MANTECH
ALWAYS ADVANCING

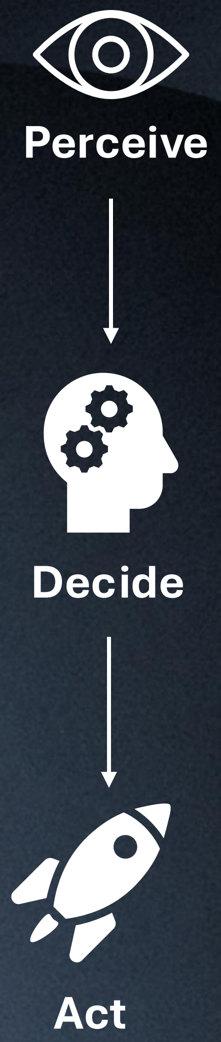
A Modular Agent-Based Architecture for Digital Engineering

Nicole Manno

Data Scientist

September 17th, 2025

Agents are having a ***moment***



Perceive

Decide

Act

- Industry is piloting end-to-end agent features
- Analysts and roadmaps highlight ‘agentic’ apps
- Open-source frameworks lowered the barrier (AutoGen, LangChain)

Impact depends on the boundary where agents meet data

How Agents Become Brittle

Today

- ✗ One-off scripts tied to tool/export/API
- ✗ Small differences stack up when chaining
- ✗ Weak governance

Boundary-first

- ✓ One-off scripts tied to tool/export/API
- ✓ Small differences stack up when chaining
- ✓ Weak governance

Technology Agnostic Integration Architecture

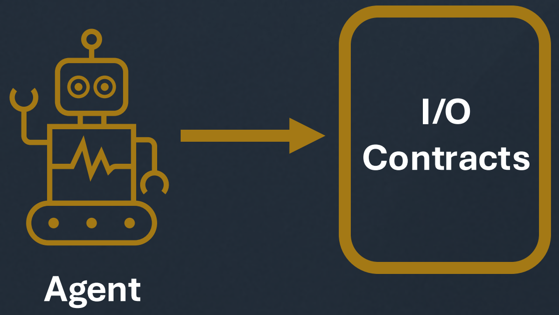
Commitment 1

Meaning-First Access

Agents connect by meaning, not data shape

Commitment 2

Self-Contained Agents



Agent

I/O Contracts

Ingest ⟶ Process ⟶ Publish

What this buys you

- **Plug-in agents** – no parsers per tool
- **Safe composition** – contracts + checks
- **Governance by default** – run-level provenance

What we Borrowed vs. What we Added

Multi-Agent Coordination

Blackboard, OAA, Cougar

Borrow

- Modular agents, distributed problem solving

Add

- Ontology-typed I/O contracts
- Design-time schema checks

Semantic Interoperability

Smart-M3

Borrow

- Concept-level IDs, not files

Add

- Apply to digital engineering
- Bind concepts to agent contract

LLM Agent Toolkits

LangChain/LangGraph

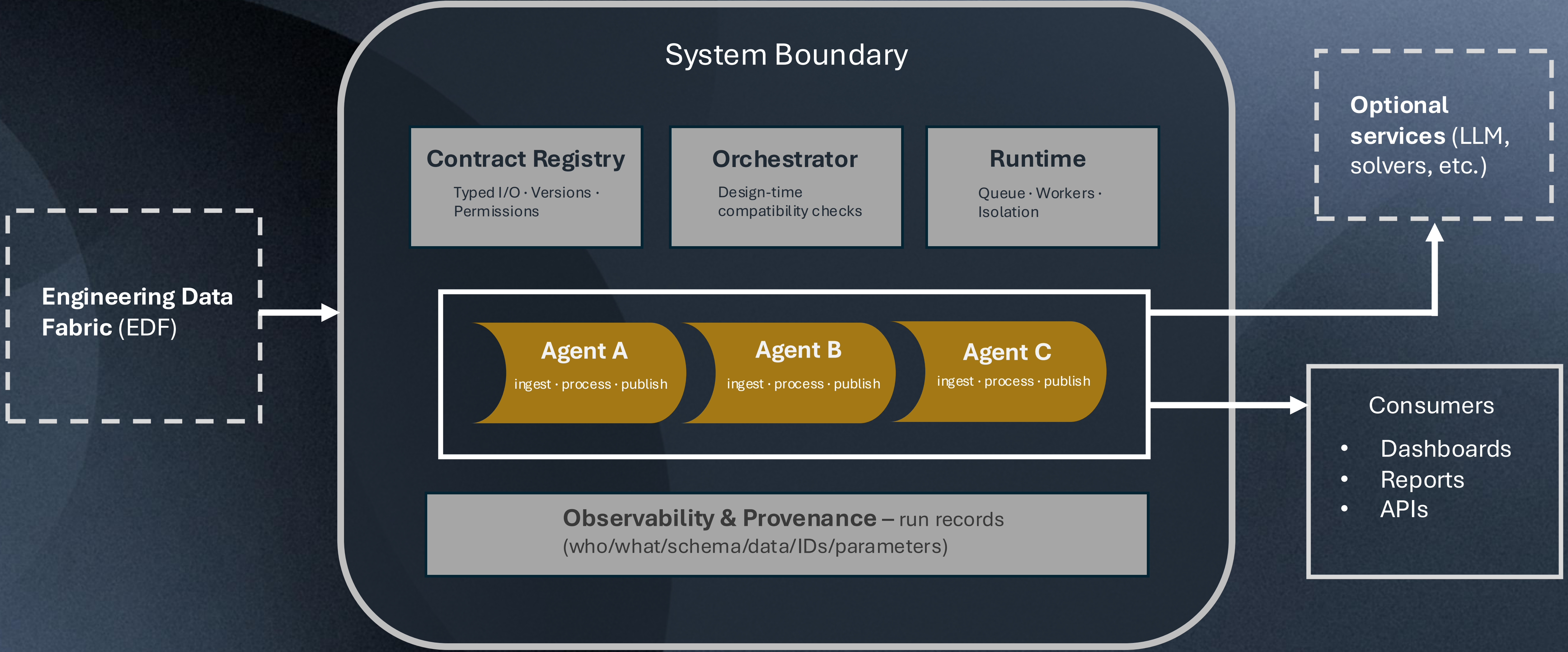
Borrow

- Graphs, multi-agents, HIL controls

Add

- Publish-time schema gating
- Ontology-tied I/O between agents

Boundary-First, Technology-Agnostic


System Boundary
Contract Registry
Typed I/O · Versions · Permissions
Orchestrator
Design-time compatibility checks
Runtime
Queue · Workers · Isolation
Optional services (LLM, solvers, etc.)
Engineering Data Fabric (EDF)
Agent A
ingest · process · publish
Agent B
ingest · process · publish
Agent C
ingest · process · publish
Consumers
- Dashboards
- Reports
- APIs
Observability & Provenance – run records (who/what/schema/data/IDs/parameters)

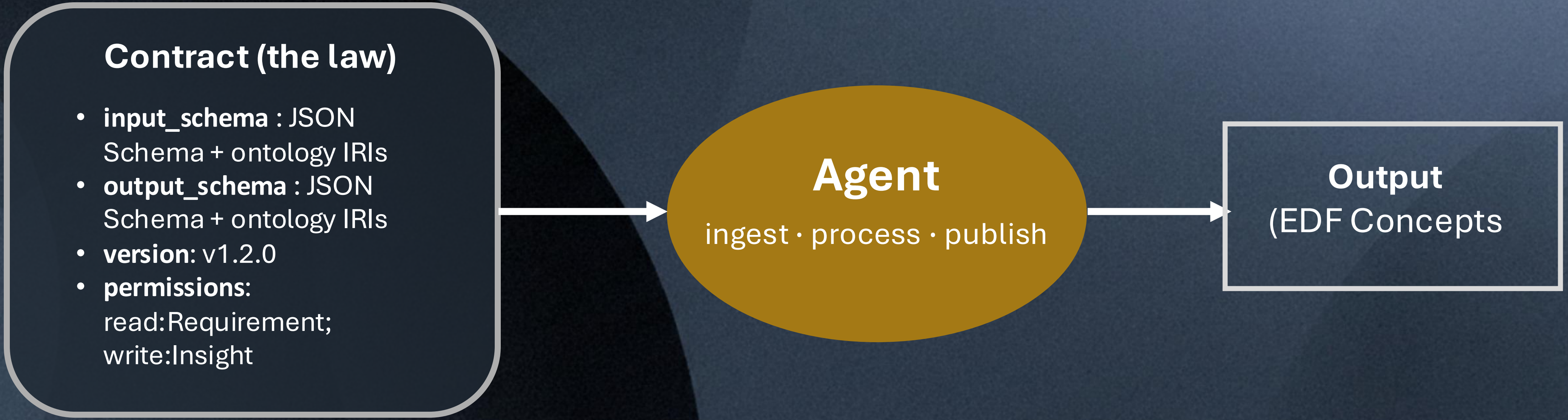
Engineering Data Fabric (EDF)

Ontologically Backed Digital Thread
A digital thread that leverages ontologies to semantically integrate and manage data across the product lifecycle

The Importance

The EDF is designed to streamline engineering activities by reimagining processes and removing data access roadblocks aiding decision making during system development

Agents as Black Boxes



Contract (the law)

- **input_schema** : JSON Schema + ontology IRIs
- **output_schema** : JSON Schema + ontology IRIs
- **version:** v1.2.0
- **permissions:** read:Requirement; write:Insight

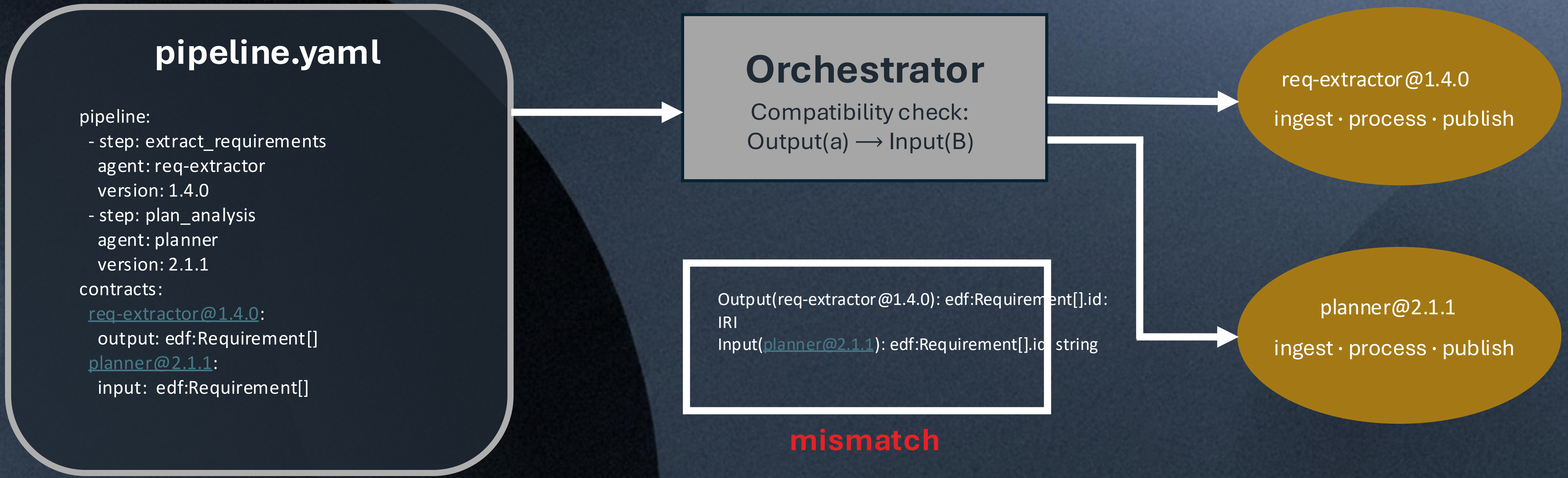
Agent

ingest · process · publish

Output
(EDF Concepts

Agents are swappable if they honor the contract

Contracts & Orchestrator



pipeline.yaml

```
pipeline:
  - step: extract_requirements
    agent: req-extractor
    version: 1.4.0
  - step: plan_analysis
    agent: planner
    version: 2.1.1
contracts:
  req-extractor@1.4.0:
    output: edf:Requirement[]
  planner@2.1.1:
    input:  edf:Requirement[]
```

Orchestrator

Compatibility check:
Output(a) ⟶ Input(B)

req-extractor@1.4.0
ingest · process · publish

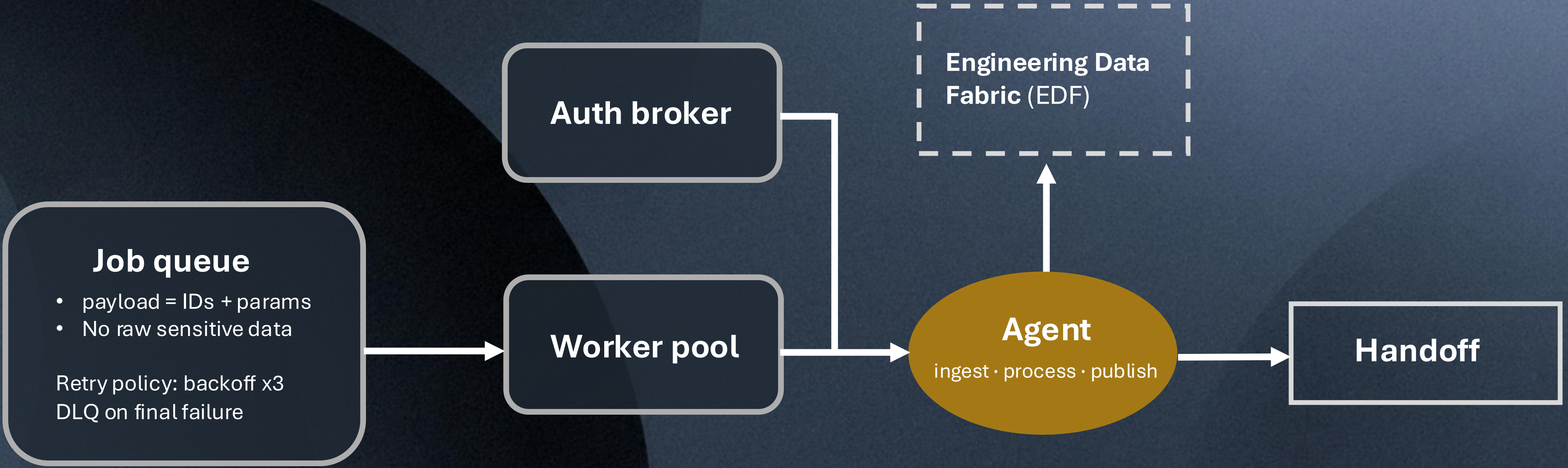
planner@2.1.1
ingest · process · publish

Output(req-extractor@1.4.0): edf:Requirement[].id: IRI
Input(planner@2.1.1): edf:Requirement[].id: string

mismatch

Pipelines declare steps + pinned versions. Orchestrator validate schemas at publish time

Runtime – Queues, Isolation, Short-Lived Tokens



Provenance (run record): who, agent image digest, contract versions, EDF concept IDs, timestamps, etc.

Governance – Observability & Provenance

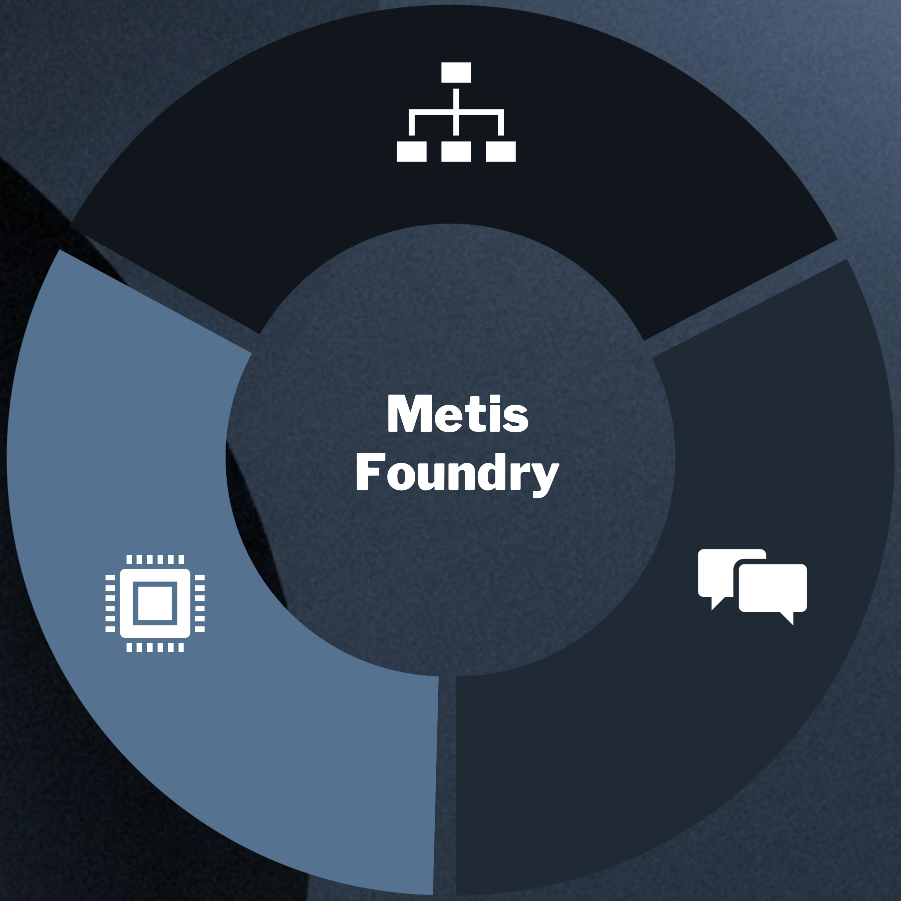
Provenance Record

- run_id
- triggered_by
- agent_image
- contract_versions
- edf_reads
- edf_writes
- params
- started_at
- ended_at
- status

What this enables

- **Audit instantly**: who/what/data/when
- **Reproduce**: re-run same image + inputs
- **Compare**: drift checks across versions

Example Usage: Metis Foundry



Capability Optimization

Identify gaps and overlaps of capabilities within a system architecture; provide reallocation based on optimization function

Obsolescence Management

System searches through user uploaded artifacts and the web to extract end-of-life dates and recommended replacement parts

Digital Thread Chatbot

Ask natural language questions about integrated data in a digital thread or a data fabric

Conclusion – *Standardize the Boundary*

- **Meaning-first access**: agents read/write concepts from the EDF
- **Black-box agents**: implement ingest, process, publish with typed I/O contracts
- **Safe composition**: design-time checks block mismatches
- **Predictable runtime:** queues, isolation, short-lived tokens
- **Governance by default**: run-level provenance

Contracts · Checks · Runtime · Provenance

Thank you!

For more information, contact us:

Nicole Manno | nicole.manno@mantech.com

Dr. Douglas Orellana | douglas.orellana@mantech.com

Patrick Keen | william.keen@mantech.com

References

1. Nii, H. P. (1986). **The blackboard model of problem solving**. *AI Magazine,* 7(2), 38-53.
2. Maritn, D. L., Cheyer, A. J., & Moran, D. B. (1999). **The Open Agent Architecture: A framework for building distributed software systems**. *Applied Artificial Intelligence,* 13(1-2), 91-128.
3. Honkola, J., Laine, H., Brown, R., & Tyrkko, O. (2010). **Smart-M3 information sharing platform**. In *2010 IEEE Symposium on Computers and Communications (ISCC).* IEEE.
4. Mavroudis, V. (2024). **LanChain v0.3** (preprint). *Preprints.org.*