

WORK THAT SHIPS

— FIELD GUIDE • AGENTIC ENGINEERING

The workspace is the product.

Models are rented. Harnesses are swappable. What compounds is the place they work in: its instructions, memory, procedures, structure, and the colleagues beside them.

• SEVEN ACTS • ABOUT FORTY-FIVE MINUTES • PRESS → TO BEGIN, G FOR THE GUIDE

FILES

MEMORY

WORKSPACE

SKILLS

AGENTS

— THE THESIS

Give the agent a filesystem and a shell.

A model guesses when it can't look. A workspace lets it look, run, and write down what it found.

Chat only

Filesystem + shell

Explicit. Every input is a file you can open, diff and cite.

Open. Bash runs whatever the harness allows: search, tests, scripts.

Durable. What it learns is written back, so the next run starts ahead.

Structure and philosophy decide how it works there. That is harness engineering.



— THE OPERATING MODEL

Seven questions. One system.

Each layer answers one question an agent asks before it can do useful work.

01

Workspaces

Where does it live?
Primer, policy, record
and sandbox.

02

Context

What reaches the
window, and how
hooks make recall a
reflex.

03

Second brain

How the graph is
structured, read and
grown.

04

Skills

How we do this kind
of work, loaded on
demand.

05

Structure

Which bytes it may
touch, and where
they go.

06

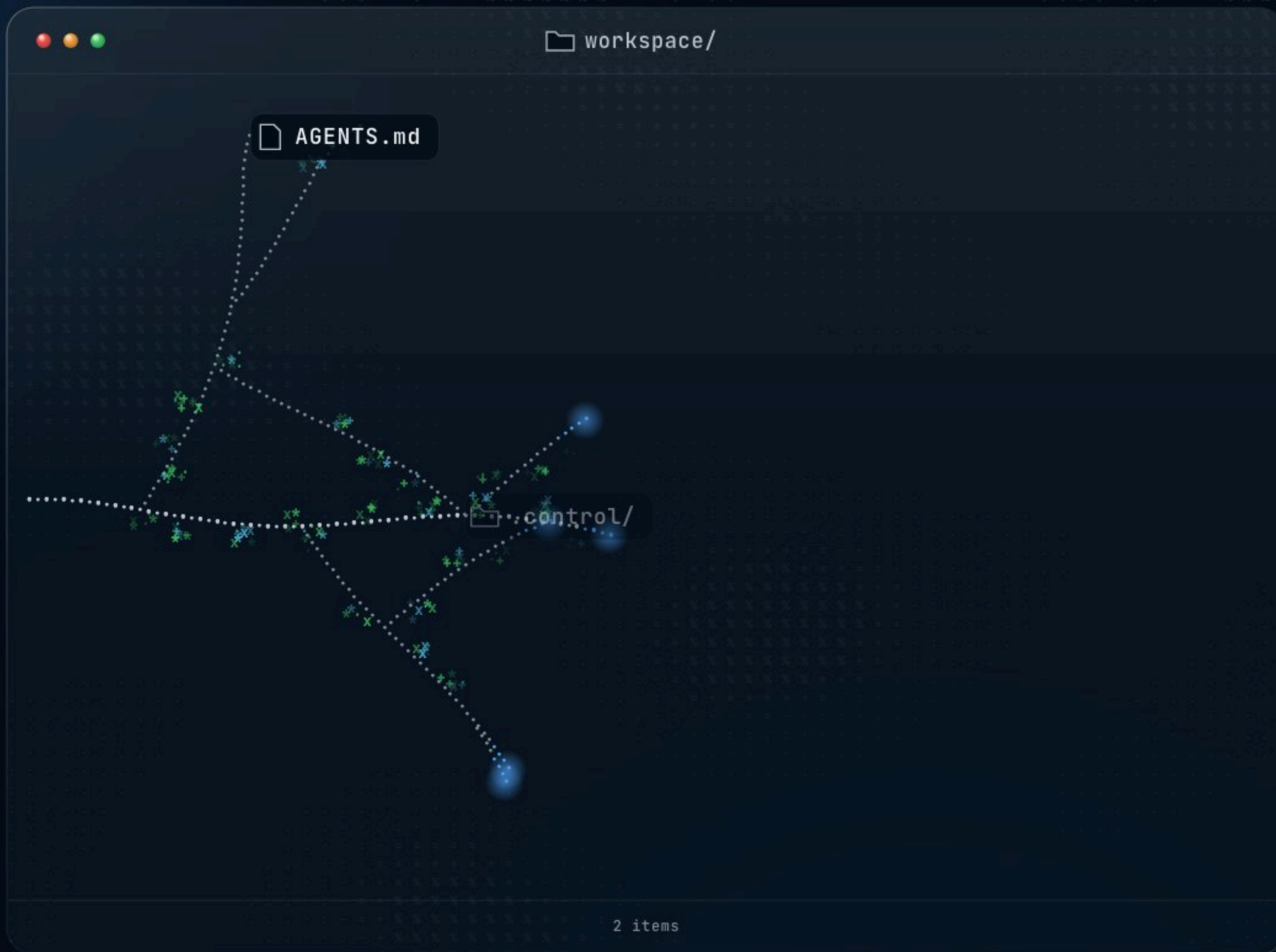
Fleets

Who else is here, and
who checks whom.

07

Leadership

What all of this asks
of people.



ACT I

Workspaces

Where does the agent live?

A folder is not a workspace.

The difference is what survives the session.

Agent workspace

Agentic workspace

● AGENT WORKSPACE

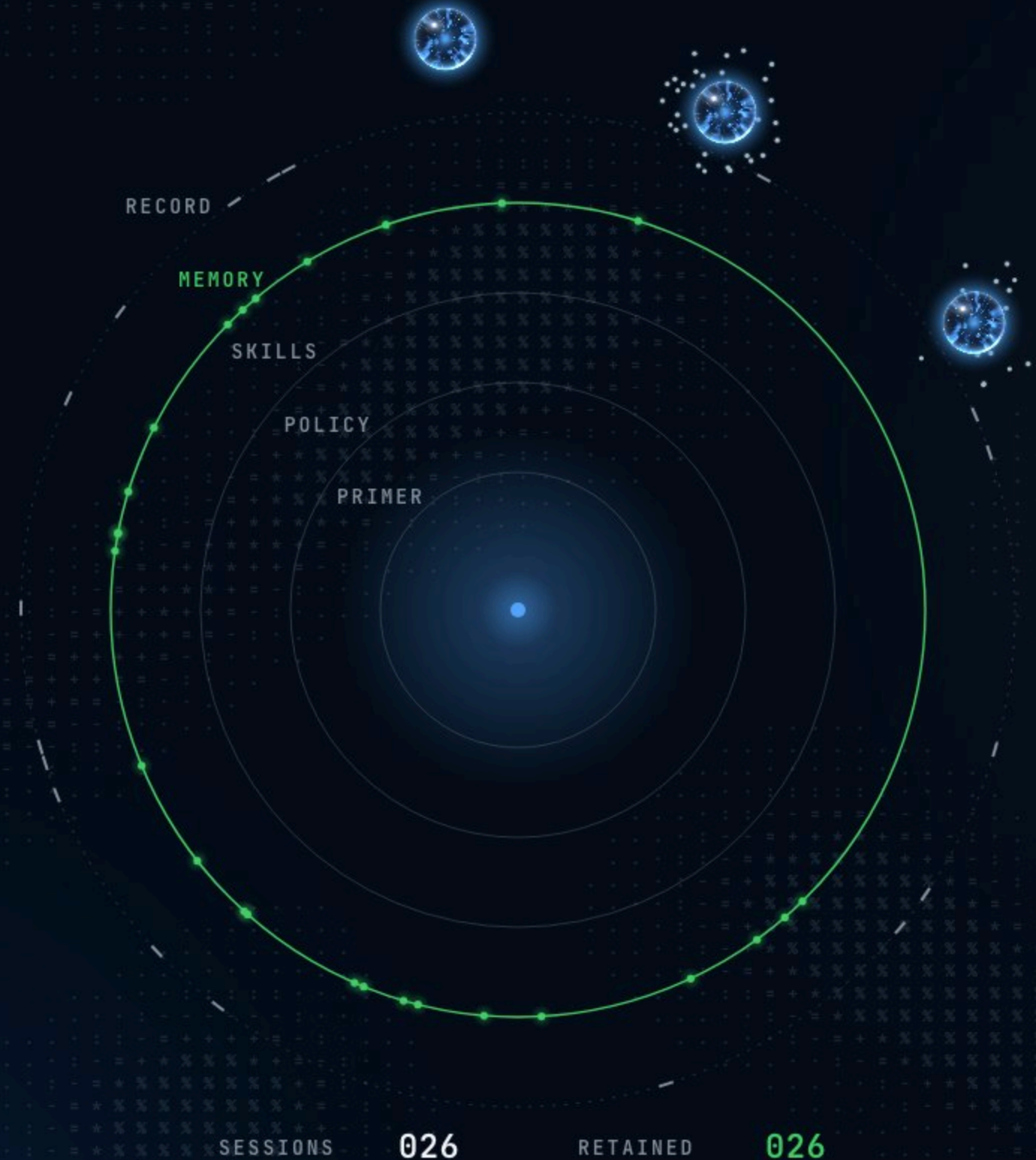
A working directory and a toolset for one agent on one task.

Useful. But context evaporates when the session ends, and the next agent starts cold.

● AGENTIC WORKSPACE

A governed place that many agents work through.

Instructions, policy, memory, skills and a record outlive every session, so each run leaves the place a little smarter.



Six layers an agent reads before it writes.

● LAYER 01

Primer

AGENTS.md ≡ CLAUDE.md

Philosophy, routes and hard boundaries. One file, byte-identical copies for every harness.

RULE Route, don't dump. Point to protocols instead of inlining them.

READ ORDER ↓

01 Primer

AGENTS.md ≡ CLAUDE.md

02 Policy

.control/policy.yaml

03 Skills

skills/<category>/<name>/

04 Knowledge

research/entities/ · knowledge-index.md

05 Record

docs/conversations/ · .control/initiatives/

06 Work

repos · worktrees · sandboxes

WRITE PATH ↑

— I · WORKSPACES

Sandbox, harness, wire.

Three layers, three different answers. Buy the box, swap the brain, and don't bet on the wire yet.

Claude Code

Codex

Gemini CLI

Your own

UNSOLVED

Wire

phone · desktop · chat



ACP: stdio-first · remote transport in draft

SWAP

Harness

the agent loop



your own loop

BUY

Sandbox

microVM · persistent volume

microVM

/workspace (volume)

exec · files · tools

MCP SERVER + CLI

the integration point never moves when the harness does

Medallion layers live in the filesystem.

Each workspace keeps its own bronze, silver and gold as plain folders. A shared ledger records what was decided.

One workspace

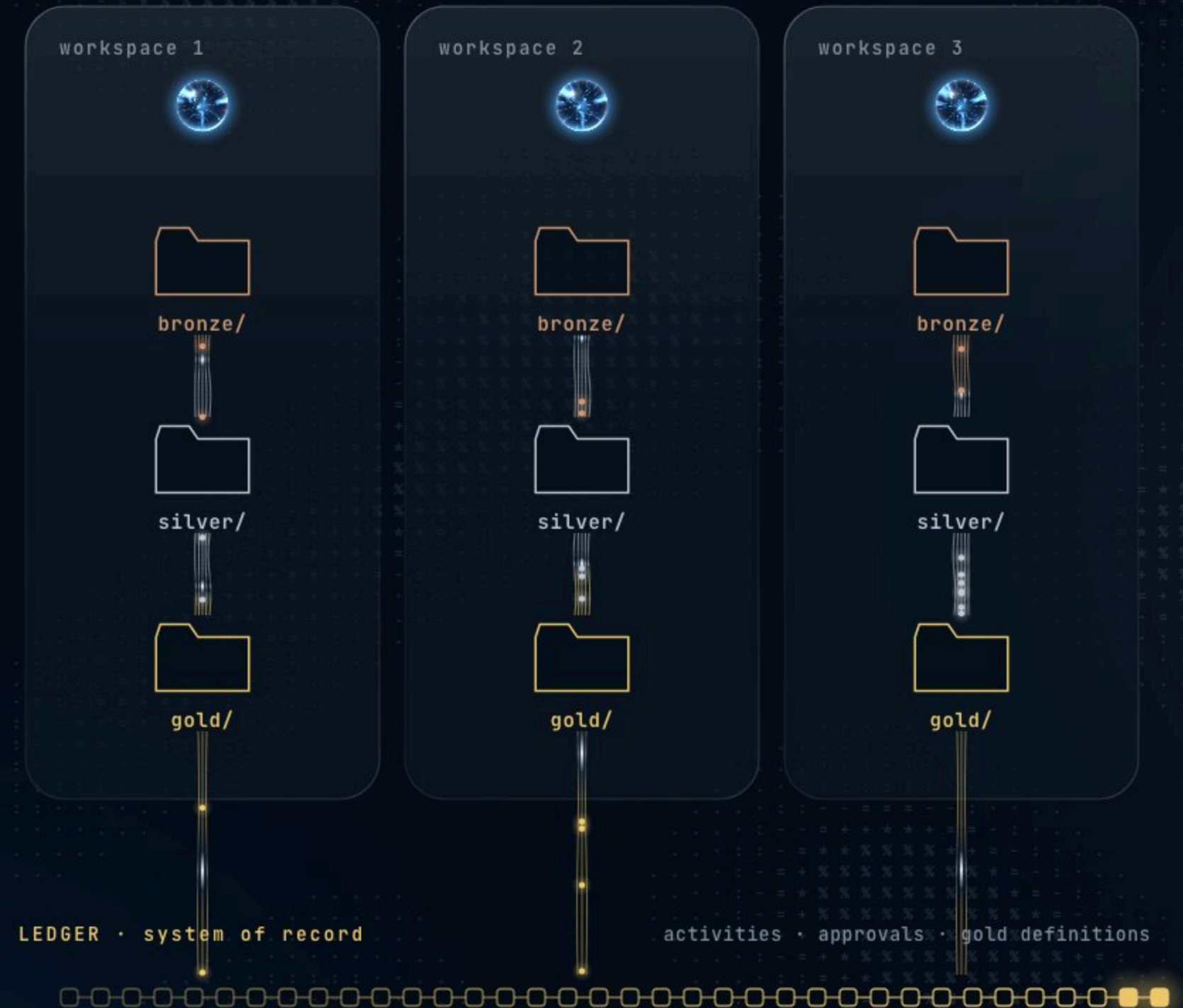
Many workspaces

bronze/ Inputs as received. Append-only, hashed, never edited.

silver/ Typed and validated. Every value cites its bronze source.

gold/ Decisions and definitions the organization trusts.

Files are the agent's working memory. The ledger is the organization's.



A review system, built on the pattern.

Sandbox

Readers

Judges

Ledger

• BEAT 04 · LEDGER

The ledger is the system of record

Activities, approvals and gold definitions go to one shared Postgres ledger outside every sandbox. Files are the working memory; the ledger is what the organization trusts.

case packet

untrusted

SANDBOX · one per case

reader subagents

harness agent · review skill

code verifies + judges

bronze/

silver/ claims

gold/

unverified quote → stays a claim

LEDGER · Postgres · system of record

ACT II

Context

What reaches the window, and when?

How an agent's context is built.

Everything the model knows this turn is what reached the window. The workspace decides what reaches it.

Bare session

Workspace session

Hover any layer of the window. Each one has a source on disk, a moment it loads, and a budget.

Less guessing is the product of more grounded layers, not of a bigger model.

Harness system prompt
built into the harness

Primer
CLAUDE.md ≡ AGENTS.md

Memory index
MEMORY.md → one file per fact

SessionStart hooks
board facts · freshness · metrics

Skill descriptions
one line per installed skill

Your message
what you typed

Intake hook
lens · graph constraints · entities

Tool results
files read · bash output

Conversation
prior turns

CONTEXT WINDOW

guessing

every turn

session start

session start

session start

session start

each turn

each prompt

while working

grows each turn

more grounded layers, less guessing

proportions illustrative, not measured

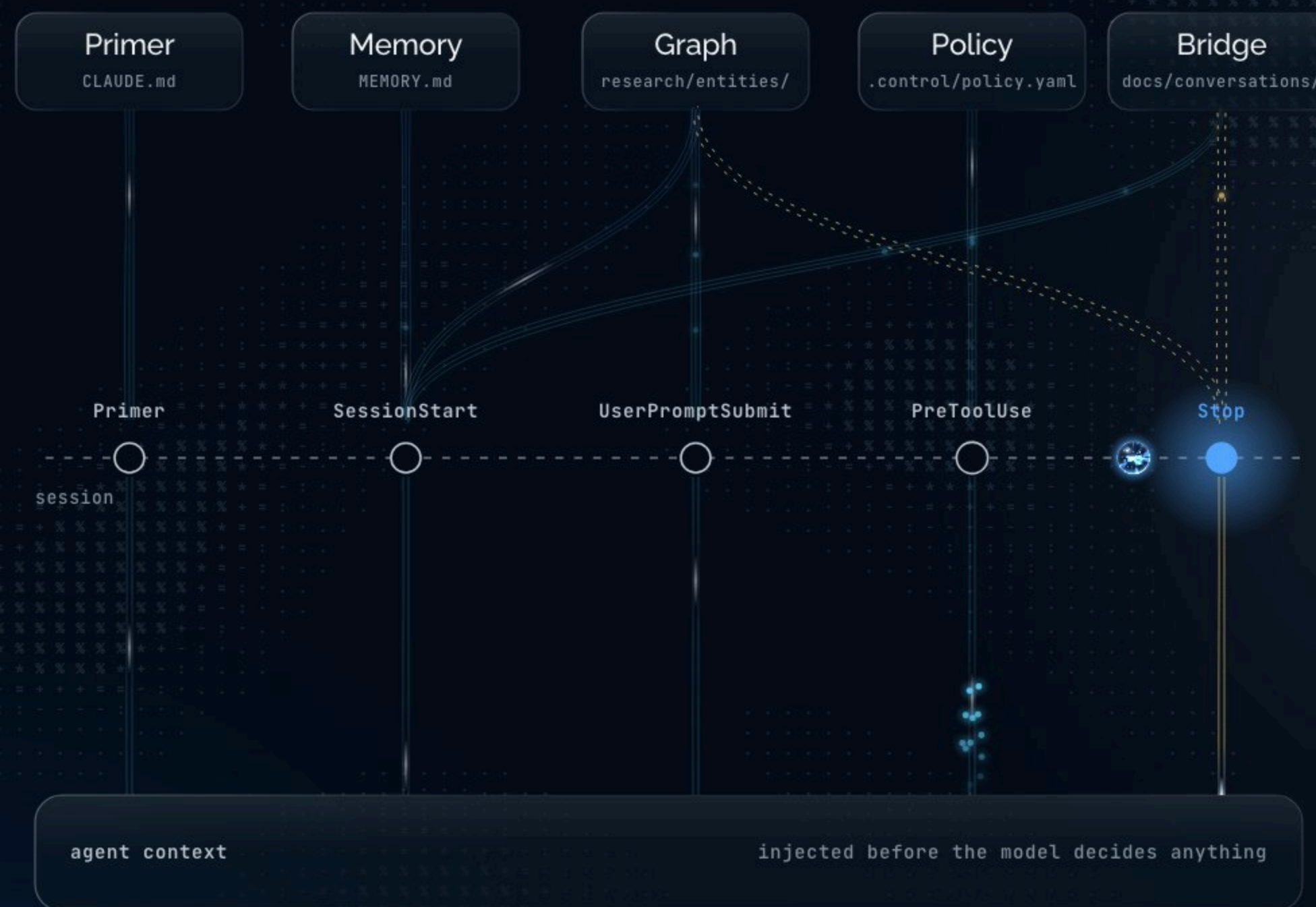
Hooks turn recall into a reflex.

• HOOK 05 OF 05 • BRIDGE • PROVENANCE

Stop

When the session stops, the bridge writes it to docs/conversations and provenance is stamped, so the next session can search what this one did.

docs/conversations/session-2026-10-26-....md



PRIMER

Asks for it

SKILL

Knows how

HOOK

Runs every time

Files first. Vectors when words diverge.

All Local Semantic **Live**

Exact names, paths, claims you can grep

RG + INDEX

What this relates to, one hop out

GRAPH LINKS

Same idea, different words; corpora past the window

EMBEDDINGS

Fresh or external state

MCP · APIS

At 1,430 entities the compiled catalog is about 260K tokens: it fits a 1M-token window, so the catalog routes and the model reads. Add vectors when vocabulary diverges or the catalog outgrows the window.



ACT III

Second brain

How is knowledge read, and how does it grow?

@ the reader

Most second brains are write-only.

Today

Wired at intake

• THE FIX IS ROUTING, NOT VOLUME

Put recall on the startup path.

The primer routes to it. An intake hook surfaces matching entities.
[/kg load](#) runs before planning. And when reads lag, prune
production. Don't add more writers.



simulated · reads per write 8.4%

Front matter is the API.

Files

Index

Edges

Synonyms

```
research/entities/pattern/  
llm-as-index-architecture.md
```

```
---
```

```
type: pattern    status: entity
```

```
core_claim: "At sub-10k-entity scale the  
substrate fits in a 1M-context  
model, so the LLM IS the query..."
```

```
tags: [architecture-decision, knowledge-graph, ...]
```

```
aliases: [kg routing, no embeddings retrieval]
```

```
related: [[knowledge-graph]], [[bstack-engine]] ...
```

```
sources: [2026-05-21-...-synthesis, BR0-1223]
```

```
---
```



docs/knowledge-index.md

13 entities · 23 edges

- grounded-vs-ungrounded-...
- multi-tier-dreaming
- recursive-controlled-sy...
- auto-compact-catalog
- bstack-engine
- entity-page
- haystack-benchmark-patt...
- knowledge-graph-enginee...
- llm-as-index-architectu...
- cognee
- kepano-obsidian-skills
- knowledge-graph
- open-knowledge-format

```
/kg load "knowledge graph" --terms "no embeddings retrieval" --expand 1
```

An entity is a claim with receipts.

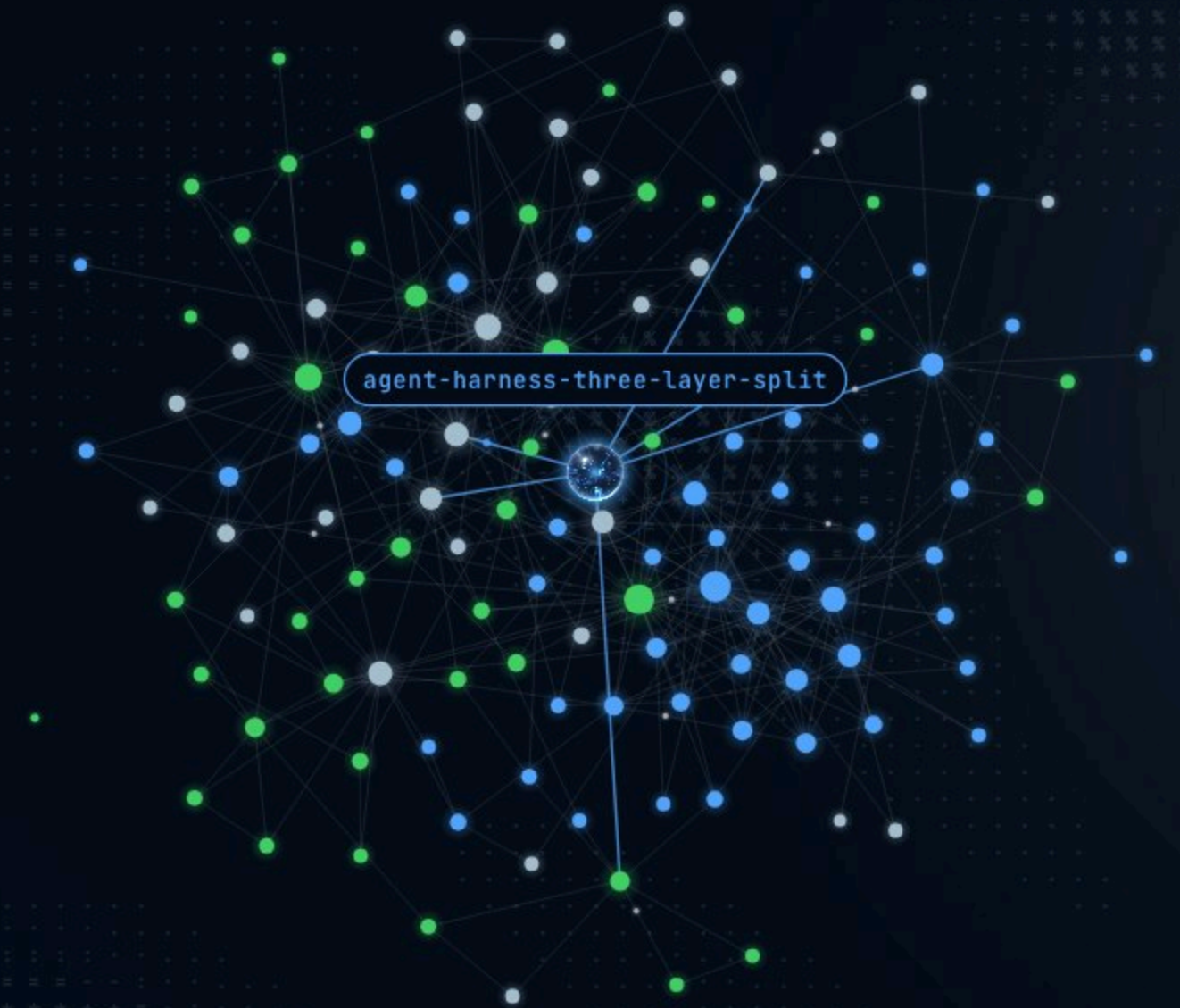
```
---
# research/entities/concept/agent-harness-three-layer-split.md
type: concept  status: entity
core_claim: "Agent harnesses split into sandbox,
orchestrator and UI wire: the sandbox
is buyable, the wire has no stable
remote transport."
sources:
- url: agentclientprotocol.com/.../transports
  note: "Read verbatim 2026-09-26 ..."
scoring: {novelty: 2, specificity: 3, relevance: 3}
related: [[buzz-acp-harness]], [[display-server-for-agents]]
---
```

— One falsifiable sentence

— Read verbatim, dated

— Scored before promotion

— Typed links become edges



1,430 entities 6,076 edges .md canonical · index of 2026-10-25

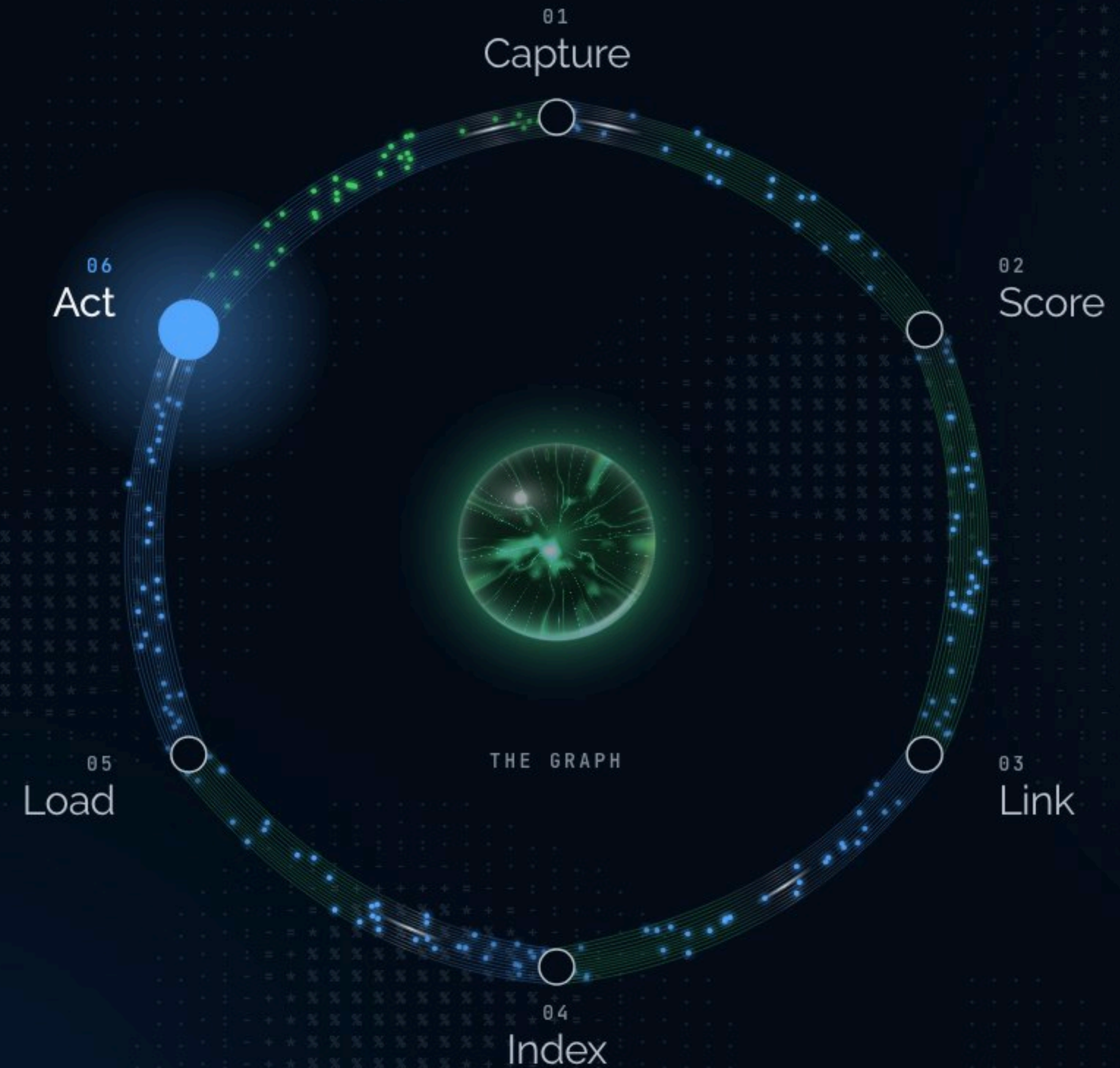
The loop that makes knowledge **readable**.

● STATION 06 OF 06

Act

cite → decide → capture

Outputs cite entities. New findings re-enter the loop.



Memory holds facts about the owner. The graph holds claims about the world. The ledger holds what is in flight.

Insight hardens into mechanism.

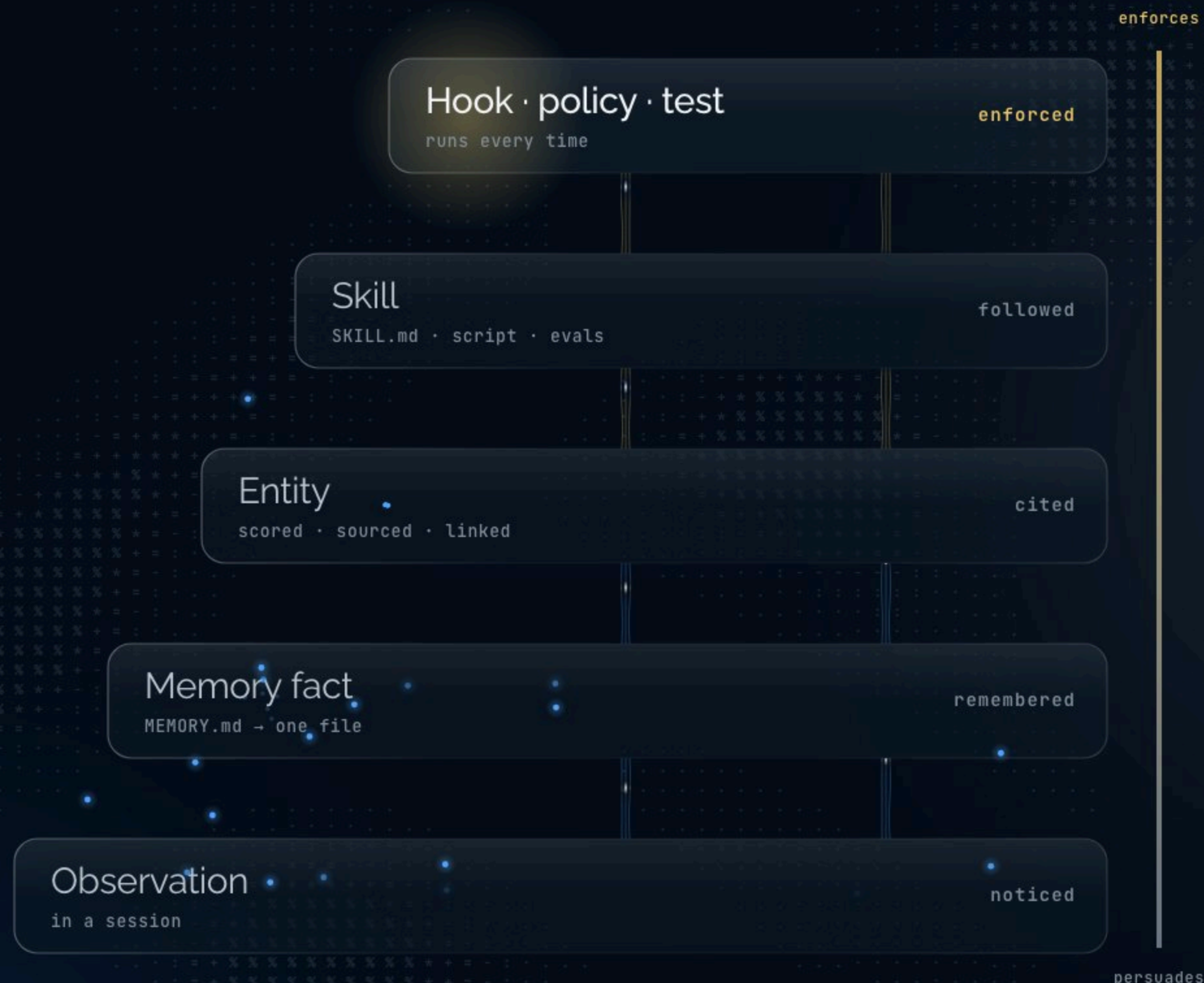
• LEVEL 05 OF 05 · ENFORCED

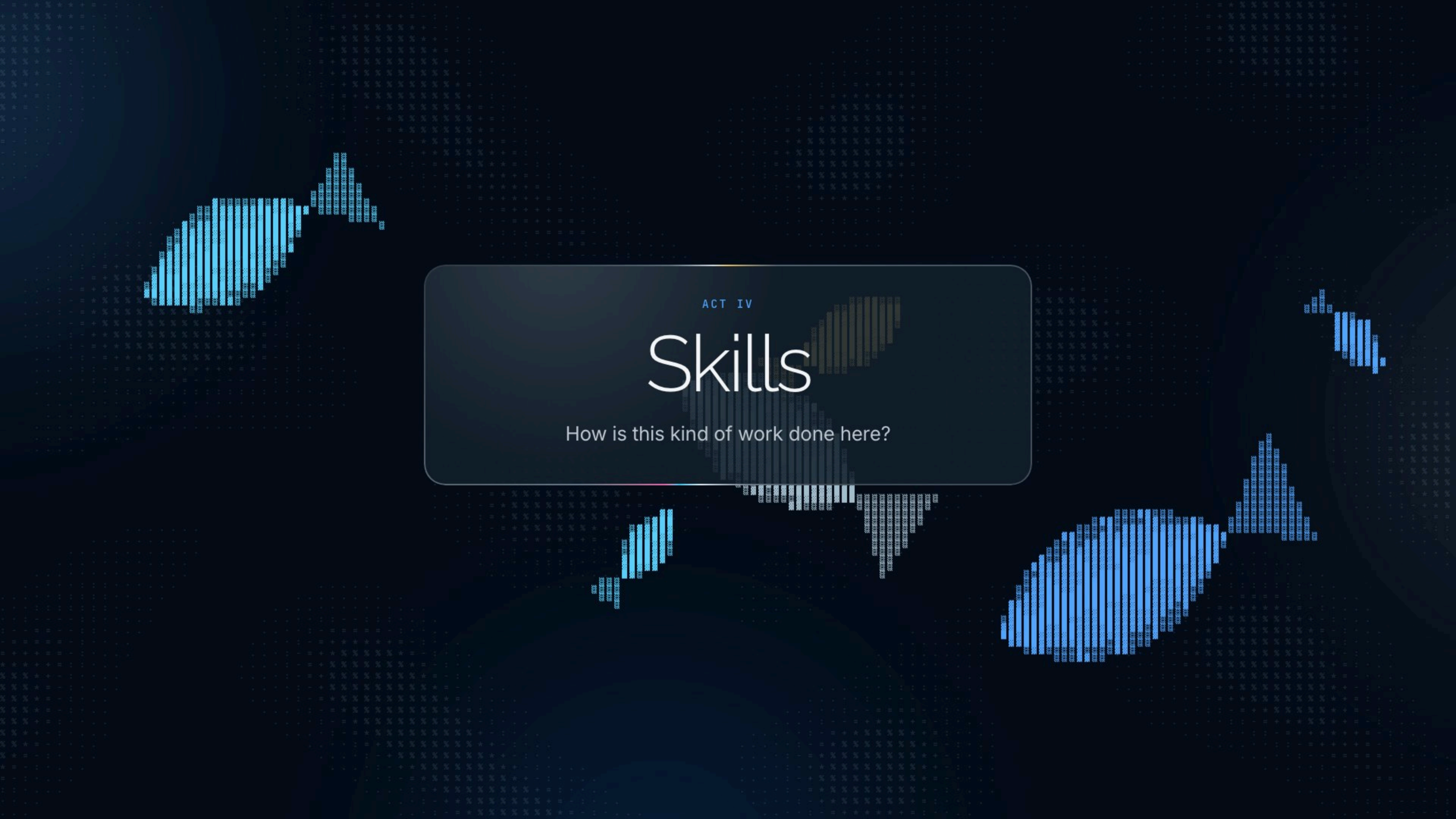
Hook · policy · test

A rule that must never be broken stops depending on anyone remembering it. It executes.

Prose persuades. Executables enforce.

Promote on the third instance, with a concrete mechanism, an invariant and a failure mode. Name what the new rule retires.





ACT IV

Skills

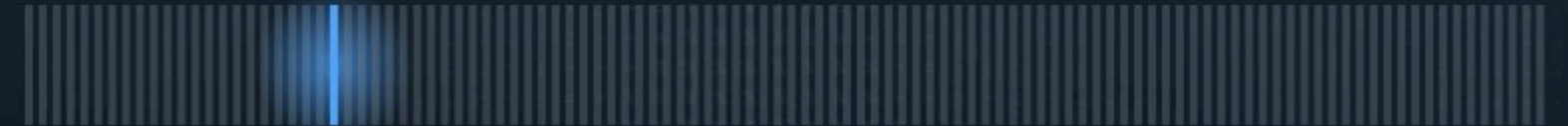
How is this kind of work done here?

A skill is a procedure, not a prompt.

```
skills/knowledge/kg/  
├ SKILL.md      description is the router  
├ scripts/      deterministic core  
└ evals/        routing + behavior tests  
  
---  
name: kg  
description: Load graph context for a topic...  
  USE WHEN ... NOT FOR ...  
---
```

CONTEXT WINDOW

ALWAYS LOADED · every skill's description



"what do we know about sandboxes?"

→ kg

ON TRIGGER · SKILL.md body

SKILL.md · kg

ON DEMAND · scripts/ · references/

a body loads only when its description matches

Skill, CLI, MCP.

One knows. One does. One reaches.

All

Skill

CLI

MCP

Repeatable judgment and steps

SKILL

Deterministic, local, testable work

CLI

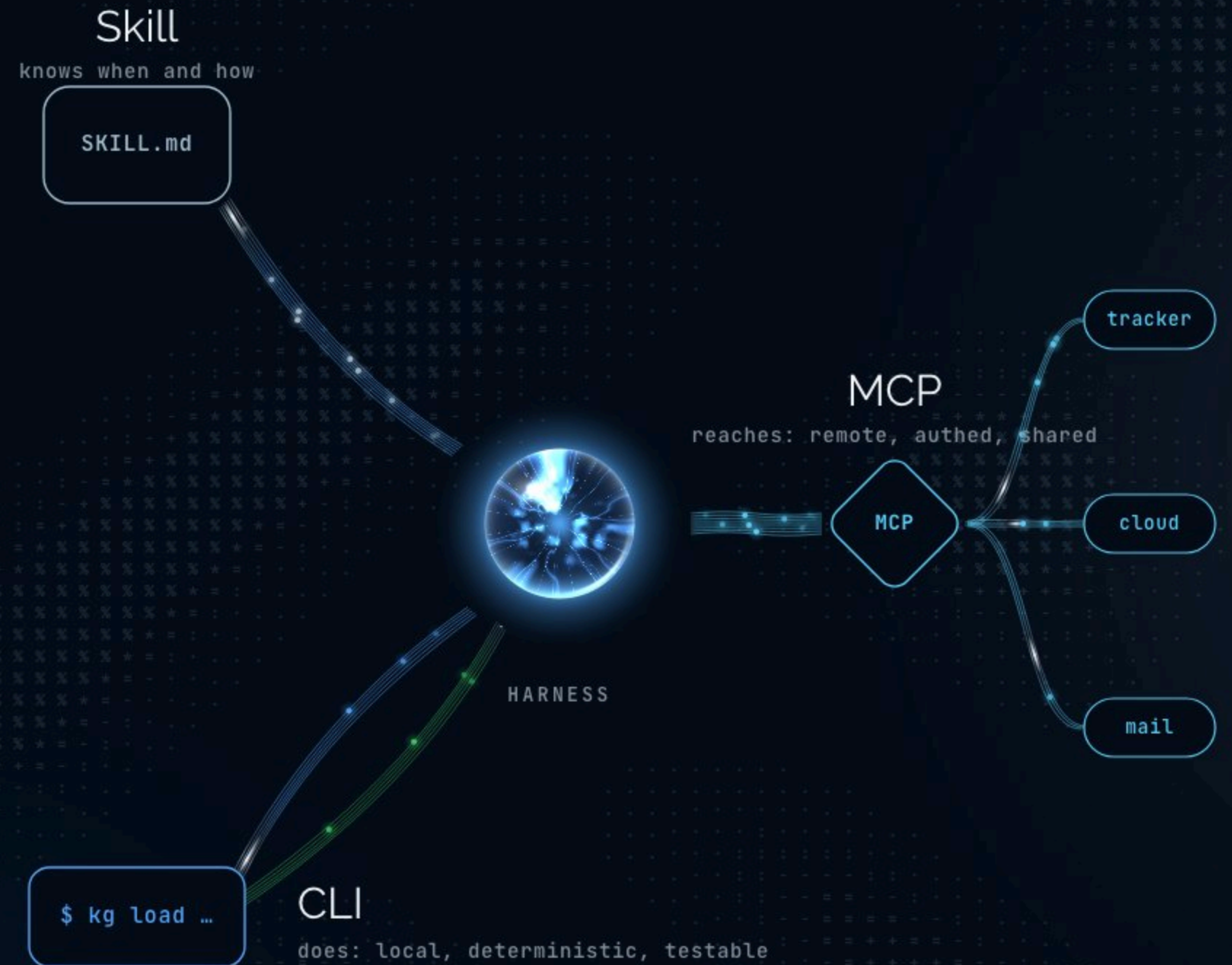
Remote system, auth, shared across harnesses

MCP SERVER

A one-off

JUST ASK

A skill that drives a small CLI beats a wide MCP surface: fewer tools in context, and a core you can test.



Author once. Fan out. Ration hard.

Fan out

Ration

115

AUTHORED

23

CATEGORIES

31

HARNESS FOLDERS

- ◆ One canonical source, in its own repo.
- ◆ Install once, symlink into every harness. Never copy.
- ◆ New skills ship evals: it fires when it should and stays quiet when it shouldn't.
- ◆ Retire before you add. The roster is a budget.

457 INSTALLED

26 KEPT A DESCRIPTION

the rest were bare names: they could not trigger

```

| *      ab9ed2  fix(bridge): capture session on stop
|\      385e34  docs(handoff): pickup for tomorrow
*      779a59  fix(bridge): capture session on stop
|/      920e00  feat(policy): gate destructive bash
| *      3060dc  merge: review strata passed
*      64a169  feat(skills): add routing evals
*      da5c51  fix(hooks): intake hook stays under 300 ms
|/      c342b9  merge: review strata passed
| *      49afd6  fix(hooks): intake hook stays under 300 ms
| *      4b0a25  docs(kg): score and link three entities
|\      3d705e  fix(hooks): intake hook stays under 300 ms
*      73c479  chore(janitor): sweep merged worktrees
|/      e2128a  feat(policy): gate destructive bash
| *      df1de8  refactor(primer): route, don't dump
*      b9b666  docs(kg): score and link three entities
*      52ada8  feat(index): compile front matter
|/      fdb864  merge: review strata passed
| *      487e4a  feat(policy): gate destructive bash
| *      feddfe  feat(index): compile front matter
|\      fdce48  docs(handoff): pickup for tomorrow
*      c240e3  feat(skills): add routing evals
|/      b5aff1  docs(handoff): pickup for tomorrow
| *      94e055  refactor(primer): route, don't dump
*      545467  chore(janitor): sweep merged worktrees
*      fd7947  feat(index): compile front matter
|/      e619de  fix(bridge): capture session on stop
| *      904005  refactor(primer): route, don't dump
| *      334d21  refactor(primer): route, don't dump
|\      7e32c4  docs(handoff): pickup for tomorrow
*      90db0e  merge: review strata passed
|/      a35eb1  feat(policy): gate destructive bash

```

ACT V

Structure

Which bytes may the agent touch?

One place, many repos.

workspace/	governance repo
├ AGENTS.md ≡ CLAUDE.md	one primer, every harness
├ .control/	policy · initiatives · asks
├ roles/	lenses: context per kind of work
├ skills/	canonical skills, own repo
├ research/entities/	the graph, as markdown
├ docs/	sessions · plans · handoffs · ADRs
├ apps/ core/ work/ ...	product repos, own CI
└ .worktrees/	one per task, disposable

- SHARED one copy of the operating context, reviewed like code
- SOVEREIGN 66 product repos under apps/, core/ and work/, each with its own history
- DISPOSABLE worktrees created 7 · merged 0 · swept 0



One writer per tree.

Parallel agents never share a checkout. They share main, through review.



01

Branch is the first action

02

Reviewers read. They never write

03

A live thread's worktree is the thread

04

Unfinished work becomes a draft PR

05

Janitor after merge, dry-run first

ACT VI

Fleets

Who else is here, and who checks whom?



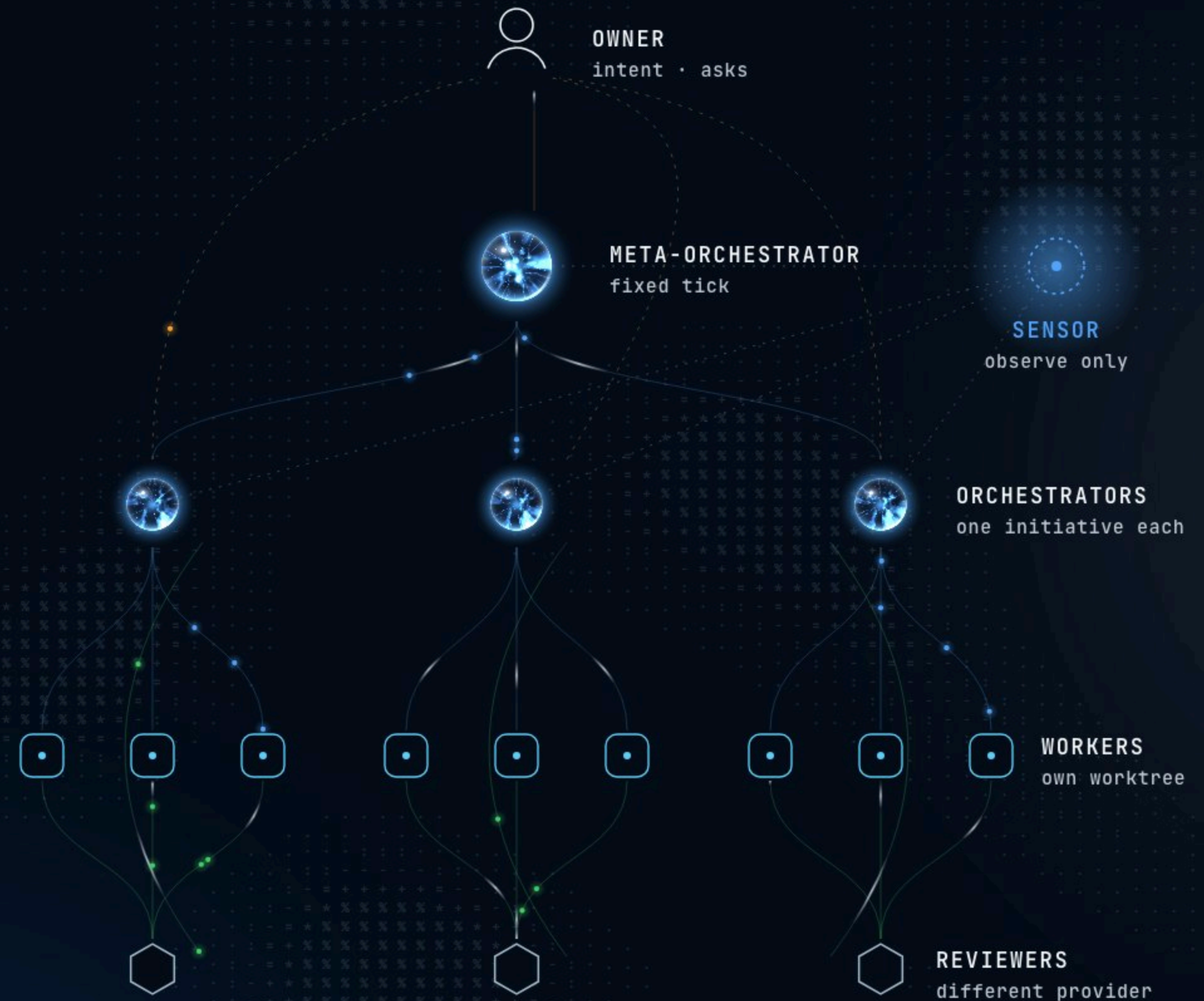
Roles, not agents.

• ROLE 06 OF 06

Sensor

Observes and files: capacity, drift, stalls.

NEVER Acting on what it measures.



Spawn, reuse, or hand off.

Independent of what I know?

Spawn a child task in its own worktree.

Needs my context?

Continue here, or fork the thread.

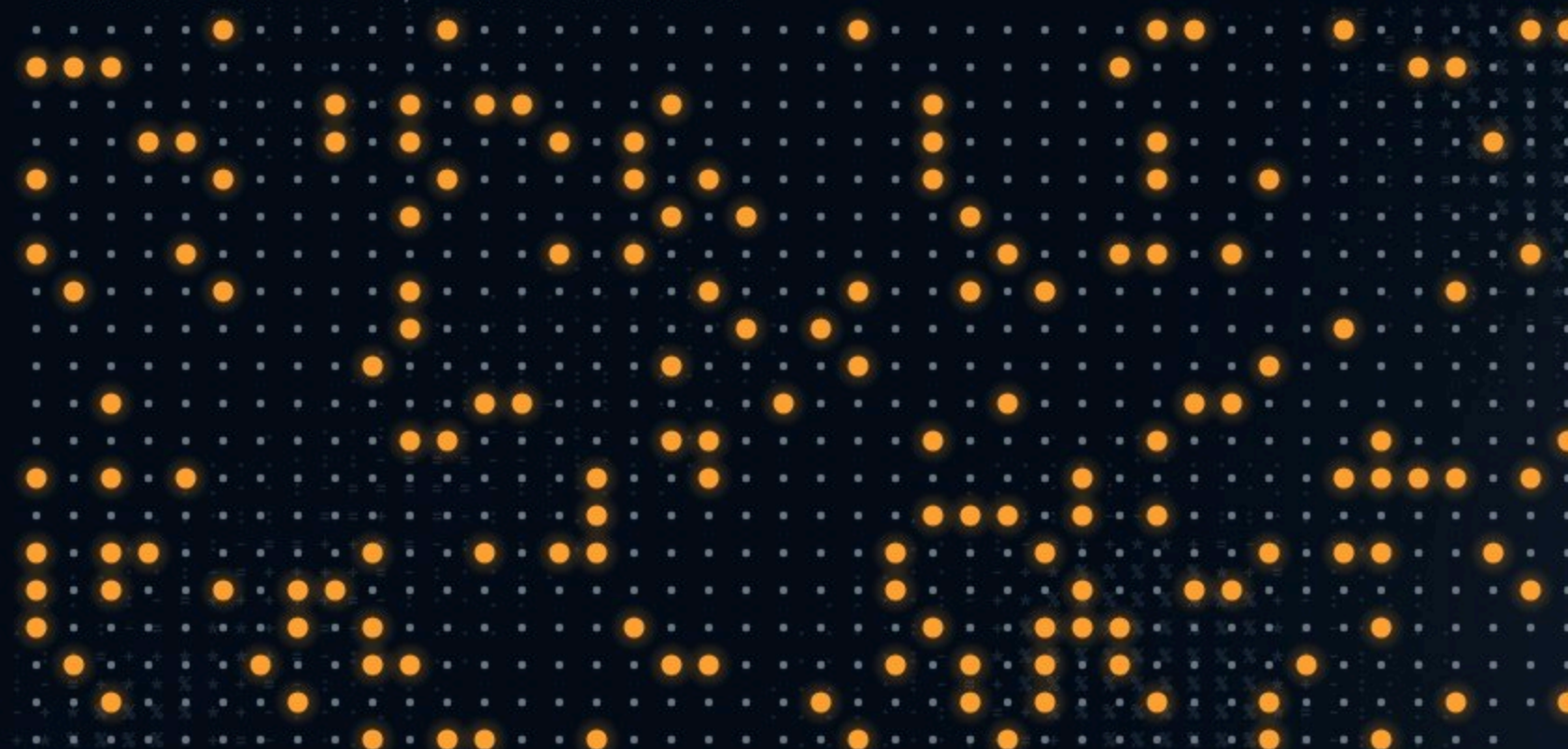
Outlives this context window?

Write a persistent brief with stop conditions.

Only the owner can decide?

File an ask: context, options, a recommendation and a safe default.

838 main sessions · 30 days · measured 2026-10-10



Long sessions carry most of the token cost. Roll over with a handoff.

Handoff

narrative for the next agent

Handback

message for the human

Brief

loop state across contexts

Ask

a decision, with its context

Receipt

evidence it actually works

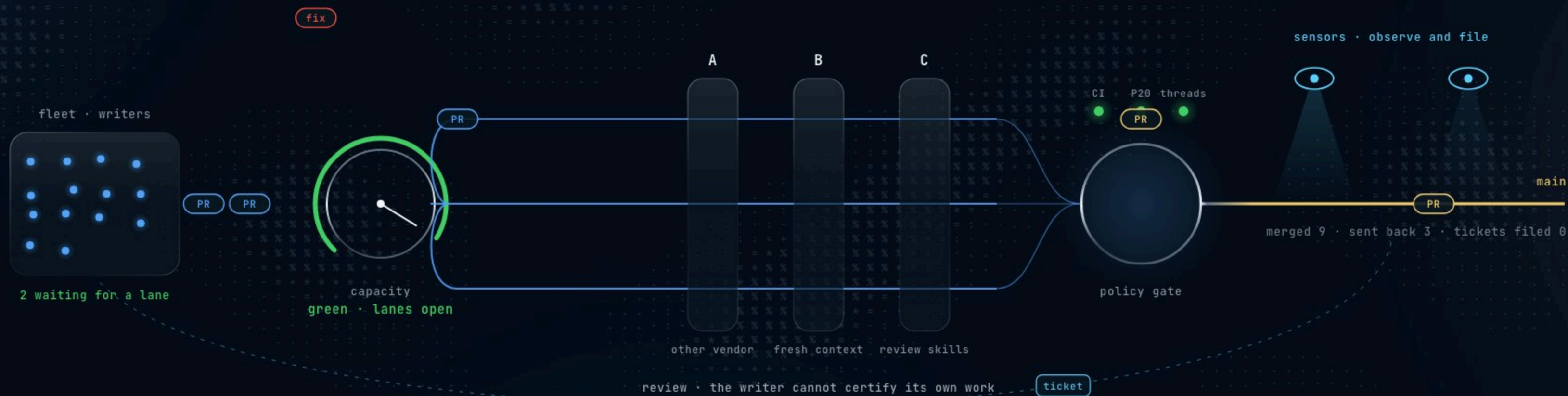
Governance that scales with the fleet.

Pace

Review

Policy

Sensors



PACE

From measured capacity, not hope

REVIEW

The writer cannot certify its own work

POLICY

Executable gates. Prose is only advice

SENSORS

Observe and file. Agents act

ACT VII

Leadership

What does this ask of people?

The unit of work is now a loop.

When an attempt costs minutes, making gets abundant and
deciding gets scarce.

Task

Loop

Every question is something you can start today.

Every repeated task is something you can automate or improve.

Every piece of work leaves the system better: a note, a skill, a
check.

The question moves from “can I do this?” to “should this exist, and
how will I know it's right?”



making

deciding

scarce

decisions taken 2

Everyone leads a fleet.

• TERM 05 OF 05

Review

who judges the result before it ships

You do not certify your own work: a separate session or a different model does.



What each person brings to their fleet is taste and judgment: knowing what good looks like, and deciding what to accept.

Attention is the new ration.

Every thread pings you

The gate calls you

Reversible drafts, research, branches: act and note it.

Costly long compute, many iterations: only inside a budget.

Irreversible or external spend, publish, merge outside policy, contact: a yes first, every time.

The measure of an autonomous system isn't how much it does without you. It's how precisely it calls you.

40 threads

gate

1 · merge to production?

2 · three GPU hours?

3 · publish the post?



reaching you: three ordered asks

budget

- reversible · act and note
- costly · inside a budget
- irreversible · needs a yes

Goalposts teach waiting. Bars rise in the open.

Goalpost moves

Bar rises

People learn to wait when proposals get overridden, the leader decides anyway, the bar moves silently, or a failed attempt costs more than a passive miss.

Make everything queryable: decisions, standards and why they changed live in the same shared graph that people and agents read.

SHARED CONTEXT · decisions/bar.md
read by every person and every agent

● v4 · + every claim cites a source
why: sources are one query away

● v3 · + review by a second model
why: cross-review got cheap

● v2 · + a dogfood receipt
why: agents can drive the app now

● v1 · tests pass
the first bar

query beam

the bar

● people ● agents

climbing 7/7

EPILOGUE

Better agents come from better harnesses.

● LAYER 06 OF 06

Workspace

The place that holds all of it and keeps what every session learns. This is where the harness compounds.

MODEL the same weights in every layer

OUTCOME illustrative, not measured

evals · 78 passed · 13 failed

workspace · 009 deposits kept



— EPILOGUE

Twenty primitives. One loop.



The workspace is the product.

01

Give agents a place before you give them more autonomy.

02

Make the graph readable at intake, or stop writing to it.

03

Scale with roles, isolation and review. Not with more prompts.