Digital Reasoning Thread: A Unified Reasoning Layer for Industrial Applications

Vlad Larichev
Industrial AI Lead
Accenture Industry X
Aachen, Germany
vlad@example.org

***Abstract*—Industrial systems often record what happened, not why it happened. The Digital Reasoning Thread (DRT) proposes a persistent reasoning layer that connects context, assumptions, logic, and outcomes across engineering, manufacturing, and operations. DRT complements the digital thread by preserving rationale and intent across tools and teams, enabling traceable and auditable decisions for humans and AI agents. This paper introduces DRT, outlines a reference architecture, discusses interoperability with existing standards and protocols, and highlights early use cases and evaluation metrics.**

***Index Terms*—Industrial AI, Digital Thread, Reasoning, Agentic AI, Manufacturing, Supply Chain, PLM, MES, ERP, UNS, OPC UA, MQTT, MCP, A2A, Governance, Traceability, Explainability, Knowledge Graphs, Vector Search**

Version 0.1 - August 2025 - License CC-BY 4.0

I. Introduction

Most industrial systems record what happened - not why it happened.

The industrial value chain can be viewed as a continuous flow of materials over time, shaped by both manufacturing processes and human decisions. As these flows move from engineering to production to service, they traverse many organizational and technical layers.

Each layer - PLM, MES, ERP, CMMS, simulation tools, document systems - operates in its own domain, with its own language, owners, syntax, and logic for representing reality.

As a result:

- Assumptions are lost at handovers
- Analysis is repeated across teams
- Audits become slow and incomplete
- Trust in AI outputs remains low

We introduce the Digital Reasoning Thread (DRT) - a persistent, cross-domain logic layer that connects assumptions, constraints, evidence, decisions, and outcomes into a consistent flow. The thread is designed to be used by both humans and AI agents, enabling more transparent, explainable, and traceable industrial systems.

II. The Digital Thread - Current State After 20 Years

The Digital Thread emerged in aerospace and defense and later spread to automotive, energy, and other complex industries. Its ambition was end-to-end traceability - linking requirements, system models, design artifacts, manufacturing steps, and field service events across the lifecycle. The goal was to maintain a unified view of product evolution.

Advances in PLM (Product Lifecycle Management) and MBSE (Model-Based Systems Engineering) provided structure and governance. Vendors and standards connected CAD, BOM systems, MES, ERP, and CMMS into a more coherent ecosystem. Digital twins benefited by referencing the thread for historical context.

Today, many firms run some form of digital thread supporting:

- Change impact analysis
- Variant and configuration management
- Compliance and certification tracking
- Faster onboarding and reuse of design components

Limitations:

- Strong for structured data like parts, systems, BOMs, workflows
- Weak for rationale that lives in unstructured artifacts like emails, slides, notes, tickets, and prompts
- Assumptions get lost at handovers and context fragments across teams
- Analyses repeat, misunderstandings grow, decision latency increases

AI adoption makes this gap more visible. Models and agents act, but the reason behind actions can be hard to audit. Knowledge graphs and Unified Namespace (UNS) architectures improve connectivity and semantics, yet reasoning remains implicit - hidden in scripts, notebooks, and human memory. This is the gap DRT aims to close.

III. Introducing Digital Reasoning Thread (DRT)

> DRT is a persistent layer that carries context and rationale across data, tools, and decisions. It turns reasoning into an asset that can be traced, audited, and reused.

Definition. The Digital Reasoning Thread is a persistent, context-aware, and auditable representation of the reasoning artifacts that drive industrial decisions across systems, agents, and humans.

Goal. Connect perception, knowledge, reasoning, simulation, execution, and feedback.

Layers.

- **Perception** collects signals from sensors, logs, images, and documents.
- **Knowledge** organizes schemas, ontologies, and catalogs.

TABLE I
Pattern mapping: where each protocol fits

Layer	Primary choice	Alternatives
Telemetry	UNS, OPC UA	REST, gRPC
Edge messaging	MQTT	NATS, Kafka
Tool control	MCP	Custom REST
Agent-to-agent	A2A	gRPC, WebSockets
Audit trail	Event store	Data lake

- **Reasoning** captures goals, assumptions, constraints, plans, and traces.
- **Simulation** explores what-if scenarios and digital twins.
- **Execution** applies plans in MES, PLCs, robots, and enterprise systems.
- **Feedback** compares intent and outcome and updates context.
- **Governance** spans all layers with identity, policy, lineage, and safety rails.

IV. Interoperability

DRT aligns with existing standards and buses.

Data and events. UNS and OPC UA for structured telemetry and hierarchy; event logs for state transitions and process mining.

Tool control. MCP to expose tool capabilities to reasoning agents; A2A for agent discovery, identity, and task negotiation.

Edge and shopfloor. MQTT topics for low-latency publish-subscribe at cells with defined QoS and retention policies.

V. Typical Use Cases

A. Engineering handover

- Input: constraints from Polarion, mesh features from HyperMesh, BoM from PLM
- Reasoning: check compliance, simulate risk, propose parameter windows
- Action: generate MES work instructions, set safe parameter ranges
- Outcome: traceable handover and faster ramp-up

B. Predictive maintenance with scheduling

- Input: time-series via UNS, maintenance logs, shift plan
- Reasoning: predict failure windows, optimize downtime slot
- Action: create work order and material pick list
- Outcome: lower unplanned downtime and fewer last-minute changes

C. Supply chain exception management

- Input: supplier ETA change, inventory, order priorities
- Reasoning: evaluate options, simulate penalties and service levels
- Action: replan allocations and trigger customer updates
- Outcome: consistent decisions and audit-ready explanations

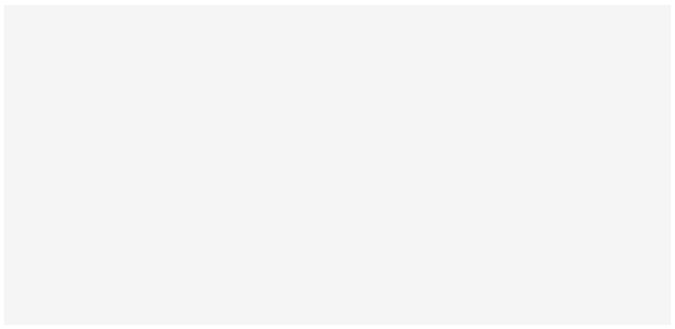

Fig. 1. Reference stack and flows

VI. Reference Architecture

The DRT architecture separates concerns and enables governance.

VII. Get Involved

The DRT concept is evolving through research, experimentation, and industry feedback. We are shaping its architecture, vocabulary, and applications across domains.

If you are interested in contributing, validating use cases, or collaborating on implementation patterns:

- Open a discussion or issue on GitHub: link(“https://github.com/vlarichev/digital-reasoning-thread”)
- Fork the repository to propose ideas, diagrams, or examples
- Submit pull requests for improvements or extensions
- Connect on LinkedIn to exchange ideas or explore joint opportunities: link(“https://linkedin.com/in/vlarichev”)

References