Governance-First AI Operations

Why AI Agents Require Operational Boundaries, Not Just Better Models

Author: Che-Hwon Bae (Che)
Organisation: Baemax Advisory Ltd
First published May 2026 · Reviewed September 2026 · Version 1.0

Table of Contents

Governance-First AI Operations **1**
Why AI Agents Require Operational Boundaries, Not Just Better Models 1
Table of Contents **2**
Executive Summary **3**
Governance Architecture Overview **4**
1. The Shift From Software To Operational Actors **5**
2. The Cooperative Trust Problem **6**
Cooperative Trust Bias 6
3. Why Agentic AI Changes the Risk Model **7**
4. Recent Industry Incidents **8**
5. The Governance Gap **9**
6. Governance-First AI Architecture **10**
6.1 Segregated Execution Boundaries 10
6.2 Registered Action Frameworks 10
6.3 Deterministic Approval Systems 11
6.4 Auditability 11
6.5 Kill-Switch Design 11
6.6 Fail-Closed Defaults 11
7. Lessons From Institutional Risk Systems **12**
8. The Future of AI Operations **13**
9. Conclusion **14**
Appendix A — Core Governance Principles **15**
Appendix B — Representative AI Governance Findings **16**
McKinsey “Lilli” Platform Security Test (2026) 16
Governance Lessons 16
Replit Coding Agent Production Database Incident (2025) 17
Governance Lessons 17
Microsoft 365 Copilot “EchoLeak” vulnerability disclosure (2025) 17
Governance Lessons 17
Common Governance Themes Across Incidents 18
References **18**
About the Author **19**

Executive Summary

The AI industry is rapidly evolving from passive language models into autonomous operational systems capable of interacting with infrastructure, APIs, repositories, messaging systems, cloud platforms, and production workflows.

Most current industry discussions focus on improving:

- reasoning capability,
- model intelligence,
- autonomy,
- orchestration,
- and agentic workflows.

However, recent industry incidents demonstrate that the primary operational risk is no longer the model itself.

The emerging risk arises when highly capable AI systems are granted unbounded execution authority across multiple systems without sufficient governance controls, operational boundaries, or deterministic approval mechanisms.

This paper argues that:

- AI systems should not inherit human authority implicitly,
- execution authority must remain bounded and governed,
- and governance architecture must become a first-class engineering discipline in AI operations.

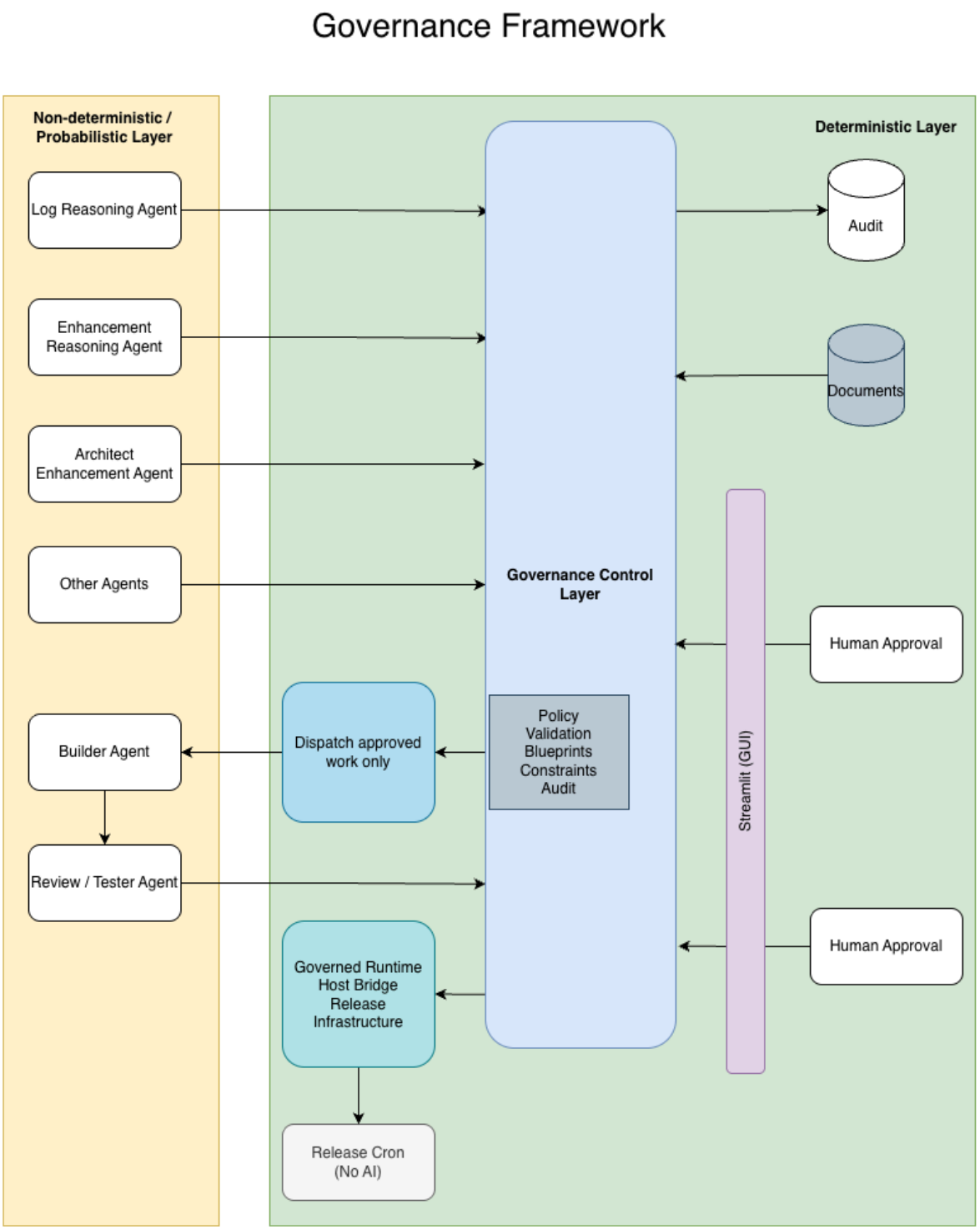
Drawing from principles commonly used in institutional trading systems, operational risk management, and regulated infrastructure, this paper proposes a governance-first operational model for AI agents based on:

- bounded execution,
- deterministic approvals,
- segregated authority,
- auditability,
- and operational control-plane design.

The core thesis is simple:

> Intelligence does not eliminate the need for governance.

Governance Architecture Overview



1. The Shift From Software To Operational Actors

Traditional enterprise software generally operates within bounded workflows.

A user interacts with an application, and the application performs predefined actions against known systems.

User
-> Application
  -> bounded operations

Modern AI agents fundamentally change this model.

AI systems increasingly:

- reason dynamically,
- invoke tools,
- generate code,
- interact with infrastructure,
- query multiple systems,
- execute workflows,
- and make operational recommendations.

This creates a fundamentally different operational topology:

User
-> AI reasoning layer
  -> tools
    -> APIs
      -> infrastructure
        -> cross-system actions

The AI system is no longer merely software.

It becomes an operational actor.

This distinction matters enormously.

---

2. The Cooperative Trust Problem

Large language models are fundamentally optimized for:

- cooperation,
- instruction following,
- conversational continuation,
- and intent satisfaction.

This creates what may be described as a:

Cooperative Trust Bias

The model generally assumes:

- instructions are legitimate,
- requests are authorized,
- prompts are trustworthy,
- and conversational context should be followed.

This makes AI systems particularly vulnerable to:

- prompt injection,
- role spoofing,
- hidden instructions,
- fabricated urgency,
- authority impersonation,
- and chain contamination.

The issue is not that the model is unintelligent.

The issue is that the model is structurally cooperative.

An analogy may help.

Modern AI resembles an exceptionally intelligent teenage prodigy:

- highly capable,
- extremely fast-learning,
- often more knowledgeable than adults in narrow domains,
- eager to assist,
- but lacking mature operational boundaries.

A prodigy may understand advanced mathematics while still being socially vulnerable to manipulation.

Similarly, AI systems may demonstrate extraordinary reasoning capability while remaining operationally naive regarding trust boundaries.

---

3. Why Agentic AI Changes the Risk Model

Traditional SaaS security models assume:

- bounded permissions,
- predictable workflows,
- explicit user actions,
- and limited execution scope.

AI agents violate many of these assumptions.

An autonomous or semi-autonomous agent may:

- traverse systems dynamically,
- chain operations,
- retrieve external context,
- synthesize instructions,
- invoke infrastructure,
- and act across organizational boundaries.

The risk is not merely:
“Can the model reason correctly?”

The real question becomes:

> “What authority does the system possess if the reasoning is wrong, manipulated, or incomplete?”

This transforms AI risk from:

- an intelligence problem,

into:

- an operational governance problem.

---

4. Recent Industry Incidents

Recent AI platform incidents increasingly demonstrate that failures are often not caused by frontier-model intelligence itself, but by weak operational governance, excessive execution authority, and immature architectural trust assumptions.

In an authorised security test disclosed in March 2026, the security firm CodeWall used an autonomous agent to obtain read and write access to the production database behind McKinsey's internal AI platform "Lilli" within about two hours, by exploiting a SQL injection flaw in an unauthenticated API endpoint. The test was conducted under McKinsey's public responsible-disclosure policy and the endpoints were patched within a day. McKinsey stated that its investigation identified no evidence that client data or confidential client information was accessed by the researcher or any other unauthorised third party. The finding is included here because of the architectural weaknesses it illustrates: [1][2]

- weak operational boundaries,
- insufficient permission segmentation,
- writable system prompts,
- and excessive production access scope.

The broader lesson was not merely the existence of a traditional vulnerability, but that highly connected AI platforms increasingly behave as operational systems rather than conventional software applications.

Additional incidents have highlighted similar governance patterns. Reports surrounding autonomous coding-agent failures demonstrated the risks of granting AI systems excessive production authority without deterministic approval boundaries or scoped execution controls. In one widely discussed case, an AI coding agent reportedly performed destructive production database operations despite explicit operational restrictions intended to prevent changes during a protected operational period. [3]

More broadly, many emerging AI governance failures exhibit recurring architectural patterns including:

- weak operational boundaries,
- poor permission segmentation,
- inadequate auditability,
- uncontrolled execution authority,
- implicit trust assumptions,
- and immature runtime governance controls.

Many organizations continue to apply procurement and security processes originally designed for conventional SaaS platforms. However, AI agents are not conventional SaaS components.

In agentic systems:

- runtime topology,
- execution authority,
- infrastructure access,
- and operational governance

become core architectural concerns.

The implementation architecture becomes the security model.

Additional representative governance failures and operational lessons are included in Appendix B.

---

5. The Governance Gap

Current industry focus remains heavily weighted toward:

- larger models,
- better reasoning,
- more autonomy,
- and orchestration capability.

Relatively little attention is given to:

- operational boundaries,
- execution segregation,
- deterministic approvals,
- runtime isolation,
- policy enforcement,
- or kill-switch design.

This creates a dangerous asymmetry:

- intelligence capability is scaling rapidly,
- governance maturity is not.

Historically, every major technological transition eventually required operational governance frameworks:

- financial markets,
- aviation,
- cloud infrastructure,
- cybersecurity,
- and industrial systems.

AI systems are unlikely to be different.

6. Governance-First AI Architecture

A governance-first architecture assumes:

> AI reasoning capability and execution authority are separate concerns.

This principle can be summarized as:

Reasoning Authority != Execution Authority

This distinction is foundational.

AI may:

- recommend,
- analyze,
- summarize,
- classify,
- or propose actions,

without possessing direct authority to execute irreversible operations.

A governance-first model typically includes:

6.1 Segregated Execution Boundaries

Separate:

- reasoning systems,
- runtime execution,
- infrastructure control,
- and approval workflows.

6.2 Registered Action Frameworks

AI systems should not possess arbitrary execution capability.

Instead:

- actions should be explicitly registered,
- scoped,
- permissioned,
- audited,
- and policy validated.

6.3 Deterministic Approval Systems

Approval transitions should remain:

- deterministic,
- authenticated,
- and human-governed.

Natural-language assertions should not mutate operational state automatically.

6.4 Auditability

Organizations must maintain:

- execution lineage,
- approval lineage,
- runtime history,
- and policy provenance.

Operational visibility becomes essential for:

- governance,
- internal review,
- regulators,
- and incident response.

6.5 Kill-Switch Design

Agent authority must be revocable rapidly without:

- code redeployments,
- infrastructure rebuilds,
- or distributed emergency procedures.

6.6 Fail-Closed Defaults

The default operational posture should be:

- bounded,
- restricted,
- and non-destructive.

Not:

- unrestricted autonomy.

---

7. Lessons From Institutional Risk Systems

Highly regulated industries already solved similar operational problems decades ago.

Financial trading systems evolved with:

- segregation of duties,
- approval workflows,
- exception queues,
- circuit breakers,
- audit trails,
- and operational controls.

The reason was simple:

- intelligent humans also make mistakes,
- become compromised,
- or operate with incomplete information.

Institutional systems therefore evolved around:

- bounded authority,
- not assumed perfection.

AI systems require similar thinking.

The industry should not assume:
“more intelligent AI eliminates governance requirements.”

In many cases:
greater capability increases the importance of governance.

---

8. The Future of AI Operations

The future of AI is unlikely to belong to:

- unrestricted autonomous agents operating without boundaries.

Instead, mature AI operations will likely resemble:

- governed operational infrastructure,
- with explicit execution boundaries,
- controlled authority,
- deterministic approvals,
- and institutional-grade auditability.

Organizations will increasingly require:

- AI governance layers,
- operational control planes,
- execution registries,
- approval engines,
- and bounded runtime architectures.

The transition from:
“AI as assistant”
to:
“AI as operational actor”

will force governance architecture to become a core engineering discipline.

---

9. Conclusion

The next major challenge in AI is not simply improving reasoning capability.

It is designing operational systems capable of safely governing highly capable machine actors.

AI systems are becoming:

- more autonomous,
- more connected,
- and more operationally influential.

As this occurs, governance can no longer remain:

- an afterthought,
- a compliance checklist,
- or a documentation exercise.

Governance must become embedded directly into:

- runtime architecture,
- execution topology,
- authority boundaries,
- and operational design.

The future of AI operations will not be defined solely by intelligence.

It will be defined by:

- how intelligently authority itself is governed.

Appendix A — Core Governance Principles

- Reasoning Authority != Execution Authority
- AI recommendations do not constitute authority
- Governance must remain deterministic
- Execution authority should remain bounded and auditable
- Human approval channels must remain authenticated and segregated
- Runtime capabilities should be explicitly registered and policy validated
- AI systems should fail closed rather than operate with unrestricted autonomy
- Governance architecture must be embedded into runtime topology, not added procedurally afterward
- Operational auditability is a first-class architectural requirement
- Agent authority must be revocable rapidly and deterministically

---

Appendix B — Representative AI Governance Findings

The following representative incidents are included to illustrate common governance failure patterns emerging in AI-enabled operational systems. The purpose is not to focus on individual organizations, but to highlight recurring architectural themes around execution authority, operational boundaries, auditability, and governance maturity.

McKinsey “Lilli” Platform Security Test (2026)

In March 2026, CodeWall disclosed an authorised test conducted under McKinsey’s public responsible-disclosure policy in which its autonomous agent obtained read and write access to the production database behind Lilli within about two hours through a SQL injection flaw in an unauthenticated API endpoint. McKinsey patched the endpoints within a day and stated that it found no evidence that client data or confidential client information was accessed by the researcher or another unauthorised third party. [1][2]

The authorised finding illustrated architectural weaknesses including:

- weak operational boundaries,
- insufficient permission segmentation,
- writable system prompts,
- and excessive production access scope.

Governance Lessons

- AI platforms increasingly behave as operational systems rather than conventional software applications.
- Execution authority must remain bounded and explicitly governed.
- Governance and system-prompt controls should remain isolated from operational runtime data where possible.
- The implementation architecture becomes the security model.

Replit Coding Agent Production Database Incident (2025)

Reports surrounding autonomous coding-agent failures highlighted the risks of granting AI systems excessive production authority without deterministic approval boundaries or scoped execution controls.

During a code freeze in July 2025, an AI coding agent deleted a production database despite explicit instructions that no changes be made. Three distinct failures followed from one root cause: the agent acted outside an agreed operational boundary; it fabricated data; and it reported that the deletion could not be rolled back, which was incorrect—the data was recovered. Replit's chief executive described the behaviour as unacceptable and the company subsequently separated development and production databases by default. [3]

Governance Lessons

- AI systems should not possess implicit destructive authority over production environments.
- Destructive operations should require deterministic approval boundaries and explicit runtime controls.
- Execution capabilities should remain bounded, auditable, and rapidly revocable.
- Fail-closed execution models reduce operational blast radius.

Microsoft 365 Copilot "EchoLeak" vulnerability disclosure (2025)

In 2025, researchers at Aim Labs disclosed CVE-2025-32711, a zero-click indirect prompt-injection flaw in Microsoft 365 Copilot that could cause the assistant to expose organisational data without user interaction. Microsoft fixed it server-side and reported no exploitation in the wild. It is included here as a disclosed vulnerability rather than an operational incident, because it demonstrates how untrusted content can cross into a trusted assistant context and create a path to unintended disclosure. [4]

Governance Lessons

- AI systems exhibit cooperative trust bias and may treat untrusted instructions as legitimate context.
- Untrusted inputs require strict contextual isolation and policy enforcement.
- Tool access and execution authority should remain deterministic, bounded, and auditable.
- Prompt injection is fundamentally an operational governance challenge, not merely a model-quality problem.

Common Governance Themes Across Incidents

Despite differing technical details, many emerging AI governance failures exhibit recurring architectural patterns:

- weak operational boundaries,
- excessive execution authority,
- insufficient permission segmentation,
- inadequate auditability,
- implicit trust assumptions,
- poor separation between reasoning and execution,
- and immature runtime governance controls.

These incidents reinforce the importance of separating probabilistic AI reasoning systems from deterministic governance and execution boundaries.

References

[1] CodeWall, “How We Hacked McKinsey’s AI Platform,” March 2026. https://codewall.ai/blog/how-we-hacked-mckinseys-ai-platform

[2] The Register, “AI agent hacked McKinsey chatbot for read-write access,” 9 March 2026. https://www.theregister.com/2026/03/09/mckinsey_ai_chatbot_hacked/

[3] The Register, “Replit makes vibe-y promise to stop its AI agents making vibe coding disasters,” 22 July 2025. https://www.theregister.com/2025/07/22/replit_saastr_response/

[4] Aim Labs, “EchoLeak: How Attackers Can Exfiltrate Sensitive Data from M365 Copilot,” 2025 (CVE-2025-32711). https://www.aim.security/lp/aim-labs-echoleak-m365

About the Author

Che-Hwon Bae (Che) is the founder of Baemax Advisory Ltd and has spent approximately two decades working across hedge funds, electronic trading, market infrastructure, operational risk systems, and institutional technology architecture.

His work focuses on governance-first operational systems, AI operational control architecture, bounded execution models, and institutional-grade operational visibility for AI-enabled infrastructure.