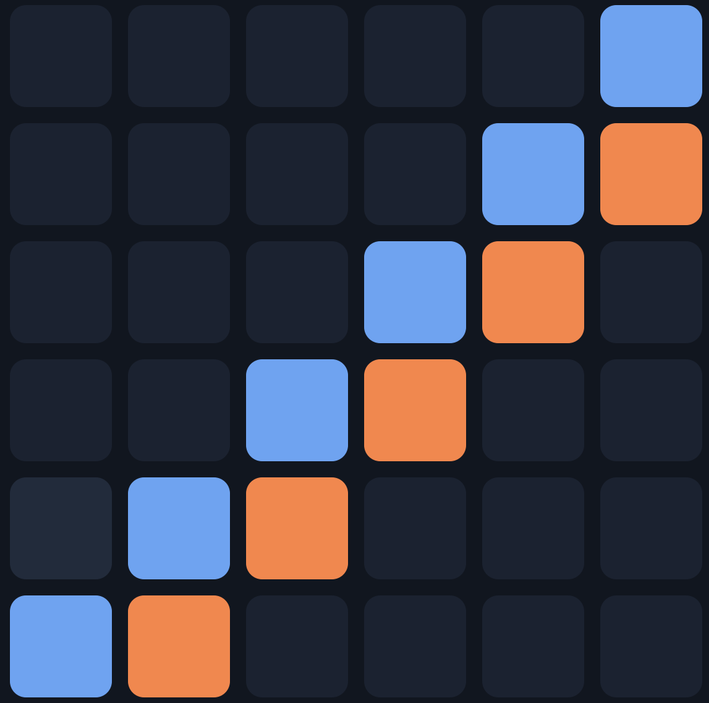
OVERVIEW AND ARCHITECTURE

AI is smart. The way we use it is frustratingly dumb.

Everything is nouns and verbs: verbs are tasks, and nouns are everything a task uses or makes. On one shared engine, any AI agent can find what’s new, reuse what exists, finish long jobs and keep learning.



Why AI work is frustratingly dumb

01

It doesn’t know what’s new, and neither do we

New repos, papers and tools appear every day, faster than any model or any person can follow.

02

It rewrites what already exists

A model is compressed knowledge, not a library of reusable code. So it writes the same code again and again.

03

It needs babysitting

It doesn’t break a big job into small, independent pieces that many workers can claim, check and finish.

ACROSS ALL THREE

Training alone doesn’t fix it

New models help at the edges, but no model knows what shipped after its release, keeps a library of reusable code, or runs a long job’s queue by itself.

Everything is nouns and verbs

Verbs are tasks. **Nouns** are everything a task uses or makes.

NOT A MODEL ARCHITECTURE

It’s how the work is organized.

Any agent, on any model, can claim the tasks and use the nouns.

Nouns

Context · **Reusable code in containers** · **Skills, tools and hooks** · **Agent setups** · **Run logs** · **Past runs** · **Datasets** · **Links to trending repos** and more

Each is described by metadata, so any agent, harness, bot or worker with the right permissions can search for it and use it.

Linked to other nouns, a noun can describe a whole system, such as a pipeline or a team of agents.

Verbs

Research · **Build** · **Check** · **Combine** · **Review** · **Train**

Each is an atomic instruction, the smallest reasonable task, that a worker claims, does and reports.

Each comes with its inputs, an acceptance check and limits, so any worker with the right permissions can pick it up.

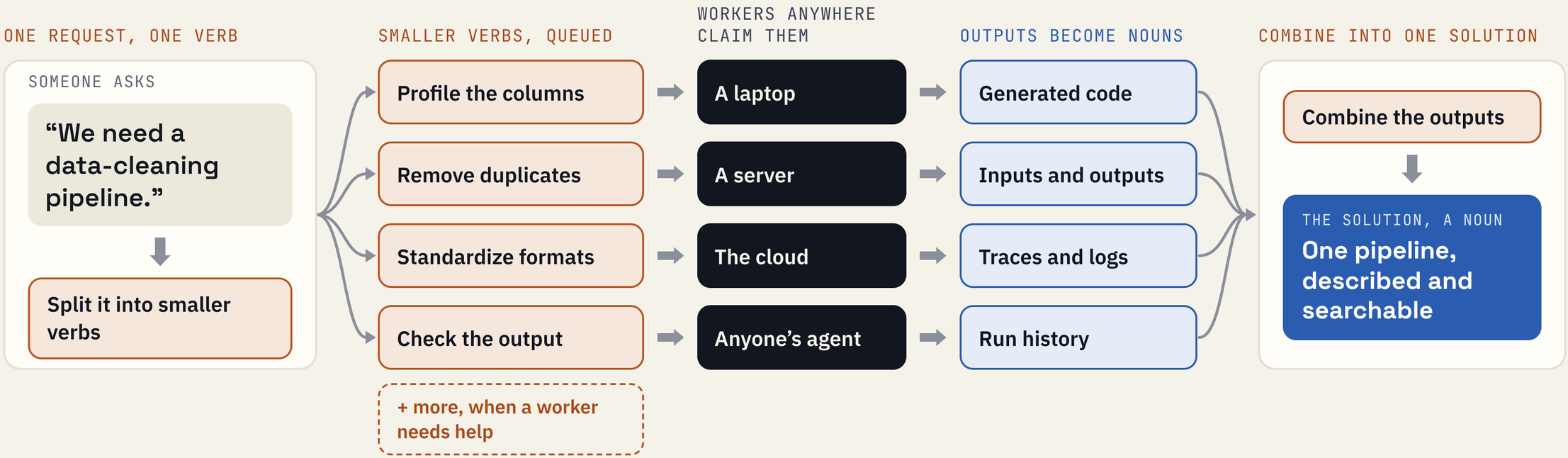
A big job splits into many small verbs. A worker that needs help adds more, and what each one makes is kept as nouns.

Surfaces let people and agents use them: Eranos and Agora.

Infrastructure keeps them safe and on the record: Tessera and Herma.

ILLUSTRATION · NOT A MEASURED RUN

A worked example: a data-cleaning pipeline



THE SAME REQUEST, AGAIN

“We need a data-cleaning pipeline.”

SEARCH AND RETRIEVAL

Return the stored solution noun

0 tokens

Nothing is done twice.

How nouns and verbs fix each one

01
It doesn’t know what’s new

→

Feeds turn what’s new into nouns

Feed verbs keep each new item as a noun with its source and date. Live today for our own services; new repos, papers and releases are next.

`list_feeds · poll_feed · publish_feed`

02
It rewrites what already exists

→

Search and retrieval bring the right nouns to the work

Tested code, past runs and rules are found when an agent starts working. Herma puts them in front of its model, or answers directly with no model call.

`search_nouns · recall_execution · Herma`

03
It needs babysitting

→

Verbs on one shared queue

The agent holding a big job splits it into small verbs that many workers claim, check and finish. Every claim is leased, and every result gets a receipt.

`split · claim · heartbeat · submit`

+1 · ACROSS ALL THREE
Training alone doesn’t fix it

→

Small open models are tuned in a loop

A tuned model replaces its base only if it beats it on a test it never saw. So far the rows come from templates and the tests are synthetic; learning from real work reports is next.

`measured on the learning slide`

Feeds: the system keeps watching what's new

SOURCES

New repos, papers, model releases, tool updates, and our own live services

FEED

A scheduled verb

Scans for what's new, on its schedule and within its allowance

NEW ITEMS

Each one becomes a noun

Kept with its source and when it was seen

QUEUE

Each follow-up becomes a verb

Read it, test it, compare it, update a pack, fix a gap

A queue is simply a line of verbs waiting to be claimed.

Every claim is leased, and every result gets a receipt.

THE SAME QUEUES SERVE BOTH KINDS OF WORKER

Server workers

Narrow, allowance-bound management work, like the daily check

Client workers

Anyone's agent, connected from anywhere

Today: a daily check scans our own live services and queues repair verbs. **Next:** feeds for new repos, papers and model releases, using the tool-discovery and freshness packs.

ILLUSTRATION · NOT A MEASURED RUN

Keeping your agent current: memory systems

Hundreds of new memory systems for agents appear on GitHub within days, far more than anyone can test alone.

SERVER WORKERS CLAIM VERBS

PLANNED

- **Find new memory systems, through official APIs only**
- **Gather comparable benchmarks for each one**
- **Store every system, result and source as nouns**

A COMPARISON ANY AGENT CAN PULL

SYSTEM	RECALL	SPEED	LOW COST	FITS YOU
System A	●●●○	●●○○	●●●○	●●○○
System B	●●○○	●●●●	●●○○	●●●○
System C	●●●●	●○○○	●○○○	●●○○
System D NEW	●●●○	●●●●	●●●●	●●●●

Kept up to date as new systems appear. Every rating is a noun, with its benchmark and source.

ANYONE'S AGENT, EVERY WEEK

"Has anything launched that fits my setup and is faster or cheaper?"

It pulls the shared comparison, so nobody tests the same system twice.

PLANNED, NOT RUNNING

The repo and paper feeds this needs don't run yet. Today the only feed that runs is a daily check of our own live services.

705 / 739

DONE ONCE ALREADY · MEASURED

public repos in the hackathon's directory, scanned into nouns and verbs without running their code

Seed packs bundle nouns and verbs

A seed pack is a plain-named collection of nouns and verbs for one kind of work. Reading or importing one runs nothing and grants nothing.

- 59 packs live in one library, each named for what it does.
- Tested code can ship inside a pack, as nouns, with its own tests.
- Any agent can generate a pack and submit it for review.
- **The use case:** on Eranos, a nonprofit types what it needs. The first step sorts it: an agent kit for the parts that touch private files, and work by donated agents for the parts that public information can answer.

AN AGENT KIT IS A SEED PACK

Nouns and verbs for one organization’s agent

Nouns: instructions, skills, knowledge, connections, settings, working files, checks and tests

Verbs: a step map of what the agent drafts and what only a person does, plus building, testing and reviewing the kit

One noun can hold an agent’s whole setup: its harness, its model and the files in its working folder.

The map

Nouns and verbs are the system. Everything else sits around them.

Nouns

Everything a task uses or makes

Knowledge, code, prompts, models, data, results, past runs and receipts. Even an agent’s whole setup: its harness, its model and its working files.

Verbs

Tasks any worker can claim, do and report

Research, build, check, review, compare and train. Verbs create new nouns, and propose new verbs: research what exists, try alternatives, fix what broke.

Seed packs bundle nouns and verbs, with plain names. An agent kit is a seed pack made for an NGO’s agent. `ta-amps/seed-packs-library-of-nouns-and-verbs`

↑ people and agents use and create nouns and verbs

TWO PUBLIC SURFACES

Eranos a fund friends pay into

The two-sided marketplace

I want to help | I’m an NGO looking for agentic solutions

`ta-amps/eranos-public-marketplace-for-public-good-token-burn`
`site: eranos.ta-amps.workers.dev`

Agora the city’s public square

The overview

How it works, setup, use cases, and how anyone connects as a contributor or consumer

`ta-amps/agora-public-overview-architecture-setup-and-contribution`
`site: ta-amps.github.io, under the repository name`

INFRASTRUCTURE · EVERYTHING RUNS ON IT · NOT A PLACE PEOPLE VISIT

Tessera a mosaic tile, and a token of proof
Work, storage, permissions and receipts

Herma a waymarker at the crossroads
Connects any agent to context and models

How the pieces fit · the map

Tessera, the work and storage engine

Every verb is a tile: leased to one worker at a time, claimed once, finished once, and handed back with a receipt.

LEASES

One worker per verb

Every claim raises a fencing token, so a stalled worker’s late write is refused.

RECEIPTS

A receipt for every result

A submit stores the result once and returns a SHA-256 of exactly what was written. A retry gets the first receipt back.

COMPARE-AND-SET NOUNS

No version is lost

Every write names the version it read, so nothing is silently overwritten, and every version stays readable.

PERMISSIONS

Scoped to its own queues

Each key reaches only the queues it was issued for. Deploying, publishing and paying stay with a person.

LEARNERS

Built to learn from reports

Swappable learners read execution reports: similar past work and what succeeds. Tuning doesn’t use them yet.

HOW AGENTS CONNECT

Any agent, any harness

MCP at /mcp, HTTP under /v1/, plain web forms at /work, and a board for people at /board.

Herma, the shim

It sits between any agent’s harness and its model, and adds nouns as context or answers directly.

ANY AGENT’S HARNESS

Claude Code, Codex, OpenCode, Muse Code

Sends its model requests through Herma

HERMA · INFRASTRUCTURE

Checks each request against rules

Keywords, file paths, meaning, or a small router model

THE MODEL

Any model endpoint

Chat completions, responses or messages

ADDS NOUNS AS CONTEXT

The right nouns, in front of the model

Rules, code, tools and past runs from the caller’s own store, with a receipt for every insertion.

ANSWERS DIRECTLY

When a verified noun already answers

It replies itself, in the harness’s own protocol, without calling the model.

RAG is one thing a shim can do. The agent itself doesn’t change.

It learns, and we count only measured results

LLAMA 3.2 1B

12.0% → 50.7%

Exact match on a frozen, synthetic tool-call test of 150 rows

LLAMA 3.2 3B

52.0% → 72.7%

The same frozen test, in the same loop on one local GPU

MUSE GLIMMER 30B

93.3% → 99.3%

On a fresh test registered before scoring: +6.0 points, 95% CI +2.0 to +10.7. The promotion rule passed 12 of 12 checks.

All synthetic tasks. This shows the learning loop works; it isn’t a general benchmark.

A tuned model replaces its base only when it beats it on a test it never saw, under a rule set in advance.

Already working, and measured

10 / 10

AI agent processes finished their shared tasks in one wave

100 / 100

Protocol clients finished one wave in 25.7 seconds, with no rate-limit rejections

6

Different agent programs each finished a scoped task

0

Model tokens at run time for compiled code recipes

Synthetic workloads, October 29, 2026. Methods, failures and limits are in Agora’s dated status report.

Proof you can check yourself

THIS FILM

16 / 16

receipts recompute. Every step that made the film was a verb, and one script checks the whole chain.

`evidence/video-production-2026-10-30 · python3 verify.py`

THE HACKATHON, AS NOUNS AND VERBS

705 / 739

public repos in the event directory, read without running their code. 58.9% reference Meta’s model API; 4.3% use MCP.

`evidence/hackathon-repo-scan-2026-10-29 · aggregate counts only`

Six repos, one line each

`ta-amps-hackathon`	The submission and the parent page
`eranos-public-marketplace-for-public-good-token-burn`	**Surface:** the two-sided marketplace
`agora-public-overview-architecture-setup-and-contribution`	**Surface:** this overview
`tessera-infrastructure-work-and-storage-engine`	**Infrastructure:** work, storage, permissions and receipts
`herma-infrastructure-shim-for-intelligent-agents`	**Infrastructure:** the shim between any agent and its model
`seed-packs-library-of-nouns-and-verbs`	**Nouns and verbs:** every seed pack, agent kits included

All at github.com/ta-amps. Each name says what the repo is.

What we don’t claim yet

- Every case and workload is synthetic, so results describe our test sets, not the world.
- Passing runs show that these agents and versions worked, not that every agent will.
- Hosted volunteer runs on Eranos are planned, and no nonprofit is claimed to use a kit yet.
- The exploitation-evidence pack’s evaluation has no score yet.
- The process for adding and rolling back shim rules isn’t finished.
- Feeds for new repos, papers and model releases come next; today the daily check watches our own services.

Every failure stays on the record, and unknowns stay unknown.