Nick Clark · Published April 25, 2026 · PDF

How to Keep an AI Agent's Decisions Consistent With Its Own Past Behavior

If your agent behaves well in one session and contradicts itself the next, per-request checks are not enough: you need to gate each decision against the agent's accumulated behavioral record, not just the current input. This guide describes an architecture for doing that, disclosed in United States Patent Application 19/647,395, built around what the filing calls the Integrity and Coherence inventive step. It is an architecture you implement yourself, not a shipping library.

What You Are Building

You are building an agent that stays consistent with itself over time. Not consistent with a single prompt, not consistent with a static rulebook, but consistent with the actual pattern of choices it has already made and recorded. When a decision comes up, the agent should be able to ask: does committing to this action fit the trajectory of how I have behaved, or does it break from it? And it should be able to answer that question from its own durable record rather than from the mood of the current request.

This is the problem developers hit once an agent runs long enough to accumulate a history. A coding agent that promised thoroughness in the morning quietly ships a stubbed-out function in the afternoon. A support agent that guarded a confidentiality scope in one delegation leaks adjacent context in the next. Each individual step looks

locally reasonable, so per-call guardrails wave it through, and the incoherence only becomes visible in aggregate. The goal here is to make that aggregate visible to the agent at decision time and to let it gate on that.

The approach described below is disclosed in United States Patent Application 19/647,395, in the material the filing groups under the Integrity and Coherence inventive step. It is an architecture, not a package you can install.

Why the Obvious Approaches Fall Short

The common approaches are legitimate and worth understanding on their own terms; the point is where each one structurally stops.

A **system prompt or constitution** states the values but does not measure adherence. It tells the agent what to be, and it re-states that on every turn, but it holds no record of whether yesterday's outputs actually honored it. Nothing accumulates.

A **per-request classifier or guard model** scores the current input or output in isolation. This is genuinely useful for catching individual bad actions, but by construction it sees one decision at a time. It cannot notice that this decision is the fifth small compromise in a widening pattern, because each call starts fresh.

Cryptographic provenance and signature checks (the kind of lineage validation used to verify an agent is who it claims to be) confirm that a history is authentic, continuous, and untampered. That is necessary but not sufficient here: a lineage can be perfectly signed and continuous while the behavior it records drifts steadily away from the agent's declared norms. Authenticity of the record is a different question from coherence of the conduct inside it.

The structural gap common to all three is that none of them carries an accumulated, first-class measure of normative consistency that the agent evaluates its next decision against. The filing's contribution is to make that accumulated measure a component of

the agent's own state and to feed it into the decision, so that gating happens against the established pattern rather than only against the current signature.

The Architecture

The core construct in Application 19/647,395 is an **integrity field**: a component of the agent's operational state that encodes, as a continuous gradient rather than a binary label, the alignment between the agent's declared operational values and its actual behavioral record as preserved in its lineage. It captures magnitude, direction, and rate of change of that alignment. It is self-referential: the agent computes it from its own actions, its own declared values, and its own policy constraints, as a first-class cognitive operation performed by an integrity engine. External systems may audit it, but they do not compute it for the agent.

Three independent domains. The filing structures integrity along three separately tracked axes. *Personal* integrity is self-referential alignment: does the agent's behavior match its own declared values. *Interpersonal* integrity is relational consistency: does the agent honor commitments made in delegations and interactions, including confidentiality scopes. *Global* integrity is alignment with broader systemic and societal norms, evaluated against system-level policy and projected downstream consequences. Each domain keeps its own current score, its own trajectory, its own baseline, and its own policy-defined bounds. They are independent by design: an agent can be true to itself yet unreliable relationally, or beneficial to itself yet harmful to the wider system. For gating decisions the three are combined into a **composite integrity score** using domain weights specified by policy, and the weighting is deterministic and policy-specified, not something the agent negotiates for itself.

The integrity trajectory. The load-bearing idea for consistency-over-time is the *trajectory*: the accumulated pattern of normative consistency recorded in the lineage. The lineage records each action; the integrity engine evaluates that action against declared values; the evaluation is written back into the lineage; and the accumulated

sequence of those evaluations constitutes the agent's integrity trajectory. This is the object you gate against. It is what lets the filing describe an agent evaluating even a governance claim against "its own integrity trajectory, the accumulated pattern of normative consistency recorded in its lineage," rather than against the current signal alone.

A prospective deviation function. Consistency is enforced before commitment, not audited after. The filing defines a deterministic deviation-likelihood metric, D = (N - T) / (E x S), evaluated at each decision point. N is the agent's current need (unmet-requirement urgency), T is the ethical threshold below which deviation is not structurally available, E is empathy weighting (how much projected harm to others the agent internalizes), and E times S, where S is a self-esteem score capturing self-assessed alignment, forms the deviation *resistance* in the denominator. Deviation pressure exists only when N exceeds T. The threshold T itself carries a *historical adjustment* reflecting the agent's recent deviation history, which is one concrete way past behavior conditions the present decision. The function produces a continuous output evaluated continuously as part of the cognitive cycle, so gradual accumulation of pressure is detected rather than missed.

Gating and cross-primitive consequence. A proposed mutation to the agent's state is evaluated against the integrity model before commitment, and its projected impact on each domain is computed. If committing would drop the composite score below a policy-defined threshold, the governance gate (the composite admissibility evaluation integrating integrity, confidence, affect, capability, and personality signals) receives that impact assessment as input. The consequence is structural: a degraded integrity score lowers the agent's computed confidence, and if that falls below the execution threshold the agent transitions to a non-executing cognitive mode where it still forecasts and plans but does not commit actions until integrity is restored.

Trajectory validation and forecasting. Two further mechanisms operate on the trajectory itself. *Integrity-aware trust slope validation* adds trajectory continuity as a validation criterion on top of standard authenticity checks, flagging anomalies such as a reported high-integrity state contradicted by numerous unaccounted deviation events, disappearing or systematically downgraded deviation entries, or self-esteem that stays flat despite a pattern of deviations. It emits an integrity trust score that supplements the standard trust score. *Moral trajectory forecasting* projects the trajectory forward and classifies it into archetypes (redemption, stabilization, radicalization, containment), surfacing recommended interventions to governance rather than enacting them autonomously.

How to Approach the Build

The following is an ordered path to implement the architecture yourself. The interface sketch below is illustrative only and faithful to the filing; it is not a library.

1. **Make declared values a structured artifact.** Personal integrity is defined as the comparison of behavior against a declared value set held in the policy reference field. If your values live only in a prose system prompt, you have nothing to compute against. Represent them as data the engine can evaluate.
2. **Give the agent a durable, append-only lineage.** The trajectory is derived from lineage entries, so every action, delegation, and governance decision needs a recorded entry, and evaluations must be written back. Treat it as append-only: the filing notes the agent cannot silently omit or retroactively alter integrity events without producing a detectable discontinuity.
3. **Build the integrity engine as three domain evaluators plus a policy-weighted combiner.** Score personal, interpersonal, and global separately, each carrying its own current value and trajectory, then combine with policy-specified weights only where a single composite is needed (gating, trust-slope validation, confidence).

```
# illustrative, per US 19/647,395 — not a shipping API
personal      = score_against(declared_values, lineage)
interpersonal = score_against(relational_commitments, lineage)
global        = score_against(systemic_norms, projected_consequences)
composite     = policy_weights.combine(personal, interpersonal, global)
```

4. **Compute deviation prospectively at each decision point.** Evaluate D = (N - T) / (E x S) before committing a mutation, and fold the recent-history adjustment into T so past behavior conditions the present threshold. Evaluate it continuously, not as a periodic audit, so slow drift is caught.
5. **Wire integrity into the gate, not just into a log.** A finding that does not change what the agent is allowed to do is only observability. Feed the composite score and the projected per-domain impact into the admissibility gate, and couple a degraded score to reduced confidence and the execution pause described above.
6. **Add trajectory validation and forecasting once the loop is stable.** Layer trust-slope trajectory-continuity checks to catch record manipulation and decoupling, and forecasting to classify the arc and recommend interventions to your governance layer.

What This Does Not Give You

This is an architecture, not a drop-in library. There is no package to install and nothing here "just works" out of the box; every component above is something you design and build for your own agent, data model, and policy.

It is disclosed in a patent filing, not benchmarked or shipped as a product. The filing does not state accuracy figures, latency, or production results, and neither does this guide.

Its quality is bounded by your inputs. Integrity is computed against *declared* values and a *recorded* lineage; if your value set is vague or your lineage is incomplete, the trajectory it produces will be too. It measures consistency with the agent's own declared norms, which is a different thing from those norms being correct: an agent can be perfectly coherent with a bad value set. The domain weights are policy decisions you must make deliberately, since they determine which kind of inconsistency the gate actually catches. And the approach targets agents that accumulate a history; for a stateless single-shot call with no durable record, there is no trajectory to gate against and this architecture does not apply.

Disclosure Scope

The architecture described in this guide is disclosed in United States Patent Application 19/647,395. The mechanisms attributed here to that filing (the integrity field, the three-domain model, the integrity trajectory, the deviation function, integrity-aware trust slope validation, and moral trajectory forecasting) are described in that application, and this guide is an educational explanation of that disclosed approach. It is not a warranty, a specification, or an offer of software, and nothing here should be read as a claim that a benchmarked or production implementation is being provided. You implement any such system yourself.

Integrity & Coherence (/integrity-coherence) All 40 steps → (/inventive-steps)

Track normative consistency. Detect deviation. Self-correct.

Chapter 3 (/patents/19-647395/chapters/integrity)

PRIMARY TECHNICAL DISCLOSURE

– The Coherence Trifecta: Empathy, Integrity, and Self-Esteem as a Unified Control Loop (/articles/the-coherence-trifecta-empathy-self-esteem-and-integrity-as-a-unified-control-loop)

SECONDARY TECHNICAL

- Three-Domain Integrity Model (/articles/integrity-coherence/three-domain-model)
- Deviation Function D=(N-T)/(ExS) (/articles/integrity-coherence/deviation-function)
- Self-Esteem as Internal Validator (/articles/integrity-coherence/self-esteem-validator)
- Deviation as Deterministic Semantic Mutation (/articles/integrity-coherence/deviation-mutation)
- Integrity Structural Placement (/articles/integrity-coherence/structural-placement)
- Empathy as Distributed Moral Load (/articles/integrity-coherence/empathy-mechanism)
- Coherence Trifecta Control Loop (/articles/integrity-coherence/coherence-trifecta)
- Coping Intercept Patterns (/articles/integrity-coherence/coping-intercepts)
- Integrity Deviation Logging (/articles/integrity-coherence/deviation-logging)
- Integrity Collapse Detection (/articles/integrity-coherence/collapse-detection)
- Redemption Engine (/articles/integrity-coherence/redemption-engine)
- Moral Trajectory Forecasting (/articles/integrity-coherence/moral-trajectory)
- Integrity-Aware Trust Slope Validation (/articles/integrity-coherence/trust-slope-integrity)
- Integrity-Confidence Cross-Primitive Coupling (/articles/integrity-coherence/confidence-coupling)
- Integrity-Modulated Discovery Traversal (/articles/integrity-coherence/discovery-integrity)
- Integrity-Aware Multi-Agent Negotiation (/articles/integrity-coherence/multi-agent-negotiation)
- Biological Signal Coupling for Integrity (/articles/integrity-coherence/biological-integrity)
- Policy-Based Integrity Constraints (/articles/integrity-coherence/policy-constraints)
- Integrity Field Portability (/articles/integrity-coherence/field-portability)
- Predictive Deviation Alerting (/articles/integrity-coherence/predictive-alerting)
- Governed Forgetting (/articles/integrity-coherence/governed-forgetting)
- Predictive Social Modeling (/articles/integrity-coherence/predictive-social-modeling)
- Refusal as First-Class Observation (/articles/integrity-coherence/refusal-as-observation)
- Historical Policy-Version Reconstruction (/articles/integrity-coherence/historical-policy-version-reconstruction)

APPLICATIONS · GENERAL

- Autonomous Vehicle Ethical Decision-Making Through Computable Integrity (/articles/integrity-coherence/autonomous-vehicle-ethics)
- Detecting Strategy Drift in Algorithmic Trading Agents With Computable Integrity (/articles/integrity-coherence/trading-normative-consistency)

- How to Keep a Legal AI Agent's Advice Consistent With Precedent and Its Own Prior Positions (/articles/integrity-coherence/legal-advisory-agents)
- Government AI Policy Agents: Consistency, Equity, and Statutory Alignment by Design (/articles/integrity-coherence/government-policy-agents)
- AI Editorial Agents for Newsrooms: Consistent Standards and Bias-Drift Detection by Design (/articles/integrity-coherence/journalism-editorial-agents)
- Integrity and Coherence for AI Environmental Compliance Agents: Consistent Regulatory Interpretation Across Facilities and Jurisdictions (/articles/integrity-coherence/environmental-compliance)
- Integrity and Coherence for Insurance Underwriting Agents (/articles/integrity-coherence/insurance-underwriting)
- Integrity and Coherence for Social Media Moderation Agents (/articles/integrity-coherence/social-media-moderation)
- Legal-Evidence Reconstruction for Autonomous-Incident Litigation: Court-Admissible Policy-Version Lineage (/articles/integrity-coherence/legal-evidence-reconstruction)
- Regulatory Audit Replay With Historical Policy Versions (/articles/integrity-coherence/regulatory-audit-replay)

APPLICATIONS · SPECIFIC

- Waymo vs a Governed Integrity Layer for Autonomous Behavior (/articles/integrity-coherence/waymo)
- Cruise vs Governed Autonomy: Why AV Safety Needs a Computable Integrity Field (/articles/integrity-coherence/cruise)
- JPMorgan Trading Compliance vs Governed Agents: The Integrity Field Gap (/articles/integrity-coherence/jpmorgan)
- Palantir Governance Alternative: Adding a Computable Integrity Field to Government Analytics (/articles/integrity-coherence/palantir)
- Does the Aurora Driver maintain normative memory across decisions? (/articles/integrity-coherence/aurora-innovation)
- Nuro Alternative: Governed Autonomous Delivery Beyond Per-Trip Safety Records (/articles/integrity-coherence/nuro)
- Zoox vs Governed Autonomy: Tracking Normative Drift the Planner Cannot See (/articles/integrity-coherence/zoox)
- Motional Alternative for Governed Normative Trajectory in Autonomous Driving (/articles/integrity-coherence/motional)
- Argo AI Legacy vs Governed Autonomy: The Missing Deviation Function (/articles/integrity-coherence/argo-ai-legacy)

- comma.ai openpilot and the Governed-Behavior Layer: Integrity Coherence for Learning-Based Driving (/articles/integrity-coherence/comma-ai)
- Apache Iceberg Time Travel vs Governed Replay: Data Without Policy (/articles/integrity-coherence/apache-iceberg-time-travel)

Integrity & Coherence overview → (/integrity-coherence)