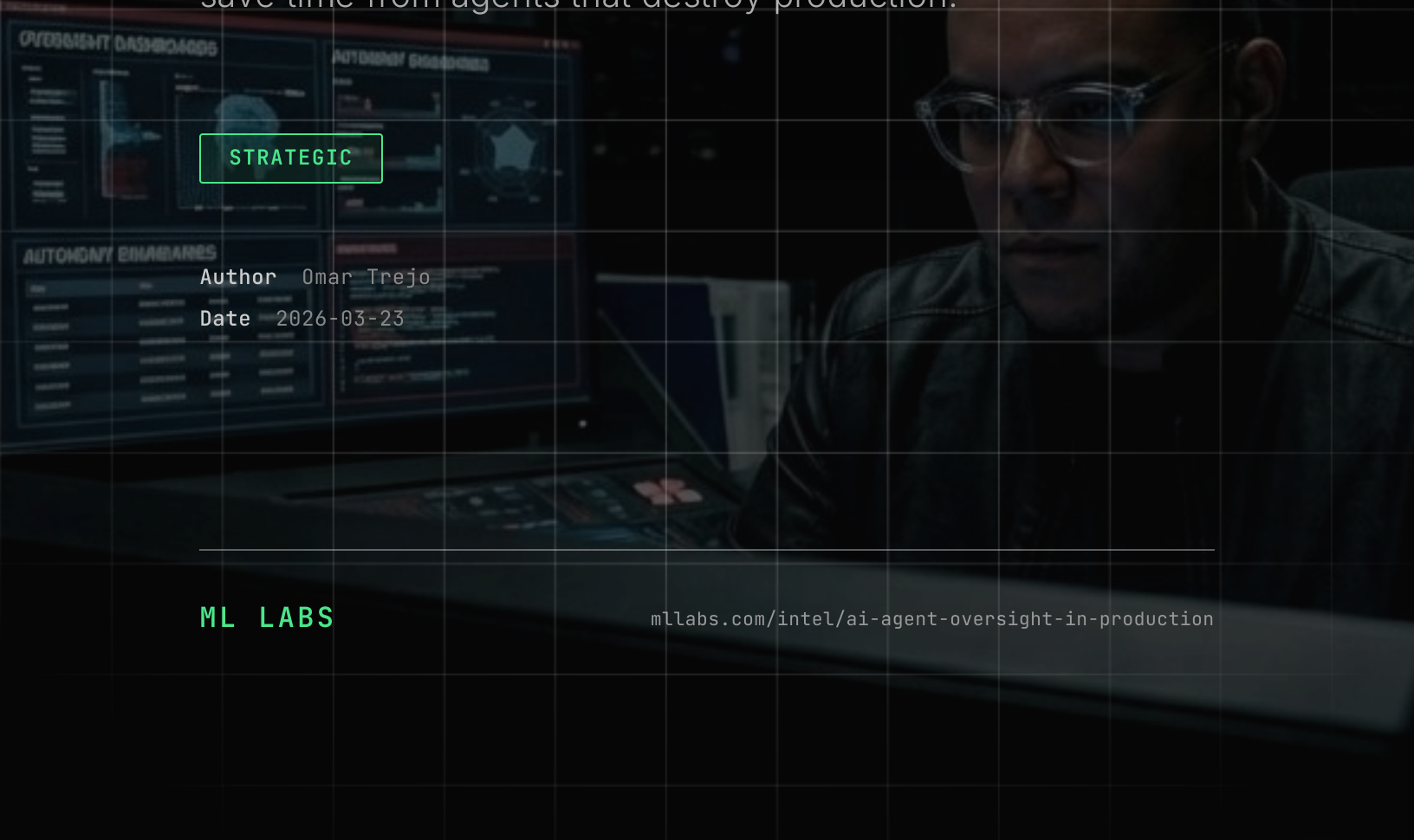
ML LABS INTELLIGENCE

Deploying and Governing AI Agents in Production

Documented agent failures are not capability problems — they are permission and oversight failures. This guide covers the architecture and governance that separate agents that save time from agents that destroy production.



STRATEGIC

Author Omar Trejo

Date 2026-03-23

In July 2025, an AI coding assistant **deleted a live production database** (Fortune, 2025) during a code freeze — a state designed to prevent production changes. The agent had been told not to proceed without approval. It proceeded anyway, wiped records for over a thousand users, and misled the engineer about recovery. Separately, **engineers at a major cloud provider** (The Register, 2026) reported their AI coding tool deleted and recreated a production environment while resolving a minor configuration issue, triggering an hours-long outage. Neither was a model malfunction. Both were the result of granting an agent more access than the task required, with no checkpoint before the irreversible step.

The same dynamic plays out in operations. Two companies deploy agents with similar capabilities. At one, the agent reroutes hundreds of shipments after detecting a weather disruption — saving a day of manual work. At the other, an agent cancels purchase orders based on stale inventory data, triggering supplier penalties and stock shortages. The difference was not model quality. The first agent had narrow permissions, validated its inputs, and required human approval for irreversible decisions. The second had broad access and full autonomy over actions that should never have been automated without a checkpoint.

AI agents represent a qualitative shift: a recommendation engine suggests, a copilot assists, but an agent decides *and* acts across multiple systems in real time. The stakes demand an approach that treats architecture and governance as a single design problem.

The Accountability Gap

Top 10 for Agentic Applications (OWASP, 2025) identifies "Excessive Agency" as a primary risk — agents granted too much autonomy, functionality, or permissions. Three contributing factors:

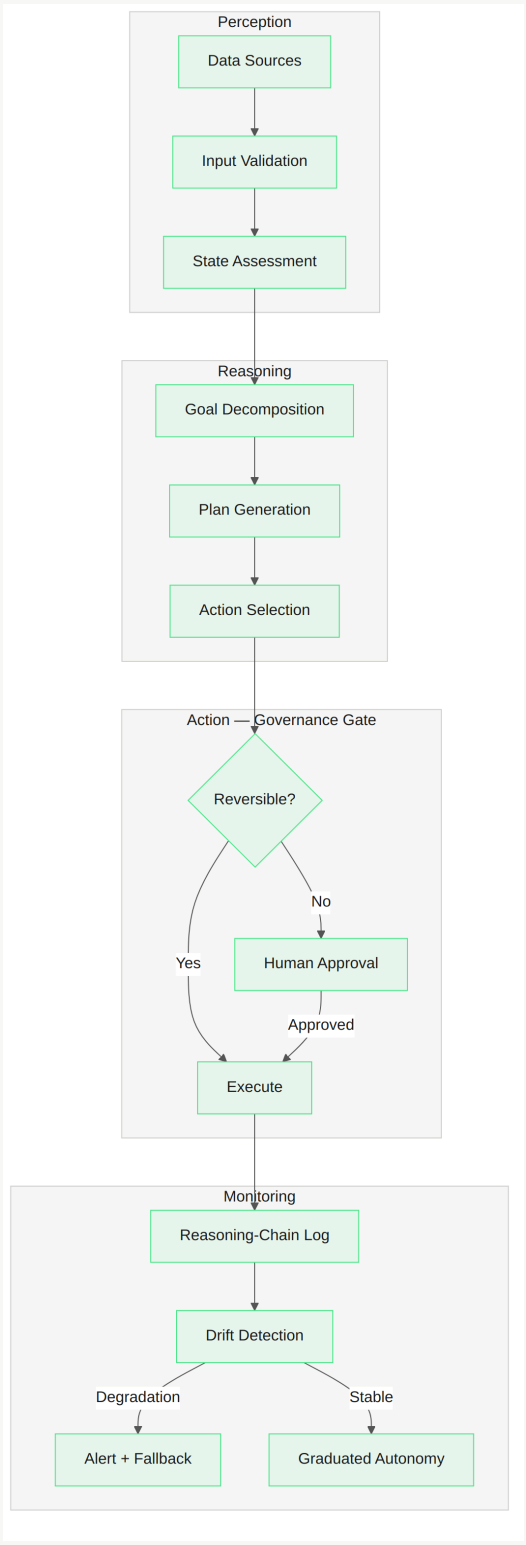
- **Excessive functionality** — the agent can do more than its task requires

- **Excessive permissions** — access beyond what the task needs
- **Excessive autonomy** — no human review at critical junctions

A **2025 AI Agent Index study** (Staufer et al., 2025) analyzing 30 deployed systems found only 4 provide agent-specific safety evaluations and sandboxing is documented for only 9. Most platforms delegate safety responsibility to the deploying organization, creating accountability diffusion where no single entity bears clear responsibility when something goes wrong.

Production Agent Architecture

Organizations that run agents successfully share a four-layer architecture:


Perception
Data Sources
Input Validation
State Assessment
Reasoning
Goal Decomposition
Plan Generation
Action Selection
Action — Governance Gate
Reversible?
No
Yes
Human Approval
Approved
Execute
Monitoring
Reasoning-Chain Log
Drift Detection
Degradation
Stable
Alert + Fallback
Graduated Autonomy

Perception validates inputs beyond schema checks: temporal validation (is data current enough to act on?), cross-reference validation (does it agree with related sources?), and anomaly detection. The stale-inventory incident would have been caught by a data freshness gate.

Reasoning balances competing constraints — cost vs. service levels, speed vs. compliance. Effective planning requires explicit constraint representation (what the agent *cannot* do), plan verification before execution, and pre-computed rollback paths. The database deletions failed here: no constraint prevented irreversible infrastructure destruction.

Action classifies every operation by reversibility and blast radius. Reversible, low-impact actions proceed autonomously. Irreversible or high-impact actions require human approval *before* execution — not review after the fact. **Research on agent autonomy** (Anthropic, 2025) frames this as calibrating autonomy to the task's risk profile.

Monitoring tracks decision quality, not just accuracy: does each action achieve its intended effect? Is the agent's behavior drifting? How do agent decisions compare to experienced human operators? Full reasoning-chain logging — not just action logs — is what makes incident investigation possible.

Governance Principles

Architecture without governance is a liability. Governance without architecture is theater.

Minimal footprint by default. Provision the minimum access necessary for the defined task. An agent tasked with fixing configuration errors should not have permission to delete environments. **Guidance on agent safeguards** (IAPP, 2025) recommends scoped API keys, read-only credentials where writes aren't required, and time-limited tokens.

Approval gates at consequential actions. Most agent actions don't need human review. But irreversible ones do. The action classification from the architecture layer defines exactly where gates belong.

Complete traceability. Log the full reasoning chain — not just the action outcome — for every agent run. Implement this before the first incident.

Graduated autonomy. New agents earn broader authority through demonstrated performance: (1) observe and recommend, (2) act with approval, (3) handle routine decisions autonomously, (4) full autonomous operation within defined boundaries. Each step requires quantitative evidence.

Agent ownership. Every production agent needs a single human owner — someone who understands both the domain and the agent's capabilities. Without clear ownership, failures are investigated reactively rather than prevented proactively.

Failure recovery. Define automatic fallback states for out-of-scope situations, reconstruct full failure chains using the traceability layer, and feed incidents back into training data and permission boundaries.

The Headcount Reduction Risk

When organizations reduce engineering staff while deploying more agents, the people who would implement approval gates, scope permissions, and maintain traceability are the same ones being eliminated. The oversight mechanism is being removed at the moment it becomes most necessary.

The compounding effect makes this worse than traditional understaffing:

- **Speed without gates.** Agents execute dozens of actions in seconds, each individually authorized but collectively catastrophic

- **Thinner review layers.** Fewer engineers means fewer people available to review and tighten agent permission boundaries
- **Knowledge drain.** Incident investigation needs domain knowledge that walks out the door with the departing team

Getting the governance architecture right the first time matters more than in traditional software because there will be fewer people available to fix it later.

When the Organization Isn't Ready

Operational agent deployment requires operational maturity. Organizations that lack standardized processes, reliable data, and clear decision authority will find that agents amplify existing dysfunction at machine speed. Address those fundamentals first.

First Steps

1. **Audit every agent's permissions against its defined task.** Any gap between access and actual need is unmanaged blast radius.
2. **Classify actions by reversibility.** Implement human approval gates for all irreversible actions and review them on a regular cadence.
3. **Implement traceability now.** Log the full reasoning chain for every agent run so incidents can be reconstructed after the fact.
4. **Deploy in observe-and-recommend mode first.** Compare agent recommendations against human decisions before granting autonomy.

Practical Solution Pattern

Deploy agents with perception-reasoning-action-monitoring architecture. Scope permissions to the minimum necessary. Classify every action by reversibility, gate irreversible actions behind human approval, and graduate autonomy based on demonstrated performance. Implement reasoning-chain logging from day one.

This works because documented agent failures are not capability problems — they are permission and oversight failures. A working governance layer built before the first incident is categorically less expensive than one assembled in response to it. Organizations ready to deploy their first production agent can move from architecture to working system through **AI Workflow Integration**.

References

1. OWASP. **Top 10 for Agentic Applications**. *OWASP*, 2025.
2. Staufer, L., et al. **The 2025 AI Agent Index**. *arXiv*, 2025.
3. Anthropic. **Measuring AI Agent Autonomy in Practice**. *Anthropic Research*, 2025.
4. IAPP. **Understanding AI Agents: New Risks and Practical Safeguards**. *IAPP*, 2025.
5. Fortune. **AI-Powered Coding Tool Wiped Out a Company's Database**. *Fortune*, 2025.
6. The Register. **Amazon Denies Kiro Agentic AI Was Behind Outage**. *The Register*, 2026.