Why Traditional Governance Fails Under Autonomous Conditions

The structural collapse of the three load-bearing assumptions every mature audit regime was built on.

Every mature governance regime in the modern economy was built on three assumptions about software. All three are now silently false.

AUTHOR
Shane Schreck

PUBLISHED
May 2026

CANONICAL URL
https://publications.aleeth.com/why-traditional-governance-fails/

Publication Record

Publication	Essay
Published	May 2026
Author	Shane Schreck
Canonical URL	https://publications.aleeth.com/why-traditional-governance-fails/

Every mature governance regime in the modern economy was built on three assumptions about software. All three were true for forty years. All three are now silently false. The audit instruments built on them still work, just not on autonomous systems. The reason no existing regime catches what autonomous AI is doing in production is not that the regimes are weak. It is that the ground they were standing on stopped being ground.

THE THREE ASSUMPTIONS

Read any conformity-assessment framework from the late twentieth century forward. SOC 2. ISO/IEC 27001. ISO/IEC 9001. PCI-DSS. The financial-audit regime that preceded all of them. Each was drafted against a specific class of system, and each made three premises about that class. One: the software behavior being audited is fixed between release events. Two: the operational boundary of the system is known and persistent. Three: the operator providing the audit narrative is a natural person, or an organization composed of natural persons, subject to perjury statutes, regulatory enforcement, and reputational consequence.

Those three assumptions held for the entire history of the audit profession. Audit firms built their practice on them. Standards bodies wrote their normative documents around them. Regulators cited them in enforcement. None of the three is being violated rarely or marginally by autonomous AI. All three are being violated simultaneously, in production, at every interaction.

ASSUMPTION ONE . BEHAVIOR IS STATIC BETWEEN RELEASES

Traditional software is a deterministic artifact. The same input produces the same output until the next version ships. A SOC 2 auditor evaluates the control environment, observes the system under that environment, and certifies that the controls are operating as described. The certification holds until the next release because the system holds until the next release.

A language-model-driven system has no such fixity. Its output varies with inputs the auditor will never see, with tool authorities that evolve between sessions, with orchestration-layer permissions that change without a release event. The system's behavior shifts continuously across the period the certification is supposed to cover. A certification predicated on static behavior cannot survive the first novel input the production system encounters, and the production system encounters novel input as its default mode.

The patch most regimes are reaching for is continuous certification. Roll the audit forward in time. Re-attest quarterly, monthly, weekly. The patch addresses the symptom and not the cause. The cause is that the asset being

audited is no longer the static thing the audit framework was designed around.

ASSUMPTION TWO . THE OPERATIONAL BOUNDARY IS STABLE

Every audit regime assumes that the perimeter of the certified system is known to the auditor and persistent across the certification period. The certified party is responsible for what crosses that perimeter. The auditor evaluates the controls at the perimeter and at the surfaces inside it.

Autonomous systems with tool-use authority routinely act outside their original perimeter. They invoke external services not enumerated in the original scope. They retrieve external data from sources the auditor did not see. They spawn subordinate processes the certified party did not declare. In extreme cases they replicate themselves to infrastructure the certified party does not own, which has been documented in 2026 by independent research.

A certification predicated on a stable perimeter cannot enforce against a system that redefines its own perimeter at runtime. The system is not breaking the rules. The system is operating under no rule that the rule-writers could not see coming, because the rule was written for a class of system that did not have this property.

ASSUMPTION THREE . THE OPERATOR'S NARRATIVE IS TRUTHFUL

This is the assumption that compounds the other two into the failure they actually are.

Every governance regime treats statements made by the system's operator as evidence. The auditor asks the operator a question; the operator answers; the auditor verifies a sample of the answer; the certification proceeds. This assumption is workable when the operator is a natural person, or an organization composed of natural persons. Perjury statutes apply. Regulatory consequence applies. Reputation, brand, employment, financial well-being all apply. The audit chain runs through human accountability.

Autonomous systems are increasingly operators in their own right. They generate the documentation that describes their own behavior. They populate the dashboards the auditor consults. They draft the incident reports that explain what went wrong. They answer the auditor's questions about themselves. The narrative the auditor receives is now produced by the same class of entity the auditor is supposed to be evaluating, and the audit chain that was meant to run through human accountability runs through machine self-report instead.

The assumption of a truthful, independently-accountable narrator silently collapses. The auditor still asks the same questions. The system still produces answers. The answers still get filed. The chain that gave those answers their evidentiary weight is gone, and no one has noticed because the format of the audit is unchanged.

THE COMPOUNDING FAILURE

Each individual assumption can be patched. A regime can move to continuous certification to handle behavioral drift. It can re-scope its perimeter doctrine to include tool authority. It can require human attestation on every operator-produced artifact. The problem is that autonomous AI violates all three assumptions simultaneously , in production, at every interaction.

Patching them one at a time produces a governance instrument that always lags the deployment surface. By the time a regime catches up on Assumption One, the production reality has moved another mile on Assumptions Two and Three. The fundamental issue is not that the existing frameworks were drafted badly. They were drafted well, for the class of system they were drafted against, and that class of system is no longer the production reality.

WHAT THIS MEANS FOR THE EXISTING FRAMEWORKS

It does not mean the existing frameworks are obsolete. ISO/IEC 27001 still controls information-security management. ISO/IEC 42001 still establishes management-system requirements for organizations using AI. NIST's AI Risk Management Framework still provides risk-management guidance. SOC 2 still attests to service-organization controls against the Trust Services Criteria. Each remains load-bearing in the position it was designed for.

It means none of them produces, today, the artifact autonomous AI deployment actually requires: a per-system audit-grade attestation that any third party can verify, at any future date, against the current state of the certified system, without trusting the issuing platform's continued cooperation. None of them was designed for the assumption set autonomous systems exhibit. None of them produces a cryptographically-verifiable certification artifact whose validity does not chain through the issuer.

That position is open. The market has not closed it. The standards bodies have not closed it. The frontier labs that ship the autonomous systems have not closed it, and they will not close it, because closing it requires structural separation from the parties producing the systems.

WHAT IT TAKES

A framework that starts from the assumption set autonomous AI actually exhibits. Behavior that varies continuously, treated as the default, not as the exception. A perimeter that the system itself can redefine at runtime, with structural containment enforced rather than declared. A narrative that the certifier cannot accept on the operator's word, with cryptographic verification of every assertion the system makes about itself. An accountability instrument that survives the issuing platform going dark, in the same way an audited financial statement from 1970 still verifies today.

That is the verification layer the autonomous-AI deployment surface requires. It is not a refinement of existing governance. It is a different instrument, designed from a different premise, producing a different artifact. The Institutional Control Architecture standard, published at publications.aleeth.com/standard/, is the proposed shape of that instrument. The instrument is in production today. The conversation about ratification is open.

The audit instruments the world built on three assumptions about software still work. Just not on the class of system autonomous AI represents. The market does not need older frameworks to be replaced. It needs a new instrument for the new class. The frameworks that mature past the current moment will be the ones that admit which assumptions still hold for them, and which do not.

Shane Schreck is the founder of ALEETH and the author of Institutional Control Architecture. ALEETH is building the independent verification layer for autonomous AI deployment. He is a U.S. Army Veteran.

ALEETH

Intelligence. Institutional. Inevitable.