HAAK

A platform for institutional intelligence

Zachary F. Mainen · Champalimaud Research · 2026

The core problem is not capability — it is memory and coordination. A mind, however capable, operates within a bounded context: it forgets between sessions, cannot act in parallel, and has no way to accumulate across its own instances. Scientific institutions exist for exactly this reason. A lab, a journal, a funding agency is not a smarter individual — it is a coordination structure that lets bounded minds accomplish what none of them could alone. The insight behind HAAK is that AI agents face the same structural problem, and the same structural solution applies.

Two theoretical foundations ground the platform. The first is the **Library Theorem**: indexed external memory reduces retrieval cost from linear to logarithmic time. An agent with access to a well-organised external record can find what it needs in O(log N) steps regardless of how large the corpus grows. Without indexing, every retrieval is a scan. The institution is not merely convenient — it is the formal precondition for scaling intelligence. The second is a **relational ontology**: every entity in the system — a person, a paper, a claim, a project, an institution — exists through its relations to other entities, not through intrinsic properties. This shapes everything from how data is stored to how agents reason about what they know and how confident they should be.

Architecture

Work in HAAK is organised along three axes. *Actors* are agents — AI instances and humans — each carrying a mandate defining their role, authority, and scope. *Methods* are the procedures they follow: readable protocol documents that describe how a task is done, step by step, with decision points and error vocabularies. *Domains* are the knowledge territories they operate in. Every piece of work sits at the intersection of all three.

Agents are organised in tiers. **Standing agents** hold permanent roles — librarian, archivist, steward, chief of staff — with memory that persists across sessions through externalised records. **Domain agents** are scoped to a project and dissolve when it completes. **Sub-agents** are ephemeral, spawned by any agent for mechanical work and dissolved on return. The tiers allow concurrent work without coordination overhead:

standing agents maintain continuity, domain agents carry execution, sub-agents absorb parallelism.

Everything the system does is externalised — written to files, committed to a version-controlled repository, auditable in full. Agents are mortal: each session ends, context is lost. Externalisation is what allows mortal instances to participate in an institution that persists. A new agent inherits context from the record; an outgoing agent hands off to the record. The institution survives the instance.

A constitutional layer governs the whole: what agents can do without approval, what requires human sign-off, how self-modification works, how privacy is handled. The constitution is itself a document in the repository — readable, forkable, amenable to amendment through the same governance process it defines.

HAAK is domain-agnostic and open source. The same platform that indexes a scientific corpus indexes an art market or a clinical trial register. The methods differ; the architecture is the same.

Application — Scientific claim verification

A scientific paper is a narrative — evidence integrated into an argument optimised for human reading. It is not structured for machines, for systematic comparison, or for panel-level verification. HAAK's claim graph format makes the structure explicit.

The method

An 8-step protocol document describes the ingestion process. Any agent can follow it. Steps 1–4 cover access and extraction: locate the paper, download the full text, run three independent reading passes — one agent reads only results prose, one reads only figure captions, one reads only methods and code — then reconcile into a candidate claim list. Step 5 is a mandatory human review gate: nothing is written until the claim table is approved. Steps 6–8 cover dependency mapping, file writing, and verification against the deposited data.

The claim file

Each claim lives in a single `.md` file with structured YAML frontmatter. A claim is a proposition, not a figure panel. The panel is a property of the *assertion* — the relation between a proposition and a specific paper. An assertion records which paper asserts the claim, in which panel, with which analysis script and data deposit. A reproduction records what happened when someone tried to re-run that analysis.

```
uuid: 26819b09-5b20-4c90-b9f3-8bdd29a2a57c
slug: distal-inhib-drops-firing-02hz
claim: Doubling distal dendritic inhibition reduces somatic firing
       from 5.5 Hz to 0.2 Hz by suppressing Ca²⁺ and NMDA spikes,
       not by raising AP threshold.
epistemic: strong
belongings:
  - relation: requires
    target: l5-model-single-cell-scope
  - relation: supports
    target: beta-gates-distal-apical-inputs
assertions:
  - paper-slug: headley-2026-inhibitory-rhythms
    panel: fig4
    analysis: scripts/Fig4.ipynb        ← original author script, linked directly
    dataset-doi: 10.5061/dryad.v6wwpzhb8
reproductions:
  - status: verified
    script: verification/headley/verify.py
    original_script: https://github.com/dbheadley/.../Fig4.ipynb
```

```
    script_execution: unmodified        ← author's code run unchanged
    time_fast: "~2 min"
    time_full: "~6 hrs (NEURON + 1.88 GB Dryad)"
    notes: control=5.5 Hz, distal=0.2 Hz — exact match. Mechanism confirmed.
```

The dependency edges (`requires`, `supports`) connect claims into a directed graph. If a required claim fails to reproduce, the failure propagates to every downstream claim. Epistemic weakness is structural, not annotated.

The verification script

`verify.py` downloads data from the public deposit, runs the specific analysis, and prints a structured PASS/FAIL comparison. It has a fast mode (pre-computed outputs, minutes) and a `--full` mode that executes the original pipeline end-to-end with the time estimate and requirements stated upfront. The author's original notebook is linked in the claim file — we link to their code, not our version of it.

What this reveals

Applied to ten papers provided by eLife, agents extracted 173 claims and ran verification against deposited data. Two findings would not have surfaced through standard peer review: a case where a deposited statistical map shows activation in occipital cortex where the paper describes fear network activation; and a case where a preregistration was submitted after the manuscript, using timestamp evidence the paper does not disclose. Both findings are now in the claim record, with the original figure, the reproduced figure, and the verification code displayed side by side on a public website: https://zmainen.github.io/elife-claim-trees/