reuse via Definition Registries

Enterprise-scale decision management requires semantic consistency across programs. If each program independently defines objectives, metrics, uncertainty models, and evaluators, cross-program analysis becomes unreliable and governance becomes inconsistent. The schema therefore supports a Definition Registry containing canonical, versioned definitions referenced by stable identifiers, including:

- **Metric Definitions:** name, units, directionality, aggregation, and computation type
- **Objective Definitions:** structured value models with stable identifiers and parent–child relationships
- **Product Definitions / Alternative Blueprints:** reusable option building blocks with compatibility metadata
- **Evaluator Definitions:** executable assessment interfaces (e.g., lookup, code, RPC, digital twin adapters, optional MBSE adapters) with qualification status
- **Bias Playbooks:** machine-readable mappings from decision classes to required checks and planner tasks

Decision instances reference these definitions and may apply controlled local overrides (e.g., weights or targets) with explicit justification. This architecture supports consistent reporting, durable auditability, and systematic reuse across programs, addressing challenges associated with heterogeneous tool ecosystems and inconsistent semantics in digital threads [8]. It also aligns with decision management objectives emphasizing standardized decision data elements in DE contexts [7].

4.4. Structured Representation of Uncertainty and Risk

Uncertainty and risk are represented as structured objects rather than narrative qualifiers. Uncertainty representations may include scenario sets, probability distributions, and bounded intervals, supporting deterministic, stochastic, and robust analyses. Risk elements connect uncertainties to consequences through owners, mitigation plans, monitors, and acceptance criteria consistent with integrated risk management guidance and approaches to decision-making under deep uncertainty that emphasize performance across plausible futures [21], [22]. Because risks and uncertainties are first-class objects in the graph, they can be shared across decisions and propagate through decision dependencies, reducing duplication and improving coherence in program-level risk management.

4.5. Bias Governance & Auditable Requirement

Bias governance is treated as an explicit component of decision quality. The decision analysis literature documents how cognitive and motivational biases can influence problem framing, model selection, and interpretation of evidence, particularly under uncertainty and organizational pressure [18], [23]. The schema therefore associates decisions with a BiasModel and supports policy-driven enforcement of required debiasing actions (e.g., premortems and outside-view forecasting) [24]. These requirements are instantiated as Action objects that produce Evidence artifacts, making bias management observable and auditable rather than procedural guidance. This is particularly relevant in concept development, where anchoring effects, incomplete evidence, and schedule constraints can substantially affect decision quality.

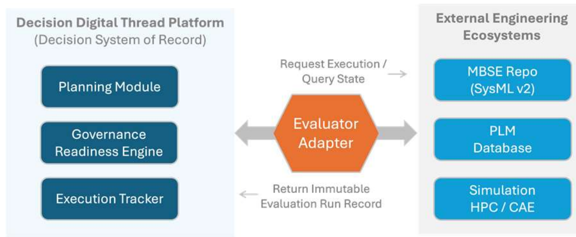

Figure 2: Operational Integration Architecture via Adapters.

4.6. Federation & Analytical Rigor Extensions

Consistent with recent work on ontology-based digital threads and decision knowledge graphs, the schema supports extensions that improve interoperability and governance without coupling to any single toolchain. First, decision-to-decision edges may carry explicit interaction semantics (e.g., weak/strong coupling; partition/synthesis) and typed information flow items describing the content exchanged across hierarchical decisions [25]. Second, parameter definitions include roles and validity regions, and evaluators may declare input/output bindings, enabling automated consistency checks across analyses and trade studies [5]. Third, external references support stable URIs with optional baseline/stream/changeset context and link-validity states, enabling federated digital threads and drift management as upstream artifacts evolve [8]. Fourth, ingestion provenance records whether decision artifacts were authored manually or derived through import or NLP-assisted extraction, preserving traceability to the originating sources [29]. These extensions preserve the core purpose of the schema—decision as a governed digital artifact—while aligning with best practices in federated digital engineering and decision knowledge management.

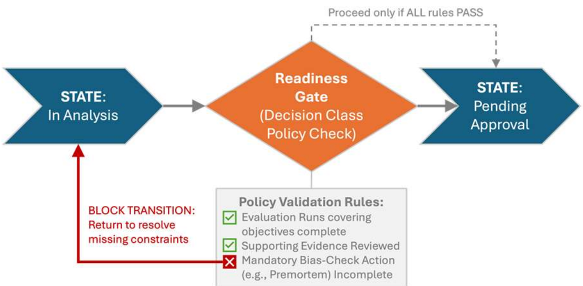

Figure 3: Policy-Driven Readiness Gating Mechanic.

5. OPERATIONALIZING THE SCHEMA: GOVERNANCE MECHANICS

Figure 2 summarizes the adapter-based integration architecture: the DDT platform remains the decision system of record while MBSE/PLM/simulation repositories remain authoritative for their native artifacts.

As shown in Figure 3, decision representation creates value only when it is used as an operational contract for platform behavior. In many systems, the schema functions primarily as a storage model, while workflow state, approvals, and analytical logic are implemented in application code, spreadsheets, and tool-specific integration layers. This separation introduces untracked "side state," in which decisions appear complete in the user interface despite critical dependencies—evidence status, evaluation parameters, model baselines, or governance checks—existing only in external systems. The result is fragmented audit trails, inconsistent decision quality, and limited ability to reproduce the basis for commitments during later reviews. These outcomes are inconsistent with digital engineering objectives for authoritative digital artifacts and lifecycle traceability [1], [8].

Accordingly, the Decision Digital Thread schema is intended to serve as the decision system of record. In enterprise deployment, it should function as the formal interface among (i) platform modules (planning, decision framing, evaluation orchestration, governance), (ii) external engineering ecosystems (MBSE repositories, PLM, simulation services, test and cost systems), and (iii) enterprise policy (roles, approvals, review requirements, bias checks, and gate readiness criteria). A fundamental design constraint follows: all decision-relevant state must be represented explicitly in the schema as nodes, edges, or typed substructures. Modules may maintain internal operational data (e.g., caches or UI preferences), but they should not maintain independent state that affects decision integrity.

5.1. Planning as Executable Governance and Traceable Execution

Planning is operationalized as part of the decision thread by representing work items as **Action** artifacts linked to decisions through typed relationships (e.g., implements, mitigates, produces). The platform planner uses the schema in two complementary ways.

First, it performs policy-driven task instantiation. **Decision Class Policy** definitions specify required evaluations, evidence reviews, stakeholder participation, and bias checks prior to readiness transitions. The planner instantiates these requirements as Action objects (e.g., premortem workshop, outside-view assessment, Monte Carlo evaluation, independent review). This approach addresses a documented limitation of debiasing practices: effectiveness depends on enforced process and documentation, not solely on awareness [19], [24].

Second, the planner records execution status directly into the graph. As actions progress, the platform updates Action state and timestamps, links produced Evidence, and records review outcomes. This makes governance questions queryable from the decision record (e.g., which required checks were completed, what evidence they produced, and which mitigations remain open).

Because decision knowledge frequently originates in presentations, minutes, and reports, the platform should support document-ingestion workflows that generate draft graph objects with explicit ingestion provenance (source artifact identifier + location pointers, extractor/tool or agent identity and version, extraction timestamp), durable links to the source artifacts, and an explicit human acceptance event that promotes extracted content into the authoritative decision record [29].

5.2. Need Framing as a Chartered Entry Point

Need framing provides a structured entry point that reduces "analysis drift" and establishes governance expectations. The framing module produces a **Need** (decision charter) and instantiates an initial Decision draft. The charter anchors the decision program in four elements:

- **Scope boundary:** what is included and excluded, including the intended gate or lifecycle phase for commitment.
- **Authority and accountability:** stakeholders, roles, decision board membership, and delegation boundaries.
- **Success criteria:** measurable objectives and metrics defining decision quality.
- **Hierarchy binding:** the program, subsystem, or component scope governed by the decision.

This is also the appropriate point to bind the decision to the Definition Registry by instantiating default objective hierarchies, metric definitions, common risk registers, and standard scenario sets. Early binding promotes semantic consistency and reduces omissions that later compromise decision comparability and governance [7], [8].

5.3. Evaluator Integration and Immutable Evaluation Runs

Reproducibility requires that the platform preserve the evaluator identity, version, inputs, and outputs used to generate evidence. Integration is operationalized using two paired constructs.

- **Evaluator Definition** specifies how an evaluation is executed (e.g., code, lookup, RPC call, digital twin adapter, structured rubric) and declares its inputs and outputs through registry identifiers. Evaluators are treated as controlled enterprise assets with qualification status (e.g., draft, validated, qualified).
- **Evaluation Run** is an immutable record of a specific execution event. It captures evaluator identity and version, parameter bindings, input data references, scenario set versions (for stochastic runs), output artifacts with integrity hashes, and summarized metric outputs. Treating runs as immutable supports audit-grade reconstruction of evidence production and aligns with established provenance principles [18], [20].

This approach accommodates both computational and human-centered evaluations. In structured reviews, the "run" may reference the rubric, the reviewed evidence set, and reviewer identity, preserving comparability and auditability without relying on informal notes.

5.4. Readiness Enforcement Beyond Structural Validation

Schema validation (e.g., JSON Schema) establishes structural correctness but does not ensure decision readiness. Governance requires semantic rules specifying what must exist and what statuses must hold before lifecycle transitions occur. A readiness engine enforces these transitions by evaluating Decision Class Policy rules against the graph state. Appendix D provides a minimal DecisionClassPolicy example encoding readiness gates, blocking behavior, and explicit role/agent authority boundaries.

For example, a decision should not be permitted to transition to COMMITTED unless: (i) Evaluation Runs cover primary objectives and constraints; (ii) supporting Evidence for key assumptions and results has the required review status; and (iii) required bias checks are complete with resulting artifacts captured as reviewed Evidence [19]. At the moment of transition, the platform records a Commitment object capturing approver identity/role, rationale, and any conditions or stop rules. When conditions are unmet, the engine blocks the transition and returns explicit blockers (e.g., missing evidence review status, missing evaluator coverage for a primary objective). This converts governance requirements into enforceable controls and supports digital engineering traceability expectations [1], [11].

Decision lifecycle states should be defined explicitly to avoid ambiguous governance. A minimal state model suitable for concept decisions includes: DRAFT (framing incomplete), IN_ANALYSIS (evaluations and evidence being generated), PENDING_APPROVAL (readiness gate passed; awaiting approver action), COMMITTED (binding Commitment recorded), EXECUTING (authorized Actions in progress), and CLOSED (outcomes captured and learning archived). Exceptional states such as VOID (commitment invalidated by a stop rule) and SUPERSEDED (replaced by a later decision revision) support audit reconstruction without rewriting history. Each state transition is a governed event that must record the triggering actor, the policy version evaluated, and the set of satisfied/unsatisfied readiness rules at the time of transition.

AI-agent participation must be constrained by explicit authority boundaries. The platform should treat agent outputs as first-class artifacts with ingestion provenance, but should restrict agents from creating or modifying governance-critical artifacts and fields. A practical boundary is: agents may create Evaluation Runs, Evidence attachments, and draft Actions; agents may not create Commitments, and agents may not advance Decision lifecycle state. This separation prevents automation from bypassing approvals while still enabling agents to accelerate evidence discovery and evaluation execution. Readiness gating then becomes the enforcement point that converts agent-produced artifacts into decision-ready inputs only after required reviews and checks are satisfied.

5.5. Revision, Lineage, and Audit Reconstruction

Enterprise use requires explicit revision semantics. Without them, the decision record cannot support compliance or retrospective review. The schema should therefore support: (i) graph-level revision history (who changed what and why), (ii) stable node identities across revisions so references remain valid, and (iii) immutable evidence artifacts with integrity hashes and review status linked to the Evaluation Runs and decisions they support. These mechanisms enable reconstruction of “what was known when,” and they also support learning by comparing predicted and realized outcomes across decision revisions and programs [6], [8].

5.6. Federation, Configuration Context, and Link Drift Management

Federated digital threads span MBSE repositories, PLM systems, simulation services, and test databases. References to external artifacts therefore require explicit configuration context (e.g., baseline, branch/stream, changeset) and link-validity semantics so that upstream changes can be detected and managed [8]. Marking references as suspect when dependent artifacts change enables governance review of whether a prior commitment remains supported by current evidence.

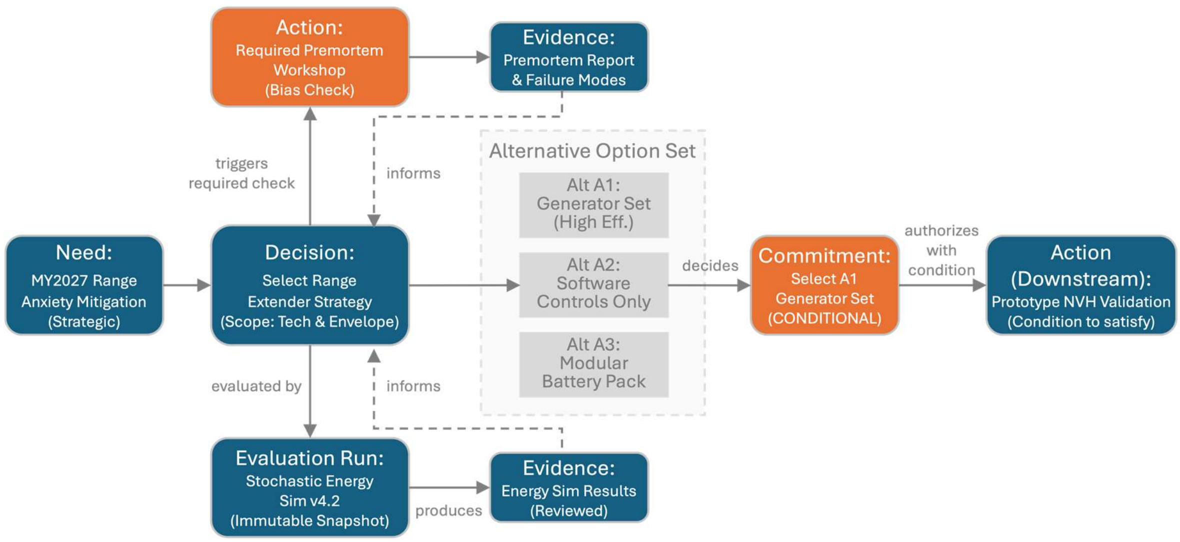

Figure 4: BEV Generator Concept Decision Instantiation.

Policy also evolves over multi-year product cycles. When Decision Class Policy requirements change (e.g., additional bias checks), the platform must reconcile in-flight decision programs without rewriting history. Synchronization mechanisms should instantiate newly required actions while archiving or superseding obsolete ones, preserving a traceable record of the requirements that applied at the time of prior approvals.

6. CASE STUDY: ILLUSTRATIVE CONCEPT DECISION INSTANTIATION IN A BEV PICKUP PROGRAM

This case study is an illustrative example designed to demonstrate how the proposed Decision Digital Thread schema represents a concept development decision as a governable, auditable digital artifact. The vehicle program, organization, and scenario are not based on a specific manufacturer or proprietary dataset. Parameter values (where referenced) are representative and used solely to support explanation of schema structure, evidence lineage, and readiness gating. Figure 4 instantiates the schema for the range-extension concept decision, showing how Need/Decision framing, Evaluation Runs, reviewed Evidence, governance Actions, and a conditional Commitment connect into an auditable decision thread.

6.1. Program Context and Decision Trigger

An automotive manufacturer is developing a MY2027 battery-electric pickup intended to maintain competitive towing capability and cold-weather usability while meeting cost and schedule constraints typical of a major vehicle program. Early concept exploration identifies a persistent customer-facing risk: perceived usable-range shortfalls under demanding conditions such as towing, low ambient temperature, and high accessory loads. The program further recognizes two limiting tendencies in early-phase reasoning: (i) addressing the issue solely by increasing battery capacity can drive cascading impacts on mass, cost, packaging, and manufacturing feasibility; and (ii) addressing the issue solely through software may be bounded by physical constraints in the most demanding duty cycles.

Entering concept development, the program initiates a focused decision: select a primary range-extension strategy that is feasible within program constraints and that reduces range anxiety under representative and stressing use scenarios. The decision scope includes range extension with or without thermal support and explicitly excludes detailed embodiment decisions that are not appropriate at this stage (e.g., packaging location and integration architecture for generator systems). If a generator-based strategy is selected, a subsequent decision is triggered to resolve packaging and integration trade-offs under a narrower scope and with additional evidence.

6.2. Decision Framing as a Governed, Scoped Artifact

The decision is instantiated as a **Need** artifact and a corresponding **Decision** artifact linked to the vehicle program entity. The Need captures the initiating concern (range anxiety risk under stressing conditions), boundary conditions (battery-electric primary customer experience), and the targeted commitment point (concept lock / early refinement).

To prevent scope creep and analysis drift, the Decision encodes explicit scope boundaries:

- **Included:** selection among a defined set of candidate strategies and specification of the intended operating concept (range extension only versus range extension plus thermal support).
- **Excluded:** detailed physical packaging, location selection, and integration architecture for generator systems.
- **Excluded:** detailed calibration design and software architecture (represented as follow-on actions and later decisions after strategy selection).

Decision objectives are instantiated by referencing canonical definitions in the Definition Registry rather than embedding free-text goals. For this decision, the objective hierarchy includes: (i) reduction of range anxiety under demanding scenarios, (ii) schedule feasibility to the target model year, (iii) development and validation cost burden, (iv) unit cost and pricing implications, and (v) thermal-management contribution in cold-weather operation. This structure supports comparability across alternatives and enables policy-driven checks tied to objective coverage.

6.3. Alternatives and Trade Space Definition

The Decision enumerates three alternatives corresponding to distinct levers for range extension. Each alternative is represented as an option with a minimal parameter set sufficient for evaluation and governance. The trade space is deliberately limited to avoid uncontrolled expansion of analysis.

- **A1: Generator set (range extender; potential thermal support).** Introduces an onboard generator set as an energy extender. At this stage, the program does not decide packaging location; it commits only to the strategy and a defined operating envelope. Key considerations include integration complexity, regulatory and certification pathways, cost, and customer acceptance.
- **A2: Software-only controls (energy management optimization).** Pursues range improvement via control strategies (e.g., predictive energy management and thermal controls) and can also affect range anxiety through forecast credibility and driver guidance. Key considerations include performance bounds in worst-case duty cycles and validation requirements across customer variability.
- **A3: Increased battery capacity (hardware energy increase).** Extends range through increased stored energy, improving robustness across many duty cycles. Key considerations include pack engineering and supplier readiness, manufacturing impacts, mass and packaging constraints, and cost/pricing pressure.

Across all alternatives, the schema requires explicit linkage between each option and the relevant objective and metric definitions, ensuring that evaluation results can be compared on consistent semantics.

6.4. Evaluation Plan and Evidence Lineage

The decision is governed by an explicit evaluation plan that specifies what must be established prior to commitment. Because the criteria include schedule and cost—beyond physical performance—the plan mixes quantitative analyses with structured assessments. Each assessment event is represented as an immutable **Evaluation Run** and anchored to **Evidence** artifacts carrying review status and provenance.

The evaluation plan includes:

1. **Range effectiveness under representative duty cycles.** Scenario-based energy simulation and/or vehicle energy modeling across stressing and representative scenarios (e.g., towing-heavy, cold-weather, mixed use). Alternative-specific assumptions are encoded explicitly (e.g., generator duty assumptions for A1; efficiency and prediction-credibility impacts for A2; mass/energy coupling for A3).
2. **Schedule feasibility and development risk.** Structured assessment of integration, validation, and (where relevant) regulatory/certification timeline risk. Where possible, a reference-class baseline is used to reduce optimistic bias.
3. **Development and unit cost implications.** Structured estimates of engineering effort and validation burden, BOM deltas, and sensitivity to pricing and take-rate uncertainty.
4. **Thermal-management contribution (when material).** Alternative-specific analysis of cold-weather implications and thermal interactions, captured as evaluation outputs tied to metric definitions.

Each Evaluation Run records evaluator identity and version, input bindings, scenario set version, and hashes of output artifacts. Evidence objects reference authoritative sources (datasets, reports, simulation outputs) and carry review status. This enables post hoc reconstruction of the evidence basis for the commitment without reliance on informal documentation.

6.5. Uncertainty and Risk Representation for Robustness

The decision is shaped by uncertainty in both usage and program execution. Key uncertainties include duty-cycle distributions (towing frequency and load levels), ambient temperature regimes and accessory loads, supplier lead times, regulatory timing (particularly for generator-based approaches), and customer acceptance and willingness-to-pay uncertainty.

In the schema, uncertainties are represented as structured objects (scenario sets, bounded intervals, or probability distributions) and linked to risk objects with explicit owners, mitigations, monitors, and acceptance criteria. Because risks are first-class objects in the graph, they can be shared across dependent decisions, enabling coherent program-level risk reasoning rather than duplicated narrative statements.

6.6. Governance Controls: Bias Checks as Enforceable Actions with Evidence

This concept decision is susceptible to framing effects, confirmation bias in evidence selection, and overconfidence in schedule and integration assumptions. The decision class policy therefore mandates bias-governance checks prior to readiness for approval. These checks are represented as **Action** objects that produce **Evidence** artifacts, enabling verification that the checks were executed and reviewed.

Required checks include: (i) anchoring control through structured comparison across the alternative set, (ii) an independent review focused on disconfirming evidence, and (iii) a premortem workshop to identify plausible failure modes and link them to mitigations [24]. A readiness gate blocks transition from "in analysis" to "pending approval" unless these required actions are complete and the resulting evidence artifacts have the required review status.

6.7. Commitment, Conditions, and Sequencing of Follow-On Decisions

When evaluation evidence is sufficient and governance checks are satisfied, the program records a **Commitment** object selecting one primary strategy (A1, A2, or A3). The Commitment captures approver identity and role, approval timestamp, rationale, and structured conditions and stop rules so that downstream satisfaction can be tracked programmatically.

The Commitment also encodes follow-on decision sequencing:

- If **A1** is selected, a subsequent decision is triggered to determine generator placement and integration architecture.
- If **A2** is selected, follow-on decisions address software architecture and validation strategy across usage variability.
- If **A3** is selected, follow-on decisions address supplier strategy, manufacturing implications, integration impacts, and pricing position.

In all cases, the Commitment links to implementation Actions and Outcome tracking. Captured outcomes support later comparison of predicted versus realized performance and enable reuse of validated evaluators and definitions in subsequent programs.

6.8. Walkthrough: Gate Readiness and Conditional Commitment Trace

To illustrate how governance becomes enforceable rather than narrative, consider a concrete trace through the decision lifecycle. The decision begins in IN_ANALYSIS with three candidate strategies instantiated as an option set and an evaluation plan generated from the decision class policy. The platform executes a scenario-based energy simulation Evaluation Run for each alternative and records immutable run provenance (evaluator identity/version, scenario set version, parameter bindings, and hashed outputs). The resulting simulation outputs are registered as Evidence but remain non-authoritative until reviewed.

A readiness gate is then evaluated to transition the decision from IN_ANALYSIS to PENDING_APPROVAL. The transition is blocked if (i) evaluator coverage for primary objectives is incomplete, (ii) supporting Evidence lacks the required review status, or (iii) mandated bias checks are incomplete. In this example, the first gate evaluation fails because the premortem workshop action has not been completed and its Evidence artifact is missing. The readiness engine returns explicit blockers, and the planner instantiates the missing action. After the premortem is executed, the produced Evidence is reviewed and accepted, and the readiness gate passes.

At approval, a Commitment is recorded selecting a primary strategy as a conditional commitment. Conditions and stop rules are stored as machine-actionable constraints tied to verification Actions. For example, a commitment may require an NVH threshold verification by a due date and may specify that failure voids the commitment and returns the decision to IN_ANALYSIS. This structure ensures that conditional approvals remain enforceable constraints downstream rather than informal "to-do" notes and allows the platform to block downstream actions when conditions lapse.

6.9. Platform-Level Validation Checks Illustrated by the Case Study

The illustrative case also motivates validation checks that reduce common failure modes in enterprise deployment.

1. **Evaluator semantic consistency.** The platform should validate evaluator bindings and parameter roles to detect inconsistencies that compromise interpretability and reproducibility [5].
2. **Decision interaction coverage.** In hierarchical decision programs, dependencies should be captured not only as precedence links but also as typed interaction semantics and information flow items [25]. Coverage can be measured as the fraction of key decision relationships annotated with interaction type and flow payload.
3. **Federation readiness and link-drift detection.** External references should include stable identifiers and configuration context (baseline/stream/changeset) and should support link-validity state so upstream changes can be detected and surfaced for governance review [8].

6.10. Limitations and Threats to Validity

This paper demonstrates structural and governance capabilities of the schema through a realistic but illustrative concept decision. The evaluation is therefore provenance- and policy-mechanics–based rather than empirical. The schema does not guarantee decision quality; it guarantees that the artifacts required to assess decision quality—assumptions, reviewed evidence, evaluation provenance, approvals, and conditions—are explicit and auditable. Empirical validation across multiple programs is required to quantify impacts on cycle time, rework, and audit effort. In addition, integration fidelity depends on adapter quality and on organizational discipline in avoiding untracked “side state.” These limitations are inherent to any enterprise decision system-of-record approach and motivate future multi-program deployments and observational studies.

7. CONCLUSIONS

Digital engineering initiatives seek to connect authoritative engineering artifacts across stakeholders and lifecycle phases. However, digital threads remain incomplete when they do not represent the most consequential artifacts of early product development: the decisions that convert evolving evidence into binding program commitments. A program may maintain rigorous traceability for requirements, architectures, and verification results and still be unable to reconstruct, with audit-grade clarity, which alternatives were considered, which uncertainties were acknowledged, what evidence and evaluator versions supported key conclusions, what approval conditions governed the commitment, and which downstream actions and outcomes followed. This paper argues that an explicit, machine-readable decision representation is therefore a prerequisite for decision-centric digital engineering and for governance-aligned AI-enabled decision workflows.

We introduced a Decision Digital Thread schema implemented as a typed, directed graph in which a decision is represented as a governed object connected to its motivating need, scoped alternatives, objective and metric references, uncertainty and risk constructs, evaluation runs and evidence, commitment events, authorized actions, and observed outcomes. The primary contribution is not an optimization method but an information model that makes decision-making operationally traceable and computationally interrogable across heterogeneous toolchains. This method-agnostic posture is intentional: organizations vary widely in analytic maturity and engineering ecosystems, and scalable adoption requires a durable decision record that can reference diverse evaluators—simulation, MBSE parametrics, cost models, and structured expert assessments—while preserving provenance and governance semantics.

Two design principles are central to enterprise deployment. First, evaluations are represented as immutable **Evaluation Run** records rather than mutable fields embedded within decisions. This binds results to evaluator identity and version, explicit input bindings, scenario definitions, and hashed output artifacts, enabling reproducibility and retrospective reconstruction of "what was known when," even as models and assumptions evolve. Second, the schema is intended to function as an operational governance contract rather than a passive storage format. Readiness policies can be evaluated against required node and edge patterns before lifecycle transitions are permitted; commitments can be recorded with structured conditions and stop rules; and bias-governance activities can be instantiated as required actions that produce reviewable evidence artifacts. Together, these mechanisms convert governance from informal process guidance into enforceable, auditable controls.

The fictitious concept-decision case study demonstrated how these principles operate in practice. The decision was scoped deliberately to separate strategy selection from downstream embodiment decisions, and it illustrated end-to-end representation of alternatives, uncertainty and risk, mixed-method evaluation plans, immutable run provenance, evidence review status, mandated debiasing actions, conditional commitments, and explicit sequencing of follow-on decisions. While the evaluation presented in this paper is structural and provenance-based, the case study supports a key conclusion: when decision artifacts are represented explicitly and governed through lifecycle rules, they become durable, comparable, and reusable across time and across programs—capabilities that are difficult to achieve through narrative documentation.

This work also clarifies boundaries and implications for adoption. A schema does not guarantee decision quality; it enables decision quality by requiring that evidence, assumptions, reviews, and approvals be represented explicitly and by supporting policy-based readiness enforcement. Adoption therefore depends on usability and authoring efficiency: if interaction costs are high, teams will reintroduce untracked side state. Definition registries, templates, and automated instantiation of required actions are practical necessities to reduce friction while maintaining semantic consistency. In addition, the governance framing is particularly important for AI-enabled decision workflows. AI can accelerate evidence discovery and evaluation execution, but without an explicit decision contract it can also create accountability risk. The schema supports safe integration by delineating what can be generated or updated by automation (e.g., candidate alternatives, evaluation runs, evidence attachments) and what must remain role-governed (e.g., commitments and lifecycle transitions), with readiness gates enforcing these boundaries.

Three directions follow naturally. First, multi-program empirical studies are needed to quantify effects on rework, cycle time, and audit effort using mixed-method evaluation designs appropriate for long-duration product programs. Second, continued alignment with evolving DE ecosystems—particularly SysML v2 and related configuration-management practices—will strengthen interoperability while preserving the principle that authoritative artifacts remain in their native tools and are referenced through stable identifiers and configuration context. Third, governance-aware AI behaviors should be developed in tandem with the schema, including state-transition controls, provenance verification, and bias-playbook enforcement, so that automation increases efficiency without weakening accountability.

The central conclusion is that a digital thread that represents products but not decisions cannot fully support decision-centric engineering. A Decision Digital Thread provides the missing layer that links evidence to commitments, execution, and learning, enabling governed and auditable decision-making across the lifecycle.

8. REFERENCES

[1] Department of Defense (DoD), *DoD Digital Engineering Strategy*, Office of the Deputy Assistant Secretary of Defense for Systems Engineering, 2018. [Online]. Available: https://ac.cto.mil/digital-engineering/. Accessed: Feb. 22, 2026.

[2] M. A. Bone, M. R. Blackburn, D. H. Rhodes, D. N. Cohen, and J. A. Guerrero, "Transforming systems engineering through digital engineering," *The Journal of Defense Modeling and Simulation: Applications, Methodology, Technology*, vol. 16, no. 4, pp. 339–355, 2019, doi: 10.1177/1548512917751873.

[3] A. M. Madni and M. Sievers, "Model-based systems engineering: Motivation, current status, and research opportunities," *Systems Engineering*, vol. 21, no. 3, pp. 172–190, 2018, doi: 10.1002/sys.21438.

[4] M. Helu, T. D. Hedberg Jr., and A. Barnard Feeney, "Reference architecture to integrate heterogeneous manufacturing systems for the digital thread," *CIRP Journal of Manufacturing Science and Technology*, vol. 19, pp. 191–195, 2017, doi: 10.1016/j.cirpj.2017.04.002.

[5] R. H. Gautier, N. Lepez Da Silva Duarte, A. Baker, and D. N. Mavris, "Formulation of an ontology-based digital thread for analysis and decision support activities," in *Proc. AIAA SciTech 2025 Forum*, 2025, Paper 1092, doi: 10.2514/6.2025-1092.

[6] V. Singh and K. E. Willcox, "Decision-making under uncertainty for a digital thread-enabled design process," *Journal of Mechanical Design*, vol. 143, no. 9, Art. no. 091707, 2021, doi: 10.1115/1.4050108.

[7] G. S. Parnell *et al.*, "Decision analysis data model for digital engineering decision management," *Systems*, vol. 13, no. 7, Art. no. 596, 2025, doi: 10.3390/systems13070596.

[8] E. Gery, "Digital threads—benefits, challenges, and best practices," *INCOSE International Symposium*, vol. 33, no. S1, pp. 103–119, 2023, doi: 10.1002/iis2.13118.

[9] A. Murat, R. B. Chinnam, S. Rana, S. Rapp, A. Dunn, R. Renke, J. Bechtel, and K. Hansen, "Strategic Decision Intelligence: Safeguarding Product Development Success in Complex Engineering Landscapes," in *Proc. Ground Vehicle Systems Engineering and Technology Symposium (GVSETS)*, NDIA, Novi, MI, USA, Aug. 13–15, 2024.

[10] A. Murat, R. B. Chinnam, S. Rana, S. Rapp, K. Hansen, T. Richman, and J. Bechtel, "From Concept to Execution: An AI-Agentic Decision Intelligence Framework for Product Planning and Concept Development," in *Proc. Ground Vehicle Systems Engineering and Technology Symposium (GVSETS)*, NDIA, Novi, MI, USA, Aug. 12–14, 2025.

[11] P. Zimmerman, T. Gilbert, and F. Salvatore, "Digital engineering transformation across the Department of Defense," *The Journal of Defense Modeling and Simulation*, vol. 16, no. 4, pp. 325–338, 2019, doi: 10.1177/1548512917747050.

[12] J. P. Shim, M. Warkentin, J. F. Courtney, D. J. Power, R. Sharda, and C. Carlsson, "Past, present, and future of decision support technology," *Decision Support Systems*, vol. 33, no. 2, pp. 111–126, 2002, doi: 10.1016/S0167-9236(01)00139-7.

[13] R. A. Howard and J. E. Matheson, "Influence diagrams," *Decision Analysis*, vol. 2, no. 3, pp. 127–143, 2005, doi: 10.1287/deca.1050.0020.

[14] Object Management Group (OMG), "Systems Modeling Language (SysML) v2," 2024. [Online]. Available: https://www.omg.org/sysml/. Accessed: Feb. 22, 2026.

[15] N. Jansen, J. Pfeiffer, B. Rumpe, D. Schmalzing, and A. Wortmann, "The language of SysML v2 under the magnifying glass," *Journal of Object Technology*, vol. 21, no. 3, pp. 3:1–3:15, 2022, doi: 10.5381/jot.2022.21.3.a11.

[16] Object Management Group (OMG), "Decision Model and Notation (DMN), Version 1.4," 2021. [Online]. Available: https://www.omg.org/spec/DMN/1.4/. Accessed: Feb. 22, 2026.

[17] J. Spriggs, *GSN—The Goal Structuring Notation: A Structured Approach to Presenting Arguments*. London, U.K.: Springer, 2012, doi: 10.1007/978-1-4471-2312-5.

[18] T. Lebo, S. Sahoo, and D. McGuinness, Eds., *PROV-O: The PROV Ontology*, W3C Recommendation, 2013. [Online]. Available: https://www.w3.org/TR/prov-o/. Accessed: Feb. 22, 2026.

[19] G. Montibeller and D. von Winterfeldt, "Cognitive and motivational biases in decision and risk analysis," *Risk Analysis*, vol. 35, no. 7, pp. 1230–1251, 2015, doi: 10.1111/risa.12360.

[20] P. Missier, K. Belhajjame, and J. Cheney, "The W3C PROV family of specifications for modelling provenance metadata," in *Proc. 16th Int. Conf. Extending Database Technology (EDBT)*, 2013, pp. 773–776, doi: 10.1145/2452376.2452478.

[21] ISO, "ISO 31000:2018 Risk management—Guidelines," 2018. [Online]. Available: https://www.iso.org/iso-31000-risk-management.html. Accessed: Feb. 22, 2026.

[22] R. J. Lempert, "Robust Decision Making (RDM)," in *Decision Making under Deep Uncertainty: From Theory to Practice*, V. A. W. J. Marchau, W. E. Walker, P. J. T. M. Bloemen, and S. W. Popper, Eds. Cham, Switzerland: Springer, 2019, pp. 23–51, doi: 10.1007/978-3-030-05252-2_2.

[23] A. Tversky and D. Kahneman, "Judgment under uncertainty: Heuristics and biases," *Science*, vol. 185, no. 4157, pp. 1124–1131, 1974, doi: 10.1126/science.185.4157.1124.

[24] G. Klein, "Performing a project premortem," *Harvard Business Review*, Sep. 2007. [Online]. Available: https://hbr.org/2007/09/performing-a-project-premortem. Accessed: Feb. 22, 2026.

[25] Z. Ming, G. Sharma, J. K. Allen, and F. Mistree, "An Ontology for Representing Knowledge of Decision Interactions in Decision-Based Design," *Computers in Industry*, vol. 114, Art. no. 103145, 2020, doi: 10.1016/j.compind.2019.103145.

[26] B. M. Kennedy, D. K. Sobek, and M. N. Kennedy, "Reducing rework by

applying set-based practices early in the systems engineering process," *Systems Engineering*, vol. 17, no. 3, pp. 278–296, 2014, doi: 10.1002/sys.21269.

[27] B. Toche, R. Pellerin, and C. Fortin, "Set-based design: A review and new directions," *Design Science*, vol. 6, e18, 2020, doi: 10.1017/dsj.2020.16.

[28] S. C. Sommer and C. H. Loch, "Selectionism and learning in projects with complexity and unforeseeable uncertainty," *Management Science*, vol. 50, no. 10, pp. 1334–1347, 2004, doi: 10.1287/mnsc.1040.0274.

[29] J. Hao, L. Zhao, J. Milisavljevic-Syed, and Z. Ming, "Integrating and navigating engineering design decision-related knowledge using decision knowledge graph," *Advanced Engineering Informatics*, vol. 50, Art. no. 101366, 2021, doi: 10.1016/j.aei.2021.101366.

CONTACT INFORMATION

Ratna Babu Chinnam, Ph.D.
Founder & CEO
DAOSoft Corp
1352 Ross Lane
Rochester, MI 48306
rchinnam@daosoft.ai
248-819-2874
Linked In: www.linkedin.com/in/rchinnam/

ACKNOWLEDGMENT

This work was funded by the US Army Applied SBIR Program under contract W51701-23-C0188.

ACRONYMS

AI	Artificial Intelligence
BEV	Battery Electric Vehicle
BOM	Bill of Materials
CAE	Computer Aided Engineering
DADM	Decision Analysis Data Model
DE	Digital Engineering
DDT	Decision Digital Thread
DI	Decision Intelligence
DSS	Decision Support System
GSN	Goal Structuring Notation
GVSC	Ground Vehicle System Center
LLM	Large Language Model
MBSE	Model-Based Systems Engineering
NLP	Natural Language Processing
PLM	Product Lifecycle Management
RPC	Remote Procedure Call

APPENDICES: ILLUSTRATIVE SCHEMA EXAMPLES

The following JSON snippets are simplified illustrative examples designed to demonstrate the structural properties of key schema nodes described in the manuscript. They are not exhaustive representations of the full schema implementation.

Note: IDs shown are stable identifiers; objects may be revised through append-only supersession records rather than in-place mutation.

Appendix A: The "Decision" Node Structure (Framing and Scope)

This snippet demonstrates how a Decision object acts as a governable container: it defines scope boundaries (included vs excluded), references standardized objective definitions held in an enterprise registry and optionally carries drift-resilient references to external engineering artifacts.

```
{
  "id": "DEC_RANGE_EXT_STRATEGY_001",
  "kind": "Decision",
  "schemaVersion": "DDT-1.0",
  "name": "Select MY2027 Range Extension Strategy",
  "lifecycleState": "IN_ANALYSIS",
  "decisionClassPolicyId": "DCP_CONCEPT_STRATEGY_HIGH_STAKES",
  "createdAt": "2026-02-01T12:00:00Z",
  "createdBy": {
    "actorType": "user",
    "actorId": "user_pm_smith"
  },
  "context": {
    "programId": "PROG_MY2027_PICKUP",
    "parentNeedId": "NEED_RANGE_EXT_001",
    "decisionQuestion": "Which range-extension strategy best mitigates range anxiety while meeting cost
and schedule constraints?",
    "scope": {
      "included": [
        "Select technology strategy among A1, A2, A3",
        "Define target operating envelope for concept validation"
      ],
      "excluded": [
        "Detailed physical packaging integration (downstream decision)",
        "Detailed software calibration (downstream decision)"
      ]
    }
  },
  "valueModel": {
    "objectiveRefs": [
      {
        "objectiveDefinitionId": "OBJ_RANGE_ANXIETY_REDUCTION",
        "priority": "PRIMARY"
      },
      {
        "objectiveDefinitionId": "OBJ_DEV_LEAD_TIME",
        "priority": "PRIMARY"
      },
      {
        "objectiveDefinitionId": "OBJ_UNIT_COST_IMPACT",
        "priority": "SECONDARY"
      }
    ]
```

```
  },
  "optionSetId": "OPTSET_RANGE_EXT_001",
  "externalRefs": [
    {
      "system": "MBSE",
      "artifactUri": "mbse://repo/sysmlv2/models/vehicle_architecture",
      "baselineId": "BASELINE_SYSMLV2_2026-01-15",
      "linkValidity": "VALID"
    },
    {
      "system": "PLM",
      "artifactUri": "plm://part/vehicle_program/BOM/top",
      "baselineId": "PLM_RELEASE_R2026.01",
      "linkValidity": "SUSPECT"
    }
  ],
  "revision": {
    "rev": 1,
    "supersedesDecisionId": null,
    "changeReason": "Initial decision instantiation for concept phase"
  }
}
```


Appendix B: The Immutable "Evaluation Run" Record (Provenance)

This snippet illustrates an immutable execution record. It does not contain model data itself; instead, it captures the exact recipe used to generate evidence: evaluator identity and version, execution context, precise input bindings, and hashes of output artifacts. A sealing hash is shown to illustrate how append-only immutability can be enforced.


```
{
  "id": "RUN_ENGTHERM_MC_0001",
  "kind": "Evaluation Run",
  "schemaVersion": "DDT-1.0",
  "decisionId": "DEC_RANGE_EXT_STRATEGY_001",
  "optionSetId": "OPTSET_RANGE_EXT_001",
  "evaluatorDefinitionId": "EVAL_ENERGY_THERMAL_RPC",
  "evaluatorVersion": "1.2.0_certified",
  "execution": {
    "executedBy": {
      "actorType": "service",
      "actorId": "adapter_energythermal_01"
    },
    "executionMode": "HPC_BATCH",
    "containerImageDigest": "sha256:2c26b46b68ffc68ff99b453c1d30413413422d706483bfa0f98a5e886266e7ae",
    "startedAt": "2026-02-01T14:00:00Z",
    "endedAt": "2026-02-01T16:20:00Z",
    "status": "SUCCEEDED"
  },
  "inputs": {
    "scenarioSetId": "SCEN_USAGE_TEMP_TOW_V3",
    "parameterBindings": [
      {
        "parameterDefinitionId": "PAR_GENSET_NOMINAL_KW",
        "alternativeId": "ALT_A1_GENSET",
        "value": 35
      },
      {
        "parameterDefinitionId": "PAR_PACK_CAPACITY_KWH",
        "alternativeId": "ALT_A3_LARGER_PACK",
```

```
        "value": 160
      }
    ]
  },
  "outputs": {
    "artifacts": [
      {
        "uri": "s3://secure-repo/runs/RUN_ENGTHERM_MC_0001/results.parquet",
        "hash": "sha256:a948904f2f0f479b8f8197694b30184b0d2ed1c1cd2a1ec0fb85d299a192a447",
        "mediaType": "application/parquet",
        "description": "Primary simulation output dataset"
      },
      {
        "uri": "s3://secure-repo/runs/RUN_ENGTHERM_MC_0001/summary.json",
        "hash": "sha256:5c1d8c2dd4b2dbe9f1d4f7b1c0a51e91f9a2f6b6f5d0e9a0b4d2a98a1f2c3d4e",
        "mediaType": "application/json",
        "description": "Derived performance summary and objective scores"
      }
    ],
    "evidenceIds": [
      "EVD_ENERGY_THERMAL_0007"
    ]
  },
  "immutability": {
    "sealedAt": "2026-02-01T16:25:00Z",
    "sealedBy": {
      "actorType": "service",
      "actorId": "provenance_sealer_01"
    },
    "runRecordHash": "sha256:0f7c1d2f3a4b5c6d7e8f9a00112233445566778899aabbccddeeff0011223344"
  }
}
```


Appendix C: The "Decision Commitment" Record with Conditions

This snippet shows the separation of approval from analysis. It records who committed the decision and captures approval conditions as structured, machine-actionable data that can be tracked and enforced by downstream governance services.


```
{
  "id": "COMMIT_0001",
  "kind": "DecisionCommitment",
  "schemaVersion": "DDT-1.0",
  "decisionId": "DEC_RANGE_EXT_STRATEGY_001",
  "selectedAlternativeId": "ALT_A1_GENSET",
  "commitmentType": "CONDITIONAL",
  "status": "ACTIVE",
  "approvedAt": "2026-03-10T00:00:00Z",
  "approval": {
    "approvedByRole": "Chief Engineer",
    "approverUserId": "user_jdoe",
    "policyId": "DCP_CONCEPT_STRATEGY_HIGH_STAKES",
    "policyCheck": {
      "checkedAt": "2026-03-09T23:50:00Z",
      "result": "PASS"
    }
  },
  "conditions": [
    {
      "conditionId": "COND_NVH_01",
```

```
        "type": "metric_threshold",
        "metricDefinitionId": "MET_NVH_CAB_DB",
        "threshold": {
          "operator": "<=",
          "value": "Target_X_dB"
        },
        "verificationActionId": "ACT_NVH_VAL_01",
        "dueDate": "2026-06-30",
        "stopRule": "If threshold not met, this commitment is void and the decision returns to
IN_ANALYSIS."
      },
      {
        "conditionId": "COND_REG_01",
        "type": "required_review",
        "evidenceType": "regulatory_feasibility_memo",
        "verificationActionId": "ACT_REG_REV_02",
        "dueDate": "2026-09-30",
        "stopRule": "If review not completed by dueDate, downstream actions are blocked and the decision
returns to IN_ANALYSIS."
      }
    ],
    "authorizedActionIds": [
      "ACT_NVH_VAL_01",
      "ACT_REG_REV_02"
    ],
    "rationale": {
      "summary": "Best balance of range anxiety reduction vs schedule risk, provided NVH and regulatory
conditions are satisfied.",
      "keyTradeoffs": [
        "Lower range anxiety vs added integration complexity",
        "Schedule robustness vs long-term cost impact"
      ]
    },
    "revision": {
      "rev": 1,
      "supersedesCommitmentId": null,
      "changeReason": "Initial conditional commitment at concept gate"
    }
}
```

Appendix D: The "DecisionClassPolicy" Object (Readiness Gating Rules)

This snippet provides a minimal Decision Class Policy definition that operationalizes readiness gating: it encodes a lifecycle transition gate, a set of validation rules that must pass, explicit blocking behavior on failure, and authority boundaries that prevent AI agents from bypassing approvals.

```
{
  "id": "DCP_CONCEPT_STRATEGY_HIGH_STAKES",
  "kind": "DecisionClassPolicy",
  "schemaVersion": "DDT-1.0",
  "name": "Concept Strategy (High Stakes)",
  "readinessGates": [
    {
      "gateId": "GATE_IN_ANALYSIS_TO_PENDING_APPROVAL",
      "transition": {
        "from": "IN_ANALYSIS",
        "to": "PENDING_APPROVAL"
      },
      "proceedRule": "ALL_RULES_PASS",
      "onFail": {
```

```
        "action": "BLOCK_TRANSITION",
        "returnState": "IN_ANALYSIS"
      },
      "rules": [
        {
          "ruleId": "RULE_EVAL_COVERAGE",
          "type": "evaluation_run_coverage",
          "minimumValidRuns": 1,
          "coverageTarget": "PRIMARY_OBJECTIVES"
        },
        {
          "ruleId": "RULE_EVIDENCE_REVIEWED",
          "type": "evidence_review_status",
          "requiredStatus": "REVIEWED",
          "requiredReviewerRoles": [
            "Systems Engineer",
            "Program Manager"
          ]
        },
        {
          "ruleId": "RULE_BIAS_CHECKS_COMPLETE",
          "type": "required_actions_complete",
          "requiredActionDefinitionIds": [
            "ACTDEF_ANCHORING_CONTROL_REVIEW",
            "ACTDEF_PREMORTEM_WORKSHOP"
          ],
          "evidenceRequired": true
        }
      ]
    }
  ],
  "authority": {
    "canAdvanceToPendingApprovalRoles": [
      "Decision Owner",
      "Systems Engineer"
    ],
    "canCreateCommitmentRoles": [
      "Chief Engineer"
    ]
  },
  "agentPermissions": {
    "agentMayCreateKinds": [
      "Evaluation Run",
      "Evidence",
      "Action"
    ],
    "agentMayNotCreateKinds": [
      "DecisionCommitment"
    ],
    "agentMayNotChangeFields": [
      "Decision.lifecycleState"
    ]
  }
}
```