

BUSINESS COLLATERAL

Cognous Open Control Stack

Open reference infrastructure for governed agent actions and traceable decision evidence

October 2026

Before an enterprise can rely on an AI agent's action, it needs to know what was proposed, who could authorize it, whether that authority was still current when the effect was attempted, and what the destination was actually observed to contain.

1. Executive Summary

Cognous Open Control Stack is open reference infrastructure for evaluating governed agent actions. It connects four things that agent governance depends on and that ordinary logs tend to blur together: a declaration of the action an agent may propose, institutional authority that is resolved independently of the incoming request, bounded runtime execution, and retained evidence that a reviewer can trace back to its sources.

The stack now consists of **13 components organized into six responsibility groups**, plus a cross-cutting exchange integration:

- **01 · Knowledge & Provenance.** Register evidence commitments and verify who signed a statement, under explicit trust assumptions.
- **02 · Institutional Governance.** Describe institutional authority and the Authority Context contract that runtime systems consume.
- **03 · Declaration & Policy Control.** Declare proposed actions and evaluate them against independently supplied authority, including at effect time.
- **04 · Constrained Execution.** Execute bounded operations through a constrained runtime and destination boundary.
- **05 · Replay & Evidence.** Reconstruct retained records and carry traceable evidence for review and for recipients across boundaries.
- **06 · Reasoning & Human Oversight.** Optional instruction protocols that support evidence-bounded reasoning and human review.
- **Cross-cutting: Governed Exchange (GAX/IMX).** Governed message exchange, local durable delivery and continuity records for the supported workflow.

The six groups are a responsibility map, not an execution sequence. The July 2026 description of the stack as a four-step lifecycle (Declare, Control, Replay, Evidence) is superseded: it omitted authority resolution, constrained execution, governed exchange, portable decision evidence and the optional evidence and behavioral layers, which each have distinct owners.

The stack is deliberately limited in what it claims to be. The current release is a merged, pinned **bounded synthetic reference**: one supported workflow, a synthetic refund, executed against a synthetic SQLite destination on one host. It is not a production deployment, a payment integration, a compliance certification or proof of institutional adoption. It does not run the model, and it does not replace agent frameworks. What it supplies is a concrete, reproducible artifact chain that lets engineers, security teams and governance reviewers examine how declaration, authority, execution and evidence fit together, and where they still do not.

A chatbot can be governed by reviewing what it said. An agent that acts must be governed by reviewing what it was permitted to do at the moment it acted, what it attempted, and what actually happened at the destination. The Open Control Stack keeps those as separate, connected facts rather than one retrospective story.

For enterprise leaders, the Open Control Stack is not another agent framework and not a governance platform. It is reference infrastructure for the evidence layer around agent actions: the records that security, risk, audit, legal and platform teams would need to examine before agents are allowed to touch business systems, together with an explicit account of what those records do and do not establish.

2. The Business Problem

Across enterprises, AI agents are beginning to call tools and APIs, access customer, financial and operational data, draft and send communications, propose database updates, approve or escalate workflow steps and initiate transactions such as refunds. With a chatbot, the output is the whole story. With an agent, the output is the last step of a longer chain of proposed, authorized, attempted and observed actions.

An organization therefore needs to explain more than what a model said or which tool it called. It needs to know what was proposed, who could authorize it, whether that authority was still current when the effect was attempted, what the executor reported and what the destination was independently observed to contain. Timeouts, duplicate messages and later changes in authority make these genuinely different questions. A timeout can conceal an effect that did commit. A later denial can coexist with an earlier, valid execution. A signed message can be authentic and still carry no authority at all.

In practice, the gaps show up in six places.

1. **The declaration gap.** The enterprise may know an agent's purpose but not its action surface. A tool name is not a policy boundary: the same integration can read information, draft a message, issue a payment or delete a record.
2. **The authority gap.** Approval often travels with the request, cached in a prompt, a message or an earlier decision. If authority is taken from the incoming message, anyone who can shape the message can shape the authority.

3. **The effect-time gap.** A decision made when an action is proposed may no longer hold when the effect is attempted. Grants are withdrawn, approvals expire, policies change. An earlier permit is not a reusable credential.
4. **The outcome gap.** An authorization record is not proof that an effect occurred, and a timeout is not proof that it did not. Recovery logic that treats "no acknowledgement" as "safe to retry" can create duplicate effects.
5. **The reconstruction gap.** After an incident, records from authorization, execution and recovery can disagree or omit facts. Review needs a connected record that preserves uncertainty rather than a tidy success story.
6. **The evidence gap.** Business reviewers and outside recipients need reviewable material that stays traceable to its sources and does not quietly convert declarations or self-reported results into verified facts.

Each gap could be patched in isolation. The reference exists to show them as one connected chain, with explicit boundaries between the links. It does not quantify cost savings, review efficiency, payment settlement or deployment outcomes; those have not been measured.

3. The Stack in One View

Group	Components	Question addressed	Does not establish
01 · Knowledge & Provenance	Evidence Registry (The Index); Evidence Attestation (BitRep)	What evidence was committed, and who signed this statement under which trust snapshot?	Truth of the statement, or any execution authority
02 · Institutional Governance	Institutional Governance (Alvorada), with the Authority Context contract	Who may decide, and how is that authority described to software?	Ratification or adoption by any real institution
03 · Declaration & Policy Control	Action Manifest; Control Plane	What may be proposed, and is this exact proposal permitted under current authority at effect time?	A completed effect merely because authorization passed
04 · Constrained Execution	Execution Runtime (Moltbot Safe)	Was exactly this operation attempted within the bounded destination?	Live OS or network confinement, or external settlement
05 · Replay & Evidence	Replay Bundle; Governance Evidence Pack; Open Decision Evidence Standard (ODES)	Can the run be reconstructed, reviewed and carried to a recipient with its lineage intact?	Policy reevaluation, independent assurance or compliance
06 · Reasoning & Human Oversight	Portable Reasoning Protocol (PRP); Research Intelligence; TFA	How should a model reason and interact so a human reviewer can rely on it appropriately?	Runtime enforcement, or proof that oversight occurred
Cross-cutting	Governed Exchange (GAX/IMX)	Was the message received, assessed and connected to its outcome?	Authorization or exactly-once delivery from receipt

The components are deliberately separate because the questions are separate. One rule runs through all of them: **evidence is not permission**. A valid signature, chain inclusion, message receipt,

reasoning instruction or evidence-package digest never authorizes execution. Authority must be supplied by a trusted resolver and evaluated by the Control Plane at the point of effect.

Two groups (01 and 06) are optional paths with narrower claims; they are not dependencies of the representative refund. The core workflow runs through groups 02 to 05 and the Governed Exchange integration. The hub repository, github.com/cogno-us/cognous-open-control-stack, owns the pinned integration scenarios, qualification tooling and public documentation; runtime implementations remain in their component repositories. Several repositories moved to new locations in October 2026; the four original artifact components dropped "Agent" from their titles (for example, Cognous Agent Action Manifest is now Cognous Action Manifest). Package names, schema identifiers and CLI names did not change.

4. Group 01 — Knowledge & Provenance

The Knowledge & Provenance group addresses a question that sits upstream of any decision: what evidence exists, who stands behind it, and has it changed? It forms a separate evidence path in the stack, not an authority dependency of the refund workflow.

Cognous Evidence Registry (The Index) is a protocol and reference implementation for registering claims, linking supporting and contradicting evidence commitments and preserving revision and lifecycle history. In the accepted bounded slice, a local blockchain contract is authoritative for those records, and a read-only indexer reconstructs visible state and event history. Withdrawal or supersession preserves history instead of silently replacing it.

Cognous Evidence Attestation (BitRep) is a binary-attestation protocol and reference verifier. Its v1 verification contract checks Ed25519 signatures against an operator-configured issuer-key trust snapshot and an evaluation time, including key status, audience and validity windows. No default trusted issuer is supplied.

It gives an evidence consumer a structured answer to the question: "Who signed this statement about these exact bytes, and was that signature admissible under the trust I configured?"

What it is not. A successful attestation means issuer signature only, under stated conditions; it is not factual truth or institutional authority. Chain inclusion is not truth either, and two signed assertions about the same bytes are not automatically independent corroboration. Neither component authorizes execution.

5. Group 02 — Institutional Governance

Agent governance eventually reaches a question software cannot answer on its own: who is entitled to decide? The Institutional Governance group keeps that question explicit and separates *proposed* authority from *operative* authority.

Cognous Institutional Governance (Alvorada) is an open research framework for businesses, public bodies and communities, with or without AI. It combines proposed constitutional designs, proportional governance (matching review intensity to consequence, reversibility and uncertainty), challenge and remedy mechanisms, domain guidance, decision templates and a public-stack implementation profile.

For software integration, the group supplies the **Authority Context 0.1.0** profile: the contract through which independently established institutional authority (institution, mandate, grant, approval, policy and evaluation time) is described to the runtime. In the reference, a separately configured trusted resolver supplies that context, independently of the incoming request.

It gives enterprises a structured answer to the question: "Where does the authority for this action come from, and how is it represented to the systems that enforce it?"

What it is not. Publishing or accepting the framework in a repository does not ratify a constitution or establish authority for any institution. The public reference uses synthetic authority fixtures. An actual deployment must establish institutional authority through its own processes and authenticate its resolver; the downstream controller must still enforce permissions at effect time.

6. Group 03 — Declaration & Policy Control

This group holds the two components most directly descended from the original stack's Declare and Control layers. Both have moved materially since July 2026.

Cognous Action Manifest

The Action Manifest is a lightweight format, Python validator and CLI (aam) for describing the actions an AI agent may propose, before runtime. It declares the tools and action types an agent may use, owners and intended scope, authority, review and reliance requirements, sensitive payload fields and redaction hints for downstream records.

The distinction between tool access and action semantics remains the heart of the manifest. Access to a payments system does not distinguish looking up an order from issuing a refund. **Manifest 1.1** adds deterministic runtime bindings: stable action and adapter identifiers, finite targets, effect limits and Authority Context references. A downstream controller can therefore compare a live proposal with one specific declared action rather than with a loose description.

It gives enterprises a structured answer to the question: "What, exactly, is this agent allowed to propose?"

Example. A reviewer declares a routine synthetic refund, its permitted adapter, its target and its effect limits. At runtime the proposal is bound to that manifest commitment.

What it is not. Matching the declaration never authorizes the refund. The manifest is declaration and validation, not a security boundary or access-control system; it does not establish current approval, a completed effect or compliance. Redaction hints must be applied by the exporting system.

Cognous Control Plane

The Control Plane is a Python reference implementation for bounded agent authorization, effect-time revalidation and persistent runtime records. It sits beside the agent, not inside it. It evaluates a Manifest-bound proposal against the Authority Context supplied by the trusted resolver and explicit policy, and records the decision.

Two capabilities are new since the original collateral:

- **Effect-time revalidation.** Immediately before the supported execution path reaches the destination, the Control Plane rechecks the exact institution and authority domain supplied by the trusted resolver, and the action, target and payload, together with the current grant, policy and required evidence. If an approval is withdrawn between decision and dispatch, the changed request is refused. An earlier permit is not a reusable capability.

- **Persistent record transactions.** Decisions, attempts, observations and reconciliation are stored as distinguishable records, including unknown acknowledgements, and appended under a stable Linux file lock with atomic replacement. Cooperating same-host writers on documented local Linux filesystems are supported, and persistence failures are explicit rather than silent.

It gives enterprises a structured record of what was proposed, what authority was resolved, what was permitted at the moment of effect, and what happened next.

What it is not. In the default path, revalidation happens *before* dispatch; it is not atomic with the destination commit, so a check-to-commit race remains (see Section 14). Individual record transactions are protected, but the whole decide, execute and reconcile workflow is not atomic. Production resolver authentication, distributed budgets, remote finality and exactly-once delivery are not established. The Control Plane is not an agent framework, a dashboard or a substitute for application authorization.

7. Group 04 — Constrained Execution

The original four-layer stack stopped at the decision. The current stack carries the action through to a bounded effect, because an authorization record is not proof that an effect was performed, and a timeout is not proof that it was absent.

Cognous Execution Runtime (Moltbot Safe) provides the stack's bounded synthetic destination and adapter boundary through its Python execution layer. Execution Runtime and Moltbot Safe name the same repository. It:

- snapshots the request content before callbacks and binds the effect to that exact operation;
- constrains institution, authority domain, action, adapter, target, amount, unit and effect count;
- serializes same-effect suppression, rejects conflicting content under the same effect identifier and counts cumulative local effects in a synthetic SQLite destination;
- exports versioned evidence (Execution Envelope 0.2.0 and executor producer profile 2.0.0) with separate attempt and observation history.

It gives enterprises a structured answer to the question: "Was exactly the authorized operation attempted, once, against the bounded destination, and what was observed there?"

What it is not. The destination is a synthetic SQLite refund table, not a bank, card network or payment processor. Application-level restrictions do not establish operating-system confinement of the whole repository; a sufficiently privileged host process can bypass the Python layer. The repository also retains an upstream TypeScript application that sits outside the reviewed Python execution boundary. Live OpenShell confinement is not established. A separately accepted, historical protected-worker campaign qualified only a fixed synthetic worker on one recorded Ubuntu 22.04 and

bubblewrap environment; that result is not transferred to the current selected pins or to arbitrary agents.

8. Group 05 — Replay & Evidence

This group holds the descendants of the original Replay and Evidence layers, joined by the Open Decision Evidence Standard, which is now a component of the stack rather than an adjacent proposal.

Cognous Replay Bundle

The Replay Bundle is a portable reconstruction format, importer, validator and CLI for agent-run evidence. **Reconstruction Bundle 0.2.0** imports the actual records of exact supported Control Plane and executor revisions, and rejects unknown producer revisions rather than guessing compatibility from shape. It preserves proposal, decision and effect identity while keeping Control Plane and executor attempt namespaces separate; it retains unknown acknowledgements, rejected or null observations and evidence-only recovery lineage; and it represents declared coverage and missing information explicitly. Legacy bundle formats and historical provenance remain distinct. Validation, summaries, redaction and export-integrity helpers remain available.

The distinction from logging still matters. A log records events; a reconstruction bundle connects what was proposed, decided, attempted and observed, including what remains uncertain, as one portable record.

What it is not. Reconstruction is non-effecting: it does not rerun the action, reevaluate policy, renew authorization or independently verify delivery. Reconstruction completeness is not destination finality. A valid digest or HMAC is not institutional authority, and the reference does not provide production key management.

Cognous Governance Evidence Pack

The Governance Evidence Pack is a reference format, validator, traceable importer and Markdown renderer for business-facing review. Its importer transforms Manifest and Reconstruction Bundle inputs with explicit source references and transformation versions, so a reviewer can follow each summarized decision or outcome back to the record it came from. It applies **assurance separation**: declared controls, source-asserted runtime facts, executed tests and independent verification are labeled as different things. Continuity checks validate source identities, commitments and producer compatibility.

It gives enterprises a structured answer to the question: "What evidence do we have about this agent's actions, and how much of it was actually verified?"

What it is not. Import success does not establish that a test ran, that an institution authorized current reliance or that a control is effective in production. An evidence pack is not certification, legal advice, an independent audit or a runtime enforcement layer.

Open Decision Evidence Standard (ODES)

ODES is an open, vendor-neutral discussion draft and reference tooling for portable decision evidence. Its v0.2 narrative and implementation profile retain the pder - v0.1 record schema. An ODES package carries authority basis, machine role, human disposition, model and policy context and evidence references across an organizational boundary, bound to explicit integrity commitments. The recipient validates the package and evaluates it under its own policy.

What it is not. ODES carries coordinates for a recipient's own reliance decision; it does not make that decision. Integrity is not authenticity, package validity cannot fill in missing authority evidence, and a valid package grants no execution authority. Reference compatibility is narrower than standards adoption.

9. Group 06 — Reasoning & Human Oversight

The sixth group addresses the human and model side of oversight. Its three components are optional instruction protocols, built as reusable SKILL.md packages, that a deployment can apply to general-purpose AI work.

- **Portable Reasoning Protocol (PRP)** asks a model to calibrate rigor to consequence, separate evidence from inference, state material uncertainty, avoid fabricating facts, citations, tool results or completed actions, and preserve the user's agency.
- **Research Intelligence** structures research work in two components: Discovery, which separates observation from interpretation and prioritizes discriminating experiments, and Abstractor of Abstractors (AoA), which proposes transferable structural invariants with explicit limits.
- **TFA** is a lightweight behavioral protocol built on three rules: say what is true; ask for nothing; protect their next move.

These protocols give reviewers a shared vocabulary for what good model reasoning and interaction should look like before a human relies on it.

What it is not. These are behavioral instructions, not runtime enforcement and not authority. Their checks in the reference are instruction and schema checks; model-behavior efficacy has not been measured. They do not establish that human oversight occurred. If a consequential action follows a reasoning conclusion, the deployment must still check authority and constrain execution separately.

10. Cross-Cutting Integration — Governed Exchange

Agents increasingly act on messages from other systems. Message receipt, understanding, acceptance, authorization, execution, observation and verification are different events, and they must stay different when messages are duplicated, acknowledgements are lost or a workflow resumes under changed authority.

Cognous Governed Exchange (GAX/IMX) is an experimental reference that connects governed messages to recipient assessment, bounded execution and retained evidence. Within it, **GAX** supplies exchange semantics (assessing message structure, profile, identity binding, references and freshness), **LocalDurableTransport** supplies local delivery and redelivery, and **IMX** supplies continuity records. The recipient adapter routes a Manifest-bound proposal through the trusted Control Plane and executor using a caller-supplied resolver and policies. On redelivery it returns the retained original Replay, ODES and IMX artifacts when available rather than executing again; if evidence was never retained, evidence-only recovery creates an explicitly derived record without a replacement effect.

What it is not. It is a bounded refund profile, not full GAX conformance, a production messaging network or institutional authentication. It is separate from the Institutional Governance repository: it implements exchange behavior, not an institution or a constitution. Message identifiers and receipt never authorize an action.

11. How the Components Work Together: One Synthetic Refund

The stack is easiest to understand through its one supported workflow. Suppose a synthetic refund request arrives from another system.

1. **Declare.** Before runtime, a reviewer has published an Action Manifest (1.1) declaring the refund action, its permitted adapter, target and effect limits. The declaration grants no permission.
2. **Receive and assess.** LocalDurableTransport delivers the refund message to the Governed Exchange recipient adapter. GAX assesses it and produces a Manifest-bound proposal. The message carries no authority; neither would a valid signature on it.
3. **Resolve authority.** A separately configured trusted resolver supplies the current Authority Context (institution, mandate, grant, approval, policy, evaluation time), independently of the request.
4. **Decide and revalidate.** The Control Plane evaluates the exact proposal and records its decision. Immediately before dispatch it revalidates the tenant, action, target, payload and current grant, policy and evidence. In the default path this is pre-effect revalidation, not destination-commit atomicity.
5. **Execute within bounds.** The Execution Runtime attempts one identified local effect, E1, against the synthetic SQLite destination, bound to the exact operation content.
6. **Handle uncertainty.** Suppose the caller times out before receiving an acknowledgement. The outcome is **unknown**: timeout is not failure, and it does not authorize a retry. Reconciliation reads the destination independently of the executor's acknowledgement, using E1. If E1 is present and applied, the historical effect is retained even if current authority is later denied. If E1 is absent at one observation, that is point-in-time evidence only; an in-flight effect can still commit, so no safe retry or settlement is inferred.
7. **Reconstruct.** The Replay Bundle imports the actual Control Plane and executor records, keeping decision, attempt, acknowledgement and observation as separate facts with their sources.
8. **Package for review and exchange.** The hub consumes the original retained Replay, ODES and IMX artifacts for that delivered operation and produces a Governance Evidence Pack. It checks effect content and count, decision, effect and attempt continuity, original artifact commitments and recipient assurance limits. It does not substitute an unrelated direct execution for evidence of the delivered message.

One negative control matters for business readers. A second valid proposal with a different effect identifier, E2, but the same underlying refund intent is a separate risk. Default effect-ID deduplication suppresses repeat delivery of E1; it does **not** deduplicate equivalent business intent across E1 and E2. A refund-intent profile exists in the selected runtime as a separate optional profile, but the default path does not enable it.

The business outcome. At the end of the run, a reviewer can see what was declared, what authority was resolved and from where, what was permitted at effect time, what was attempted, what the destination was observed to contain, what remains uncertain, and how each summary traces to an original record. Every answer is a record with stated limits, not a recollection.

12. What Makes the Integrated Stack Valuable

The value claims below describe what the reference is designed to make examinable. None has been measured as an enterprise outcome.

Declaration before deployment. Review can start from a validated, runtime-bindable action surface instead of a purpose statement or a tool list.

Authority that does not travel with the request. Independent resolution means that controlling a message does not mean controlling the authority behind it.

Authorization that stays current. Effect-time revalidation narrows the window in which a withdrawn grant or approval can still be acted on, while being explicit that it does not close it.

Honest outcomes. Decision, attempt, acknowledgement and observed destination state are kept apart. Unknown remains unknown; a later denial does not erase an earlier valid effect.

Traceable review. Evidence Pack summaries trace to original records, and declared controls are labeled differently from verified ones.

Portable evidence. ODES lets a recipient evaluate decision evidence under its own policy, without mandatory disclosure of raw evidence.

Executable qualification. Required scenarios resolve to actual test evidence, run in two isolated repetitions, with known limitations characterized rather than hidden.

Separation of responsibilities. Agent teams keep their frameworks and models. Each component has one named responsibility and an explicit statement of what it does not establish, which makes the chain easier to review and to extend.

13. How It Differs from Adjacent Categories

The Open Control Stack is complementary to most adjacent categories. The differences are worth stating precisely.

Category	What it does	What it misses	How the stack differs
Agent frameworks	Plan, call tools, orchestrate	Independent authority and effect-time records	Sits beside the framework; binds, decides and records
Tool lists and prompts	Name systems; describe intent	Validated action semantics and limits	Manifest 1.1 binds proposals to declared actions
Logs and observability	Record events	Separation of decision, attempt and observation	Keeps those as distinct, reconcilable records
Policy documents	State what should happen	Evidence of what was permitted at effect time	Records revalidated decisions per operation

Category	What it does	What it misses	How the stack differs
GRC checklists and AI governance platforms	Program-level inventories and workflows	Run-level, source-traceable artifacts	Supplies artifacts a platform could consume
Signatures and ledgers	Prove origin or inclusion	Authority to act	Treats them as evidence only, never permission

ODES no longer belongs in this table as an adjacent category: it is part of the stack (Group 05). Its role is still carefully bounded. ODES is a vendor-neutral discussion draft for portable decision evidence; the stack's reference tooling exports ODES packages from exact supported Manifest and Replay inputs through a documented mapping. That is reference compatibility, not a claim of standards adoption or universal interoperability.

14. Current Evidence and Release Status

Release boundary. Three levels need to be kept apart:

- **CO, the selected default.** Authority Context is resolved independently, and decision-critical inputs are revalidated immediately before dispatch. This is pre-effect revalidation, not destination-commit atomicity: a grant, approval, policy or evidence state can change after the final check and before the effect commits.
- **C1, optional and local only.** A separately qualified same-host profile orders cooperating authority mutations, claim and budget consumption and the synthetic effect through one participating SQLite transaction boundary. Its qualification run passed 73 tests, separately from the release run. It is not enabled by default, does not cover external processors and does not establish distributed atomicity.
- **C2 and C3: no claim.** No remote commit protocol, distributed coordinator, external settlement finality or production exactly-once guarantee is established.

Selected generation. The hub's component lock selects profile merged-producers-v1, pinning all 13 components to exact revisions. Newer component releases are not selected automatically.

Current release evidence. CI run 37694032916 resolved all **35 acceptance scenarios**, with required suites executed twice. A fresh reproduction on 13 October 2026, reviewed for the publication checkpoint, ran 1,009 Python testcases in each of two repetitions plus 20 separate Node tests, with 34 invariants passing and one characterized limitation (the business-intent case in Section 11). Read-only inspection of the retained destinations confirmed one synthetic effect for the representative and lost-acknowledgement cases, and zero effects where the resolver was missing or authority was revoked. An earlier 915-test figure belongs to a previous persistence generation and is historical; it is not added to, or substituted for, current counts. Separate profile populations (such as C1's 73 cases) are not added into a single qualification total.

Release state. Release closeout froze feature intake on 13 October 2026. The publication checkpoint accepted the frozen-snapshot and reproducible-example gates for the named synthetic evidence;

licensing, publication safety, disclosure and reporting gates remain partial. Public preview approval is pending. Final publication and operational trust (tracked as #30) remain on **HOLD**.

15. What the Stack Is Not

Scoped claims are part of the design. Cognous Open Control Stack is not:

- an agent framework or a model runtime;
- a hosted dashboard or a complete enterprise AI governance platform;
- a production deployment, a payment integration or evidence of external settlement;
- a compliance certification product, legal advice or an independent audit;
- proof of institutional adoption or ratification;
- a security boundary by itself, or a substitute for application authorization;
- a guarantee of rollback, exactly-once delivery or destination-commit atomicity;
- a guarantee that model outputs are correct or that human oversight occurred;
- a proprietary trust network.

Several deployment prerequisites remain outside the release and must be qualified separately: production resolver authentication, identity and credential custody, remote revocation propagation, live OpenShell confinement, arbitrary-agent confinement, deployment-wide bypass resistance, distributed budgets, external payment settlement and independent operational review. Human-review burden, model-behavior improvements and enterprise benefit have not been measured. The quality of any real deployment's records will depend on its upstream systems, authority processes, security controls and implementation discipline.

16. Why Open Source Matters

For artifacts whose purpose is trust and review, inspectability is a design requirement, not a distribution choice. Publishing the formats, schemas, validators, reference behavior, examples and tests lets enterprise teams inspect exactly what each record contains, run the synthetic workflow themselves and compare expected with observed results before committing to any platform.

Publishing openly provides:

- **Transparency.** Reviewers can see what a manifest, decision record, reconstruction bundle, evidence pack or ODES package contains, and what each one disclaims.
- **Reproducibility.** The qualification tooling runs pinned sources in isolated repetitions, so claims can be checked rather than taken on trust.
- **Vendor-neutral evaluation.** Teams can assess the artifact chain on its merits, without a procurement commitment.
- **Integration clarity.** Published contracts show how other systems can produce and consume the artifacts.

- **Shared vocabulary.** Open reference artifacts help define what declaration, authority, execution and evidence records should contain.

The open layer is intentionally narrow. Commercial systems can compete above it on hosted services, integrations, enterprise storage, identity integration, approval workflows, compliance mappings, viewers, dashboards, deployment support and service levels. The reference defines the records; it does not attempt to be the platform.

Licensing is component-specific. Existing licenses and notices apply to each repository, and the selected stack should not be described as uniformly licensed. Distribution rights for some selected optional components are still under review as part of the publication gates.

17. Strategic Summary and Practical Next Step

Enterprises are moving from AI experimentation to agents that act. The risk shifts from what a model says to what an agent does, and the governance progression follows: chatbots require output review; agents require action governance; governed agents require authority that is current at the moment of effect; and enterprise programs require evidence that stays honest about uncertainty.

Cognous Open Control Stack does not claim to solve AI governance. It claims something narrower and more testable: that declaration, independently resolved authority, bounded execution and traceable evidence can be connected in one reproducible reference, with each boundary stated explicitly. Its practical value today is as an evaluation instrument for teams designing their own controls.

To evaluate it, start with the hub's "Start here" guide ([docs/start-here.md](https://github.com/cogno-us/cogno-us/blob/main/docs/start-here.md)) at [github.com/cogno-us/cogno-us-open-control-stack](https://github.com/cogno-us/cogno-us/cogno-us-open-control-stack). Developers can reproduce the reference on a disposable Linux checkout with Python 3.11 and Node 20:

```
git clone https://github.com/cogno-us/cogno-us-open-control-stack.git
cd cogno-us-open-control-stack
python3.11 -m venv .venv
source .venv/bin/activate
python tools/reference_release.py run --results-dir results/reference
```

Review begins with `results/reference/scenario-results.json`. Governance reviewers can follow the governance quickstart ([docs/governance-quickstart.md](https://github.com/cogno-us/cogno-us/blob/main/docs/governance-quickstart.md)), which walks through one synthetic refund question by question. For any real pilot, separately qualify identity and credential custody, independent authority resolution, destination semantics, revocation propagation, recovery and operational observation.

Before enterprises can rely on governed AI agents, they need more than prompts, logs and dashboards. They need records that keep proposal, authority, effect and evidence distinct, and an open reference that shows exactly where those records stop.

Cognous · cogno.us · github.com/cogno-us/cogno-us-open-control-stack