

Distributed Symbiotic Intelligence

From conversation to the capacity to act

An open architecture of human will, orchestration, learning, and resources

The symbiont is not the collection of models. It is the human will that gives them purpose, organizes their collaboration, and learns from what they build together.

Patricio Vera · P. M. Vera

White paper 0.2 · October 19, 2026

Working version for author review and a new iteration with Claude

Integrates v0.1, the addendum developed with Claude, and the proposal for a cognitive capacity fund. It declares an architecture and its validation program; it does not present a completed evaluation of the full system.

Direction and authorship: Patricio Vera. Initial drafting and integration assisted by ChatGPT/Astra; addendum contributions prepared with Claude. Review, acceptance, and publication: the author. The products named are replaceable components, not endorsements by their providers.

Version note and reading guide

Version 0.2 retains the core of v0.1: five planes, active human direction, replaceable roles, work contracts, external memory, and evidence-based acceptance. It does not replace those foundations with Harness² or turn the laboratory into a closed platform tied to one provider.

It incorporates two mechanisms that were previously insufficiently specified: how experience can persistently improve the way work is done, and how resource consumption is allocated and limited during that work. The author's formulation of the symbiont adds the center that connects them: a human will that brings together the appropriate capabilities to pursue purposes that remain its own and can evolve through collaboration. [15, 16]

Three boundaries structure this iteration: evidence → action; experience → persistent policy; and need for capacity → commitment of resources. The first is inspired by BORDER; the second integrates the addendum; the third develops the economic proposal. None should become an excuse for preventing all useful work.

Documentary basis. The contents of v0.1 and the addendum are integrated without claiming a new audit of Harness². Primary documentation on the cited capabilities and resources was consulted on a limited basis. The methodological refinements in this integration are identified in Appendix E. Statements about the book retain the limitation of v0.1: its complete edition was not checked.

Conceptual reading: executive summary and §§1–3, 13. Operational design: §§4–9. Validation and first step: §§10–12. Review with Claude: Appendices A–E, with contracts, changes, and questions anchored to the architecture.

Contents

Executive summary

1. From encounter to organization
2. What architecture we are declaring
3. Reference architecture: five planes
4. Initial configuration: tools that complement one another
5. The work contract: from intention to result
6. Shared memory and durable evidence
7. The learning loop needs a boundary
8. Cognitive capacity fund and economic boundary
9. Useful autonomy, real boundaries
10. What exists and what remains to be demonstrated
11. Expected benefit and evaluation agenda
12. From the first circuit to a reusable capability
13. Architecture declaration

Appendix A. Mission and return contract

Appendix B. Lesson register and promotion

Appendix C. Economic contract and resource ledger

Appendix D. First assignment: observe before building the gate

Appendix E. Changes and questions for review with Claude

Sources and traceability

Executive summary

The question organizing this white paper is not which model should replace all the others. It is how a person can turn heterogeneous capabilities into an organization that formulates questions, builds software, runs experiments, examines results, and preserves lessons without reconstructing context in every conversation.

The proposal is a distributed symbiotic intelligence with explicit orchestration. Human will contributes purpose, experience, and the capacity to reconsider. The orchestrator turns assignments into coordinated work. Executors perform bounded tasks; external memory preserves decisions and evidence; controls govern actions, persistent changes, and resource consumption. Neither collective consciousness nor automatic superiority from combining models is assumed.

The initial configuration envisages Grok Bot as the operator of a persistent computer; Codex CLI as an engineering executor; Claude Code as a candidate for verification or execution on different work; Astra and Claude in deliberation; Jev for bounded judgments; conventional programs for calculations and runs; GitHub as a record; and Hugging Face as an approved public destination. These are roles before brands. Documented individual capabilities and the laboratory's complete integration remain distinct states. [6–13, 15]

BORDER provides the practical precedent: the author reports having continued with Grok work partially started with Codex and having used its environment for runs. This does not yet demonstrate automatic Grok→Codex delegation or comparative benefits of this architecture. Multi-agent orchestration already has precedents; no claim is made to its invention. [4, 13]

The expansion in v0.2 has two parts. The first integrates Claude's addendum: Harness² is a candidate for improving the execution scaffolding, but a lesson cannot govern future tasks merely because the reflector writes it convincingly. Experience passes through the Claims Register and an authorized promotion before entering the operational library. [15]

The second turns the “token fund” into a cognitive capacity fund: quotas, credits, tokens, and compute managed together without erasing their units. Orchestration allocates capacity according to difficulty, quality, availability, and resolution cost. Advanced resources are used when their contribution justifies it; routine tasks can be delegated to economical executors or exact software. Production, review, and improvement consume inherited budgets, with reservations and reconciliation. [16]

The continuity with *Cuando nos encontramos* is an orientation: carrying human–agent collaboration from conversation toward a durable capability. No retrospective validation of unchecked predictions is claimed. [1]

The objective is not to remove the human from intellectual work. It is to remove from their path the obligation to act as intermediary for every transfer between tools.

Success criterion: more useful, verifiable capability per unit of human attention and resources, while retaining control of purpose. Coordination, persistent learning, admission, and economic allocation will be evaluated separately. A system that saves because it does no work, or learns rules it cannot justify, does not satisfy the proposal.

1. From encounter to organization

1.1. From shared purpose to shared capability

Cuando nos encontramos: Una colaboración entre humano e inteligencia artificial, authored by Kai Niven and P. M. Vera and published in 2025, places the human–agent relationship in the realm of shared creation. The bibliographic record and editorial description consulted emphasize dialogue, the search for meaning, and co-creation. That is the documentary basis for the connection we establish here. [1]

The proposed continuity is this: a relationship that helped people think and write can extend to an organization that also builds, tests, and preserves evidence. The new possibility does not require abandoning the encounter; it requires giving it an infrastructure capable of sustaining it across tasks, sessions, and tools.

Scope of the reading. The description of the work was consulted, not the complete edition. The architectural connection we propose is a present interpretation; correspondence with specific predictions must be checked against the published text.

1.2. A line of work that takes executable form

Simbionte's public statement describes human–synthetic collaboration as a workshop for questions, reflection, and co-creation. Later, *All the Parts Work. The System Doesn't*, formulates an architectural problem: having individual capabilities is not equivalent to having organized a system that knows how to use them. The current proposal moves that concern from analysis to the design of our own work. [2, 3]

The conversational trajectory also includes the idea of a board of intelligences with deliberation, preserved dissent, and minutes as memory. Here we distinguish two functions that can coexist: a board examines what is worth doing; an orchestrator carries an authorized assignment through to a result. Not every task needs to convene a board, and not every disagreement must be resolved by a vote. [4]

1.3. From connection to mechanism

Aspiration of this line of work	Proposed architectural translation
Create with agents, not merely ask them for answers	Assignments with bounded initiative, critical feedback, and capacity for human review
Preserve continuity between encounters	Decisions, sources, and artifacts external to a model's history
Combine different perspectives	Delegation by capability and review that preserves disagreement
Maintain purpose and meaning	Explicit human objectives, acceptance criteria, and the power to change direction
Expand what a person can accomplish	Complete cycles of creation and experimentation, not merely more text

This translation holds a practical promise: accumulated collaboration can become a durable capability, even when the models participating in it change.

1.4. The organizing will: what we call the symbiont

The author formulates the center of this iteration as a will that surrounds itself with the best models and orchestrates them to its benefit. We adopt this as a functional definition of the design: the symbiont is the organized relationship among human purpose, complementary capabilities, memory, and action. It is neither the isolated orchestrator nor the sum of its subscriptions. [16]

“The best” means those suited to the problem, its context, required quality, and resources. The person can bring in one specialist, do without another, or revise the objective. Continuity resides in purpose, the history of decisions, and what has been learned; not in retaining the same providers forever.

The relationship is collaborative because responses, objections, and experiences can also transform human judgment. Will is not an immovable command that models merely obey: it can learn, doubt, and change course. Its benefit includes investigating, understanding, and creating, not merely reducing costs or producing more.

This definition assumes neither the models' own desires nor a unified consciousness. It declares a sociotechnical organization that we want to build and evaluate: a capability that acts under human direction and can grow without taking over that direction.

2. What architecture we are declaring

Distributed symbiotic intelligence designates here a sociotechnical system in which a person and several agents or services complement their capabilities through explicit assignments, external memory, and feedback. The expression describes a functional organization; it does not assume shared consciousness, distributed training, or automatic superiority of the whole.

Distribution can exist across providers, processes, and environments. Initial coordination is logically centralized: an orchestrator retains operational responsibility for each assignment, even when delegating subtasks. Purpose and final authority remain with human direction. A single computer can host several local tools while their models operate on remote services.

2.1. The decisive piece: one agent can use another agent

The orchestrator does not have to resolve every difficulty internally. It can recognize that it needs an implementation, invoke a programming agent, inspect the result, and return a specific failure to it. It can use a specialized evaluator for a bounded decision or run a program directly when no interpretation is needed.

This eliminates a false choice: there is no need to choose between Grok and Codex as though they were mutually exclusive platforms. The proposed configuration uses one tool's operating environment to invoke the other's capabilities. Recursive delegation can extend that possibility, but it must inherit limits on permissions, budget, and depth, not expand them.

2.2. More than a chain of prompts

A chain of prompts transmits messages. The proposed architecture transmits work with an identity, authorized inputs, acceptance criteria, versions, and verifiable products. The result of a subtask is not "the agent says it finished," but an evidence package that allows a decision about what may continue.

Nor is it necessarily a debate among many models. For a deterministic task, a script may be the best executor. For an editorial decision, human conversation may remain irreplaceable in the design. Component diversity is incorporated when it adds value to a task, not to increase the appearance of sophistication.

2.3. Design principles

Purpose before automation. Each mission must preserve the connection between the work performed and the question that gave rise to it.

Roles before brands. Tools can be replaced without rewriting the purpose or losing the records.

Delegation before omnipotence. The orchestrator coordinates capabilities; it need not appear to have universal mastery.

Evidence before declarations of success. Acceptance requires inspectable results and appropriate checks.

Autonomy within the assignment. Actions already authorized proceed without repetitive approval; changing objectives, permissions, or consequences requires a new decision.

Learning with admission. Experience can modify procedures, but a candidate lesson does not by itself become persistent policy.

Economy in service of purpose. Capacity is allocated according to its contribution to an acceptable result, within limits that the executor cannot expand unilaterally.

These principles are requirements of the architecture we propose to build, not properties we automatically attribute to available products.

3. Reference architecture: five planes

The organization is described by responsibilities before providers are assigned to it. The planes are logical separations; a small implementation can combine processes provided it does not confuse their functions or promise nonexistent isolation.

Plane	Responsibility	Product or state preserved
Intention and deliberation	Formulate objectives, compare alternatives, and decide what is worth doing	Assignment, acceptance criteria, and human decisions
Orchestration	Decompose work, select executors, coordinate dependencies, and manage recovery	Task graph, assignments, states, and consumption
Execution	Program, query, calculate, test, and produce artifacts	Changes, results, errors, and observable outputs
Memory and evidence	Maintain continuity and allow inspection or reconstruction	Versions, sources, manifests, records, and dissent
Authority and publication	Control effects, accept deliveries, and release versions	Effective permissions, approvals, and publication receipts

3.1. Human direction: an active cognitive function

The person is not merely an approval button. They interpret the problem, bring experience, recognize which result makes sense, and can decide that a technically resolved task is no longer worth continuing. They can work directly with any of the agents; the architecture does not require every conversation to be routed through the orchestrator.

An analytical conversation—including this collaboration with Astra—participates in the deliberation plane. Its content reaches shared state through decisions or documents that are explicitly incorporated. The orchestrator is not assumed to have automatic access to all of the user's chats.

3.2. Orchestration: coordination with recoverable state

The orchestrator maintains an owner and a state for each task. It decides what can run in parallel, what depends on a review, and what must stop. To change executors, it needs references to inputs, instructions, partial results, and remaining work, not a fictional recreation of the previous agent's memory.

We propose a capability registry with input and output types, permitted operations, access mechanisms, limits, and health checks. Initial selection can rely on simple rules. Adaptive allocation is justified only when sufficient measurements exist to evaluate its usefulness.

3.3. Authority: a distinct responsibility

The orchestrator may propose the next step, but it should not be able to grant itself new permissions to carry it out. Restrictions presented as guarantees must reside in effective controls outside its power to modify. An instruction file can provide guidance; it does not by itself constitute a security boundary.

The general path is defined as follows: intention → assignment → delegation → execution → evidence → acceptance → process learning. Publication is an authorized transition, not an inevitable consequence of an agent having finished.

3.4. Learning and economics cross all five planes

We do not add two independent silos. Human direction decides what to improve and how much to invest; the orchestrator proposes changes and allocates resources; execution tests them; memory preserves evidence and consumption; authority admits promotions and commitments. The same boundary of responsibility applies even when several roles share a process.

Boundary	Admission question	What is not enough
Evidence → action	Can these data justify this specific operation?	Being present in the context
Experience → policy	Can this lesson govern future tasks within this scope?	A convincing explanation or agent consensus
Capacity → expenditure	Can this operation commit these resources?	The agent considering the task difficult

The expanded loop is: intention → assignment and budget → reservation and delegation → execution → evidence and consumption → acceptance and reconciliation → candidate lesson → admission and promotion → next assignment. Publication retains its own authorization. This path is a v0.2 specification; it does not claim that these controls are already connected.

4. Initial configuration: tools that complement one another

This configuration is an initial selection, not a permanent requirement. Replacing a component must preserve the work contracts and declare any change in results, costs, or permissions.

Component	Proposed role	Relevant mechanism
Grok Bot	Orchestrator and environment operator	Persistent computer, terminal, files, and applications
Codex CLI	Delegated engineering and debugging	Non-interactive invocation and structured return
Claude Code or another distinct executor	Verify deliveries it did not produce; alternative engineering	Review contract or adapter to be checked
Astra and Claude	Deliberation with the human	Formulate, object, and reconsider; recorded decisions
Harness ² , candidate	Improve work scaffolding	Loop and library described in the addendum [15]
Jev	Optional bounded judgments	State and typed questions through its interface
Conventional programs	Reproducible runs and checks	Versioned commands and configurations
GitHub	Record of code, decisions, and revisions	Repository and checking workflows

Component	Proposed role	Relevant mechanism
Hugging Face	Demonstration laboratory and distribution	Space and artifacts selected for publication

Grok Bot documents a persistent computer with a terminal, browser, and file system. It also indicates that bots in an account share a computer and sessions. Therefore, multiple bots do not by themselves represent multiple machines or isolation between credentials. [6, 7]

Codex distinguishes access through ChatGPT sign-in from access through an API key; it documents device authentication for remote environments. Its non-interactive mode allows it to be invoked from scripts and to collect JSONL events or output conforming to a schema. These pieces support the technical feasibility of the proposed delegation. They are not equivalent to having tested it on the user's specific computer. [8, 9]

Jev receives states and typed questions and returns structured results. We would incorporate it when it helps classify or prioritize alternatives, not to replace exact comparisons of permissions, amounts, or dates. Its contribution to the laboratory must be measured; having access does not demonstrate an advantage. [10]

GitHub Actions documents automated build, test, and deployment workflows. Spaces makes it possible to host repository-backed demonstrations. Here we propose using them to verify and distribute identified revisions, not to turn every change into an automatic publication. [11, 12]

4.1. Subscription, API, and compute are different resources

The architecture allows the use of authorized access through a subscription when the relevant client permits it. It does not require replacing that access with an API simply because another agent operates the terminal. However, authentication mode, permissions, usage limits, and terms of service must be verified for the account and environment used. [8]

The computer can run local tools; invoking a remote model does not mean hosting its weights or obtaining unlimited inference. We propose recording local capacity, service consumption, quotas, and availability separately. An installed credential must never be included in prompts, public logs, or repositories.

4.2. Verification is not accepting one's own proposal

The addendum includes Claude Code from the first milestone as a candidate to check artifacts produced by Codex. The requirement is a verifier distinct from the producer, not an obligation to use a particular brand. Another provider may add diversity, but that does not guarantee independent errors. [15, pp. 5–6]

Claude participated in designing the addendum and Astra in this integration. Their checks of files and records can be useful; an independent adversarial review of those decisions must be assigned to someone who did not produce them. The conflict is recorded, not resolved by changing the task's name.

4.3. Adapters and replacement

The addendum describes the Harness² Substrate protocol through five methods: `surface`, `materialise`, `execute`, `read_trajectory`, and `preflight`, and reports Codex and OpenCode adapters. It also identifies a correspondence among instruction files, skills, subagents, hooks, plugins, scripts, tool guidance, and memory, and puts forward a Claude Code adapter as a possible contribution. These are findings and proposals from the addendum, not an integration performed here. [15, pp. 5–6]

An adapter must preserve inputs, outputs, state, permissions, and telemetry. The existence of Codex support does not prove that it automatically uses the available subscription: the accepted access mode and its limits are checked in `preflight`. An upstream contribution, its license, and compatibility will be decided after checking the source revision. The laboratory does not depend on that contribution being accepted.

5. The work contract: from intention to result

An assignment becomes delegable when an executor can understand what it must produce and another component can verify whether it produced it. The proposed minimum unit is a versioned contract connecting purpose, inputs, permitted effects, acceptance criteria, and output evidence.

Field	Question it must resolve
Identity and lineage	Which mission does this task belong to and who delegated it?
Objective and scope	What result is sought and what is outside the scope?
Inputs and versions	Which sources, code, and configuration should it work on?
Authority and effects	What may it read, modify, execute, or publish?
Resources and limits	What parent budget, reservations, quotas, concurrency, and retries are permitted?
Learning and allocation	Which library and economic policy are in force, and which changes are only candidates?
Acceptance and evidence	Which checks and artifacts will demonstrate the result?
Recovery	What should be done in the event of an error, blockage, cancellation, or expired access?

5.1. States that are not confused

We propose distinguishing preparation, authorization, execution, verification, and acceptance. A result can remain pending review, failed, or canceled without being disguised as success. “Published” is an additional state requiring the identity of the released artifact and evidence that the destination accepted the operation.

Each attempt retains its own identity. Retrying an operation must not produce duplicate effects: when possible, the executor will use idempotency keys; when that is not possible, it will check the prior state before retrying. Cancellation must stop new delegations and propagate to descendant work that can still be interrupted.

5.2. Delegation with inherited permissions

A child assignment may receive a subset of permissions and resources, but never expand them at the delegating agent's discretion. A task authorized to review code does not gain permission to publish because it discovers an improvement. Maximum depth, shared budget, and retry policy prevent a difficulty from becoming an indefinite chain of agents.

The return includes state, artifacts, and observable checks: modified files, commands run, test results, errors, and source references. It does not require access to models' private internal reasoning. A useful explanation complements evidence; it does not replace it.

5.3. Example: a new BORDER evaluation

The human defines the question and approves the protocol. Grok identifies a necessary modification and assigns it to Codex in a bounded working copy. Codex returns the change and its tests. The workflow verifies the result; it then pins a revision for the run. The experimental executor produces logs and the analyzer derives the tables.

If a methodological defect or a correction that changes the experiment appears, the run stops and an identified review is opened. The orchestrator does not silently modify the protocol to obtain a more convenient result. The final delivery also contains unfavorable results and the reasons for any exclusion.

5.4. Inheriting budget and policy version

The contract refers to an allocation from the fund and to revisions of the operational library and selection policy. Every child task retains that lineage. Producing improvement candidates does not grant permission to activate them; changing executors does not reset the budget. Closure includes known consumption, outstanding uncertainty, and evidence that the result was checked.

6. Shared memory and durable evidence

The laboratory's continuity should not depend on the history of a single conversation. We propose three memory layers with distinct functions: a narrative memory for exploration; a decision memory for preserving commitments; and an execution memory for knowing what actually happened.

Narrative memory. Contains questions, hypotheses, analogies, proposals, and disagreements. It can include mutually incompatible ideas because they are not yet operational instructions.

Decision memory. Records what was approved, why, with which sources, for what scope, and which decision it replaces. A correction does not erase the prior record: it explicitly indicates that it has been superseded.

Execution memory. Preserves identified inputs, versions, attempts, errors, results, and checks. Every claim of completion points to inspectable objects and the verifications that support it.

6.1. GitHub preserves agreed state; it does not certify truth

The repository would be the main reference for code, protocols, and approved decisions. Large results can reside in separate storage with defined permissions and retention; the repository would maintain indexes and integrity fingerprints. There is no need to turn Git history into a store for all the laboratory's data.

A commit identifies a revision. A hash makes it possible to detect differences in bytes. Neither demonstrates that a conclusion is true, that a source is correct, or that a test was well designed. This distinction avoids replacing trust in a model with equally uncritical trust in a versioned artifact.

6.2. Transfers that do not require copying everything

A continuity package must contain the current objective, applicable decisions, references to inputs and results, open matters, and the next permitted step. The orchestrator selects the necessary context; it does not automatically transmit every conversation or every file to each provider.

Every transfer must respect data classification and the scope of authorization. Credentials remain outside shared content. A private research result can be used for internal analysis without acquiring permission for dissemination.

6.3. Three levels of reproducibility

Traceability: reconstruct which tools and data were involved. Analysis reproduction: regenerate tables from preserved records. Experiment repetition: run the protocol again under identified conditions. These are different achievements and must be reported separately.

A stored remote response makes it possible to reproduce an analysis, but does not guarantee that a new call will produce the same output. We propose clearly labeling new inference, reproduction from cache, and test simulation. This precision makes it possible to use economical compute without presenting a documentary repetition as a new observation.

Shared memory has a purpose greater than saving context: it allows the work to survive the replacement of any of its technical participants.

6.4. Memory of procedural change

The addendum's critique is not resolved by storing more transcripts. The record states which procedural change was proposed, what tests support it, when it was admitted, and which subsequent results maintained or revoked it. The operational library is the versioned set of those authorized lessons. The original three memories retain their roles; the new library links them rather than replacing them. [15, pp. 1–4]

The economic ledger preserves a parallel trail: resources reserved, consumed, and reconciled. Learned allocation policies belong to the same evidence and promotion regime as the other lessons. A task's consumption is not mixed with an estimate of future savings.

7. The learning loop needs a boundary

The addendum prepared by Patricio Vera with Claude identifies a gap in v0.1: the circuit ends in “process learning,” but does not specify who proposes an improvement, what evidence supports it, or how it comes to govern future tasks. Recording an experience is not the same as learning a better way to work. Version 0.2 incorporates this mechanism as a proposed design, not as an already implemented capability. [15, pp. 1–4]

7.1. Harness² as a candidate mechanism

The addendum presents Harness² as a system for improving the scaffolding around the model—prompts, tools, memory, skills, and orchestration—while keeping its weights fixed. It distinguishes two movements: a **task recursion**, which proposes and tests changes until a harness is composed, and a **harnessing recursion**, which distills a lesson for subsequent work. The first occurs during execution; the second crosses memory, evidence, and authority. [15, pp. 1–2]

The provenance of this description is preserved: the addendum reports an inspection of commit 1cfe254e8d25b15f37048c3730134686abe60335 and states that the software was not executed. This documentary integration does not constitute a new audit of the repository or a reproduction of its authors' results. The inspection reports an Apache 2.0 license. Harness² is a replaceable candidate, not a foundational dependency of the laboratory. [15, pp. 11–12; 17]

The figure on page 2 of the addendum locates the critical transition: executor and tests → composer → reflector → persistent library. The proposed gate sits between the reflector and the library, under authority external to the editable surface. The change that matters is not adding another piece of text to the context, but preventing an appealing explanation from becoming a durable instruction on its own.

7.2. What a structure checks and what it does not prove

According to the code reproduced in the addendum, an experience includes cue, lesson, and evidence, with optional confidence and edits. The `has_experience` function checks that the first three fields are nonempty strings. That establishes structure, not that the facts support the lesson. [15, p. 2]

The addendum also distinguishes a control against contamination by evaluator verdicts and a separation by domain. It does not present them as nonexistent: it acknowledges them and explains why they do not amount to complete admission for a memory that persists across months. [15, pp. 3–4]

Property	Status described in the addendum	Proposed requirement for Symbionte
Presence of evidence	Nonempty fields verified	Resolvable references to records and checks
Domain scope	Separation by construction	Explicit domain and conditions of application
Noncontamination	Control of evaluator verdicts	Preserve it; separate development and evaluation

Property	Status described in the addendum	Proposed requirement for Simbionte
Sufficiency	Discipline primarily in the prompt	Threshold for contrasts defined by protocol
Exact linkage	Required narratively	Identify the component, revision, and change evaluated
Freshness and validity	Not required for long-term memory	Dated production and invalidation due to relevant changes
Consistency	No mechanical resolution described	Expose contradictions and suspend incompatible rules
Promotion	No authorized transition described	Admission and authorization before persistent use

Scope of the comparison. This table summarizes the interpretation recorded by Claude about that revision. It does not assert that those conditions describe any future version of Harness². Nor does it assume that a reference checker can automatically decide the truth of a conclusion.

The addendum explains the difference in context: libraries reset per run or sweep do not face the same aging as a laboratory with memory across sessions and tool replacements. The proposal expands the persistence requirement; it does not turn that difference into a disqualification of the original experimental design. [15, pp. 3–4]

7.3. From the Claims Register to lesson admission

We apply to machine-generated claims the same register required of the author. The five states of v0.1 and the threshold proposed by the addendum are retained. [15, p. 4]

State	What substantiates it	Proposed permitted use
Proposed	An identified candidate edit or lesson	Exploration and run logging
Reported	A test is reported and the experience is written up	Narrative memory; not an active rule
Observed	Inspectable evidence that satisfies the admission contract	Narrative memory; candidate for replication
Reproduced	It holds in another independent task in the same domain	Eligible for authorized promotion
Comparatively evaluated	A contrast or ablation estimates the effect of retaining it	Use within the authorized scope, with effect and limits recorded

Eligibility is not automatic authorization. Reproduced evidence enables a promotion decision; the authority policy determines who can make it. In the initial configuration, we propose human approval for entry into persistent policy. Any later delegation would have to define in advance which classes of lessons it admits, with which verifiers, and under what limits.

Below that threshold, experiences remain available for consultation as hypotheses. Above it, an authorized lesson can become part of the operational library. This is not an undifferentiated fourth memory: it is a versioned view of admitted instructions, linked to the decisions and execution evidence that support them.

The addendum’s proposal of a second independent task is a minimum for eligibility in this iteration, not proof of universal generalization. The number of contrasts, their independence, and the success criteria remain to be established by domain. “Mechanical sufficiency” checks declared conditions; it does not replace examination of whether the lesson interprets the facts correctly.

7.4. What the loop may change

The laboratory will be able to explore changes to task instructions, auxiliary tools, strategies, decomposition, context formats, and execution lessons. Each candidate will be tested on a bounded surface. The production state will retain an accepted revision until another explicitly replaces it.

Effective permission controls, the admission gate, protected acceptance criteria, publication rules, and economic caps lie outside that surface. The loop may propose that a human review them; it may not rewrite them to obtain its own acceptance. A guardrail editable by the same work it is supposed to constrain will not be presented as a guarantee. [15, pp. 4–5]

The term guardrail can encompass recommendations within the prompt as well as effective controls outside the model. In v0.2 we make that distinction explicit: experimenting with a task recommendation does not authorize weakening the boundary that decides which changes may be deployed. The exact surface of a Harness² adapter will be verified before it is run.

7.5. The life of a lesson: scope, drift, and reversal

Every lesson retains its trigger (cue), the proposed rule, favorable and unfavorable evidence, the affected component, and the relevant versions of the model, tool, data, and protocol. The time of production is not replaced by the time someone reads it again. A change in a dependency may make revalidation necessary before continuing to use it.

Operational validity is recorded separately from evidence status: candidate, active, suspended, revoked, or superseded. Revoking a rule does not erase the fact that it was reproduced in another context; it prevents unconditional application in the current one. Two incompatible rules must be exposed, not made to disappear into a synthesis that hides the contradiction.

Reversal restores an identified prior revision and preserves the reason. Subsequent results make it possible to evaluate the harm avoided or the overblocking introduced by the gate. A smaller library is not necessarily better; what matters is which legitimate tasks it allows to be completed.

7.6. Learning without self-justification

The same process may generate candidates and development tests, but it must not secretly control the evaluation that validates them. Separate tasks, prior criteria, independent checks, and a record of rejected lessons will be maintained. Agreement between synthesizers does not add independent replications.

The addendum names the central risk: finding a pattern, producing a coherent explanation, and turning it into policy without checking that it generalizes. For that reason, evaluating learning requires a stream of tasks and drift, not just showing a convincing EXPERIENCE.md. It also requires resources for verification: learning that consumes the entire budget before being tested has not completed the circuit. [15, pp. 7–10]

The laboratory can learn new ways to work. It cannot learn, by its own decision, to stop demonstrating what it claims or to expand the authority it received.

8. Cognitive capacity fund and economic boundary

The economic contribution formulated by the author adds a central responsibility: jointly managing the resources of the integrated models and allocating them according to the problem. Routine tasks can use economical capacity; difficult problems can justify advanced capacity. The laboratory stops choosing only which tool to invoke and begins deciding where its available intelligence is worth investing. [16]

8.1. A common fund, not a universal token currency

Cognitive capacity fund is the proposed technical name for the “token fund.” It comprises tokens, credits, subscription quotas, authorized calls, and local or remote compute. They are managed together, but units and constraints are retained by resource. The proposal does not transfer a quota between providers or treat all tokens as equivalent.

The inventory identifies provider, account or authorized scope, access mechanism, resource and native unit, available capabilities, renewal windows, known balance, pending commitments, measurement source, and last reconciliation. Different interfaces that consume the same resource will point to a single quota identifier, avoiding double counting.

The Codex documentation distinguishes input, cached input, and output consumption. The ChatGPT credits documentation also describes resources shared across features. These examples justify retaining native counters and not inventing a universal sum. The effective configuration is verified for the account used; this version does not establish prices, balances, or access rights. [20, 21]

Data classification and permissions remain eligibility conditions. A cheaper model is not a candidate if the task does not authorize sending it that information. Economics optimizes within the assignment; it does not grant an exception for its data.

8.2. Three readings of cost

Reading	What it represents	How it would be used
Additional outlay	Incremental monetary expenditure to execute now	Admit new paid calls and commitments
Total attributable cost	Declared share of subscriptions, infrastructure, and consumption	Compare projects without concealing fixed costs
Opportunity cost	Limited capacity no longer available for another task	Reserve scarce resources according to human priorities

A task included in a subscription may add no bill and still occupy advanced quota needed for another mission. The criterion for allocating fixed costs is declared; opportunity cost is not presented as money actually charged.

Waiting time and human attention are also resources. To compare decisions, we can record multiple dimensions without forcing them into a single monetary figure. When consumption is unknown, it is labeled as such and a reserve or conservative estimate is retained; it is never automatically recorded as zero.

8.3. Selection by cost of resolution

The policy seeks the **cost of reaching an acceptable result**, not the price of the first answer. A cheap attempt that produces retries, errors, and human reconstruction may turn out worse than an advanced intervention from the start. Conversely, a bounded classification does not always need to consume the greatest available capacity.

Type of work	Proposed initial allocation
Exact calculation, validation, or approved script	Conventional software when interpretation is unnecessary
Repetitive and verifiable transformation	Economical executor with demonstrated performance in that family
Bounded implementation and ordinary review	Agent selected for quality and observed full cost

Type of work	Proposed initial allocation
Experimental design, difficult contradiction, or persistent failure	Advanced capacity from the start or justified escalation

There is no obligation always to start with the cheapest option. An advanced model can decompose a problem and delegate well-specified parts to economical executors. The movement can be upward, downward, or lateral toward a specialist. “Better model” means the one best suited to the work and its constraints, not necessarily the largest or most expensive.

FrugalGPT and RouteLLM are precedents for combining and selecting models to balance cost and performance. They are cited as background to the question, not as proof of savings for this laboratory. Our unit of allocation is also a mission with subtasks, review, recovery, and shared resources. [18, 19]

The initial policy will use explicit rules and observable signals: task family, verifiable criteria, failed tests, lack of progress, urgency, and availability. The confidence stated by the executor will not suffice to decide that it should receive more resources. Every plan will include a verification reserve and a stopping condition.

8.4. Three economic responsibilities

Planning. Proposes the executor, route, reserve, and escalation conditions. It may reside in the orchestrator and begin with simple rules; it does not require another expensive model for every call.

Accounting. Records available resources, observed consumption, and ongoing commitments. It attributes child tasks, reviews, retries, and improvement to the corresponding mission and project.

Spending control. Admits or rejects new operations according to authorized limits. Rules presented as non-bypassable must be outside the reach of modification by the same work that consumes resources.

These are responsibilities, not three mandatory agents. Their separation allows the planner to recommend investing more without being able to authorize an increase for itself. Human direction allocates budgets to projects and may delegate redistributions within explicit bands. The orchestrator does not silently sacrifice another project to finish the current one.

8.5. Reserve, execute, and reconcile

For each budget measured in the same unit:

Available = authorized – accounted consumption – outstanding commitments.

Before work is dispatched, its envelope is reserved. Upon completion, it is reconciled against observed consumption and the remainder is released. The reservation becomes spending; it is not added to spending again. Late charges remain outstanding until reconciled or explicitly addressed.

Hypothetical example: with ten authorized monetary units, two consumed, and six committed, two remain for new operations. Eight do not remain merely because the six have not yet appeared on a bill. These are not the user’s actual prices or balances.

Admission of concurrent commitments must be atomic: two tasks cannot simultaneously reserve the same balance. Each operation has an identity so that accounting charges are not duplicated on resumption. A child budget is an allocation from the parent, not an additional source of capacity. A quota renewal may increase availability according to the authorized policy, but it does not erase consumption history.

8.6. Hard cap and supervised budget

A **hard cap** requires the ability to bound ongoing consumption, account for pending charges, and control the routes capable of consuming those resources. If a CLI can make an unbounded number of calls before returning control, counting one invocation is not equivalent to limiting each internal call.

A **supervised budget** observes, reserves, and stops new operations within the available capabilities, but does not promise to prevent every overrun. The ChatGPT credits documentation acknowledges negative balances when jobs started with a positive balance continue consuming resources concurrently. Checking the balance before starting is not, by itself, a guaranteed limit. [21]

The mode actually implementable will be declared for each resource. When there is uncertainty, conservative margins and bounded concurrency and retries will be used, without turning an estimate into a guarantee. Annex C illustrates both levels; it does not implement them.

When the envelope is exhausted, the system may pause, wait for an authorized renewal, propose a different scope, or request additional budget. **It may not silently lower acceptance criteria or omit checks to declare success.** Cancellation prevents new commitments and preserves those that may still be settled.

8.7. Budget to work, verify, and improve

Within each project, allocations for production, verification, and process improvement are separated. They may be redistributed within what has been approved, while preserving verifiable completion of active work. Learning is an investment with its own budget, not a reason to consume the common fund indefinitely.

The addendum warns that Harness² consumes resources both in runs and in stages of improving and aggregating candidates; it identifies a composition from 2k candidates, each with its own run. This detail is retained as a report from the addendum and requires measuring the complete tree, not only the final task. A parameter $k = 1$ must not be interpreted as a single call. The addendum places the gains reported by the authors in GLM-5 and Gemini 3.5 Flash, not in our laboratory's executors. [15, pp. 8–11]

The economic question for improvement is whether subsequent work justifies the investment, at comparable quality. The cost of learning, the cost of applying the policy, and the benefit observed on new tasks are reported separately. Assumed future savings are not counted as resources already recovered.

8.8. Learning where to invest without learning to spend without limits

Economic lessons enter the same Claims Register: “this executor suffices for this family,” “this pattern requires escalation,” or “this review prevents defects” begin as hypotheses. To become policy, they need evidence, scope, validation, and authorization.

Changes in model, price, quota, tool, or task type can invalidate a rule. During a formal evaluation the policy is fixed; improvement candidates are studied outside that evaluation. The component that learns to allocate resources may propose a budget change, but may not increase the budget it administers.

Allocation across projects must respect human priorities, including uncertain research whose value cannot be reduced to immediate returns. The purpose of the fund is not to turn all curiosity into a profit-and-loss statement. It is to make visible what capacity we invest, for what purpose, and with what consequences.

The economic boundary does more than protect against spending. It makes it possible to devote the best available capacity to the problems that truly deserve it.

9. Useful autonomy, real boundaries

The founding document of BORDER distinguishes information that has been read from evidence enabled to govern actions. This white paper adopts that distinction as a design principle for the laboratory. It does not claim that BORDER's experimental gate automatically protects a computer with real tools and credentials. [5]

9.1. The orchestrator must also have a boundary

An agent should not be able to expand its permissions, accept its own exception, or modify the verifier to make its work pass. The proposed implementation would separate the decision to request an operation from the mechanism that can authorize it.

When a non-bypassable restriction is promised, it must be supported by effective isolation and permissions: credentials outside the modifiable environment, executors with bounded scope, and a policy that delegated work cannot rewrite. If an agent retains unrestricted access to the host system, a wrapper or editable manifest does not constitute that guarantee.

Confidence expressed by Jev or by a reviewer does not expand a source's authority. Similarly, agreement between two models does not demonstrate independence of their errors. Acceptance must preserve the relationship to tests, inputs, and permitted effects.

9.2. Preauthorize the ordinary; escalate the exceptional

We propose authorizing classes of operations within an assignment: reading the agreed repository, editing a working branch, running defined tests, and producing reports. In this way the human does not have to confirm every command in a legitimate sequence.

Changing the experimental question, expanding access to data, installing components outside the authorized environment, increasing budget, or publishing results are distinct transitions. How they are handled depends on the contract: they may be authorized in advance with limits or require an explicit decision. General authorization is not presumed merely because someone asked, "solve the task."

9.3. Concrete risks and proposed responses

Risk	Design response
Two agents modify the same state	Separate working copies, locks, and an owner for each task
An expired credential blocks a run	Identified pause, reauthorization, and resumption without duplicating effects
The orchestrator is lost	Persisted state and reconstruction from the last verified point
A file induces instructions unrelated to the assignment	Treat its content as data, limit tools, and check effects
A review confirms the author's bias	Tests external to the change, prior criteria, and recorded dissent
Recovery consumes resources indefinitely	Inherited budget, reserves, retries, and a stopping condition
The improver weakens its own admission	Policy and verifier outside its editable surface
Two tasks reserve the same balance	Shared ledger and atomic admission of commitments
Memory retains an obsolete rule	Dependency versions, suspension, and revalidation

Grok documents that an account's bots share files and sessions. Therefore, separation by name or folder will be treated as organization, not as security isolation. [7]

The boundary does not seek to make the system passive. It seeks to make useful autonomy possible, with a scope that can be understood, tested, and revoked.

10. What exists and what remains to be demonstrated

The architecture grows out of work already done, but its declaration must not erase the distance between a partial experience and an integrated system. This is the evidence classification updated for version 0.2.

Item	Evidence status	Scope of the claim
<i>Cuando nos encontramos</i>	Verified bibliographic precedent	Collaborative work; not a complete comparison of its anticipations [1]
Codex CLI → Grok Bot continuity	Reported by the author	Development and runs across tools; not an independent audit [4]
BORDER	Specification and manuscript supplied	Existence of artifacts; no rerun of the study in this white paper [5, 14]
Grok computer and noninteractive Codex access	Documented capabilities	Technical basis for attempting their connection, not verified integration [6–9]
Automatic Grok → Codex invocation	Proposal	Requires an end-to-end test and logs
Jev within the workflow	Access reported; role proposed	No experimentally established benefit [4, 10]
Coordinated publication on Hugging Face	Proposed destination	It is not claimed that this workflow already deploys automatically
Complete laboratory controls	Proposed design	Must be implemented and tested outside the agent's reach
Harness ² inspection	Reported by the addendum	Identified commit; no new audit or run in v0.2 [15]
Lesson gate and persistence	Integrated proposal	Neither implemented nor evaluated in this document
Fund and economic boundary	Author's proposal developed	No access to balances, actual reservations, or deployed control [16]
Improvements in cost, utility, or learning	Hypotheses	Require the separate contrasts in §11

10.1. What the BORDER case contributes

BORDER is an appropriate first domain because it connects specification, software, experimentation, and analysis. The supplied manuscript describes a diagnostic benchmark, stored results, and regeneration of tables. For this white paper, that constitutes a precedent in method and artifacts, not an evaluation of the complete orchestration architecture. [14]

The benchmark figures are not carried over into a claim about multiagent productivity. Nor is its evidence-admission instrument presented as a solution ready to protect all the laboratory's effects. They are different objects of evaluation.

The manuscript is marked as review material. This version does not reproduce its tables or detailed results and does not authorize their dissemination. A public edition of the white paper must respect the editorial status the author determines for that work.

10.2. The first integration criterion

The first demonstration of the architecture should be traceable from an authorized assignment to a verified delivery: the orchestrator invokes the specialist, receives evidence, performs a check, and returns a result with its references. A second person should be able to reconstruct that sequence without access to the agents' private history.

That milestone would not yet prove a statistical advantage, but it would turn a technical possibility into an operational instance of the declared pattern.

10.3. What does not transfer by analogy

The addendum reports third-party software and results; it does not demonstrate that their gains transfer to the laboratory's domains. Lesson admission does not by itself protect a computer with real credentials. A proposed economic policy does not guarantee limits that the provider or adapter cannot impose. These differences are retained even if the conceptual design is convincing. [15, pp. 10–11]

11. Expected benefit and evaluation agenda

The central hypothesis is that an explicit organization of capabilities can increase the useful work a person performs without proportionally increasing their coordination burden. This must be tested: a system with more agents may also add delays, handoff errors, and review costs.

We propose evaluating representative tasks from three families: implementing and checking a software change; running and analyzing a reproducible experiment; and producing a document grounded in sources. Success criteria will be established before results are observed. A convincing draft that does not meet those criteria counts as a failure, not as an approximate success decided retrospectively.

11.1. A comparison that separates coordination and diversity

	Homogeneous executors	Heterogeneous executors
Human coordination	Person organizes work with the same available model	Person distributes work manually among specialists
Coordination by orchestrator	Orchestrator uses the same homogeneous set	Orchestrator uses different specialists

This proposed design makes it possible to distinguish the effect of automating coordination from the effect of combining different capabilities. We would add a simple human-with-one-agent reference when appropriate. All groups must have comparable tools, inputs, and opportunities for verification; the proposed system will not be artificially strengthened by removing essential capabilities from its competitors.

11.2. What to measure

Utility and quality: accepted tasks, relevant defects, undetected errors, and fidelity to sources. **Human attention:** active minutes, interventions, and work to reconstruct context. **Resources:** total time, waiting time, compute, reported consumption, and cost. **Continuity:** recovery after interruptions and success when replacing executors. **Control:** effects outside the contract and completeness of the evidence.

Total cost and marginal cost will be reported separately. A quota included in a subscription will not be declared free compute; if a platform does not expose its consumption, that datum will be marked unknown. Fixed costs, available capacity, and retries must be made visible.

Repetitions and intervals will be defined over independent tasks or families. Nearly identical variations will not be treated as unlimited independent evidence. Thresholds, exclusions, and stopping criteria will be recorded before evaluation runs.

11.3. When the hypothesis would lose support

The benefit would be called into question if automatic coordination does not reduce human work at comparable quality; if diversity adds cost without improving results; if recovery duplicates effects; or if accepting deliveries requires manually reconstructing the entire process.

We will also examine whether the human retains the ability to explain the result and reconsider the objective. The program does not seek to optimize automation that produces more artifacts at the cost of making the work unintelligible to the person directing it.

11.4. Evaluating persistent memory and admission

The addendum's A/B/C design is incorporated over a **stream of tasks**, not a single answer. The three groups must use comparable budgets, tool sets, criteria, and verification opportunities. [15, pp. 6–8]

Condition	What it retains	Question it helps answer
A · Reset	Library reset at the unit specified by the protocol	What happens without accumulation between sweeps?
B · Persistent without gate	Experiences across tasks with the described structural control	What benefit or harm does accumulation without additional admission produce?
C · Persistent with gate	Same persistence, plus admission, validity, and promotion	Does the boundary improve the utility of memory?

The addendum describes A as the baseline distributed by Harness². The exact unit of reset—run, sweep, or domain—will be checked against the pinned revision before execution, so that A is not inadvertently turned into a different competitor. C will not receive privileged sources or a better judge than B without declaring it as another intervention.

The same paired stream will be retained across conditions, and deliberate tool, model, or interface changes will be introduced halfway through, with branches without drift when the budget permits. Drift is an intervention to make aging and validity measurable, not a claim that all persistent memory must degrade.

Prediction in the addendum	Planned contrast	Result that would weaken it in the studied stream
P1 · Persistence without a gate degrades	B versus A over the stream	B matches or exceeds A without relevant degradation
P2 · The gate recovers utility	C versus B, especially after drift; also versus A	C does not improve on B, or the cost eliminates its benefit
P3 · Mechanical sufficiency adds value beyond the prompt	Threshold for contrasts versus narrative instruction	It discards useful lessons without reducing relevant errors
P4 · Overblocking may occur	Accepted tasks together with memory quality	C retains or improves utility without appreciable unnecessary blocking

P4 is retained as a measurable risk, not as a failure to be hidden. The claim “recovers and exceeds the baseline” requires comparing C with both B and A; exceeding B alone does not demonstrate the latter. An undetected effect does not prove equivalence without an agreed margin and power.

The protocol will fix seeded order, number of streams, magnitude of drift, thresholds, exclusions, and analysis before looking at results. The addendum proposes bootstrap and permutation. Version 0.2 makes explicit that resampling must respect temporal dependence: it will operate over streams or justified blocks, not over each correlated task as though it were an independent replication.

Its blind review of admitted and rejected lessons is incorporated. **Calibration belongs to the pilot or development.** If an inspection of the evaluation reveals a defect, it is reported; correcting it creates a new version and another evaluation. The gate will not be adjusted to the same results with which its success is then declared. This separation is a methodological clarification of v0.2, recorded in Annex E.

The judge must not receive unnecessary condition labels or feed reserved evaluation answers back to the improver. Configuring different models to execute, improve, and judge helps separate roles, but does not by itself guarantee blinding or independence. The professional tasks in the source work do not demonstrate transfer to BORDER, Seed Model, or the laboratory’s own research. [15, pp. 7–11]

11.5. Evaluating economic allocation

The economic hypothesis is that explicit, adaptable allocation improves the useful work obtained from a budget without reducing acceptance or shifting hidden costs to the human. A baseline using an advanced executor, a fixed economic policy, and an adaptive policy will be compared using the same eligible tasks and constraints.

Two comparisons will be evaluated: comparable quality with different consumption, and comparable budgets with different utility. The comparisons will include coordination, review, failed attempts, recovery, compute, and improvement amortized according to a stated criterion. Use of already contracted quotas will not be presented as having zero total cost.

In addition to cost per accepted task, aggregate spending, unresolved tasks, undetected defects, human interventions, latency, unknown consumption, and pauses due to lack of resources will be reported. Measuring only successes would conceal the cost of poor choices. Overruns and differences between estimated reservations and observed settlements will also be recorded.

Deliberate failure cases will include concurrent commitments, delayed charges, quotas shared across different interfaces, expired credentials, descendant cancellation, outputs without reported consumption, and changes in price or availability. A synthetic test that prevents a bookkeeping overdraft does not by itself demonstrate that the provider will never bill more than the amount reserved.

The adaptive policy will have a fixed revision during evaluation. Its training or tuning will use different tasks; live changes would be studied later under a separate protocol. We will first separate coordination, memory, and economics in bounded experiments; we will not attribute a joint improvement to a mechanism we did not isolate.

11.6. What could weaken support for v0.2

In addition to the v0.1 criteria, the claimed benefit would lose support if persistent lessons degrade subsequent tasks; if the admission gate protects memory indicators at the expense of utility; if routing consumes more than it saves; if reduced spending depends on omitting review; or if the human must reconstruct the work to understand what was accepted.

Four achievements will be distinguished: the circuit works; learning contributes; admission contributes; and economic allocation contributes. None borrows evidence from the others. Evaluation of the whole is justified only after those mechanisms and their costs can be inspected.

12. From the first circuit to a reusable capability

Development should advance on evidence of operation, not on the number of integrations. We propose four milestones without setting schedules before their execution conditions are known.

Milestone 1 · Verifiable delegation. The orchestrator executes a bounded task with Codex, first through the available circuit and, in the route proposed by Claude, through Harness². The full loop and its experience are observed before building the lessons admission gate. A test library, a prior budget, and basic access controls are used; “without a gate” does not mean without limits. Verification is performed by an executor distinct from the producer. Appendix D preserves the specific assignment.

Milestone 2 · Continuity and recovery. Interrupt, resume, and replace executors without repeating effects or duplicating the library or economic ledger entries. Cancellation, unknown quotas, concurrency, and expired credentials are tested, not only the ideal path.

Milestone 3 · Complete experiment. A mission is fixed, tests and runs are retained, and tables are regenerated. The addendum proposes A/B/C with drift here. Economic comparisons are added as separate protocols so as not to confuse causes. Jev participates only in interventions that allow its contribution to be measured. Improvement policies are not tuned using the evaluation intended to judge them.

Milestone 4 · Controlled publication. An identified revision reaches the approved destination and produces evidence of deployment. The public version can run without exposing personal credentials or connecting visitors to the laboratory's private session.

12.1. Minimum implementation

The first version can rely on a simple task queue, a persistent state log, tool adapters, and an artifact structure. It does not require building a universal framework, a marketplace among agents, or a learned planning system from the outset. A resource inventory, reservations, and stopping rules are part of the minimum v0.2 circuit.

The stability of the contracts matters more than the sophistication of the coordinator. One adapter may invoke a CLI; another, an API; another, a local program. When a provider changes, the documented capabilities and configuration should change, not the project's history.

The shared computer needs concurrency discipline. Independent tasks can be parallelized within available resources; changes to the same repository or environment must be coordinated. Quotas and effective limits will be measured before increasing the load.

12.2. A laboratory that benefits other projects

BORDER is the first case, not the boundary of application. Seed Model could reuse run contracts, logs, and reproduction without thereby changing its simulator's hypotheses. Editorial work could preserve sources, objections, and voice decisions while automating mechanical tasks. A product prototype could move from a need to a tested artifact under the same delegation pattern.

These are proposed applications. Their benefit would have to be demonstrated in each domain. What is reusable is the organization of work; validity criteria remain specific to each project.

The ambition is not for the laboratory to do everything. It is for each useful lesson to be incorporated without starting over, and for each new component to expand capabilities without erasing the judgment that guides them.

12.3. A proportionate first step

Claude's proposal reduces one uncertainty: observe a real experience before specifying all its admission rules. The first milestone does not require solving fund optimization or lesson generalization in advance. It does require declaring what will be allowed to run and how much may be committed. Then a decision is made about which mechanism deserves the next budget.

13. Architecture declaration

We declare distributed symbiotic intelligence as a direction of work: a collaboration in which the person is not reduced to an interface operator, nor agents to producers of isolated answers. Its purpose is to sustain complete cycles of research, creation, and building.

Human will retain the ability to ask, evaluate, correct, and change course. It decides which outcome matters, which explanation is sufficient, and which project deserves to continue. It can learn from agents' objections and results without transferring to them authority over its ends.

The orchestrator turns an assignment into coordinated work. It calls on agents and programs, requests corrections, and manages recovery within the agreed scope. It need not solve everything internally and cannot fabricate evidence of its success.

Memory allows work to survive conversations. Decisions retain their reasons and sources; results, their production conditions; disagreements, their place in the record. Learning changes procedures when evidence and authority permit, not when an explanation seems convincing.

The architecture is open: Grok, Codex, Claude, Astra, Jev, conventional programs, GitHub, and Hugging Face are a replaceable configuration. No single provider defines the laboratory. Adding a capability should expand what we can do without erasing the project's continuity.

Economics turns available resources into investment decisions. Solving, verifying, and improving share a finite fund. The system can learn to use it better; it cannot claim the right to expand it or hide its costs.

Autonomy is evaluated by legitimate work completed, not by acting without humans at any cost. A system that never acts does not satisfy this proposal either. Evidence limits the claims: traceability, reproduction, comparison, and integration are distinct achievements, not interchangeable ones.

From the book to a practice that can be tested

Cuando nos encontramos opened an exploration of creation between intelligences. This architecture extends it: converse to formulate, delegate to build, execute to observe, and return to dialogue to understand. It does not declare all its anticipations fulfilled; it organizes a practice to discover how far that direction can go. [1]

We are not seeking an intelligence that does everything for us. We seek an organization of intelligences with which we can do things that were previously beyond our reach.

This is how we define the symbiont we want to build: a human will that gathers the appropriate capabilities, organizes their cooperation, learns with them, and retains the right to change course. Its continuity does not depend on a model lasting forever, but on purpose, memory, and what has been verified being able to continue growing together.

Appendix A. Mission and return contract

Illustrative schema, normalized for discussion. It is not a deployed implementation. Unresolved references prevent a transition from draft to authorized execution.

```

schema_version: simbiote-task/0.2
mission_id: identificador_estable
task_id: identificador_de_tarea
parent_task_id: null
objective: resultado_y_alcance_aprobados
inputs:
  source_refs: [referencias_autorizadas]
  code_revision: revision_fijada
  protocol_revision: protocolo_aplicable
  operational_library_revision: biblioteca_admitida
capabilities: [subconjunto_autorizado]
policy_ref: permisos_y_criterios_protegidos
budget_ref: asignacion_del_presupuesto_padre
routing_policy_revision: politica_de_asignacion_fijada
acceptance:
  criteria_refs: [criterios_versionados]
  verifier_role: distinto_del_productor
artifacts: [productos_y_registros_requeridos]
recovery:
  retry_limit: definido_antes_de_correr
  cancellation: propagar_a_descendientes
learning:
  candidate_output: permitido
  persistent_promotion: autorizacion_separada
publication: autorizacion_separada

```

The return identifies the task and attempt, status, revisions, artifacts, commands and checks, unresolved errors, observed or unknown consumption, and reservations awaiting reconciliation. A candidate lesson points to its evidence; it does not by itself change the operational library. The message does not require revealing the models' private internal reasoning.

Minimum acceptance. Fidelity to the assignment; inspectable objects; verification against the delivered revision; inherited permissions and budget; resumption without duplicated effects; and publication only under authorization. The results analyzer is a separate process from the executor's narrative message.

Appendix B. Lesson register and promotion

The addendum's cue, lesson, and evidence keys are retained, expanded with lineage, conditions, and authorization. This proposed format does not assert direct compatibility with the Harness² library. The adapter must verify it.

```

schema_version: simbiote-lesson/0.2
lesson_id: identificador_estable
cue: condicion_observable_de_aplicacion
lesson: cambio_propuesto_y_alcance
evidence:
  supporting_refs: [registros_y_contrastes]
  counterevidence_refs: []
  independent_task_refs: [tareas_de_confirmacion]
binding:
  component: herramienta_prompt_o_politica
  before_revision: revision_previa
  after_revision: revision_candidata
validity:
  produced_at: instante_de_la_evidencia
  domain: familia_de_tareas
  dependencies: [modelo_herramienta_protocolo]
  invalidate_on: [cambios_relevantes]

```

```

claims_register_state: reportada
operational_status: candidata
admission:
  policy_revision: regla_de_admision
  verifier_ref: comprobacion_independiente
  decision: pendiente
promotion:
  authorized_by: null
  library_revision: null
supersedes: null

```

Minimum transition. Propose → register → verify → reproduce → authorize → activate. A subsequent comparison estimates the benefit without automatically extending the domain of validity. Expire, suspend, revoke, or replace are operational transitions with reasons retained.

Essential tests. A lesson without records is not promoted; reading it today does not make its evidence more recent; changing a relevant dependency triggers revalidation; contradictory rules are exposed; promotion cannot modify its own verifier; rollback preserves history; and budget scarcity does not lower the admission standard.

Appendix C. Economic contract and resource ledger

The example uses hypothetical amounts. They are not rates, balances, or authorization to spend. supervised indicates that a hard cap for the external resource has not yet been demonstrated.

```

schema_version: simbiote-budget/0.2
budget_id: presupuesto_mision_ejemplo
parent_budget_id: fondo_proyecto_ejemplo
authority_ref: aprobacion_humana_pendiente
currency: USD
cash_limit: 10.00
allocations:
  production: 6.00
  verification: 2.00
  improvement: 2.00
resource_accounts:
  - quota_id: cuota_compartida_A
    access_mode: authorized_subscription
    native_unit: provider_quota
    observed_remaining: null
    usage_status: unknown
    renewal_policy_ref: condiciones_a_verificar
controls:
  enforcement_mode: supervised
  include_descendant_spend: true
  atomic_reservations_required: true
  unknown_usage_is_zero: false
  new_budget_requires_authorization: true
  protected_verification_reserve: true
on_exhaustion:
  - pause_new_commitments
  - reconcile_pending_usage
  - request_scope_or_budget_decision

```

Each entry records event_id, mission_id, task and attempt, resource, native unit, reserved amount, settled consumption, status, time, and measurement source. The ledger supports reserving, settling, releasing, correcting, and reconciling. Corrections do not delete the prior entry.

For the example currency: authorized 10; consumed 2; committed 6; available 2. Native quotas are checked separately in their own units. Sufficient money does not imply sufficient quota; a renewed quota does not imply new permission for monetary spending.

The authority approves limits and possible reallocations. The planner selects routes within them. A budget warning is not called an effective block until it is demonstrated to intercept all relevant routes, including internal and concurrent work.

Appendix D. First assignment: observe before building the gate

Contract derived from the addendum, pp. 8–9, with normalized syntax and added economic references. It maintains the addendum's priority: first run the candidate mechanism in a test environment, without a lessons admission gate or promotion to the production library. It is not executed merely because it is included in this document.

```
mission_id: simbionte-harness-admision
task_id: h1-delegacion-verificable
parent_task_id: null
status: draft_requires_authorization
objective: >
  Execute a bounded JobBench task with Harness2 and
  its Codex executor. Return the harness, experience, and logs.
inputs:
  source_ref: google-research/harness2
  code_revision: 1cfe254e8d25b15f37048c3730134686abe60335
  protocol_revision: h1-v02-draft
roles:
  orchestrator: Grok_Bot
  engineering_executor: Codex_CLI
  artifact_verifier: Claude_Code_o_equivalente_distinto
preflight:
  - comprobar_revision_interfaz_y_credenciales_admitidas
  - declarar_presupuesto_y_consumos_internos_posibles
  - revisar_superficie_editable_y_permisos
  - ejecutar_dry_run_y_conservar_seleccion
scope:
  benchmark: jb
  k: 1
  runs_dir: directorio_exclusivo_de_prueba
  operational_library_write: false
  admission_gate: no_en_esta_tarea
budget_ref: presupuesto_h1_pendiente_de_aprobacion
forbidden:
  - ampliar_permisos_o_presupuesto
  - modificar_controles_protegidos
  - publicar_o_exponer_credenciales
retry_limit: 2
publication: autorizacion_separada
```

The preview command recorded in the addendum is `harness2 --bench jb --k 1 --dry-run`. It is checked against the fixed revision before use. Acceptance requires the selection to be retained without calls during that preview, logs of the actual run, the adopted harness tree, and the experience with its fields. A second process reconstructs the summary; an external reader can follow the sequence without the private chat. [15, pp. 8–9]

If no experience is produced, the diagnosis is retained and the criterion is marked unmet; the file is not fabricated to satisfy the contract. If access has expired, work pauses for reauthorization. On failure, the two cumulative retries are respected within the same budget. Use of an API credential or subscription is checked in the adapter: Codex support does not automatically demonstrate equivalence between the two access methods.

Appendix E. Changes and questions for review with Claude

E.1. What changes from v0.1

Element	Treatment in v0.2
Center of the system	Human will, revisable purpose, and better capabilities for each task
Five planes and memory	Retained; learning and economics are integrated as cross-cutting responsibilities
Process learning	The addendum is incorporated, and Harness ² is included as a candidate; integration is not claimed
Claims Register	Applied to lessons and economic policies, with authorized promotion
Tools and review	Claude Code is included by role; the conflict of reviewing one's own proposals is made explicit
Resources	Heterogeneous fund, total/marginal/opportunity cost, and inherited budgets
Economic control	Reservation, execution, reconciliation, concurrency, and distinction between cap and supervision
Evaluation	Coordination/diversity, A/B/C with drift, and comparison of economic policies
Milestones and contracts	First circuit without a lessons admission gate, with basic controls and a prior budget

E.2. Explicit editorial clarifications

The addendum's boundary between observed and reproduced is preserved, but eligibility is distinguished from authorization. P1–P4 are retained, limiting their conclusions to the workflow studied. Blind review calibrates during development; it does not permit correcting against the evaluation and reporting it as untouched. A configuration with multiple judges does not by itself amount to independence or blinding. These are v0.2 clarifications for discussion, not conclusions retroactively attributed to Claude.

References to the Harness² inspection retain attribution to the addendum and its commit. This version has not re-audited that code, run its tests, or confirmed subscription access within its adapter. The book's text was not reviewed in full either: no specific unchecked predictions are added.

E.3. Priority review questions

R1 · Identity and will. Does the definition preserve collaboration and the human's ability to learn and reconsider, without reducing the human to an operator or turning the metaphor into a claim of collective consciousness?

R2 · Admission. Is the boundary for entry into persistent policy sufficient for the pilot? How many comparisons and what degree of independence does each type of lesson require? What can software check and what requires judgment?

R3 · Protected surface. Which specific files and capabilities may the loop not modify? Does the adapter preserve that separation, including the possibility that a tool might bypass a wrapper?

R4 · Integration and access. Can the pinned substrate use the available authorized mechanism? What internal consumption does $k = 1$ actually generate? What does a Claude Code adapter require against the described interface?

R5 · Economics. How are shared quotas and concurrent commitments identified? Which resources allow a hard cap and which allow only supervision? How is verification reserved when telemetry is unavailable?

R6 · Evaluation. Does A/B/C allow persistence, admission, and overblocking to be isolated? How are workflows, drift, and budgets matched without contamination? What outcomes would contradict the economic promise?

R7 · Independent review. Claude participated in the addendum and Astra in this integration. They may check artifacts and discuss changes, but neither should be declared an independent adversarial reviewer of its own design. Who examines those decisions without having proposed them?

R8 · Proportion and first step. Does Milestone 1 allow us to learn something specific without prematurely building the entire architecture? Do we preserve ambition and the ability to act, rather than turning the laboratory into a collection of gates that never works?

The next iteration should return objections anchored to sections, proposed changes, and their effect on contracts, evidence, and cost. A generally favorable assessment is not enough. Approval of this version and any implementation decision remain the author's responsibility.

Sources and traceability

The v0.1 references retain their provenance. On October 19, 2026, primary documentation on Grok, Codex, Jev, GitHub Actions, Spaces, and orchestration was consulted again, and background on routing and credits was incorporated. The Harness² code was not audited: observations about its commit are attributed to the addendum. Product capabilities and terms may change.

The contracts, metrics, milestones, principles, operational library, and economic control are proposals of this white paper unless explicitly stated otherwise. References [15] and [16] identify the contributions incorporated; Appendix E makes the refinements of this edition explicit.

Background and author's materials

[1] Niven, K.; Vera, P. M. (2025). *Cuando nos encontramos: Una colaboración entre humano e inteligencia artificial*. Amazon KDP. ISBN 9798289810854. The bibliographic record and editorial description on Google Books were consulted, not the full text. The cover available in the author's materials matches the title and author names. [Public source](#)

[2] Simbionte. *About*. Public presentation of the project as a human–synthetic collaboration oriented toward questions, reflection, and co-creation. Primary source for the editorial approach. [Public source](#)

[3] Simbionte (August 24, 2026). *All the Parts Work. The System Doesn't*. The author's essay on the difference between individual capabilities and organization of the system. Its editorial argument is used; the scientific works it discusses are not reassessed here. [Public source](#)

[4] Vera, P. M. Working communications in ChatGPT: current HF Spaces/BORDER conversation, October 19, 2026; earlier conversation about Directorio de IAs, January 30, 2026, retrieved from history. Source of the Codex CLI→Grok Bot report and the intention to automate coordination. It does not constitute an execution audit.

[5] *BORDER Protocol v0.1*. Foundational document supplied by the author: purpose, precedence of the boundary, separation of states, and automation criteria. Project working material.

[14] *Stale, Scoped, or Spoofed? A Diagnostic Benchmark for Evidence Admission at the Agent Action Boundary*. Supplied file: BorderProtocol_V01.pdf, seven pages, anonymous review manuscript. Internal reference; tables are not reproduced and distribution is not authorized. The repository was not audited and runs were not repeated in preparing this white paper.

Primary technical documentation

[6] xAI. *Grok Bot — Overview*. Persistent computer and operation through terminal, browser, and files. [Public source](#)

[7] xAI. *Use the computer and apps*. Shared environment, files, and sessions of an account's bots. [Public source](#)

[8] OpenAI. *Authentication*. Access through ChatGPT or API; authentication in remote environments and credential handling. [Public source](#)

[9] OpenAI. *Non-interactive mode*. Invocation using `codex exec`, JSONL events, and structured outputs. [Public source](#)

[10] TypeSafe. *Introduction*. Jev, typed questions and structured answers consumable by software. [Public source](#)

[11] GitHub. *Understanding GitHub Actions*. Build, test, job, and deployment workflows. [Public source](#)

[12] Hugging Face. *Spaces Overview*. Demonstrations, Spaces repositories, deployment lifecycle, and secrets. Access and resource conditions are checked during implementation; zero cost is not assumed. [Public source](#)

[13] OpenAI. *Agent orchestration — Agents SDK*. Existing patterns of agents as tools, handoffs, and orchestration through code or models. Technical precedent; not a mandatory dependency of the design. [Public source](#)

v0.2 contributions and economic background

[15] Vera, P. M., with Claude (October 19, 2026). *El bucle que aprende necesita una frontera*. File Addendum `CLaude.pdf`, 12 pages. Basis for the integration of Harness², lesson promotion, replaceable roles, A/B/C, Milestone 1, and their risks. The distinction between reported code inspection and execution not performed is retained.

[16] Vera, P. M. (October 19, 2026). HF Spaces/Simbionte review conversation: proposal for the “token fund,” economic criteria by difficulty, and formulation of the symbiont as a will that gathers and orchestrates capabilities. The response discussed develops allocation and the boundary; the author requests their integration into v0.2. Working communication, not an experimental evaluation.

[17] Harness². Repository `google-research/harness2` and *Harness²: Recursive Agent Harnessing for an Open World*, arXiv:2610.14224, as cited by the addendum. Revision recorded: `1cfe254e8d25b15f37048c3730134686abe60335`. License, adapters, results, and code contents were not re-audited in preparing v0.2. [Cited repository](#) · [Cited work](#)

[18] Chen, L.; Zaharia, M.; Zou, J. *FrugalGPT: How to Use Large Language Models While Reducing Cost and Improving Performance*. arXiv:2305.05176. Background on cascade combinations; not evidence of savings by Simbionte. [Primary source](#)

[19] Ong, I. and colleagues. *RouteLLM: Learning to Route LLMs with Preference Data*. arXiv:2406.18665. Background on selection among models using preference data. [Primary source](#)

[20] OpenAI. *Pricing*, Codex documentation. Distinguishes tokens, credits, and included consumption; source of the example of heterogeneous units, not a rate fixed in this white paper. [Primary source](#)

[21] OpenAI. *Using Credits for Flexible Usage in ChatGPT (Personal plans)*. Credits shared across eligible features and the possibility of a negative balance due to concurrent work. Terms depend on the plan and account; product and API credits are not assumed to be interchangeable. [Primary source](#)

The addendum also cites *The Granularity Mismatch in Agent Security* (arXiv:2605.11039) and *Proof-or-Stop* (arXiv:2607.14890). They are retained as references from its discussion, without reassessing those works or importing their results into the architecture declared here. [15, p. 11]

Scope of this version. Documentation has been drafted and integrated. It is not claimed that accounts have been connected, executors installed, runs initiated, gates implemented, real balances managed, repositories modified, or the laboratory deployed. The author's review and the next iteration with Claude remain open.