Cognous Agent Governance Evidence Pack: Business-Facing Evidence Packages for Governed AI-Agent Deployment

André de Lima
Cognous
Seattle, Washington
andredelima@cogno.us

July 2026

Abstract

Enterprise AI agents increasingly use tools, retrieve data, propose actions, rely on external sources, trigger workflows, and produce outputs that affect business decisions. As organizations move from demonstrations to governed deployment, technical runtime records alone are not sufficient. Security, risk, compliance, audit, legal, and executive reviewers need business-facing evidence that explains what an agent is deployed to do, which systems it can access, which actions it can propose, which authority and review controls apply, what was blocked, what sources were relied on, what replay bundles exist, what validation results are available, what redacted or signed exports were produced, what risks remain open, and who reviewed the evidence.

Cognous Agent Governance Evidence Pack is a minimal public reference format for translating AI-agent runtime governance records into business-facing evidence packages. It provides a JSON-backed schema, Pydantic object model, semantic validator, markdown renderer, command-line utilities, examples, and documentation for producing reviewable governance artifacts. It is intentionally scoped: it is not an agent framework, not a model runtime, not a compliance certification product, not a hosted dashboard, not a security boundary by itself, and not a complete enterprise governance platform. Its contribution is narrower and operational: a structured evidence layer that helps teams summarize agent deployment context, action inventories, authority controls, policy controls, blocked actions, reliance records, replay bundles, validation results, redacted exports, risks, and governance review decisions in a form business stakeholders can inspect.

1 Introduction

AI-agent systems are moving from isolated prototypes toward operational deployment. In enterprise settings, agents may read customer records, search internal documents, draft communications, generate recommendations, propose updates to systems of record, and participate in workflows that require authority or review. These capabilities create a governance problem that cannot be solved by technical logs alone.

A runtime system may produce detailed records: action proposals, policy decisions, blocked actions, reliance records, replay bundles, validation reports, redaction metadata, and export signatures. These records are necessary, but they are not always sufficient for business review. A security reviewer, audit committee, risk officer, customer-facing compliance team, or executive sponsor needs a higher-level artifact that answers the business questions around a deployment.

Those questions include:

1. What agent is being reviewed?
2. What business purpose does the agent serve?
3. Where is it deployed?
4. What systems, tools, and data domains does it touch?
5. What actions can it propose?
6. Which actions require authority?
7. Which actions require human review?
8. What policy controls are in place?
9. What actions were blocked?
10. What sources did the agent rely on?
11. What replay bundles or run records exist?
12. What validation results are available?
13. What exports were redacted or signed?
14. What risks remain open?
15. Who reviewed the evidence, and what decision was reached?

Agent Governance Evidence Pack addresses this evidence-translation problem. It converts technical runtime and governance records into a structured, business-readable package. The package can be rendered as markdown, validated for internal consistency, and used as a review artifact for security, risk, audit, compliance, and deployment governance.

The guiding design principle is:

> No enterprise agent should move from demonstration to governed deployment without a business-facing evidence package that summarizes its purpose, action surface, controls, runtime evidence, risks, and review decisions.

The public reference implementation exposes a small set of public objects and utilities: `EvidencePack`, `AgentOverview`, `DeploymentContext`, `ToolInventoryItem`, `ActionInventoryItem`, `AuthorityModelSummary`, `PolicyControlSummary`, `BlockedActionSummary`, `RelianceSummary`, `ReplayBundleInventoryItem`, `ValidationSummary`, `RedactionExportSummary`, `RiskRegisterItem`, `ReviewRecord`, `ValidationIssue`, `ValidationReport`, a command-line interface, JSON schemas, example evidence packs, a markdown renderer, validation helpers, and a test suite.

2 Problem Statement

Enterprise AI-agent governance has a translation gap. Runtime systems may produce technical records, but business reviewers need evidence packages that are organized around deployment questions, risk controls, and review decisions.

This gap appears in several operational forms.

2.1 Runtime Records Are Too Technical for Business Review

Technical records are often structured around events, identifiers, traces, and payloads. They may be appropriate for developers or incident responders, but they are not always suitable for executives, risk teams, auditors, or business owners. A replay bundle may reconstruct what happened in a run, but a governance review also needs to understand what the deployment is for, what controls exist, what risks remain, and what decision was made.

2.2 Evidence Is Scattered Across Systems

Relevant evidence may live in agent manifests, control-plane records, replay bundles, validation reports, redacted exports, risk registers, approval workflows, monitoring systems, and manual review notes. Without a common evidence package, reviewers must assemble the story manually.

2.3 Governance Reviews Lack a Stable Artifact

Many organizations conduct AI reviews through meetings, spreadsheets, slide decks, ticket comments, or policy checklists. These may be useful, but they are often not structured enough to validate, render consistently, or compare across deployments. Agentic systems require review artifacts that can be produced, versioned, inspected, and updated as deployment posture changes.

2.4 Blocked Behavior Is Often Invisible

A successful deployment is not only one where harmful actions did not occur. It is one where inappropriate or unauthorized actions were proposed and stopped. Business reviewers need explicit summaries of blocked actions, including the action type, tool involved, count, reason summary, and evidence reference.

2.5 Source Reliance Is Hard to Summarize

Agents rely on documents, APIs, databases, tools, user inputs, and model outputs. Technical reliance records may exist, but business reviewers need an aggregate summary: which sources were relied on, how often, for what scope, and where the supporting records can be found.

2.6 Validation and Export Integrity Are Not Business-Visible

Validation reports, redaction metadata, and signatures are technical mechanisms. Their business relevance is that they indicate whether evidence packages are internally consistent, whether sensitive material was removed before sharing, and whether exported records can be checked for modification. These signals need to be visible in a business-facing artifact.

2.7 Risk and Review Decisions Are Detached from Runtime Evidence

Risk registers and review decisions often exist separately from runtime evidence. An evidence pack connects the review posture to the underlying records: what risks remain open, what controls support the deployment, what replay bundles exist, and who approved, rejected, or conditionally approved the deployment.

3 Design Goals

Agent Governance Evidence Pack is designed around ten goals.

1. **Business-facing evidence.** Runtime governance evidence should be summarized in a form business stakeholders can inspect.
2. **Deployment context.** The evidence package should identify the agent, business purpose, environment, systems touched, data domains, and ownership.
3. **Action inventory.** The package should summarize what actions the agent can propose and whether those actions require authority, review, or reliance records.
4. **Control summary.** Authority models and policy controls should be represented explicitly.
5. **Blocked-action visibility.** Blocked behavior should be summarized as evidence, not omitted.
6. **Reliance visibility.** Sources used by the agent should be summarized for review.
7. **Replay and validation linkage.** Replay bundles and validation summaries should be referenced in the package.
8. **Redaction and export tracking.** Redacted exports and intended recipients should be visible.
9. **Risk and review integration.** Risks and governance decisions should be captured in the same artifact as runtime evidence.
10. **Portable rendering and validation.** Evidence packs should be loadable, inspectable, validated, and rendered through a small public schema and CLI.

4 Non-Goals

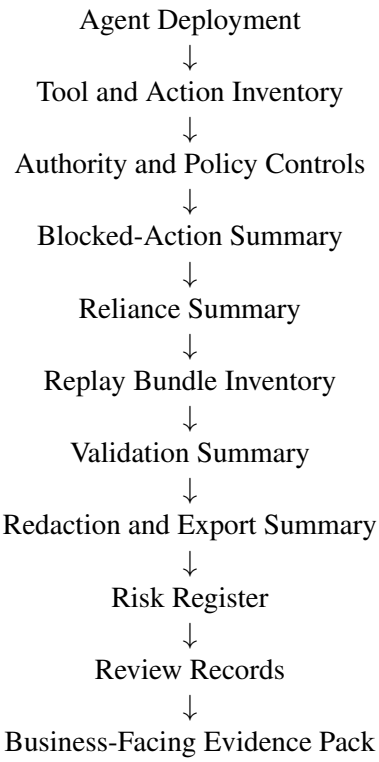
The architecture is deliberately bounded. Agent Governance Evidence Pack is not:

- an agent framework;
- a model runtime;
- a runtime control plane;
- a replay viewer;
- a hosted dashboard;
- a compliance certification product;
- a complete enterprise governance platform;
- a production authorization system;
- a security boundary by itself;
- a substitute for application-level authorization;
- a guarantee that model outputs are true, safe, or complete;
- a guarantee that regulatory obligations have been satisfied.

It summarizes available governance evidence and validates package consistency. It does not execute agents, enforce runtime access control, certify compliance, or prove that a deployment is safe.

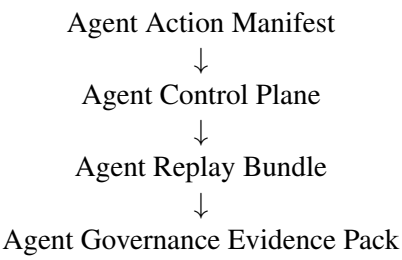
5 System Overview

The evidence packaging pattern is:



The evidence pack is created during deployment review, governance review, incident review, or periodic audit. It may be populated manually, from runtime systems, from replay bundles, from action manifests, or from other governance tooling.

In the Cognous Open Control Stack, Agent Governance Evidence Pack is the *Evidence* layer:



The manifest declares what an agent may propose. The control plane records and governs what the agent actually proposes at runtime. Replay bundles package technical run records. Evidence packs translate those records into business-facing governance review.

6 Core Object Model

6.1 EvidencePack

An `EvidencePack` is the top-level object. It describes the evidence package, review status, agent overview, deployment context, tool inventory, action inventory, authority model, policy controls, blocked actions, reliance records, replay bundles, validation summary, redaction exports, risk register, review records, and optional metadata.

Field	Meaning
`pack_id`	unique evidence pack identifier
`pack_version`	evidence pack format version
`title`	human-readable title
`generated_at`	timestamp when the pack was generated
`review_status`	draft, in review, approved, approved with conditions, rejected, or retired
`agent_overview`	agent identity and business purpose
`deployment_context`	environment, systems touched, data domains, geography, and notes
`tool_inventory`	tools and systems available to the agent
`action_inventory`	actions the agent can propose
`authority_model`	summary of authority governance
`policy_controls`	controls applied to the deployment
`blocked_actions`	aggregate blocked-action records
`reliance_summary`	aggregate source reliance records
`replay_bundles`	available replay bundles or run records
`validation_summary`	validation results across supporting records
`redaction_exports`	redacted or exported review artifacts
`risk_register`	risks, severity, status, mitigation, and owner
`review_records`	reviewer decisions and notes
`metadata`	optional extension metadata

6.2 AgentOverview

`AgentOverview` identifies the agent under review. It records the agent name, optional description, business purpose, owner, business unit, user population, and lifecycle stage.

6.3 DeploymentContext

`DeploymentContext` describes where and how the agent is deployed. It records the environment, deployment name, deployment date, systems touched, data domains, geographic scope, and additional notes.

6.4 ToolInventoryItem

`ToolInventoryItem` describes a tool or system available to the agent. It records the tool name, description, external system linkage, data classification, access mode, and control status.

6.5 ActionInventoryItem

`ActionInventoryItem` describes an action the agent can propose. It records the action name, tool name, action type, description, whether authority is required, whether review is required, whether reliance is

required, default posture, and control status.

6.6 AuthorityModelSummary

`AuthorityModelSummary` describes how privileged actions are governed. It includes a narrative summary, authority scopes, privileged action types, whether authority grants must expire, whether human approval is required, and optional notes.

6.7 PolicyControlSummary

`PolicyControlSummary` records a policy control, its description, implementation status, evidence reference, and optional notes.

6.8 BlockedActionSummary

`BlockedActionSummary` summarizes blocked behavior. It records the action name, action type, tool name, count, reason summary, and evidence reference.

6.9 RelianceSummary

`RelianceSummary` summarizes sources the agent relied on. It records the source name, source type, count, scope summary, and evidence reference.

6.10 ReplayBundleInventoryItem

`ReplayBundleInventoryItem` records available replay bundles or run records. It includes the bundle ID, run ID, status, generation timestamp, signed flag, redacted flag, validation status, and evidence reference.

6.11 ValidationSummary

`ValidationSummary` summarizes validation results across supporting replay bundles or evidence artifacts. It includes counts of valid bundles, invalid bundles, warnings, errors, and an optional narrative summary.

6.12 RedactionExportSummary

`RedactionExportSummary` records redacted exports or evidence artifacts prepared for specific recipients. It includes the export ID, export type, redacted fields, generation timestamp, intended recipient, and evidence reference.

6.13 RiskRegisterItem

`RiskRegisterItem` records a known risk. It includes the risk ID, title, description, severity, status, mitigation, owner, and evidence reference.

6.14 ReviewRecord

`ReviewRecord` records a governance review decision. It includes the reviewer, reviewer role, reviewed timestamp, decision, and notes.

6.15 ValidationIssue and ValidationReport

`ValidationIssue` records an error or warning discovered during evidence-pack validation. A `ValidationReport` aggregates validation issues and marks the pack as valid only when no error-severity issues are present.

7 Review Status

The evidence pack supports the following review statuses:

Review Status	Meaning
`draft`	evidence pack is being prepared
`in_review`	evidence pack is under review
`approved`	deployment or evidence package has been approved
`approved_with_conditions`	approval is conditional on stated review notes or mitigations
`rejected`	deployment or evidence package has been rejected
`retired`	evidence pack refers to a retired deployment

Review status is not a compliance certification. It is an evidence-package status that helps teams track review posture.

8 Deployment Environments

The public model supports the following deployment environments:

- `development`;
- `staging`;
- `production`;
- `sandbox`;
- `other`.

Production deployments receive additional warnings when review posture, redaction exports, or signed replay bundles are missing. This reflects the stronger evidence expectations that apply once agents move beyond experimental environments.

9 Action Types and Privileged Actions

The evidence pack uses action types to summarize the agent's action inventory. The current public reference implementation supports:

Action Type	Meaning
`read`	retrieves or inspects information
`write`	creates or modifies information
`external_send`	sends information outside the immediate system boundary
`delete`	removes information or records
`export`	packages or transfers information for downstream use
`purchase`	initiates or prepares a purchase action
`approve`	approves, confirms, or authorizes a workflow step
`escalate`	escalates to a human, queue, or supervisory process
`other`	action type not otherwise classified

For warning purposes, the reference implementation treats `external_send`, `write`, `delete`, `export`, `purchase`, and `approve` as privileged action types. If these actions lack authority or review indicators, the validator emits warnings. These warnings do not invalidate the evidence pack, but they surface issues for review.

10 Control Status

Tools, actions, and policy controls can be assigned a control status:

Control Status	Meaning
`implemented`	control is implemented
`partial`	control is partially implemented
`planned`	control is planned but not yet implemented
`not_applicable`	control does not apply

Control status is informational. It helps reviewers understand which parts of the governance posture are active, incomplete, planned, or out of scope.

11 Risk Register

The evidence pack includes a risk register so that governance review can connect deployment evidence to open risk.

Risk severity values are:

- `low`;
- `medium`;
- `high`;
- `critical`.

Risk status values are:

- `open`;
- `mitigated`;
- `accepted`;
- `transferred`;
- `closed`.

The validator treats critical open risk as incompatible with full approval. If an evidence pack has review status `approved` and contains a critical open risk, validation returns an error. This does not certify that all non-critical risks are acceptable; it establishes a minimal consistency rule for the public reference implementation.

12 Validation

The current public reference implementation validates evidence-pack consistency and emits a `ValidationReport`. Errors invalidate the evidence pack. Warnings indicate review concerns but do not make the pack invalid.

12.1 Error Conditions

The validator checks error conditions including:

1. `pack_id` must be non-empty.
2. `title` must be non-empty.
3. `generated_at` must be non-empty.
4. `agent_overview.agent_name` must be non-empty.
5. `agent_overview.business_purpose` must be non-empty.
6. `deployment_context.environment` must be present.
7. tool names in `tool_inventory` must be unique.
8. action names in `action_inventory` must be unique.
9. every action must reference a known tool.
10. blocked-action counts must not be negative.
11. reliance counts must not be negative.
12. risk IDs must be unique.
13. replay bundle IDs must be unique.
14. redaction export IDs must be unique.

15. `approved` or `approved_with_conditions` status requires at least one review record.
16. approved review records require a `reviewed_at` timestamp.
17. critical open risks prevent full approval.
18. authority-required actions require an authority model.
19. review-required actions require a policy control mentioning review or approval.
20. full approval requires at least one replay bundle.

12.2 Warning Conditions

The validator checks warning conditions including:

- owner is missing;
- business unit is missing;
- systems touched is empty;
- data domains is empty;
- tool inventory is empty;
- action inventory is empty;
- policy controls are empty;
- blocked actions are empty;
- reliance summary is empty;
- replay bundles are empty;
- validation summary is missing;
- redaction exports are empty;
- risk register is empty;
- review records are empty;
- production deployment has status `draft` or `in_review`;
- production deployment has no redaction exports;
- production deployment has no signed replay bundles;
- privileged action types lack review indicators;
- privileged action types lack authority indicators;
- external-system tools lack data classification.

Validation remains implementation-level. It does not prove that evidence is complete, that controls are sufficient, that runtime enforcement exists, or that compliance obligations have been satisfied.

13 Markdown Rendering

The public reference implementation includes a markdown renderer. Rendering is important because governance evidence must be readable by humans, not only consumed by machines.

The renderer produces a business-facing report with the following structure:

1. Executive Summary
2. Agent Overview
3. Deployment Context
4. Tool Inventory
5. Action Inventory
6. Authority Model
7. Policy Controls
8. Blocked-Action Summary
9. Reliance Summary
10. Replay Bundle Inventory
11. Validation Summary
12. Redaction and Export Summary
13. Risk Register
14. Review Records
15. Known Limitations

The executive summary includes counts of tools, actions, privileged actions, replay bundles, validation results, and open risks. When a pack is approved with conditions and review notes exist, the renderer surfaces that condition early and points readers to the review records section.

The rendered markdown is not a separate source of truth. It is a deterministic presentation of the underlying evidence-pack data.

14 Command-Line Interface

The `agep` command-line interface provides small utilities for local review and automation.

14.1 Implemented Commands

The current public reference implementation includes:

- `agep validate <path/to/evidence_pack.json>`;
- `agep summarize <path/to/evidence_pack.json>`;
- `agep render <path/to/evidence_pack.json> [--out output.md]`;
- `agep check-examples`.

The validation command loads an evidence pack, runs semantic validation, prints errors and warnings, and returns a process exit code. The summarize command prints compact counts and status fields. The render command produces a business-facing markdown report. The check-examples command validates and renders all included examples.

These commands are intended for local validation, examples, and lightweight workflows. They are not a hosted service, daemon, dashboard, or enterprise control surface.

15 JSON Schemas and Public Conformance

The repository includes JSON Schema Draft 2020-12 files for the public evidence-pack objects. The primary schema is `evidence_pack.schema.json`. These schemas define the public interchange boundary for records such as `EvidencePack`, `AgentOverview`, `DeploymentContext`, `ToolInventoryItem`, `ActionInventoryItem`, `AuthorityModelSummary`, `PolicyControlSummary`, `BlockedActionSummary`, `RelianceSummary`, `ReplayBundleInventoryItem`, `ValidationSummary`, `RedactionExportSummary`, `RiskRegisterItem`, and `ReviewRecord`.

The schemas serve four purposes:

1. **Interchange.** They allow external systems to understand the shape of evidence-pack records.
2. **Validation.** They support structural validation outside the Python package.
3. **Documentation.** They make the public evidence boundary explicit.
4. **Conformance.** They provide a basis for future compatibility tests across implementations.

The schema boundary is deliberately limited. JSON schemas validate structure, required fields, field types, and allowed values where specified. They do not prove that an evidence pack is operationally sufficient or that a compliance obligation has been satisfied.

16 Example Evidence Packs

The public repository includes example evidence packs for common enterprise agent scenarios.

16.1 Customer-Service Agent

A customer-service agent reads CRM records, drafts outbound responses, and records blocked send attempts. The evidence pack summarizes tools, customer-data domains, action inventory, authority and review controls, blocked email sends, reliance records, replay bundles, validation status, redaction exports, risks, and review decisions.

16.2 Internal Research Agent

An internal research agent searches documents, reads files, generates summaries, and prepares exports. The evidence pack summarizes document access, export controls, reliance sources, replay artifacts, validation status, and open review concerns.

16.3 Procurement Agent

A procurement agent compares vendor quotes, drafts purchase recommendations, proposes purchase orders, and sends vendor communications. The evidence pack summarizes authority controls, approval conditions, blocked purchase-order behavior, redacted exports, risks, and review decisions.

16.4 Finance Workflow Agent

A finance workflow agent reads invoices, generates payment recommendations, proposes approvals, and exports audit records. The evidence pack summarizes privileged action controls, approval requirements, replay bundles, validation results, redaction posture, risks, and review records.

17 Relationship to Runtime Evidence

Agent Governance Evidence Pack is not a runtime recording layer. It depends on evidence produced elsewhere, which may include action manifests, runtime control records, replay bundles, validation reports, redaction exports, signed bundles, risk records, or human review notes.

A runtime evidence layer can populate an evidence pack in several ways:

- action manifests can provide declared tools and action inventories;
- control-plane records can provide blocked-action summaries, authority records, and reliance summaries;
- replay bundles can provide run-level evidence and reconstruction references;
- validation reports can provide consistency results;
- redaction helpers can provide export metadata;
- signing helpers can provide export-integrity metadata;
- governance workflows can provide risk and review records.

The evidence pack does not require a specific upstream implementation. It can be populated manually or automatically. Its role is to provide a stable business-facing evidence format.

18 Security Considerations

Agent Governance Evidence Pack improves reviewability and governance transparency, but it should not be treated as a security product.

Important limitations include:

- the evidence pack summarizes records but does not enforce policy;
- review status is not compliance certification;

- validation checks structure and consistency, not operational sufficiency;
- blocked-action summaries depend on upstream runtime records;
- reliance summaries depend on upstream provenance capture;
- redaction export summaries do not guarantee that all sensitive data was removed;
- signed replay bundle indicators do not replace production key management;
- risk registers are only as complete as the review process that maintains them;
- production deployments still require authentication, authorization, monitoring, incident response, runtime control, and organizational governance.

The correct security posture is layered. Agent Governance Evidence Pack supplies structured business-facing evidence. It should be combined with runtime policy gates, run recording, replay bundles, redaction, signing, monitoring, application authorization, risk management, and governance review.

19 Implementation Boundary

The public reference implementation is intentionally thin. It provides enough structure to demonstrate the evidence-pack category and support real integration experiments. It does not reveal or depend on private Cognous product internals.

The public boundary is therefore:

Public Reference Layer	Private Product Layer
Evidence pack schema	hosted evidence registry
Pydantic object model	enterprise evidence store
Semantic validator	compliance workflow automation
Markdown renderer	hosted review interface
Tool inventory summaries	automated tool discovery
Action inventory summaries	enterprise action registry integration
Authority model summary	customer-specific authority workflows
Policy control summary	policy authoring and enforcement systems
Blocked-action summary	runtime analytics and dashboards
Reliance summary	production provenance pipeline
Replay bundle inventory	replay viewer and investigation tooling
Validation summary	automated evidence scoring
Redaction export summary	production export management
Risk register	enterprise risk management integration
Review records	workflow routing and approval systems
CLI utilities	platform APIs and managed services
Example packs	deployment-specific templates
JSON schemas	enterprise conformance tooling

Public software should expose the integration contract: what evidence can be summarized, what reviewers can inspect, what validators can check, and what downstream systems can render. Private product systems can add enterprise integrations, hosted workflows, identity management, richer policy authoring, automated evidence ingestion, dashboarding, workflow routing, and customer-specific governance logic.

20 Reference Implementation Status

The public reference implementation is intentionally small. It is meant to be inspectable, testable, and safe to publish.

Capability	Status	Notes
Core object model	Implemented	Pydantic v2 models
Evidence pack loader	Implemented	JSON file loading and model validation
Evidence pack dumper	Implemented	formatted JSON export
Validation reports	Implemented	error and warning issues
Agent overview	Implemented	identity, purpose, owner, business unit
Deployment context	Implemented	environment, systems, data domains, geography
Tool inventory	Implemented	tools, classifications, control status
Action inventory	Implemented	actions, authority, review, reliance, control status
Authority model summary	Implemented	scopes, privileged types, approval and expiry posture
Policy controls	Implemented	control summaries and evidence references
Blocked-action summary	Implemented	blocked action counts and reasons
Reliance summary	Implemented	source names, types, counts, evidence references
Replay bundle inventory	Implemented	signed, redacted, validation status
Validation summary	Implemented	valid and invalid bundle counts, errors, warnings
Redaction export summary	Implemented	exports, fields, recipients, evidence references
Risk register	Implemented	severity, status, mitigation, owner
Review records	Implemented	reviewer, role, timestamp, decision, notes
Markdown renderer	Implemented	business-facing report output
CLI validation	Implemented	`agep validate`
CLI summary	Implemented	`agep summarize`
CLI rendering	Implemented	`agep render`
Example validation and rendering	Implemented	`agep check-examples`
JSON schemas	Implemented	public schema boundary
Example evidence packs	Implemented	customer service, research, procurement, finance
Test suite	Implemented	model, loader, validator, renderer, CLI, examples
Hosted review UI	Extension	out of current minimal scope
Automated evidence ingestion	Extension	out of current minimal scope
Enterprise approval workflows	Extension	out of current minimal scope
Compliance mapping	Extension	out of current minimal scope

21 Test Suite and Continuous Integration

The repository includes a test suite covering the public reference behavior. Tests are not part of the evidence-pack format itself, but they are part of the public trust surface. They show which claims are exercised by executable checks.

The test suite covers areas such as:

- valid evidence-pack loading;
- invalid JSON behavior;
- model validation behavior;
- required field validation;
- duplicate tool names;
- duplicate action names;
- unknown tool references;
- duplicate risk IDs;
- duplicate replay bundle IDs;
- approval status without review records;
- critical open risk preventing approval;
- authority-required actions without authority model;
- review-required actions without review or approval controls;
- approved packs without replay bundles;
- warning/error distinction;
- production deployment warnings;
- privileged action warnings for authority and review;
- CLI validation;
- CLI summary;
- CLI rendering;
- CLI unknown-flag behavior;
- example pack validation and rendering.

A GitHub Actions workflow runs the test suite and example checks across supported Python versions. This gives the public reference implementation a basic compatibility and regression signal without implying production certification.

22 Example Use Cases

22.1 Deployment Review

An AI platform team prepares to move an agent from staging to production. The evidence pack summarizes the agent's purpose, systems touched, tools, actions, controls, blocked actions, reliance records, validation results, redacted exports, open risks, and review decisions. Instead of reviewing scattered technical artifacts, the review board receives one structured package.

22.2 Security Review

A CISO organization reviews whether an agent has appropriate controls. The evidence pack shows privileged action types, authority requirements, review requirements, policy controls, blocked behavior, data classifications, redacted exports, and signed replay bundle availability. Security can identify missing controls or approve the deployment with conditions.

22.3 Audit Preparation

An audit team needs evidence that a deployed agent was governed. The evidence pack summarizes the available runtime records, replay bundles, validation status, redaction exports, and review decisions. The evidence pack does not replace the underlying records, but it provides the index and business narrative auditors need to inspect them.

22.4 Incident Review

A customer-facing incident raises questions about an agent's behavior. The evidence pack identifies relevant replay bundles, blocked actions, reliance sources, validation status, and review notes. Investigators can move from business summary to technical evidence without reconstructing the evidence trail manually.

22.5 Executive Reporting

Executives need a concise view of agent deployment posture. The rendered evidence pack provides a business-readable summary: purpose, deployment context, number of tools and actions, privileged action count, replay bundle availability, validation results, open risks, and review decision.

23 Evaluation Criteria

The usefulness of Agent Governance Evidence Pack should be evaluated against operational criteria rather than benchmark scores.

A deployment should ask:

1. Does the evidence pack identify the agent being reviewed?
2. Does it explain the agent's business purpose?
3. Does it identify the owner and business unit?
4. Does it describe the deployment environment?
5. Does it identify systems touched and data domains?

6. Does it list tools and their control status?
7. Does it list actions and their action types?
8. Does it identify authority-required actions?
9. Does it identify review-required actions?
10. Does it identify reliance-required actions?
11. Does it summarize authority models and policy controls?
12. Does it summarize blocked actions?
13. Does it summarize reliance sources?
14. Does it inventory replay bundles or run records?
15. Does it summarize validation results?
16. Does it identify redacted or signed exports?
17. Does it include known risks with severity, status, mitigation, and owner?
18. Does it include review records and decisions?
19. Does it validate with no error-severity issues?
20. Does it produce warnings that are useful for review?
21. Can it be rendered into a business-facing report?
22. Can reviewers trace from summary claims to evidence references?

24 Roadmap

The public roadmap should remain implementation-focused and avoid overclaiming.

Implemented MVP capabilities include:

- evidence pack object model;
- agent overview;
- deployment context;
- tool inventory;
- action inventory;
- authority model summary;
- policy control summary;
- blocked-action summary;
- reliance summary;

- replay bundle inventory;
- validation summary;
- redaction export summary;
- risk register;
- review records;
- semantic validation;
- markdown rendering;
- command-line validation;
- command-line summaries;
- command-line rendering;
- example evidence packs;
- JSON schemas;
- test suite and continuous integration.

Near-term extensions may include:

- richer evidence references;
- evidence pack diffing;
- evidence completeness scoring;
- optional import helpers from replay bundles;
- optional import helpers from action manifests;
- additional rendered report templates;
- industry-specific examples;
- conformance test suite;
- machine-readable review conditions.

Longer-term work may include hosted review interfaces, enterprise evidence registries, compliance mapping, approval workflows, automated evidence ingestion, dashboarding, and integrations with governance, risk, and compliance systems. Those are product-layer concerns and should remain separate from the minimal public reference implementation.

25 Conclusion

AI-agent governance cannot end with technical runtime records. As agents gain access to tools, systems, data, and workflows, enterprises need business-facing evidence packages that summarize what the agent is for, where it is deployed, what it can do, what controls apply, what was blocked, what sources were relied on, what replay bundles exist, what validation results are available, what exports were redacted or signed, what risks remain open, and who reviewed the evidence.

Cognous Agent Governance Evidence Pack provides a minimal public reference architecture for that evidence layer. It describes agent overview, deployment context, tool inventory, action inventory, authority models, policy controls, blocked actions, reliance summaries, replay bundles, validation summaries, redaction exports, risk registers, and review records. It validates evidence-pack consistency, publishes JSON schemas for the public evidence boundary, includes example packs for common enterprise scenarios, and provides command-line utilities for local review and markdown rendering.

The contribution is deliberately narrow. Agent Governance Evidence Pack does not run agents, enforce policies, certify compliance, or prove correctness. It supplies a concrete business-facing evidence artifact for a specific operational need: making AI-agent governance review inspectable, structured, and portable.

References

[1] Cognous. *Agent Governance Evidence Pack: Business-facing evidence packages for governed AI-agent deployment*. Public reference implementation, 2026. `https://github.com/cogno-us/cognous-agent-governance-evidence-pack`

[2] Cognous. *Agent Action Manifest: Declare what actions an AI agent may propose before runtime execution*. Public reference implementation, 2026. `https://github.com/cogno-us/cognous-agent-action-manifest`

[3] Cognous. *Agent Control Plane: Runtime control and replay for AI agents*. Public reference implementation, 2026. `https://github.com/cogno-us/cognous-agent-control-plane`

[4] Cognous. *Agent Replay Bundle: Portable replay records for AI-agent runs*. Public reference implementation, 2026. `https://github.com/cogno-us/cognous-agent-replay-bundle`

[5] National Institute of Standards and Technology. *Artificial Intelligence Risk Management Framework*. NIST AI 100-1, 2023.

[6] OWASP Foundation. *OWASP Top 10 for Large Language Model Applications*. Project documentation.

[7] Leslie Lamport. Time, clocks, and the ordering of events in a distributed system. *Communications of the ACM*, 21(7):558–565, 1978.