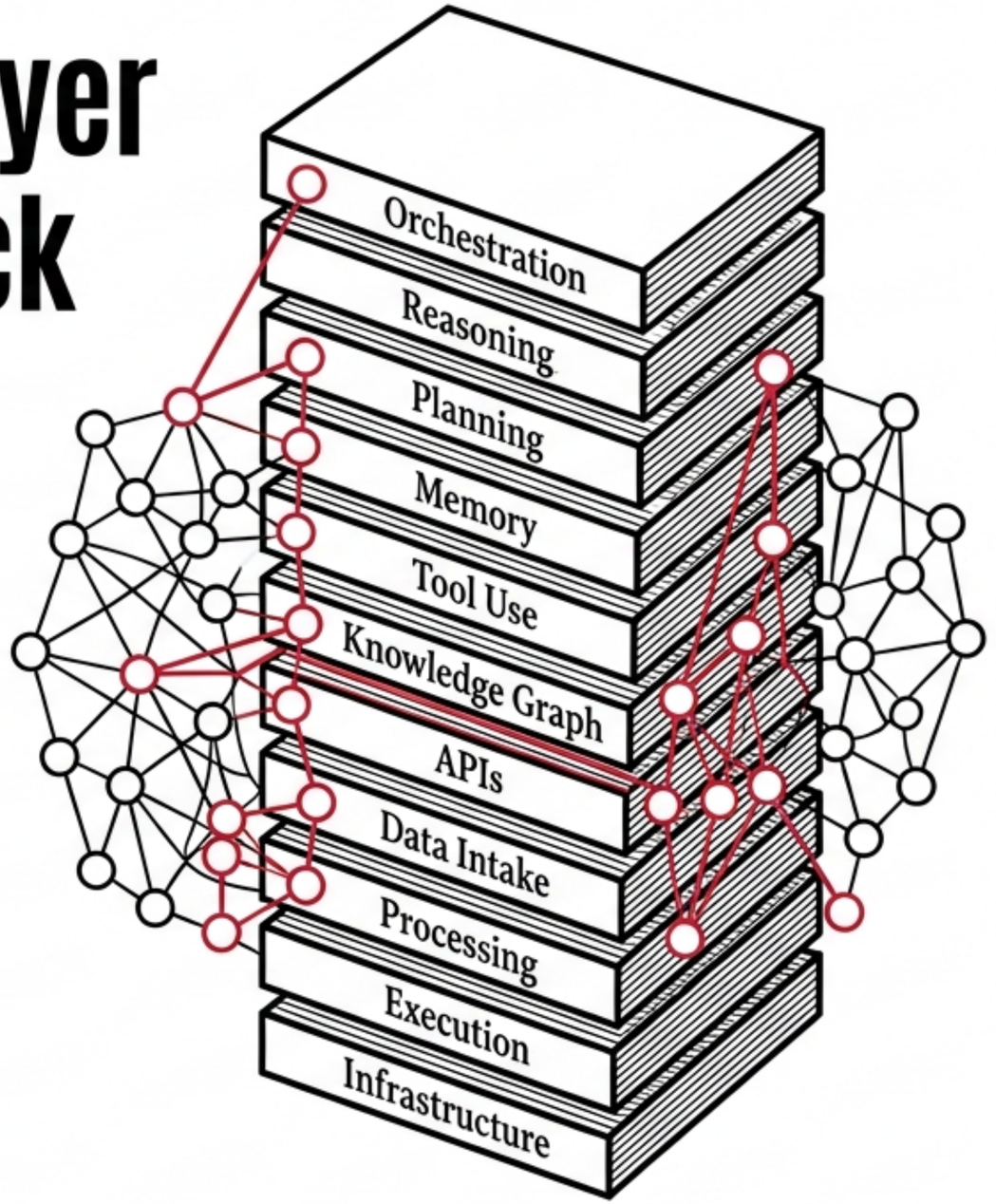
How We Mapped the 11-Layer AI Agent Engineering Stack to a Live RDF Codebase A LivE RDF

An empirical analysis of agent-rdf-memory: 11 Layers, 140+ Steps, and 25 Verified Links

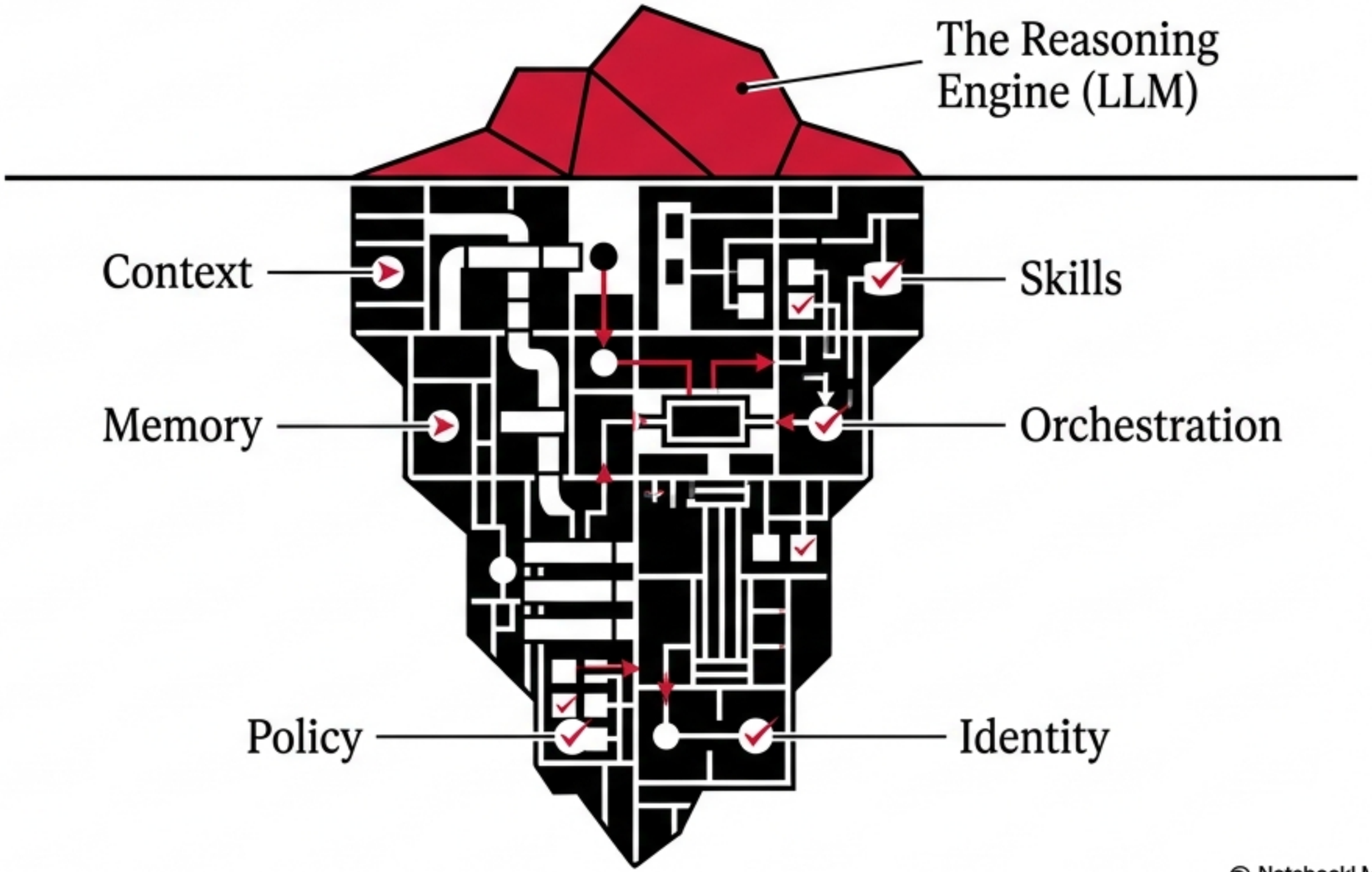


Original stack model by Brij Kishore Pandey. Knowledge Graph curated by OpenLink Software via Claude Sonnet 5, document-to-kg-skill, and rdf-infograph

THE REASONING MODEL IS ONLY 10% OF AN AI AGENT. THE OTHER 90% IS ENGINEERING.

What happens when an LLM redraws an agent architecture diagram—and then has to defend its claims against a real GitHub repository?

This analysis proves that a reliable AI agent operating environment is built on purpose-specific Semantic Web engineering, not just API calls."



The Implementation Ledger: Scoring the Stack Against Reality

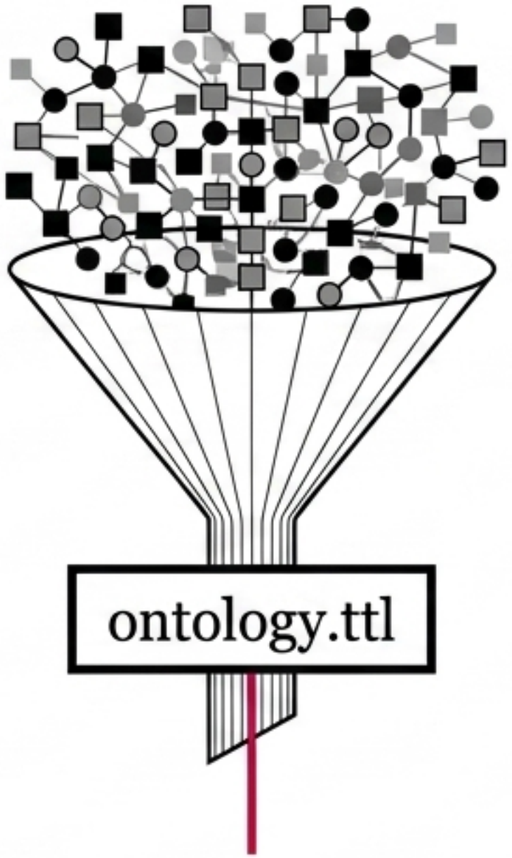
Layer	Status	Code Artifacts
1. Model	✗	Out of Scope (External)
2. Context	✓	`ontology.ttl`
3. Memory	✓	`sessions/` , `entities/`
4. Tools	○	`scripts/` , `load_memory.py`
5. Skills	✓	`howto/`
6. Orchestration	✓	`README.md` , `AGENTS.md`
7. Identity	✓	`core.ttl`
8. Policy & Guardrails	✓	`preferences.ttl`
9. Observability	○	`sessions/` , `index.ttl`
10. Evaluation	✓	`scripts/`
11. Runtime	○	`load_memory.py`

7 layers fully implemented via **RDF-Turtle rules.** 3 layers defer execution to the host harness. **1 layer (Reasoning)** is **strictly external.**

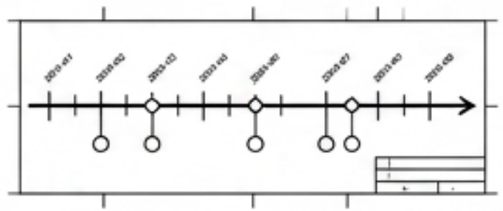
Layers 2 & 3: Giving the Agent Contextual Awareness and Perfect Recall

Layer 2: Context

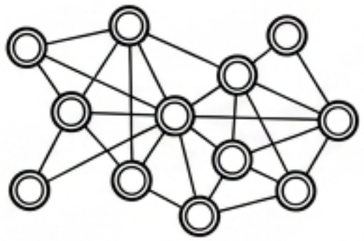
Prompt-Intent Classes



Layer 3: Memory



Episodic: sessions/
(YYYY-MM-DD format)

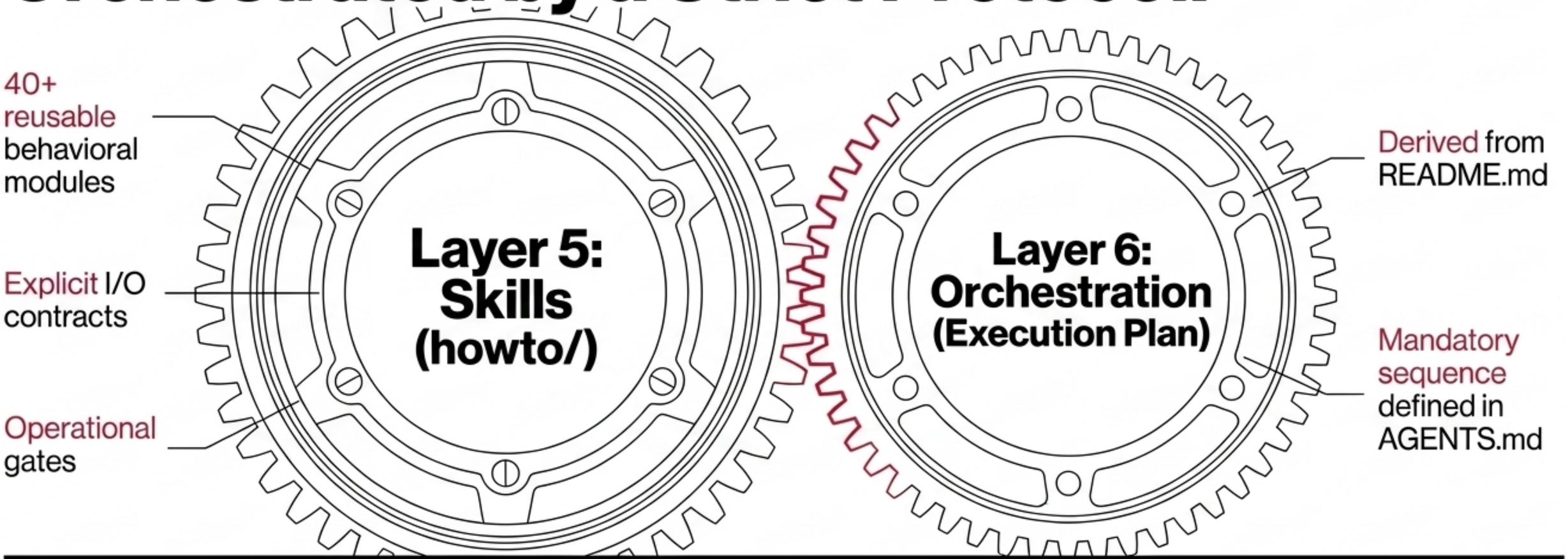


Semantic: entities/
(people, tools, concepts)

Behavioral:
preferences.ttl
(16-step sub-HowTo)

Context is treated as a strict relevance budget, not a full data dump. The memory layer defines episodic, semantic, and long-term behavioral recall using Linked Data principles

Layers 5 & 6: 40+ Composable Abilities Orchestrated by a Strict Protocol.



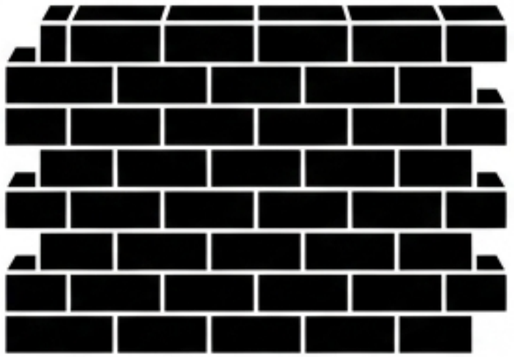
The howto/ folder serves as a comprehensive skill library. Every composable ability is governed by a mandatory retrieval and orchestration plan; the repository does not allow allow improvised execution.

Layers 7, 8 & 10: Fail Closed, Not Open

Identity

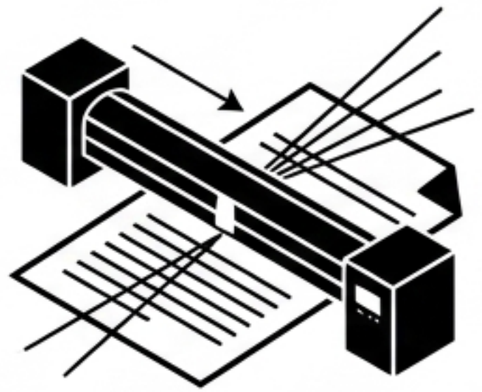
core.ttl
(WebID & openssl certificate-modulus checks)

Policy



preferences.ttl
(140+ HowToSteps as hard blocking gates with public/private boundary)

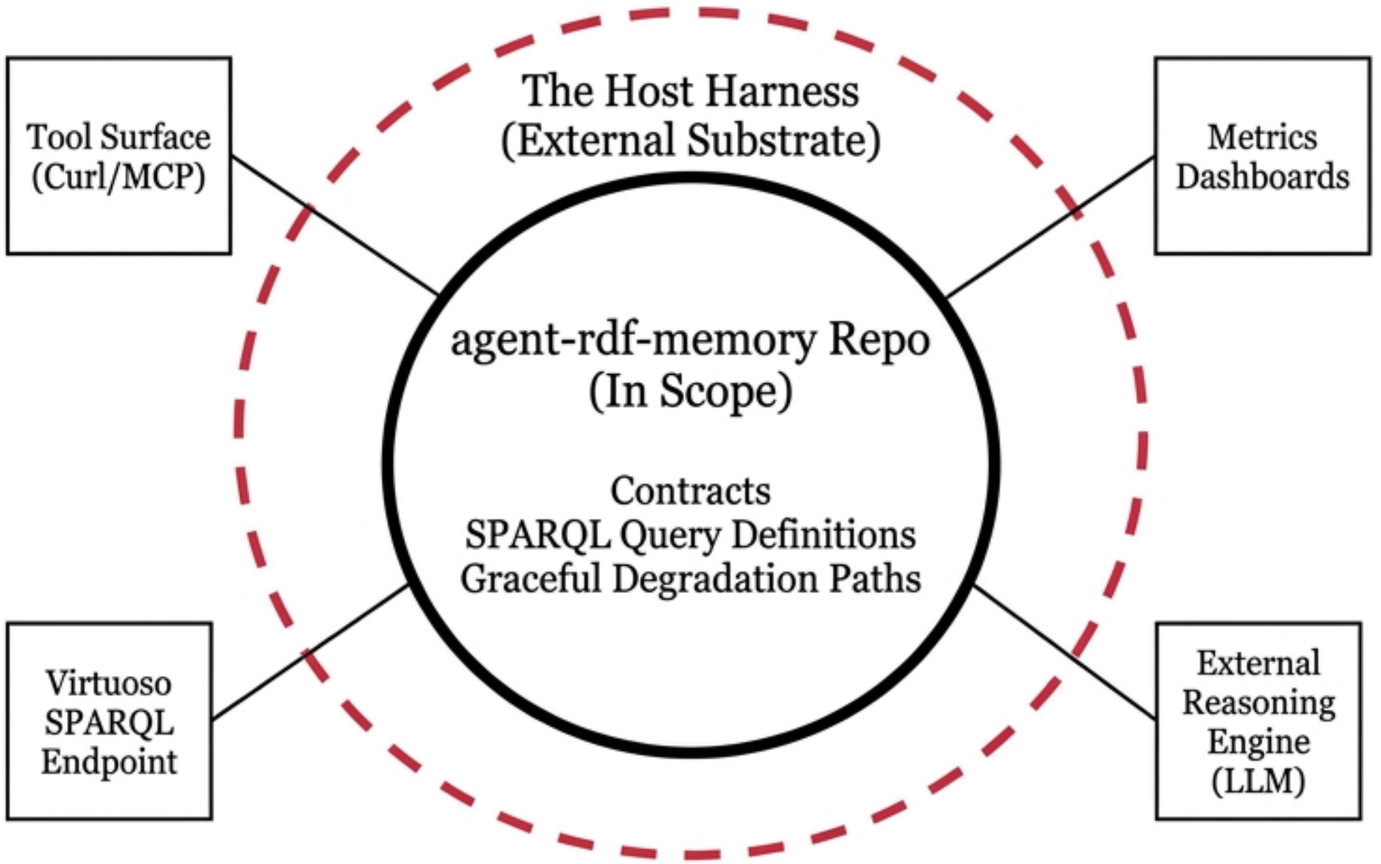
Evaluation



scripts/
(Post-session continuous audit scripts for Turtle-validity)

"Continuous evaluation is live-tested: A July 2026 edit to preferences.ttl was is live-tested: A July 2026 edit to preferences.ttl was blocked by the system until it fully complied with the policy layer."

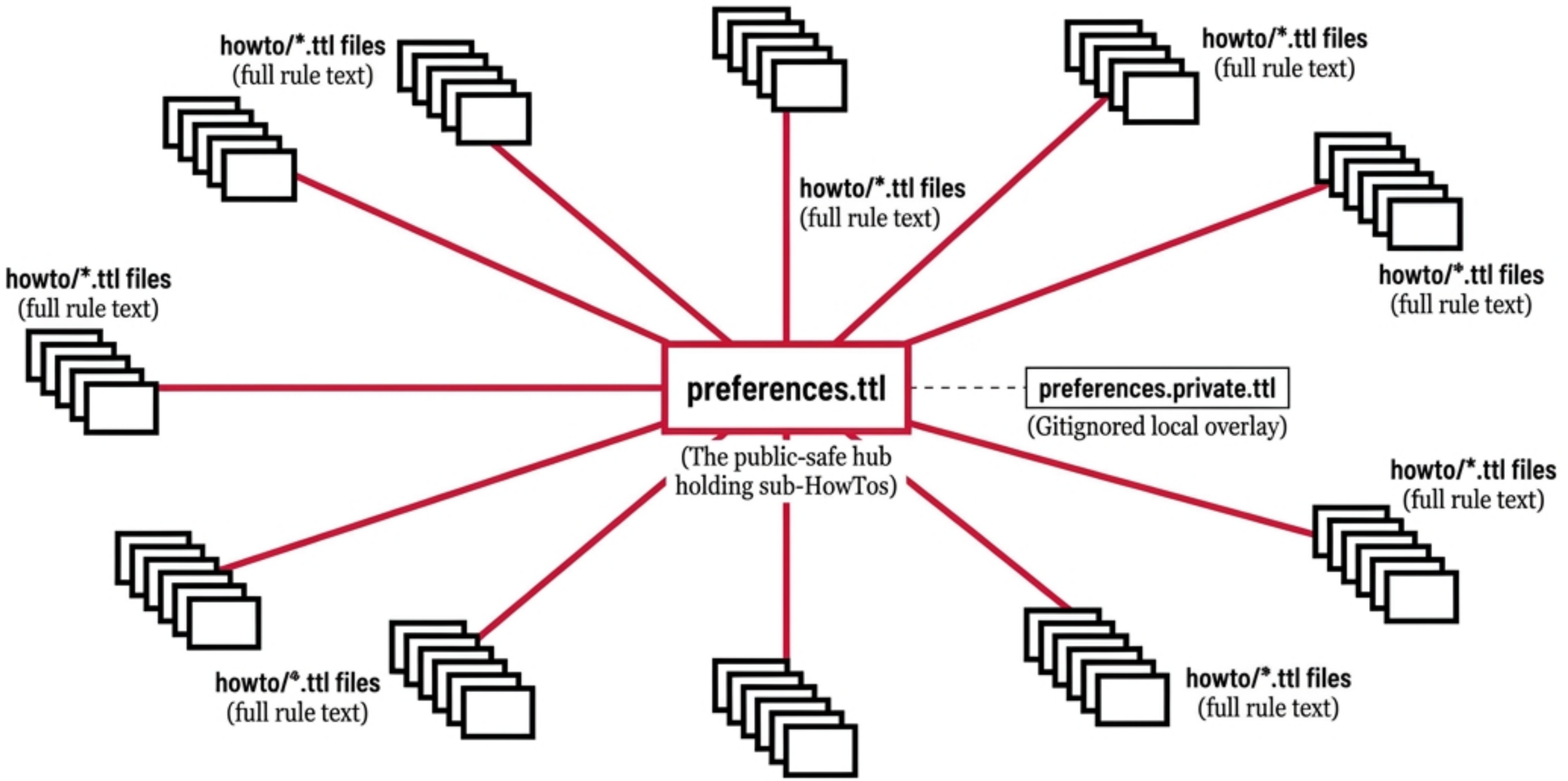
The Boundary Map: What Lives Outside the Repository.



Why are Tools, Observability, and Runtime only partially covered? Because the repository defines the behavioral contracts, but the execution substrate lives entirely in the host harness.

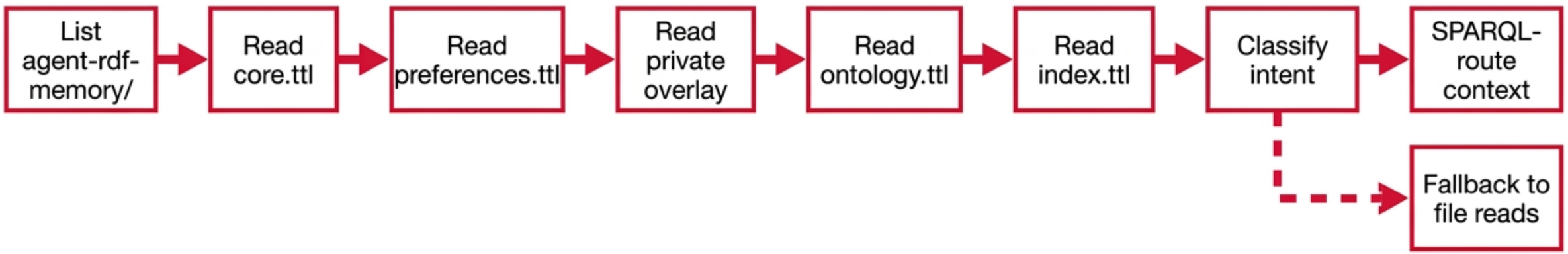
The Reasoning Model itself is strictly out of scope—the repo identifies and routes to the LLM, but the reasoning is always external.

The Hub-and-Spoke Architecture of an RDF Agent



preferences.ttl acts as the central hub, utilizing sparse pointers (name + description). All rationale, code, gates, and incident notes are strictly isolated in the companion howto files.

The Mandatory Session-Start Retrieval Protocol



Before any task begins, the agent must execute this exact pathway. If the ontology-routed SPARQL context selection fails, the system enforces a strict fallback to direct file reads.

The Methodology of Truth: A 5-Step Verification Chain

1. Read Core Docs First
(core.ttl, AGENTS.md, README.md)

2. Map Layers to Files
Classify 11 stack layers against repository paths.

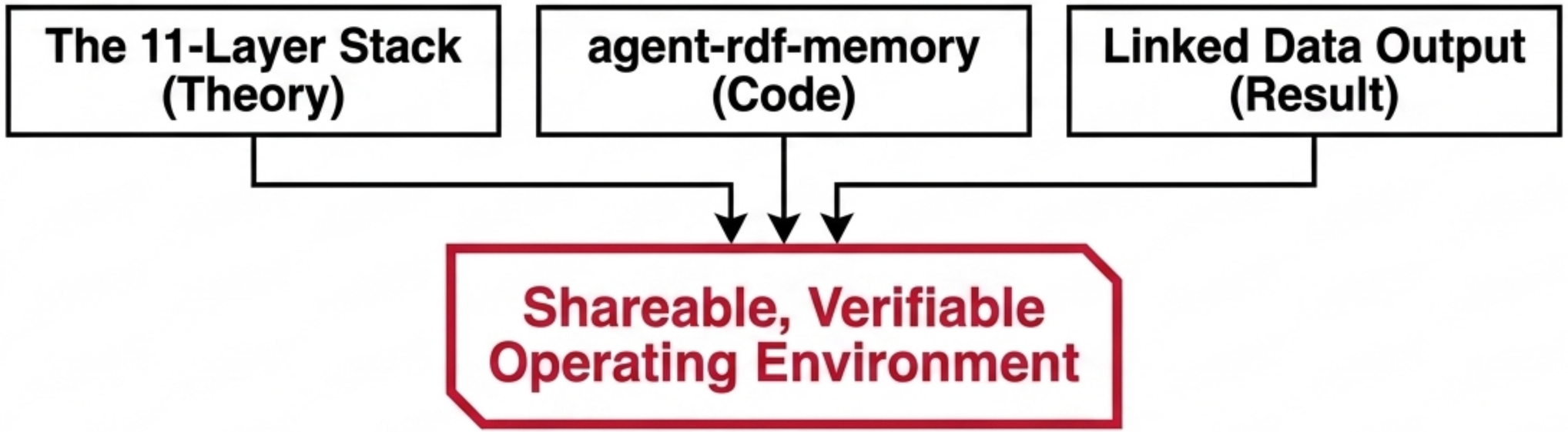
3. Verify Links via API
Fetch full file tree via GitHub Git Trees API.
Reject non-existent/gitignored paths.

4. Link Inline, Once
Scan prose in reading order; link first occurrence only.

5. Generate Linked Data
Transform verified content into RDF.

We didn't guess paths. Every link mapped in this analysis was verified against the live GitHub repository tree on July 21, 2026, before publication.

The Blueprint for a Purpose-Specific Semantic Web



By enforcing Linked Data principles and a strict 11-layer architecture, agent-rdf-memory replaces flat markdown memory stores with queryable behavioral contracts. The result? Better environments yield infinitely better agents.

Review the companion RDF data at agent-rdf-memory-stack-alignment-inline-links.ttl or visit the OpenLink Software GitHub repository to explo