CAES

Constitutional AI Execution Standard

Normative Specification

Version	v0.2.0 (Draft)
Status	Public Review
Working Group	CAES-WG
Category	Normative Specification
Supersedes	v0.1.1

A standard for pre-execution authorization, cryptographic proof, and fail-closed AI execution.

Public Review Draft | March 2026

Document Map

This professionally formatted edition preserves the normative content of the CAES v0.2.0 public review draft while improving readability for review, circulation, and archival use.

Overview

CAES (Constitutional AI Execution Standard) defines the minimum structural requirements for AI systems that execute actions with legal, financial, regulatory, or operational consequence.

A CAES-conformant system does not merely log what happened. It proves - before execution occurs - that the action was authorized.

This repository contains the normative specification, version history, and working group materials for CAES.

Why CAES Exists

AI systems have moved beyond advisory roles. They now:

- Commit transactions
- Modify infrastructure
- Trigger automated workflows
- Interact with physical systems
- Influence consequential human decisions

Yet there is no widely adopted standard defining what it means to **authorize** an AI action and **prove** that authorization occurred.

CAES addresses this structural accountability gap.

Scope

CAES applies to any AI system that crosses an **Effect Boundary**, including:

- Digital state mutation
- External system invocation
- Communication with consequential impact
- Behavioral influence in regulated contexts
- Physical actuation (robots, vehicles, automation)
- Cross-boundary access to regulated data
- Behavioral output that influences a decision with material consequence
- Recording, observation, or retention of personal or sensitive data

If an AI system can produce consequential effects without independent human mediation, CAES scope applies.

Purely advisory systems are out of scope.

Core Principles

A conformant CAES implementation MUST provide:

- Deterministic policy evaluation
- Pre-execution authorization receipts
- Cryptographically verifiable evidence

- Append-only causal event recording
- Fail-closed enforcement
- Offline-verifiable artifacts
- Explicit authority delegation for high-stakes actions
- Decision resolution following **CPP - Constitutional Policy Protocol**

Governance applies to any output capable of producing consequential effect, including behavioral outputs.

CAES defines structural requirements - not implementation architecture.

Effect Boundary Definition

An **Effect Boundary** is crossed whenever an AI system produces an output that can cause a consequential change in state outside the system itself.

Effect Boundaries MUST be treated as governed action triggers. The following categories are normatively recognized:

Category	Description
ExternalSideEffect	Network calls, filesystem writes, API invocations, database mutations
HumanFacingOutput	Messages, instructions, recommendations, or communications with material consequence
GovernanceRelevantState	Policy changes, permission changes, authorization scope changes
SafetyCriticalActuation	Physical actuation beyond defined safety parameters
WorkflowTransition	Workflow gate passage, run state transition, process-level commitment
BehavioralInfluence	AI-generated output that influences a user decision with legal, financial, or regulatory consequence
SensoryCapture	Recording, observation, or retention of audio, video, personal, or sensitive data

Implementations MAY define additional Effect Boundary categories. Implementations MUST NOT narrow or exclude the categories above.

Internal computation, model inference, and non-effecting reasoning are out of scope unless they cross one of the above categories.

Fail-Closed Semantics

Fail-closed is the mandatory default behavior under uncertainty, failure, or missing information. It is not configurable.

A system is fail-closed if and only if all of the following hold:

21. **Absence of authorization is denial.** An action MUST NOT proceed if a verified Decision Receipt cannot be produced. There is no implicit authorization path.
22. **Verification failure is denial.** If receipt verification fails - for any reason including infrastructure failure, timeout, or mismatch - execution MUST halt.
23. **Policy evaluation failure is denial.** If policy cannot be evaluated to a deterministic result, the disposition MUST be Denied or RequiresHumanAuthorization. No action proceeds.

24. **No silent fallback.** A system MUST NOT fall back to a permissive state under failure. Degraded operation that permits execution without receipts is non-compliant.
25. **Failure is recorded.** Failures that trigger fail-closed behavior MUST produce a recorded denial event. Silent failure is non-compliant.

A fail-closed event MUST:

- produce a denial record equivalent in structure to a Denial Receipt
- append that record to the Governed Spine
- surface a structured error indicator to the caller

Uncertainty is denial. Silence is non-compliant.

Authorization-Execution Separation

CAES requires strict temporal separation between the authorization decision and the execution of the governed action.

Requirements:

26. **Decision precedes execution.** A Decision Receipt MUST be produced, persisted, and verified before any effect-bearing action is initiated.
27. **Receipt is the gate.** The execution layer MUST check for a verified Decision Receipt as its first action. No receipt means no execution.
28. **No reconstruction after the fact.** A receipt produced after execution has occurred does not satisfy this requirement. Post-hoc receipts are non-compliant.
29. **Write-then-verify persistence.** Receipt persistence MUST include an immediate readback and byte-level verification step. Acknowledgment from a storage system is not sufficient. Verification MUST confirm durable storage before execution is authorized.
30. **Receipt bound to action scope.** The receipt MUST uniquely identify the specific action authorized. A receipt for Action A cannot authorize Action B, even if the same policy was evaluated.

Non-compliant patterns:

- Recording a receipt at execution time rather than before execution
- Using a previously issued receipt to authorize a new action
- Treating a storage acknowledgment as equivalent to verified persistence
- Allowing execution to proceed when receipt verification is unavailable

Decision Dispositions

Every governance evaluation MUST yield exactly one of the following dispositions. Dispositions MUST be explicitly named - implicit or inferred dispositions are non-compliant.

Disposition	Meaning	Execution Outcome
Approved	Action is authorized as proposed	Execution proceeds
Modified	Action is authorized with a policy-defined transformation	Execution proceeds with transformed payload
Denied	Action is not authorized	Execution does not proceed; Denial Receipt produced
RequiresHumanAuthorization	Action requires explicit human authorization before proceeding	Execution suspended; escalation pathway invoked

Requirements:

- A Modified disposition MUST specify the transformation applied and bind it to the receipt
- A Denied disposition MUST produce a Denial Receipt with equivalent persistence and signing requirements as an approval receipt
- A RequiresHumanAuthorization disposition MUST define an explicit escalation pathway; timeout or failure of that pathway MUST resolve to Denied
- Implementations MUST NOT define additional dispositions that bypass execution gating

Core Primitives

CAES formalizes three foundational objects. All three MUST be present for Level 2 conformance. Level 1 requires the Decision Receipt at minimum.

1. Decision Receipt

A Decision Receipt is a cryptographically signed, pre-execution authorization artifact that proves a specific action was evaluated against a specific policy before execution occurred.

Required properties:

- **Pre-execution:** MUST be produced before the governed action executes. A receipt produced after execution is non-compliant.
- **Cryptographic signature:** MUST be signed using a verifiable asymmetric signature scheme. The signing algorithm MUST be explicitly specified in the Conformance Statement.
- **PolicyHash binding:** MUST include the PolicyHash computed at evaluation time. Execution MUST fail closed if the PolicyHash is absent or cannot be verified.
- **Unique action binding:** MUST uniquely identify the specific action authorized, including sufficient context to prevent replay against a different action or scope.
- **Disposition inclusion:** MUST record the governing disposition (Approved, Modified, Denied, RequiresHumanAuthorization).
- **Timestamp:** MUST record the time of evaluation in a verifiable, non-repudiable form.
- **Write-then-verify persistence:** MUST be written to durable storage and verified via immediate readback before execution proceeds. Acknowledgment alone is insufficient.
- **Append-only ledger entry:** MUST be appended to the Governed Spine as an authoritative event.

Denial Receipts:

A Denial Receipt is a receipt with disposition Denied. Denial Receipts carry the same signing, persistence, and ledger requirements as approval receipts. A Denial Receipt is evidence that the constraint functioned - not an error.

2. PolicyHash

A PolicyHash is a deterministic cryptographic fingerprint of the canonical policy state active at the time of evaluation.

Required properties:

- **Deterministic canonicalization:** MUST use a defined, reproducible canonicalization method before hashing. The same policy inputs MUST always produce the same PolicyHash regardless of key ordering, whitespace, or encoding. The canonicalization method MUST be specified in the Conformance Statement.
- **Collision-resistant hash function:** MUST use SHA-256 or a stronger approved algorithm. The algorithm MUST be specified.
- **Computed at evaluation time:** MUST be computed at the moment of policy evaluation. It MUST NOT be precomputed and reused across evaluations unless the policy state is provably unchanged.
- **Immutable after issuance:** MUST NOT be recomputed or replaced after the receipt is signed. If policy changes between evaluation and execution, the receipt is invalid.
- **Bound to receipt:** MUST be embedded in the Decision Receipt, not referenced by pointer.
- **Offline recomputable:** A verifier with access to the original policy definition MUST be able to independently recompute the PolicyHash and verify match without live system access.

The PolicyHash is the proof that a specific policy version governed the decision. It MUST NOT be a pointer, a label, or a version string alone.

3. Governed Spine

The Governed Spine is an append-only, causally ordered, tenant-scoped record of all governed events.

Required properties:

- **Append-only:** Entries MUST NOT be modified or deleted after creation. Revocation, correction, and tombstoning are explicit append events - not mutations.
- **Causal ordering:** Events MUST be ordered within their partition scope. Ordering MUST be assigned by the platform at ingestion - not by the emitting actor.
- **Tenant scoping:** Every event MUST carry a tenant identifier. Cross-tenant access MUST be structurally impossible, not merely policy-restricted.
- **Fail-closed on append failure:** If a spine append fails, the associated action MUST NOT proceed. No partial state. No silent continuation.
- **Canonical identifiers:** All spine objects MUST carry platform-assigned identifiers. Actor-assigned or self-generated identifiers as primary spine keys are non-compliant.
- **Causal linkage:** Each event MUST carry a reference to its causal parent where one exists. Events that begin a new causal chain MAY carry a null parent reference, but MUST NOT fabricate a causal link.
- **Dead Letter handling:** Events that cannot be successfully appended after bounded retry MUST be routed to a Dead Letter mechanism. DLQ entries are evidence of system boundary encounters and MUST be accessible for audit.

Compliance Levels

CAES defines three conformance tiers. Each tier is a strict superset of the previous.

Conformance claims MUST specify:

- The CAES version
- The claimed level
- A complete Conformance Statement (see Conformance Claims section)

Partial conformance claims are non-compliant.

Level 1 - Receipt-Bounded Execution

Minimum requirement: All effect-boundary actions require verified Decision Receipts. Fail-closed enforcement.

Testable requirements:

Requirement	Test
Decision Receipt produced before execution	Verify receipt timestamp precedes execution timestamp
Receipt persisted with write-then-verify	Demonstrate readback verification step in implementation
Execution blocked if receipt absent	Inject missing receipt; verify execution does not proceed
Execution blocked if receipt unverifiable	Corrupt receipt; verify execution does not proceed
Denial produces Denial Receipt	Trigger denial; verify denial record exists in ledger
Fail-closed on verification failure	Simulate storage failure; verify execution halts
No silent degradation path	Audit all execution paths; confirm no receipt-free path exists

Exclusions from Level 1: PolicyHash canonicalization, cryptographic signing of receipts, spine ordering, and offline verification are not required at Level 1. Implementations SHOULD include them, but they are not Level 1 blockers.

Level 2 - Verifiable Evidence Chain

Minimum requirement: All Level 1 requirements plus cryptographic signing, PolicyHash canonicalization, append-only spine, and offline-verifiable sealed artifacts.

Testable requirements (in addition to Level 1):

Requirement	Test
Receipts signed with named algorithm	Inspect receipt; verify signature and algorithm declaration
PolicyHash present in every receipt	Inspect receipt; verify PolicyHash field present and populated
PolicyHash deterministic	Evaluate same policy twice; verify byte-identical hash
PolicyHash recomputable offline	Provide policy definition to verifier; verify independent hash matches
Spine is append-only	Attempt modification of existing entry; verify rejection
Spine ordering is platform-assigned	Inspect sequence field; verify actor cannot control it
Evidence Pack is sealed and offline-verifiable	Export pack; verify on air-gapped machine with zero network calls
Pack tamper-evident	Modify one byte in sealed pack; verify signature invalidation
Decision dispositions explicitly named	Inspect all governance outcomes; verify each maps to a named disposition
Policy system is CPP-compliant	Verify PolicyHash is canonical, versioned, deterministic, and evaluation output includes matched rules and disposition per CPP spec

Level 3 - Full Constitutional Conformance

Minimum requirement: All Level 1 and Level 2 requirements plus human authority delegation, complete causal invariants, effect classification, chaos attestation, and structured error codes.

Testable requirements (in addition to Level 1 and Level 2):

Requirement	Test
Human authority delegation produces binding artifact	Trigger human authorization; verify DelegationChain artifact in pack
Delegation chain depth enforced	Exceed configured max depth; verify hard rejection with structured error
Delegation chain expiry enforced	Use expired delegation; verify rejection with structured error
Delegation scope enforced	Use delegation for out-of-scope action; verify rejection
Delegation binding target mandatory	Omit target binding; verify rejection with BINDING_MISSING error
Fail-closed under 8 chaos modes	Execute chaos harness; verify each mode produces documented error, no silent pass, no state corruption
Chaos modes covered: DB write failure, key rotation conflict, clock skew, network partition during pack export, disk full during pack write, duplicate receipt injection, PolicyHash mismatch, delegation chain expiry	Each mode independently tested
Evidence Pack contains PolicyHash artifact	Inspect exported pack; verify PolicyHash manifest present
Evidence Pack contains DelegationChain artifact	Inspect exported pack for delegated actions; verify artifact present
Evidence Pack contains ChaosTestAttestation	Inspect exported pack; verify attestation covering all 8 chaos modes
Cross-pack provenance chain	Link packs across operations; verify broken link produces structured error
All public-surface failures produce structured error codes	Enumerate failure paths; verify 100% emit typed error codes, no raw exceptions
Evidence Pack export is deterministic	Export same scope twice; verify byte-identical output
Offline verification requires zero network calls	Air-gap verify; confirm zero outbound packets
Effect Boundary categories declared	Inspect Conformance Statement; verify all governed categories listed

Level 3 structured error code requirement:

Implementations MUST define a typed error code namespace. Every failure mode reachable from the public surface MUST map to a named code in that namespace. Raw exceptions or untyped error strings surfacing to callers are non-compliant.

Non-Compliant Patterns (Explicitly Disallowed)

CAES prohibits the following patterns. An implementation exhibiting any of these patterns MUST NOT claim CAES conformance at any level.

Authorization Failures

- Prompt-only "policy" enforcement without deterministic evaluation
- Post-hoc logging presented as authorization proof
- Non-deterministic model-only policy evaluation without deterministic verification layer
- Receipts produced after execution rather than before
- Reuse of a prior receipt to authorize a new action
- Treating storage acknowledgment as equivalent to verified persistence

Evidence Integrity Failures

- Mutable logs masquerading as evidence ledgers
- PolicyHash implemented as a version label, pointer, or name rather than a cryptographic hash of canonical policy inputs

- PolicyHash recomputed at execution time rather than bound at evaluation time
- Evidence artifacts that require live system access for verification

Enforcement Failures

- Silent degradation under failure conditions
- Fallback to permissive execution when receipt verification is unavailable
- Implicit authorization paths not gated by receipt verification
- Invocation-layer governance bypass (governance that applies only to some invocation surfaces)
- Governance that applies only when explicitly enabled

Behavioral and Output Failures

- Post-hoc content filtering presented as governance of behavioral outputs
- Prompt-only behavioral constraints without deterministic evaluation
- Silent modification of outputs without audit trace
- Emission of behavior when policy evaluation fails or is unavailable

Recording and Privacy Failures

- Recording without verifiable authorization
- Passive or continuous recording without policy evaluation
- Use of recorded data outside authorized purpose
- Silent collection of personal or sensitive data
- Deletion without audit trace or verification

Behavioral Governance

Overview

CAES applies not only to traditional execution actions (e.g., transactions, system mutations), but also to **effect-bearing behaviors** produced by AI systems.

An AI system may produce consequential outcomes through:

- language (responses, instructions, recommendations)
- interaction (user guidance, persuasion, escalation)
- physical actuation (robotics, embodied systems)
- social or psychological influence

These outputs may cross an Effect Boundary even when no explicit system mutation occurs.

CAES therefore requires governance of **behavioral outputs**, not just execution calls.

Behavioral Effect Boundary

A **Behavioral Effect Boundary** exists when an AI-generated output can:

- influence a user decision with material consequence
- produce legal, financial, regulatory, or safety impact
- trigger downstream actions by humans or systems
- create risk through communication, instruction, or interaction

Systems MUST treat such outputs as effect-bound.

Policy Requirements for Behavioral Governance

Policies MUST define constraints on:

- **Allowed content** - what the system may say or communicate
- **Disallowed content** - prohibited instructions, claims, or statements
- **Interaction boundaries** - escalation requirements, refusal conditions, handoff to human authority
- **Domain-specific restrictions** - regulated domains (legal, medical, financial), sensitive interactions (minors, vulnerable populations)
- **Physical interaction constraints** (if applicable) - permitted contact or actuation, safety envelopes and exclusion zones

Policies MAY include contextual or conditional rules, but MUST remain deterministically evaluable at the point of decision.

Behavioral Decision Requirement

For any behavior crossing a Behavioral Effect Boundary, a **Decision Receipt MUST be produced prior to output emission**.

The receipt MUST:

- bind the behavior to the governing policy state (PolicyHash)
- capture the evaluated decision disposition
- include sufficient context to support audit and reconstruction

Behavioral Enforcement

Systems MUST enforce behavioral policies:

- **before output is delivered**
- **without relying on post-hoc filtering alone**

Post-generation filtering MAY be used as a secondary control, but MUST NOT replace pre-decision authorization.

If policy evaluation fails or cannot be completed:

- the system MUST fail closed
- the behavior MUST NOT be emitted

Behavioral Dispositions

Behavioral governance decisions MUST result in one of the four defined Decision Dispositions:

- **Approved** - output may be emitted as generated
- **Modified** - output must be transformed to comply with policy before emission
- **Denied** - output must not be emitted
- **RequiresHumanAuthorization** - escalation required before emission

All dispositions MUST be recorded and auditable.

Causal Recording

Behavioral decisions MUST be recorded in the Governed Spine, including:

- the evaluated behavior or output reference
- the Decision Receipt reference
- the resulting disposition
- any transformation or escalation applied

Behavioral events are first-class governed events.

Privacy, Recording, and Data Stewardship

Overview

AI systems may possess capabilities to:

- record audio, video, or environmental data
- capture user-generated content
- observe or infer personal or sensitive information

These capabilities introduce legal, privacy, and ownership obligations equivalent to or exceeding those applied to human actors.

CAES requires that such capabilities be governed as **effect-bearing operations**.

Sensory Effect Boundary

A **Sensory Effect Boundary** exists when an AI system can:

- record audio, video, or other environmental signals
- capture or store user content
- observe or infer personally identifiable or sensitive information

Crossing this boundary MUST be treated as a governed action subject to all Decision Receipt, PolicyHash, and Governed Spine requirements.

Consent and Authorization Requirements

AI systems MUST NOT:

- record audio, video, or environmental data without appropriate authorization
- capture or store personal content without a permitted legal basis
- operate recording capabilities in restricted or prohibited contexts

Authorization MAY be derived from:

- explicit user consent
- contractual agreement
- legal or regulatory allowance

The system MUST be able to:

- determine whether recording is permitted in context
- deny or disable recording if authorization cannot be verified

Failure to verify authorization MUST result in fail-closed behavior.

Use and Purpose Limitation

Captured data MUST:

- be used only for the purposes for which authorization was granted
- NOT be repurposed without reauthorization
- NOT be accessed outside defined policy scope

Policies MUST define:

- allowed use cases
- prohibited use cases
- domain-specific restrictions

Retention and Storage Governance

AI systems MUST enforce:

- defined retention policies for all captured data
- automatic expiration or deletion based on policy rules
- restrictions on duplication, transfer, or export

Retention policies MUST be:

- explicitly defined
- deterministically enforceable
- auditable

Deletion and Proof of Deletion

Data deletion MUST be treated as a **governed action**.

When data is deleted:

- the deletion MUST be executed through the governed execution pathway
- the deletion MUST produce a Decision Receipt representing the action
- the receipt MUST be recorded in the Governed Spine
- the receipt MUST be retained even after the data itself is deleted

The deletion record is permanent. The data is not.

Recording and Data Handling Dispositions

Decisions related to recording and data handling MUST result in one of the four defined Decision Dispositions:

- **Approved** - recording or usage permitted
- **Denied** - recording or usage prohibited
- **Modified** - restricted or redacted capture authorized
- **RequiresHumanAuthorization** - escalation required before capture or use

All decisions MUST be auditable and traceable.

Enforcement Requirements

Privacy and recording policies MUST be enforced:

- prior to data capture
- prior to data usage
- prior to data retention or transfer

Post-hoc enforcement alone is non-compliant.

Principle

AI systems MUST behave as legally accountable actors with respect to privacy, consent, data ownership, and content stewardship.

> ***Capabilities that observe, record, or retain data carry the same governance obligations as actions that execute.***

Conformance Claims

Implementations claiming CAES compliance MUST publish a complete **Conformance Statement** that includes all of the following:

Field	Requirement
CAES version	Exact version string (e.g., v0.2.0)
Claimed compliance level	Level 1, 2, or 3
Receipt signing algorithm	Named algorithm (e.g., Ed25519)
PolicyHash canonicalization method	Named method (e.g., JCS/RFC 8785 + SHA-256)
Fail-closed enforcement description	How the system behaves under each failure mode
Effect Boundary categories governed	Complete list of governed categories
Write-then-verify mechanism	Description of receipt persistence and verification procedure
Evidence Pack format (Level 2+)	Format and verification method
Human delegation model (Level 3)	Delegation artifact structure and enforcement
Chaos attestation coverage (Level 3)	Which chaos modes are attested
Error code namespace (Level 3)	Reference to typed error code registry

Unversioned or incomplete Conformance Statements are non-conformant.

Conformance Statements MAY be self-attested. Third-party audit is RECOMMENDED for Level 3 claims.

Reference Implementation

The CAES Working Group designates one reference implementation per version.

CAES v0.2.0 Reference Implementation: Keon Systems - Governed Execution https://keon.systems

Reference designation does not imply commercial endorsement. Other implementations may independently conform.

Working Group

CAES is maintained by the CAES Working Group (CAES-WG).

The Working Group publishes:

- Normative specification versions

- Amendment proposals
- Conformance guidelines
- Mark usage policy

Specification changes are versioned. There are no informal amendments.

Amendment Process

To propose a change:

31. Identify the normative requirement affected.
32. Describe the structural issue.
33. Document backward compatibility impact.
34. Submit as a versioned proposal.

Changes are incorporated only via published version increment.

Mark Usage

"CAES" and associated marks are used to designate conformance with a specific published version.

Use of the CAES mark requires a complete Conformance Statement.

Mark usage guidelines are provided in /assets.

License

The CAES specification text is published under a permissive documentation license to encourage adoption and independent implementation.

See LICENSE.md for details.

Status

CAES v0.2.0 is a public draft released for review and comment.

This version supersedes v0.1.1 and introduces normative strengthening of Core Primitives, Compliance Levels, Fail-Closed Semantics, Authorization-Execution Separation, Decision Dispositions, Effect Boundary Definition, Behavioral Governance, and Privacy and Recording governance requirements.

Implementations conforming to v0.1.1 SHOULD review this version for new testable requirements before claiming v0.2.0 conformance.

Feedback and formal amendment proposals are welcome.

CAES Constitutional AI Execution Standard March 2026