MOIAS Methodology

`v2.0.0`

Author: John J. McCormick · Updated: February 28, 2026 · CC BY 4.0 · aiagentgovernance.org

SUGGESTED CITATION

McCormick, J. J. "The MOIAS Methodology: A Phase-Gated Governance Lifecycle for AI Agents." v2.0.0, February 2026. aiagentgovernance.org. https://aiagentgovernance.org/framework/governance-lifecycle/

> ***Authorship context:*** *This is a practitioner's methodology, not an academic paper. The author built and operated the governed agent system described here — writing governance controls in production as agent failures occurred, documenting incidents in real time, and extracting the methodology from operational experience. The theoretical frameworks cited throughout (Dekker, Reason, Vaughan, Boyd, Weick & Sutcliffe) were identified after the governance mechanisms were built; they provide vocabulary for patterns that production operations had already discovered empirically.*

Abstract

The Methodology for Oversight of Intelligent Agent Systems (MOIAS) is an operational governance framework for supervising autonomous AI agents performing real work under delegated authority. Unlike compliance frameworks that assess AI models at a point in time, MOIAS governs the full lifecycle of agent work: from assignment through execution, review, remediation, and approval. The methodology was derived from continuous production operations in which autonomous agents performed real software engineering work under governance controls. Every governance mechanism was created in response to a documented failure. MOIAS addresses a structural gap in the current AI governance landscape: the space between model-level compliance and operational agent oversight.

Origin

This methodology emerged during the development and operation of a governed autonomous software production system designed to build regulated enterprise applications. Governance controls were introduced in response to observed operational failures within that system. The methodology documents those controls and their evolution.

MOIAS defines governance principles, lifecycle structure, and authority models. It does not prescribe implementation details or enforcement mechanisms.

---

1. The Problem MOIAS Addresses

Organizations deploying AI agents at scale face a governance gap that existing frameworks do not close:

- **Observability** shows what agents did. It does not determine whether agents should have done it.
- **Security** blocks what agents should not do. It does not ensure agents do the right work in the right way.
- **Compliance** checks that regulatory requirements are met at the model level. It does not govern the operational lifecycle of agent work.

MOIAS fills the space between these layers. It provides the operational discipline for ensuring that autonomous agents, performing real work under delegated authority, produce reliable outcomes with structured human oversight.

The need for this discipline is structural, not speculative. Research from production multi-agent operations has documented recurring behavioral failure modes that are invisible to observability tools, unaddressed by security policies, and outside the scope of compliance dashboards. These failures occur not because agents are defective, but because capable autonomous actors optimizing under pressure will drift from their directives. This is a property of agency itself, well-documented in safety engineering literature (Dekker, 2011; Vaughan, 1996).

---

2. Core Thesis

> *Autonomous AI agents operating under delegated authority will drift from their directives. This is not a bug — it is a behavioral tendency observed in governed operational contexts, consistent with drift dynamics documented in safety engineering literature (Dekker, 2011; Vaughan, 1996). The question is never whether drift occurs, but when, how fast, and whether you detect it before impact.*

Every mechanism in MOIAS exists to either prevent drift, detect drift, or contain the blast radius of drift that was not caught in time. The methodology is empirical: every governance mechanism was created in response to a real, documented failure. Mechanisms that cannot be traced to a specific incident or measurable drift pattern are questioned — governance complexity without safety value is itself a risk.

---

3. Theoretical Foundations

MOIAS draws on established frameworks from safety engineering, military operations, and organizational theory. These frameworks were identified after the governance mechanisms were built — they provide vocabulary for patterns that production operations had already discovered empirically.

3.1 Separation of Duty and Two-Person Integrity

Origin: Separation of Duty (SoD) — NIST SP 800-53 AC-5, financial controls literature. Two-Person Integrity (TPI) — U.S. Department of Defense, formalized 1962 for nuclear weapons operations.

Principle: No single actor may both propose and authorize a critical action. SoD enforces this through role separation in access control and authorization processes. TPI enforces a stricter variant: two authorized individuals must be continuously co-present throughout execution, each holding only half the authorization. The MOIAS lifecycle implements SoD; TPI provides the historical reference demonstrating that this separation-of-authority principle has 60 years of operational validation in the most demanding safety-critical context known.

Application to agent governance: The MOIAS lifecycle enforces separation of duty: no single actor can both propose and authorize a critical action. An agent that authors a work plan and then executes that plan is a single actor holding both keys. Phase gates enforce this separation: the agent proposes, a separate actor (human or validator) approves, and only then does execution proceed. This adapts TPI's separation-of-authority property without implementing its concurrent-presence or symmetric-incapacity requirements (where neither actor alone can complete the action, enforced by physical interlock design).

3.2 Reason's Swiss Cheese Model

Origin: James Reason, 1990. *Human Error*. Cambridge University Press.

Principle: Every defense layer has holes. Failures occur when the holes in multiple layers align. Safety comes from stacking layers with different failure modes, not from making any single layer perfect.

Application to agent governance: No single governance mechanism is sufficient. Agents can comply with every directive while violating the spirit of their plan. Operators can approve work correctly while missing prerequisite gaps. Each layer has a characteristic failure mode — safety comes from ensuring the layers fail independently. Note: in single-operator deployments, this independence is partial — a single human at every gate means that one cognitive failure mode can compromise multiple layers. Full independence requires distributed oversight, addressed at maturity Level 2+ through independent validation functions.

3.3 Dekker's Drift Into Failure

Origin: Sidney Dekker, 2011. *Drift into Failure*. Ashgate. Building on Rasmussen (1997) and Vaughan (1996).

Principle: Systems do not fail suddenly. They drift gradually as small deviations from expected behavior are normalized. Each deviation is individually rational — it saves time, reduces friction, and nothing bad happens. The deviations accumulate until the system crosses a safety boundary that no one noticed approaching.

Application to agent governance: This is the most dangerous failure mode for AI agents because it is invisible during the drift period. Every individual decision looks correct. The pattern is only visible in retrospect, across incidents. Behavioral pattern detection and incident-driven learning exist specifically to make drift visible before it normalizes.

3.4 Boyd's OODA Loop

Origin: Colonel John Boyd, USAF, 1970s. Developed from fighter combat analysis.

Principle: The decision cycle is Observe, Orient, Decide, Act. The critical phase is **Orient** — the mental model through which observations are interpreted. Two actors observing the same data will reach different decisions if their orientation differs.

Application to agent governance: Different actors oriented toward different questions catch different failures. A planning agent oriented toward throughput will miss compliance gaps. An operator oriented toward work quality will miss plan prerequisite gaps. Governance systems need actors with diverse orientations, not redundant actors with the same orientation.

3.5 Normalization of Deviance

Origin: Diane Vaughan, 1996. *The Challenger Launch Decision*. University of Chicago Press.

Principle: Deviant behavior becomes normalized through repeated occurrence without negative consequences. The boundary of acceptable behavior shifts incrementally until actions that would have been rejected initially become routine.

Application to agent governance: Trust calibration prevents the normalization of agent behavioral deviations over time. Every deviation is documented, even when the outcome is good. “It worked” is not evidence of correctness — it may be evidence of luck.

3.6 High-Reliability Organizations

Origin: Weick, K.E. & Sutcliffe, K.M., 2007. *Managing the Unexpected*. Jossey-Bass.

Principle: High-reliability organizations maintain safety through preoccupation with failure, reluctance to simplify, sensitivity to operations, commitment to resilience, and deference to expertise.

Application to agent governance: The governance system institutionalizes these properties: incident logs maintain preoccupation with failure, multi-layered governance resists simplification, behavioral monitoring provides sensitivity to operations, remediation cycles build resilience, and human operators provide expert judgment at decision points.

---

4. The Phase-Gated Lifecycle

The MOIAS lifecycle governs agent work through a sequence of phases, each separated by a gate that requires explicit approval before the agent may proceed. Gates are structural controls designed to prevent the executing agent from advancing its own work unilaterally.

4.1 Lifecycle Phases

```
PLAN → AUTHORIZE → EXECUTE → REVIEW → APPROVE → AUDIT
```

Phase	Purpose	Gate Requirement
PLAN	Define work scope, acceptance criteria, and execution approach	Human approval of scope and approach
AUTHORIZE	Verify that plan requirements are met and execution is permitted	Authorization from designated authority
EXECUTE	Perform the defined work within the approved scope	Completion signal with evidence of work performed
REVIEW	Independent assessment of work quality and correctness	Structured findings report with severity classification
APPROVE	Human confirmation that the work meets acceptance criteria	Human sign-off
AUDIT	Verify governance lifecycle compliance and record governance decisions	Complete audit trail of lifecycle execution

Specific implementations may define additional phases (e.g., separate engineering and architecture review phases, remediation cycles between review and approval). The lifecycle is iterative — work may cycle through Review and Execute multiple times before reaching Approve.

4.2 Gate Properties

Every phase transition passes through a gate with the following properties:

- **Explicit approval required.** No agent may advance its own work to the next phase. A separate actor (human operator or designated reviewer) must approve the transition.
- **Evidence-based.** Gate approval requires documented evidence that the current phase’s requirements are met. Claims of completion without verification evidence are a documented failure pattern (see the Behavioral Pattern Taxonomy (patterns.md)).
- **Reversible.** If a gate check reveals deficiencies, work returns to the appropriate phase for remediation. Governance lifecycles are iterative, not linear.
- **Auditable.** Every gate decision is logged with: who approved, when, what evidence was presented, and what the approval authorizes.

4.3 The Operator-as-Router Architecture

In MOIAS, a human operator sits between every phase transition. The operator is not a bottleneck — the operator is a control point.

The operator-as-router architecture provides:

- **Context preservation.** The operator carries context across agent sessions that would otherwise be lost.
- **Priority management.** The operator sequences work across multiple agents and workstreams.
- **Quality gate.** The operator reviews agent output before routing to the next phase.
- **Audit trail.** Every routing decision is a human action, traceable through the governance record.

This is a deliberate architectural choice. Full agent autonomy and full manual control are both failure modes. The MOIAS lifecycle provides calibrated delegation: agents execute within scoped authority boundaries, and humans approve phase transitions.

4.4 Impact Statements

Every phase-transition request must include a structured **Impact Statement** that provides the approving human with a decision surface:

```
Risk:          LOW / MEDIUM / HIGH / CRITICAL
Action:        What this approval authorizes
If approved:   What happens next
If rejected:   What happens instead
Reversible:    Yes / No / Partial
Blast radius: Which systems or users are affected
```

Risk levels are determined by reversibility and blast radius, not by technical complexity:

Risk Level	Condition
LOW	Review-only gate, or all prior gates already passed
MEDIUM	First review of new work, or partial prior passes
HIGH	Production deployment, data migration, or schema change
CRITICAL	Irreversible action, multi-tenant impact, or security boundary change

Impact Statements exist because human approvers face cognitive overload when reviewing dense technical content under time pressure. The Impact Statement provides a structured signal for go/no-go decisions without requiring the approver to parse every technical detail.

5. Governance Directives

Governance directives are the enforceable rules that agents must follow within the MOIAS lifecycle. Each directive was created in response to a specific, documented failure. Directives that cannot be traced to a real incident are questioned.

5.1 No Single Actor Holds Both Keys

An agent that writes a plan and then executes that plan without independent verification is a single actor holding both authorization keys. Phase gates enforce separation: the proposing agent and the approving actor are always distinct.

Derived from: Two-Person Integrity (TPI). Motivated by documented incidents where agents bypassed their own governance gates.

5.2 Role Boundary Enforcement

Agents must not perform work outside their assigned role scope. An implementation agent does not perform its own review. A planning agent does not execute its own plans. When something outside an agent’s scope needs attention, it is flagged to the operator — not performed unilaterally.

Motivated by a documented incident where an agent, upon receiving ambiguous instructions, began performing both implementation and review — roles assigned to different agents.

5.3 Instruction Mode Must Match Operation Type

Different types of operations require different instruction specificity:

Operation Type	Instruction Mode
Feature development	Descriptive (intent-based)
Infrastructure/schema changes	Prescriptive (exact specification)
Security operations	Prescriptive with verification gates
Review assignments	Structured (findings table, evidence, gate criteria)

Giving autonomous agents intent-based instructions for infrastructure operations produces improvisation and drift. Prescriptive instructions produce repeatable outcomes.

Motivated by a documented incident where an agent received a descriptive instruction for a schema change and applied invented values instead of the specified ones.

5.4 Every Deviation Is Documented

Every deviation from expected behavior must be documented in the incident log, even when the outcome is good. “It worked” is not evidence of correctness. The incident log is the design document for governance improvements.

Derived from: Dekker’s drift theory. The antidote to drift is structural commitment to never normalizing deviation.

5.5 Silent Failures Compound Undetected

Every automated operation requires a verification assertion. “Did this produce the expected result?” must be answered, not assumed. Multiple documented incidents involved systems that appeared functional while fundamentally misconfigured.

Motivated by incidents involving systems declared operational without adoption verification, cost tracking producing incorrect values silently, and automation executing unauthorized work.

5.6 Commit Hygiene

Deliverable work should be presented as logical, reviewable units. Micro-commits, auto-generated noise, and “saved progress” entries obscure the actual changes and make review more difficult. Each commit in a deliverable range should represent a meaningful unit of work.

5.7 Proactive Shared Change Management

When a governance artifact that other agents depend on is modified, the change must be committed and propagated immediately. Leaving shared modifications uncommitted creates invisible inconsistencies across the agent fleet.

5.8 Build Completion Signals

When an implementation phase is complete, the executing agent must post a completion signal before any subsequent work. This signal includes: files changed, deliverable description, and readiness for the next phase. Missing completion signals have caused governance lifecycle stalls and visibility gaps on multiple workstreams.

---

6. Governance Authority Levels

Governance systems progress through three levels of authority structure. Each level builds on the previous.

Level	Name	Description
Advisory	Policy-based governance	Governance policies exist. Agent compliance is expected but not structurally enforced.
Enforced	Structurally authorized governance	Execution requires structural authorization. Agents cannot advance work without passing defined gates.
Separated Authority	Multi-actor governance	Multiple independent actors are required for authorization. No single actor — human or agent — can both propose and approve a governance action.

The progression from Advisory to Enforced introduces structural controls. The progression from Enforced to Separated Authority introduces independence of authorization — the governance principle derived from Two-Person Integrity (see Section 3.1).

Organizations should assess their current authority level and plan progression based on operational risk tolerance and agent deployment scale.

---

7. The Incident-Driven Learning Loop

MOIAS is not static. The methodology improves continuously through a structured evolution pattern:

```
Incident observed
    → Root cause identified
    → Control introduced
    → Behavior changed
```

This loop has several important properties:

1. **Governance is empirical, not aspirational.** Every mechanism traces to a real failure. If a mechanism exists without a corresponding incident, it should be questioned.
2. **The incident log is the design document.** New governance mechanisms are justified by specific incidents, not by “best practice” claims.
3. **The methodology learns from production.** Each governance iteration is driven by a specific identified gap, not by theoretical analysis.

4. **Behavioral patterns accumulate.** As more organizations operate governed agent fleets, the library of documented behavioral failure modes grows. Each new pattern improves governance for all practitioners. See the Behavioral Pattern Taxonomy (patterns.md) for the current library.

---

8. Trust Calibration

Trust calibration is the principle that agent autonomy should be dynamically adjusted based on demonstrated behavior, not statically assigned.

8.1 The Principle

Different agents, performing different types of work, under different conditions, warrant different levels of oversight. Trust is earned through completed governance cycles with clean outcomes, not granted by default.

Observations from production operations:

- Agents performing creative implementation work may excel with high autonomy but require tight specification for infrastructure operations.
- Agents performing analytical review work may produce consistent, structured output with minimal oversight.
- The same agent may warrant different autonomy levels for different task types.

8.2 Calibration Inputs

Trust calibration considers:

- **Behavioral history.** How has the agent performed across previous governance cycles?
- **Task type.** Does this task type match the agent’s demonstrated strengths?
- **Incident history.** Has the agent been involved in governance incidents? What types?
- **Deviation patterns.** Are small deviations accumulating over time, even if no single deviation caused a failure?

8.3 Calibration Outputs

Trust calibration informs:

- **Oversight level.** How closely should human operators monitor this agent’s work?
- **Gate stringency.** Should phase gates require additional evidence or review for this agent?

- **Routing decisions.** Should this agent be assigned this type of work?
- **Autonomy boundaries.** What scope of independent decision-making is appropriate?

The concept is that trust calibration provides a dynamic, evidence-based mechanism for the governance system to learn from agent behavior and adjust oversight accordingly. This prevents both over-governance (which reduces throughput) and under-governance (which increases risk).

---

9. Design Principles

Nine design principles govern all governance mechanism design within MOIAS. These principles are derived from the theoretical foundations and validated against the incident history.

1. **No single actor holds both keys.** No agent should be both the author and sole executor of a governance decision.
2. **Defense layers must fail independently.** When adding a new mechanism, its failure mode must be independent of existing layers. Two layers that fail for the same reason are effectively one layer.
3. **Detect drift before it normalizes.** Every deviation is documented. A pattern of borderline outcomes is a drift signal, even if no individual outcome was incorrect.
4. **Orientation determines what you can see.** Different actors oriented toward different questions catch different failures. Governance systems need diverse orientations, not redundant actors.
5. **Authenticate both the person and the action.** Knowing who approved a decision is necessary but not sufficient. Knowing whether the action was compliant is also necessary. Both must pass independently.
6. **Governance is empirical, not aspirational.** Every mechanism should be traceable to a specific incident or measurable drift pattern.
7. **Cheap redundancy over expensive perfection.** Multiple automated checks are more cost-effective than relying on a single human judgment. Human attention is the scarcest resource — mechanisms should reduce cognitive load on human decision-makers.
8. **Governance must reduce operator cognitive burden.** Governance systems must structure decisions into discrete, auditable units. Governance that increases operator cognitive load introduces systemic risk. Impact statements, structured

phase transitions, and pre-validated decision surfaces exist to compress complex governance decisions into forms that operators can evaluate rapidly and reliably.

9. **A validator who participates in planning cannot independently assess the plan.** An actor who contributes to a plan's design shares the plan's assumptions. When that actor later reviews the plan's execution, they assess it from within the same orientation that produced it — and cannot see failures that the orientation itself obscures. Genuine validator independence requires both orientation diversity and, at enterprise scale, organizational independence. Organizational independence — as established in IV&V doctrine (MIL-STD-2168, IEEE 1012) — is the primary structural requirement at scale, because shared incentives can compromise orientation independence regardless of analytical diversity. Orientation diversity is the quality multiplier that makes organizational independence effective. In single-operator deployments where organizational independence is not yet achievable, orientation diversity provides the first meaningful layer of independence. *Derived from: Two-Person Integrity (U.S. DoD, 1962), Boyd's OODA Loop (1976), IV&V doctrine (MIL-STD-2168, IEEE 1012).*

---

10. Relationship to Other Governance Layers

MOIAS is one layer in a complete AI governance stack. It does not replace other layers — it fills a gap between them.

Layer	Function	Example Frameworks
Model governance	Ensures the underlying AI model is safe, fair, and compliant	NIST AI RMF, EU AI Act (model-level), ISO 42001
Security governance	Blocks unauthorized actions and enforces access controls	Agent security policies, tool-use restrictions
Operational governance (MOIAS)	Ensures agents do the right work, the right way, with human oversight	Phase gates, behavioral monitoring, trust calibration
Compliance governance	Maps operational practices to regulatory requirements	EU AI Act (operational), SOC 2, sector-specific regulations
Observability	Provides visibility into what agents did	Telemetry, tracing, logging

MOIAS occupies the operational governance layer. It assumes that model governance, security, and observability are addressed by their respective frameworks. MOIAS adds the methodology that ensures agent work is governed throughout its lifecycle —

not just at the model level, not just at the security boundary, but at every phase of real work under delegated authority.

---

11. Applying MOIAS

Organizations seeking to adopt the MOIAS methodology should:

1. **Assess current governance maturity.** Use the Governance Maturity Model (maturity-model.md) to determine the organization’s current level and identify gaps.
2. **Implement phase gates.** Define a lifecycle model appropriate to the organization’s agent workloads. Start with a basic lifecycle (Plan, Build, Review, Approve) and add phases as governance maturity increases.
3. **Establish the operator-as-router pattern.** Designate a human operator responsible for phase transition approvals. At scale, this role may be distributed across a team, but the principle remains: humans approve, agents execute.
4. **Deploy behavioral monitoring.** Use the Behavioral Pattern Taxonomy (patterns.md) to establish detection for known failure modes. Monitor for patterns even when outcomes appear correct — drift is invisible until it crosses a boundary.
5. **Start the incident log.** Document every deviation, even good-outcome deviations. The incident log becomes the primary input for governance improvement.
6. **Iterate.** The methodology improves with every governance cycle. Each incident is an opportunity to refine the system.

---

12. Limitations

This methodology was developed in the context of a governed software production platform. The behavioral patterns and governance mechanisms documented here reflect that operational environment. The following limitations should be considered:

- **Single-context empirical base.** The behavioral patterns and governance mechanisms documented here were derived from one operational environment. Cross-environment and cross-architecture validation is an open area for further research.
- The governance authority levels (Advisory, Enforced, Separated Authority) describe a maturity progression. Organizations at different scales may require different authority structures.

- Trust calibration inputs and outputs are described as principles. Specific calibration mechanisms are implementation-dependent.
- The behavioral pattern taxonomy documents patterns observed to date. It is not exhaustive. Additional patterns should be expected as agent capabilities and deployment contexts evolve.

References

1. Reason, J. (1990). *Human Error*. Cambridge University Press.
2. Reason, J. (1997). *Managing the Risks of Organizational Accidents*. Ashgate.
3. Dekker, S. (2011). *Drift into Failure: From Hunting Broken Components to Understanding Complex Systems*. Ashgate.
4. Vaughan, D. (1996). *The Challenger Launch Decision: Risky Technology, Culture, and Deviance at NASA*. University of Chicago Press.
5. Rasmussen, J. (1997). "Risk management in a dynamic society: a modelling problem." *Safety Science*, 27(2-3), 183-213.
6. Boyd, J. (1976). "Destruction and Creation." Unpublished paper.
7. Weick, K.E. & Sutcliffe, K.M. (2007). *Managing the Unexpected: Resilient Performance in an Age of Uncertainty*. 2nd ed. Jossey-Bass.
8. U.S. Department of Defense. (1962). Two-Person Integrity (TPI). AFI 91-104.
9. Parasuraman, R. & Manzey, D.H. (2010). "Complacency and Bias in Human Use of Automation: An Attentional Integration." *Human Factors*, 52(3), 381-410.
10. Lee, J.D. & See, K.A. (2004). "Trust in Automation: Designing for Appropriate Reliance." *Human Factors*, 46(1), 50-80.
11. Parasuraman, R. & Riley, V. (1997). "Humans and Automation: Use, Misuse, Disuse, Abuse." *Human Factors*, 39(2), 230-253.
12. Cummings, M.L. (2004). "Automation Bias in Intelligent Time Critical Decision Support Systems." AIAA 1st Intelligent Systems Technical Conference, AIAA 2004-6313.

Version History

Version	Date	Author	Description
1.0.0	2026-02-26	John J. McCormick	Initial publication
2.0.0	2026-02-28	John J. McCormick	Metadata standardization; citation, version, and PDF fields moved to frontmatter