Open source research lab

A research lab where the lab itself is open source. Not just the research outputs — the organizational structure, decision-making processes, resource allocation, and governance are all visible, forkable, and improvable by anyone.

The lab is managed through changes to a public management project on GitHub. Anyone can see how the lab operates, propose changes to its structure, and fork the entire organization to create their own. Offline functionality is inevitable and TBD, but the public repo is the canonical source of truth for maintaining the ethos and openness.

Why

Should also note that this isn't all-or-nothing. Move from politics hell to fully open can be done gradually.

Research labs are opaque. Funding decisions happen behind closed doors. Credit assignment is political. Access to resources depends on who you know. The knowledge produced is paywalled. The organizational knowledge — how to actually run a lab — is never written down at all.

An open source lab inverts this. The structure is the product. If someone disagrees with how resources are allocated, they can see the exact process, propose a change, or fork the whole thing and run it differently. ~~The lab becomes a protocol, not an institution.~~

Relationship to existing institutions

The lab is additive, not exclusionary. It doesn't replace universities, funding agencies, corporate labs, or governments. It runs alongside them.

Researchers can participate in the open lab while holding positions at existing institutions. Grants can come from traditional funders (NSF, NIH, DARPA, private foundations) and flow through traditional fiscal sponsors or university overhead structures. The lab's transparency is about what happens with the resources, not about requiring that resources move through new channels.

The chain layer provides mechanisms for attesting to work associated with money — who did what, funded by whom, producing what output. These attestations are useful regardless of how the money moved. People might eventually choose to move money around only with the guarantees provided by the open lab's infrastructure, but that's their choice, not a requirement for participation.

Change to say as long as alternative power structures exist

Until current systems of government are superseded — and that's a long horizon — people will have to interface with governments through existing institutions for things like taxes, employment law, visa sponsorship, and public funding compliance. The lab's design acknowledges this. A researcher receiving a stipend through the lab still files taxes through whatever legal entity employs them. A grant received by the lab still goes through a fiscal sponsor that handles reporting to the funder. The open layer sits on top of these structures, making the decisions and flows visible, without pretend-

ing the underlying structures don't exist.

The goal is not to tear down existing institutions. It's to build something better in the open, let people use both, and let the better system win over time by being more transparent, more auditable, and more aligned with how researchers actually want to work.

This is implied

Architecture

Need to integrate this with the hierarchica PM doc

Higher-level trees (the lab itself, divisions within it) have two repos: a project management repo (managed by pm) and an organization repo. Individual research projects below them may only have a pm repo and their code repo. The two-repo structure is for organizational layers that need to track more than just a tech tree.

The organization repo

A GitHub repo that is the public record of the organization. Public here means at least to all members — the org can choose to make it world-readable too. This is where things live that don't belong in any individual project but need to be tracked: documents, small scripts, spreadsheets, test programs, analyses, meeting notes, governance decisions, and anything else that should be visible and version-controlled but doesn't need wide release as a standalone project.

It also includes pointers to the other repos in docs and things so it can act like a central dispatch.

Structure:

```
org-repo/
├── docs/                   # golden copy of project state
│   ├── governance/           # decision-making processes, roles
│   ├── research-agenda/      # what the lab works on and why
│   ├── resources/            # grant tracking, allocation records
│   └── onboarding/           # how to join, how things work
├── members/
│   ├── alice/                # Alice's working directory
│   │   ├── notes/            # her meeting notes, scratch work
│   │   ├── scripts/          # one-off analysis scripts
│   │   └── proposals/        # drafts she's working on
│   ├── bob/
│   │   ├── notes/
│   │   ├── data/             # small datasets, spreadsheets
│   │   └── experiments/      # test programs, prototypes
│   └── ...
├── checks/                 # integrity checks (see below)
└── archive/                # completed work, historical records
```

Member directories: every member gets their own directory. This is their workspace within the org. They can put anything there — notes, scripts, Excel sheets, draft proposals, experimental code. It's

theirs to organize however they want. The point is that their work is visible to the rest of the organization without requiring them to publish it anywhere else.

and they can add others as reviewers for PRs when they want to share

When a member wants to promote something from their personal directory to the project-wide golden copy, they open a PR moving or copying it into docs/. This is the mechanism for requesting visibility: the PR itself is a request for the organization to recognize and adopt the work. Other members review it, and the merge is the organization saying "yes, this is now part of our shared state."

The docs/ directory: the golden copy. This is the authoritative state of the project — governance, research agenda, resource records, everything the organization has collectively agreed on. Nothing gets into docs/ without a PR. The PR history in docs/ is the lab's institutional memory.

The checks/ directory: automated integrity checks that project managers create and maintain. These run against the repo (via CI or on-demand) and surface issues that humans should look at.

should also keep a history of automated and human assisted audits

Integrity checks

Integrity checks function similarly to tests in AI-generated code. When AI agents produce code, tests check that the code does what it claims. When an organization produces documents, decisions, and resource allocations, integrity checks do the same thing: they check for ~~BS and~~ possible issues that should be surfaced to people who care.

Examples: should refer to funding more generally since it needs to be managed for both public and private organizations

- **Consistency checks**: does the resource allocation in docs/resources/ add up? Do the numbers in the budget spreadsheet match the grant amounts recorded in governance decisions?

- **Staleness checks**: are there proposals in member directories that have been sitting for months with no PR to docs/? Are there governance documents that reference people who are no longer active?

- **Completeness checks**: does every active grant have an allocation record? Does every member have an onboarding document? Does every research project referenced in the agenda have a corresponding pm instance in the tech tree?

- **Conflict checks**: are there contradictory statements across different governance documents? Did two proposals get merged that allocate the same funds differently?

- **Attribution checks**: does every work product in docs/ have clear attribution? Are there documents that reference work without crediting the member who did it?

These checks can be simple scripts (grep for inconsistencies), Claude- powered analyses (read the docs and flag things that don't make sense), or structured validators (parse YAML/CSV and verify invariants). Project managers create and tune them over time. They're not gates — they surface issues for human judgment, same as a test suite surfaces failures for a developer to evaluate.

The checks themselves are in the repo, so anyone can see what's being checked, propose new checks, or argue that a check is wrong. The checking infrastructure is as open as everything else.

This is key and should be emphasized earlier. Anyone can submit a PR and demo for organizational changes.

Automated recognition

Integrity checks are the negative side — they find problems. The same infrastructure works in the positive direction: automated dashboards that recognize significant achievements in the organization.

operate out the mesh for now. It is a way to enable cross-org data movement and manage trust but not necessary at first!

The data is already there. The org repo tracks work products, the pm repo tracks PRs and merges, the gossip layer tracks resource flows, and the member directories show who did what. Recognition is a read operation on state that's already being maintained.

Examples of what automated recognition surfaces:

- **Milestone completion**: a plan's entire tech tree reaches merged status. The dashboard highlights it — this is a significant organizational achievement, not just another PR.
- **First contributions**: a new member's first PR to docs/ gets merged. Visible on the dashboard. This matters because onboarding is hard and the organization should notice when someone crosses the threshold from observer to contributor.
- **Sustained output**: a member has been consistently contributing work products over a period. Not a leaderboard or a score — just visibility that steady work is happening, because steady work is easy to overlook in favor of flashy launches.
- **Unblocking impact**: a PR merges and unblocks several downstream projects in the recursive tech tree. The dashboard shows the cascade — the person who did that work enabled a whole layer of progress.
- **Funding milestones**: a grant's deliverables are all complete. The chain of funding → allocation → work → output is visible end-to-end. Useful for the organization's own awareness and for demonstrating impact to funders.
- **Cross-project collaboration**: work products that span multiple research projects, or a member contributing to a project outside their usual area. The organization should see when silos are breaking down.

These are generated by the same kinds of scripts and Claude-powered analyses that power integrity checks. They live in the org repo alongside the checks, are tunable by PMs, and are as visible and forkable as everything else. The recognition criteria are not hidden in someone's head — they're code in the repo that anyone can read, propose changes to, or argue about.

The dashboards (mobile web, TUI, or whatever interfaces exist) show these alongside project status. The point is that the organization notices what its members accomplish without requiring anyone to self-promote or anyone in management to remember to say something. The system does the notic-

ing. People decide what to do with it.

Also good something to emphasize this earlier as the ideal and that removes brag.

Critically, the integration with Claude Code means recognition doesn't have to target any specific metrics. Fixed metrics get gamed — lines of code, number of PRs, commit frequency, citation counts. They reward the metric instead of the work. Because everything in the org repo, pm repo, and gossip layer is readable text and structured data, anyone can ask Claude to look at what anyone else is doing and assess their contributions holistically, with context. Claude can read a member's directory, their PRs, the projects they've touched, the discussions they've participated in, and explain what that person has been contributing and where — or whether there are gaps. This is a fundamentally different thing from a dashboard counting commits. It's closer to what a thoughtful colleague would say if they'd been paying attention to everything, which no human can do at organizational scale but a machine intelligence can.

Need to explicitly mention nepotism and close affiliations.

This also means the organization doesn't have to agree on what metrics matter. Different people can ask different questions. A PM can ask "who's been quietly enabling other people's work?" A funder can ask "what did this grant produce?" A new member can ask "who should I talk to about X?" The answers come from the same data but through different lenses, and none of those lenses are baked into the system as the official way to measure contribution. The data is open. The interpretation is on-demand. Nothing is gameable because nothing is fixed.

This is also how claude is the new organizational glue so people can focus on work.

The project management repo

Also should mention how it allows people to work at any place and time.

Separate from the org repo. Managed by pm. Contains project.yaml, plans/, and the tech tree. This is where the work planning happens: what needs to be built, in what order, who's working on what.

And need to be explicit that it's not a tool for bean counting.

The pm repo's recursive tech tree connects the organizational layer to all the research projects below it. Prescriptive mode for lab-wide initiatives (the org decides something needs to happen and suggests it to a project). Descriptive mode for observing what projects are doing without directing them.

It will expose what looks like inefficiency but is actually useful. I.e., tech is already a baby UBI and that isn't bad.

The org repo and pm repo reference each other but are separate concerns: the org repo is the record of what the organization is and has done, the pm repo is the plan for what it's doing next.

But it should be explicit.

The mesh layer (OmertaMesh)

The lab runs on an OmertaMesh where researchers are peers. No central servers. No single point of control or failure.

This is only used for orgs that explicitly want to minimize human contact or allow machine participation.

The mesh provides:

- **Communication**: researchers talk to each other directly, not through a platform someone else controls
- **Discovery**: find other researchers, their projects, their published work — all through the mesh's gossip network
- **Collaboration**: share workspaces, review code, pair on problems without routing through GitHub

or any other third party

- **Resilience**: the lab continues to function if any single node goes offline, including the "founding" nodes Not a real problem

The gossip layer this is just an example of how a network could be built on top of academia. Should move to case study section.

Built on OmertaMesh's gossip network, the lab tracks:

- **Grants and funding**: who received what, from whom, for what purpose. Announced to the network when received. Allocation decisions are proposals that go through governance.

- **Gifts**: unrestricted contributions to the lab or to individual researchers. Tracked in the gossip network so the community can see the flow of resources without anyone having to report to a central authority.

- **Stipends**: regular support for researchers. Terms are public. Continuation is tied to participation (defined by governance docs, not by a manager's discretion).

- **Scholarships**: support for people learning to contribute. Tracked the same way as stipends but with different expectations (learning vs. producing).

- **Work products**: papers, code, datasets, tools, analyses. Published to the gossip network with attribution. The network maintains a living record of what the lab has produced and who contributed.

All of these are gossip-native: they propagate through the network without requiring any central database. Any node can verify the history. The data is eventually consistent — if you're offline for a week, you catch up when you reconnect.

The chain layer (OmertaProtocol) Same as above, add more exposition about the problems with academia being addressed and how this allows visibility with minimal top-down structure like academics prefer. It makes existing implicit systems explicit and directly engineerable.

Later in the tech tree, the gossip-tracked items move to OmertaProtocol. This provides:

- **Immutability**: once a grant, gift, or work product is recorded on chain, it can't be retroactively altered
- **Verifiability**: anyone can independently verify the lab's financial and research history without trusting any single node
- **Interoperability**: other labs, funders, and institutions can read the lab's records programmatically
- **Persistence**: the record outlives any individual node, any individual researcher, and potentially the lab itself

The gossip layer is the fast, informal layer — things show up quickly, propagate naturally, and can be corrected. The chain layer is the permanent record — things are committed deliberately and stay forever.

The transition from gossip to chain is not automatic. It's a governance decision: the lab decides what

gets committed to the permanent record and when. This prevents premature permanence (recording something on chain before the community has had time to review it).

The chain doesn't replace existing accounting, reporting, or compliance structures. A grant that's tracked on chain is also tracked in whatever system the fiscal sponsor uses. The chain is a parallel record that provides independent verifiability — useful for the lab's own transparency and for anyone who wants to audit the lab without relying on a single institution's books.

Tech tree

Don't need specific commands in this doc. Should use higher level diagrams instead. Should also gather data on inefficiencies in academia and orgs caused by politics and clan affiliations.

Layer 0: Foundations

```
pm plan add "Open source research lab"

pm pr add "Organization repo structure" \
  --plan plan-N \
  --description "Create the org repo with initial directory structure: docs/ (golden copy), members/ (per-member directories), checks/ (integrity checks), archive/. Seed docs/governance/ with provisional governance framework: how decisions are made, how membership works, how to change the governance itself. Seed docs/onboarding/ with how to join. Create a CONTRIBUTING guide explaining the member directory → PR to docs/ workflow."

pm pr add "Member directory workflow and templates" \
  --plan plan-N \
  --depends-on pr-A \
  --description "Templates for member directories. PR workflow for promoting work from members/ to docs/. Branch protection rules for docs/ (require review). Example: a member writes a proposal in their directory, opens a PR to move it into docs/research-agenda/, reviewers approve, it becomes part of the golden copy."

pm pr add "Integrity checks framework" \
  --plan plan-N \
  --depends-on pr-A \
  --description "Framework for checks/ directory. Checks are scripts or Claude-powered analyses that run against the repo and surface issues. Initial checks: consistency (do budget numbers add up), staleness (proposals sitting without PRs for >30 days), completeness (every grant has an allocation record), attribution (every doc in docs/ has clear authorship). CI integration so checks run on every PR to docs/. Checks surface issues — they don't block merges. PMs tune them over time."

pm pr add "Lab OmertaMesh network setup" \
  --plan plan-N \
  --description "Bootstrap the mesh network for the lab. Founding nodes. Peer discovery. Researcher onboarding process (join the mesh, verify identity, start participating)."
```

tree), resource flows (from gossip), researcher activity, org repo health (from integrity checks), and recognized achievements (from automated recognition). The dashboard does the noticing — milestone completions, new contributor arrivals, unblocking cascades, funding impact chains — so the organization sees what its members accomplish without requiring self-promotion. Read-only, device-key authenticated."

Layer 3: Permanent record (OmertaProtocol)

```
pm pr add "OmertaProtocol integration design" \
  --plan plan-N \
  --depends-on "pr-K,pr-L,pr-M" \
  --description "Design the gossip-to-chain transition. What gets committed, when, by whose authority (governance decision, not automatic). Smart contract interfaces for grants, work products, attribution. The chain record parallels the org repo and gossip layer — it doesn't replace existing accounting or compliance structures."

pm pr add "Grant and resource tracking on chain" \
  --plan plan-N \
  --depends-on pr-O \
  --description "Move grant/gift/stipend records from gossip to OmertaProtocol. Immutable funding history. Any funder or auditor can verify independently. Attestations of work associated with money — useful alongside whatever traditional accounting the fiscal sponsor uses."

pm pr add "Work product provenance on chain" \
  --plan plan-N \
  --depends-on pr-O \
  --description "Commit published work products to chain. Permanent attribution. Funding links. Citation graph that nobody controls and everybody can verify."

pm pr add "Lab history and institutional memory" \
  --plan plan-N \
  --depends-on "pr-P,pr-Q" \
  --description "The lab's complete record on chain: what was funded, what was produced, who participated, how decisions were made. Survives any single node, any single researcher, any single generation of the lab. The org repo on GitHub is a convenient interface to this record; the chain is the permanent copy."
```

What makes this different needs to be at the start

Most "open science" initiatives open-source the outputs (papers, data, code) but keep the institution opaque. The lab itself — how it decides what to work on, who gets funded, how credit is assigned — remains a black box.

This lab open-sources the institution. The governance is forkable. The resource flows are visible. The decision history is auditable. If someone doesn't like how the lab works, they can see exactly how it works, propose a change, or fork the entire thing.

The mesh and chain layers make this resilient. The lab doesn't depend on GitHub staying up, or any cloud provider, or any single country's legal framework. It's a protocol running on a peer network. The management repo on GitHub is a convenient interface, not a dependency.

Open questions

- **Legal entity**: does the lab need a legal entity for receiving grants? Probably yes, at least initially. How does a legal entity map to an open-source governance process?
- **Conflict resolution**: governance PRs can be contentious. What's the escalation path? Probably: discuss on mesh, propose on GitHub, decide by whatever process the governance docs specify (which is itself changeable via PR).

no privacy, need to be explicit that this is the trade bat it

- **Privacy**: some resource flows (individual stipend amounts, scholarship applications) may need privacy. How does this work with gossip transparency? Probably: amounts can be public while personal details are private, with the individual choosing what to disclose.

inevitable with AI aggregated data leakage anyways, just making

- **Bootstrap problem**: the first version of governance has to come from somewhere. It can't be voted on by a community that doesn't exist yet. Accept this and make the initial governance explicitly provisional — "this is the starting point, here's how to change it."

that data publicly visible

- **Forking**: if someone forks the lab, do they fork the resource history? The reputation history? Probably: they fork the governance and structure but not the resources (you can't fork money) or reputation (you can't fork trust). They start fresh with a new network that happens to use the same rules.

Give some examples of corporate backed parallel construction scenarios that will happen otherwise
- healthcare savings examples from real life already at IBM on twitter

Also need to be clear about the benefit for smaller investors. Any non-AI-visible organization is relatively much more risky. It will become a standard requirement of startups to minimize risk. Not doing so will be a signal of otherwise untrustworthy behavior.