International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18(20s) 2026 pp. 700-709
https://doi.org/10.70917/ijcisim-2026-5212
Article

The Automation Paradox: Balancing Artificial Intelligence Autonomy and Human Governance in Private Cloud Infrastructure Management

Shaileshbhai Revabhai Gothi

Independent Researcher, USA

Abstract: The proliferation of artificial intelligence-driven automation in private cloud infrastructure has created a governance challenge that the field has not yet adequately theorized: as autonomous systems gain capability, the organizational mechanisms required to oversee them grow more complex, and the human operators responsible for accountability grow less competent through disuse. This paper introduces the Autonomy-Accountability Boundary (AAB) model, which reconceptualizes this tension not as a calibration problem - the question of how much autonomy to grant an AI system - but as an architectural design problem: where to draw an explicit, enforceable, and observable boundary between machine-executable action and human-declared intent. Drawing on practitioner experience in software-defined data center (SDDC) lifecycle management and supported by a synthesis of literature spanning autonomic computing theory, human-automation interaction research, and production AIOps practice, this paper argues that desired-state management architectures provide a natural implementation substrate for the AAB model. When AI systems operate within declared desired states, autonomy is maximally productive and accountability is structurally preserved. When AI systems are permitted to redefine the operational target unilaterally - as increasingly capable agentic systems are designed to do - governance fails not because of malicious intent but because the accountability boundary has been architecturally dissolved. The paper proposes four operational governance mechanisms derived from the AAB model and discusses their implementation in enterprise private cloud environments. These findings have direct relevance for cloud infrastructure architects, platform engineering organizations, and policymakers engaged with AI governance in high-stakes operational contexts.
Keywords: AI governance, private cloud, SDDC lifecycle management, autonomy-accountability boundary, AIOps, desired-state management, human-AI teaming

1. INTRODUCTION

The deployment of artificial intelligence in enterprise infrastructure management has accelerated substantially over the past decade, driven by the convergence of increasingly capable machine learning systems, the maturation of cloud-native architectures, and the operational pressures facing infrastructure teams managing environments of unprecedented scale and complexity. Modern private cloud platforms - software-defined data centers that abstract compute, storage, and network resources into programmable infrastructure - routinely host thousands of workloads executing across heterogeneous physical substrates. The operational management of these environments has exceeded the cognitive bandwidth of human operators working without AI assistance. AIOps platforms, autonomous remediation engines, and large language model-augmented incident response tools have moved from experimental deployments to production dependencies at enterprise scale (Dang, Lin, & Huang, 2019; Notaro, Cardoso, & Gerndt, 2021).

Yet alongside these productivity gains, practitioners and researchers have identified a persistent structural tension. As AI systems assume greater operational autonomy, the organizational mechanisms necessary to govern that autonomy - to ensure accountability when systems fail, to maintain operator competence, and to preserve the human

judgment required for novel failure modes - become simultaneously more important and more difficult to sustain. Bainbridge (1983), in a prescient analysis of industrial process automation, observed that the introduction of automation to relieve operators of routine tasks produces an operator who is ill-prepared to intervene when the automation fails - precisely the moment when human judgment is most needed. Forty years later, the same irony manifests with striking fidelity in enterprise cloud infrastructure management.

This paper engages directly with this tension. The conventional framing positions AI autonomy and human governance as a dial: practitioners are advised to calibrate the level of automation, retain humans for high-stakes decisions, and implement override mechanisms. This framing, while practically useful, mischaracterizes the structural nature of the problem. The autonomy-governance challenge in cloud infrastructure is not primarily a question of calibration. It is a question of architectural design. The boundary between what an AI system may execute autonomously and what requires human-declared intent must be defined explicitly, enforced structurally, and made observable to all parties responsible for accountability. When this boundary is implicit - embedded in organizational convention rather than system architecture - it erodes under operational pressure, and governance fails not through deliberate choice but through incremental drift.

The central contribution of this paper is the Autonomy-Accountability Boundary (AAB) model, which provides a theoretical framework and practical implementation path for this architectural approach to AI governance in private cloud environments. The AAB model is grounded in the observation that desired-state management architectures - a foundational pattern in SDDC lifecycle management in which operators declare the target configuration and AI/automation systems execute convergence - provide a natural substrate for drawing and maintaining this boundary. The operator's declaration of desired state is, in effect, a governance artifact: it defines the space within which autonomy is legitimate, provides a ground truth against which autonomous actions can be evaluated, and preserves a human-authored record of intent that enables accountability after the fact.

The remainder of this paper is structured as follows. Section 2 reviews the theoretical foundations in human-automation interaction and autonomic computing. Section 3 characterizes the governance deficit in current private cloud automation practice. Section 4 introduces and formalizes the AAB model. Section 5 examines AAB implementation in SDDC lifecycle management. Section 6 proposes operational governance mechanisms derived from the model. Section 7 examines empirical evidence from production environments. Section 8 discusses implications for cloud infrastructure architects and policymakers. Section 9 concludes.

2. THEORETICAL FOUNDATIONS: HUMAN-AUTOMATION INTERACTION IN COMPLEX SYSTEMS

The theoretical preconditions for the governance challenge this paper addresses were established well before cloud computing emerged as an operational paradigm. Two foundational contributions are particularly generative.

Bainbridge (1983) identified what she termed the ironies of automation: the observation that automation systems designed to eliminate human cognitive work paradoxically require higher-order cognitive capability from human supervisors, precisely because the remaining tasks - diagnosing automation failures, intervening in novel situations, and maintaining system models accurate enough to support override decisions - are cognitively demanding tasks that routine operators are systematically underprepared to perform. Bainbridge's analysis was grounded in industrial process control, but its structural logic applies with equal force to cloud infrastructure management. An operator who delegates routine remediation to an AIOps platform is an operator whose diagnostic skills, environmental familiarity, and situational awareness are all degrading in real time. When the automation fails - and complex systems fail - that operator must intervene with degraded capabilities into a system whose state they have not directly shaped.

Parasuraman, Sheridan, and Wickens (2000) provided the field's most influential taxonomic framework for thinking about levels of automation. Their model distinguishes ten levels of human-automation interaction along a spectrum from full human control to full automation, and further distinguishes four functional domains in which automation can operate: information acquisition, information analysis, decision selection, and action implementation.

The Parasuraman model has several properties relevant to the present analysis. First, it clarifies that autonomy is not a scalar property but a multidimensional one: a system may be highly autonomous in action implementation while maintaining human authority over decision selection. Second, it implies that different automation levels carry different accountability structures: at lower levels of automation, the human operator retains clear causal responsibility for outcomes; at higher levels, responsibility diffuses across the human-system boundary in ways that complicate attribution.

Calhoun (2022) sharpens this analysis by distinguishing adaptable automation - in which the human operator retains authority to assign how automation is applied - from adaptive automation, in which the system itself determines its level of engagement in response to detected conditions. Calhoun's empirical review finds that adaptable automation produces superior outcomes on metrics including task performance, situation awareness, and perceived control. This finding has direct architectural implications: it suggests that governance designs which preserve operator authority over automation scope - rather than delegating scope decisions to the automation system itself - produce better operational outcomes by both performance and accountability measures.

The autonomic computing vision articulated by Kephart and Chess (2003) provides the engineering counterpart to these behavioral science foundations. IBM's autonomic computing initiative proposed a self-managing architecture built around a MAPE-K feedback loop: Monitor, Analyze, Plan, and Execute, operating over a shared Knowledge base. The autonomic computing vision was explicitly motivated by the growing complexity of enterprise computing environments - Kephart and Chess observed that the systems being built were already too complex for human operators to manage directly - and proposed autonomous self-management as the only viable path to operational continuity at scale. The MAPE-K model has proven enormously influential; it underlies the design of virtually every contemporary AIOps platform and cloud automation framework (Hezavehi et al., 2021).

What the autonomic computing vision did not adequately theorize was the governance implication of MAPE-K autonomy. The original formulation assumed that the knowledge base - the repository of policies, system models, and operational constraints that governs autonomous action - would be authored, maintained, and supervised by human engineers. In practice, as AI-generated knowledge and learned policies have entered the knowledge base, and as the pace of environmental change has outstripped human capacity to keep policy current, the governance assumption embedded in the original design has eroded. The system's knowledge - and therefore its behavioral constraints - may no longer reflect any specific human intent.

3. AUTONOMIC COMPUTING AND THE GOVERNANCE DEFICIT IN PRIVATE CLOUD INFRASTRUCTURE

Cloud computing realized the utility computing vision at a scale that Kephart and Chess could not have fully anticipated. Armbrust et al. (2010) characterized cloud computing as a transformation in the economics and operational model of computing infrastructure, enabling organizations to access virtually unlimited compute capacity without capital investment, and establishing the API-driven programmable data center as the new operational paradigm. This programmability is both the source of cloud infrastructure's operational power and the substrate through which AI governance challenges manifest.

Private cloud infrastructure - enterprise-operated software-defined data centers built on platforms such as VMware vSphere and vSAN, Nutanix, OpenStack, and their Kubernetes-native successors - extends the cloud model to environments where data residency, security architecture, and regulatory compliance requirements preclude public cloud adoption. Private cloud platforms implement programmable infrastructure management through a combination of declarative desired-state engines, event-driven remediation systems, and increasingly AI-augmented operational control planes. In this context, SDDC lifecycle management - the discipline of managing the birth, operational configuration, upgrade, and decommissioning of the software-defined data center platform itself - represents one of the most complex and high-stakes operational domains in enterprise IT.

The governance deficit in this domain manifests along three distinct dimensions. The first is the opacity of automated action. Contemporary AIOps platforms and autonomous remediation systems execute large volumes of individual configuration changes, resource reallocations, and remediation actions in large enterprise environments. The operational velocity that makes these systems valuable also makes the audit trail of automated actions difficult to construct retroactively. When an incident occurs, the investigation must untangle a causal chain in which many contributing actions were executed by AI systems whose decision rationale may not have been logged with sufficient fidelity to support root cause analysis (Li et al., 2022).

The second dimension is scope ambiguity. The operational boundary of an autonomous system - the set of actions it is authorized to take without human approval - is typically defined through a combination of role-based access control, policy configuration, and organizational convention. These mechanisms are designed for relatively static operational contexts. In environments where AI systems are learning and adapting their behavioral policies from operational feedback, the effective scope of autonomous action drifts over time in ways that neither the AI system nor its human supervisors may track explicitly. The result is a governance gap: the organization believes the automation is operating within sanctioned boundaries when the boundaries have in practice been dissolved through incremental scope expansion (Kreuzberger, Kühl, & Hirschl, 2023).

The third dimension is what this paper terms accountability dissolution - the progressive erosion of clear human responsibility for operational outcomes as automation assumes greater operational authority. Raisch and Krakowski (2021) identify the automation-augmentation paradox at the organizational level: automation deployed to free human cognitive resources for higher-order judgment tasks may instead produce cognitive withdrawal, as human operators disengage from operational monitoring precisely because the system appears to be managing effectively. The paradox is that the organization remains accountable for the operational outcomes produced by its autonomous systems while systematically degrading its capacity to govern those systems. The automation complacency literature documents this phenomenon empirically across multiple operational domains: automated system operators who are not required to actively engage with operational data lose the situational awareness necessary to detect system states that fall outside the automation's competency envelope (Saadeh et al., 2025; Cooke, 2024).

4. THE AUTONOMY-ACCOUNTABILITY BOUNDARY (AAB) MODEL

The Autonomy-Accountability Boundary model proposes that the governance challenge described in Section 3 can be structurally addressed by treating the boundary between autonomous AI action and human-declared intent as a first-class architectural artifact - an explicit, enforceable, and observable component of the operational system design, rather than an implicit organizational convention.

The model rests on three definitional claims.

Claim 1: Autonomy and accountability are structurally coupled, not inversely related.

The conventional framing treats autonomy and accountability as a trade-off: granting more autonomy to AI systems means accepting less accountability, and vice versa. The AAB model rejects this framing. Autonomy and accountability are structurally coupled: autonomous action is governable when the space of sanctioned autonomous action is explicitly defined and the actual behavior of the autonomous system is observable against that definition. The problem is not autonomy per se but the absence of an explicit boundary.

Claim 2: The appropriate unit of the boundary is human-declared intent, not action type or system component.

Conventional governance approaches draw the autonomy boundary at the level of action types (this category of action requires human approval) or system components (this subsystem is governed by this policy). The AAB model draws the boundary at the level of human-declared intent: the autonomous system is authorized to take any action necessary to converge the operational state toward the declared intent, and is not authorized to take actions that alter, redefine, or circumvent that intent. This framing preserves maximum operational flexibility for the autonomous

system within the declared intent space while maintaining an absolute governance boundary at the point of intent definition.

Claim 3: Desired-state management architectures provide the natural substrate for AAB implementation.

Desired-state management - the operational pattern in which human engineers declare a target configuration that the automation system continuously works to achieve and maintain - is architecturally isomorphic to the AAB model. The desired-state declaration is the human-declared intent artifact. The automation system's MAPE-K loop operates within the convergence space defined by that declaration. The difference between desired-state governance and current practice is not architectural; it is the disciplined enforcement of the constraint that the automation system may not alter the desired state without human authorization. In current practice, this constraint is routinely violated by systems that respond to operational conditions by autonomously escalating their own operational targets - systems that, for example, expand the scope of a planned maintenance operation in response to discovered dependencies, or adjust operational policy parameters in response to observed workload characteristics, without explicit human approval.

The formal structure of the AAB model can be stated as follows. Let S denote the operational state of the infrastructure. Let D denote the declared desired state, authored by human operators. Let A denote the action space of the autonomous system. The AAB boundary B partitions A into two subsets: A_sanctioned, the set of actions the autonomous system may take without human approval, defined as all actions that move S toward D without modifying D; and A_governed, the set of actions that require human authorization before execution, defined as all actions that would modify D, create new operational objectives not derivable from D, or require assumptions about operator intent not explicitly encoded in D. The governance discipline of AAB implementation consists of maintaining the clarity and completeness of B, monitoring the autonomous system's actual action distribution for drift toward A_governed, and creating organizational mechanisms that ensure human engineers are sufficiently engaged with the operational state to author and maintain D with accuracy and timeliness.

The AAB model has direct implications for the design of AI oversight mechanisms. Arrieta et al. (2020) argue that explainability is a prerequisite for responsible AI deployment in high-stakes contexts - that operators must be able to understand the reasoning behind autonomous decisions in order to maintain meaningful oversight. The AAB model clarifies what explainability is for in this context: the primary governance function of explainability is not general transparency but specific boundary enforcement. An explanation architecture designed for AAB governance focuses on answering three questions about every significant autonomous action: (1) What element of D is this action serving? (2) Does this action fall within A_sanctioned as defined at the time of execution? (3) Did this action modify, circumvent, or make implicit assumptions about D? These questions are more tractable than general explanations of AI reasoning, and they directly support the accountability attribution required when autonomous actions contribute to operational incidents.

5. AAB IMPLEMENTATION IN SDDC LIFECYCLE MANAGEMENT

SDDC lifecycle management presents a compelling implementation context for the AAB model because it combines high operational stakes with a natural desired-state management architecture. The management of SDDC platform upgrades - the process of updating the firmware, hypervisor software, storage controllers, and network fabric components that constitute the software-defined data center - requires coordinating thousands of interdependent configuration changes across a production environment that cannot be taken offline. This process has been a primary target for AI-driven automation, and it is a context in which the governance challenges of autonomous AI action are acutely visible.

Contemporary SDDC lifecycle management platforms implement desired-state architectures in which platform engineers declare a target version topology - the software versions, patch levels, and configuration profiles that represent the target operational state of the SDDC - and the lifecycle management system orchestrates the convergence

process. The governance challenge arises at the boundary between the declared topology and the discovered state: production SDDC environments invariably contain dependencies, customizations, and operational constraints that were not anticipated when the desired-state declaration was authored. An AI system that responds to these discoveries by autonomously expanding the scope of the upgrade - including additional hosts, modifying the upgrade sequence, or adjusting configuration profiles to accommodate discovered constraints - is operating outside the AAB boundary, regardless of whether its actions are technically correct.

The distinction between technically correct and AAB-compliant autonomous action is crucial. An AI system can make modifications that improve operational outcomes by conventional metrics - shorter upgrade duration, fewer configuration errors, reduced risk of instability - while simultaneously violating the governance boundary by executing actions whose scope was not sanctioned by the human engineer. The engineer's desired-state declaration encoded their understanding of the operational risk at the time of authoring. Autonomous scope expansion, even when technically sound, represents a substitution of the AI system's operational judgment for the engineer's risk assessment, without the engineer's knowledge or authorization.

The practical implementation of AAB governance in SDDC lifecycle management involves several structural components. The first is formal desired-state schema design that makes implicit operational intent explicit. Desired-state declarations in current practice are often partial specifications - they define the target version topology but leave unspecified the scope boundaries, the acceptable failure modes, and the conditions under which the automation should pause and request human input. AAB-compliant desired-state declarations include explicit scope assertions: the set of system components this operation is authorized to touch, the failure conditions under which autonomous execution should stop, and the categories of discovered constraints that require human decision rather than autonomous resolution.

The second is boundary monitoring instrumentation. The lifecycle management platform's event stream must be instrumented to detect and log autonomous actions that approach or cross the AAB boundary - actions that touch system components not listed in the scope assertion, that apply configuration changes not derivable from the desired-state declaration, or that alter operational parameters in response to discovered conditions. These events constitute boundary proximity alerts and should be surfaced to human engineers for review, even when the autonomous action was technically successful.

The third is a structured human re-engagement protocol for boundary proximity events. This protocol addresses the Bainbridge problem directly: operators whose primary contribution to the SDDC lifecycle process is desired-state declaration and boundary alert review maintain the situational awareness and environmental familiarity needed to make competent decisions when boundary alerts escalate. The review protocol is not primarily a control mechanism - most boundary proximity events will be correctly resolved by the autonomous system - but a cognitive maintenance mechanism, ensuring that the human engineers responsible for accountability remain substantively engaged with the operational environment.

6. GOVERNANCE MECHANISMS: OPERATIONALIZING THE AAB MODEL

The AAB model implies four operational governance mechanisms that translate the architectural principle into engineering practice.

Mechanism 1: Explicit Desired-State Governance Schemas

Every AI-managed operational process should be initiated by a desired-state declaration that encodes not only the target configuration but also the governance metadata required for AAB enforcement: scope assertions, authorization scope, failure boundary conditions, and the intent provenance - the operational rationale and risk assessment that motivated the declaration. Governance schemas should be version-controlled, reviewed through change management processes, and treated as first-class engineering artifacts with the same quality assurance requirements as infrastructure code. The Breck et al. (2017) ML production readiness framework provides a useful

analogy: just as production-ready ML systems require explicit monitoring, testing, and validation infrastructure, production-ready AI-managed infrastructure processes require explicit governance schemas.

Mechanism 2: Boundary-Indexed Audit Trails

The operational audit trail for AI-managed infrastructure processes should be indexed against the AAB boundary, not merely against action type or system component. Every logged autonomous action should be tagged with its boundary relationship: within declared scope, boundary proximity, or boundary violation. Boundary proximity and violation events should be escalated for human review regardless of technical outcome. This audit structure enables the retrospective analysis required for incident investigation and provides the empirical data needed to identify patterns of boundary drift over time.

Mechanism 3: Adaptable Automation Controls

Following Calhoun's (2022) distinction, operational control interfaces for AI-managed infrastructure should be designed to preserve operator authority over automation scope, not merely over individual actions. Operators should be able to adjust the authorization scope of an executing automation workflow - expanding or contracting the set of actions the system may take autonomously - without interrupting the execution of authorized actions. This design ensures that discovery of unanticipated dependencies can be handled through rapid scope adjustment by an informed operator rather than either stopping the automation entirely or allowing autonomous scope expansion.

Mechanism 4: Structured Operator Re-Engagement Protocols

Governance systems that rely on human override capacity must invest in maintaining that capacity. The operational model for AI-managed infrastructure should include structured requirements for human operator engagement with the operational environment, distinct from incident response. These may include regular review of boundary proximity event logs, participation in desired-state declaration authoring and review processes, periodic hands-on execution of infrastructure management tasks to maintain environmental familiarity, and scenario-based exercises that develop the diagnostic capabilities required to respond effectively to autonomous system failures. The site reliability engineering discipline has developed analogous practices under the banner of toil reduction balanced against operational ownership (Amershi et al., 2019).

7. EMPIRICAL EVIDENCE AND INDUSTRY OBSERVATIONS

The empirical literature on human-automation interaction in high-stakes operational contexts provides substantial support for the AAB model's premises, though direct studies of AAB governance in cloud infrastructure management contexts remain sparse - a gap this paper identifies as a research priority.

Falowo and Bou Abdo (2026) report empirical findings from a study of automation trust and framework readiness in cybersecurity incident response - a domain with structural similarities to SDDC lifecycle management. Their findings indicate that practitioners demonstrate high trust in automation systems whose decision boundaries are clearly defined and observable, and significantly lower trust in systems whose governance scope is implicit or learned. This trust asymmetry has operational consequences: practitioners who distrust automation boundaries develop compensatory manual oversight practices that consume the operational bandwidth that automation was intended to free, while practitioners who over-trust opaque automation boundaries are vulnerable to governance failures precisely in the high-stakes situations where explicit boundaries matter most.

Cooke (2024) examines the extension of individual human-automation interaction models to sociotechnical systems - the organizational and team-level dimensions of automation governance that are not captured by individual cognitive models. Her analysis finds that governance failures in highly automated systems are often attributable not to individual operator error or AI system malfunction, but to the dissolution of shared situational awareness across the human-automation system boundary. In cloud infrastructure teams, this manifests as a distribution of operational knowledge in which no individual or subgroup maintains sufficient understanding of both the automation system's

behavioral policies and the infrastructure's actual operational state to detect and respond effectively to governance boundary violations.

Wang et al. (2024) demonstrate the operational potential of large language model-augmented autonomous agents for cloud root cause analysis, reporting that their RCAgent system achieves meaningful improvements in diagnostic accuracy and response time compared to rule-based approaches. Their study also surfaces a governance observation that directly motivates the AAB model: the agent's tool-use architecture - its ability to execute diagnostic actions autonomously in response to inferred operational conditions - creates accountability challenges when the agent's diagnostic actions interact with production system state in ways that were not explicitly authorized. The RCAgent paper treats this as an engineering challenge to be addressed through careful tool permission design; the AAB model situates that engineering challenge within a broader governance architecture.

The AIOps literature more broadly documents the pattern this paper has theorized: AI systems that operate effectively within well-defined operational envelopes produce accountability structures that human organizations can sustain, while AI systems that dynamically adapt their operational scope produce governance challenges that outpace organizational response capacity (Li et al., 2022; Notaro et al., 2021). Singh and Szajnfarber (2025) provide a systems engineering perspective on this pattern, arguing that effective human-AI collaboration requires explicit allocation of decision rights between human and machine components - a principle that maps directly to the AAB model's treatment of A_sanctioned and A_governed.

8. DISCUSSION: IMPLICATIONS FOR CLOUD INFRASTRUCTURE ARCHITECTS AND POLICYMAKERS

The AAB model has several implications for the design and governance of AI-managed private cloud infrastructure that extend beyond the SDDC lifecycle management context examined in detail above.

For cloud infrastructure architects, the primary design implication is that governance must be architected into the operational model from the beginning, not added as a control layer after the fact. The desired-state management pattern that the AAB model builds upon is already present in contemporary SDDC platforms; the governance deficit arises not from the absence of appropriate architectural primitives but from the failure to enforce governance discipline at the desired-state authoring and boundary enforcement layers. Architects who treat desired-state declarations as governance artifacts - with explicit scope assertions, version control, review processes, and boundary monitoring instrumentation - are implementing AAB governance without necessarily adopting the AAB nomenclature.

The architectural implications interact with the organizational implications in important ways. The Bainbridge problem - operator skill degradation through disuse - is an architectural consequence as much as an organizational one. Governance architectures that preserve meaningful human engagement with the operational environment, as the structured operator re-engagement protocol described in Section 6 is designed to do, are architecturally encoding the organizational condition necessary for sustained accountability. Organizations that deploy AI-managed infrastructure without investing in these mechanisms are making an architectural bet that the AI system will never fail in ways that require human understanding and intervention - a bet that is not supported by the operational track record of complex systems (Breck et al., 2017).

The agentic AI governance challenge identified by Kshetri (2025) amplifies these implications substantially. Current AI governance discourse has focused primarily on the governance of AI-assisted decision support systems - systems that recommend actions for human approval. The emerging generation of agentic AI systems, including the autonomous LLM agents entering production cloud operations deployments, execute multi-step action sequences in response to high-level objectives without per-action human review. These systems represent a qualitative shift in the governance challenge: the boundaries relevant to governance are now goal-level boundaries (what objectives is this agent authorized to pursue) rather than action-level boundaries. The AAB model's formulation in terms of human-declared intent rather than action type is specifically designed to remain applicable at the goal level: an agentic system

operates within the AAB boundary when its goal pursuit is bounded by the declared desired state, and violates the boundary when it pursues self-defined operational objectives.

For policymakers and standards bodies engaged with AI governance in critical infrastructure contexts, the AAB model suggests that effective governance frameworks should require explicit desired-state declarations for AI-managed operational processes, mandate boundary-indexed audit trails, and establish accountability assignment rules for actions taken within versus outside declared desired states. These requirements are technically tractable, operationally implementable, and directly address the governance failure modes that produce accountability dissolution. They are more targeted than broad explainability requirements, which are technically demanding and often fail to produce the specific accountability information needed for incident investigation and organizational learning.

The MLOps literature has developed complementary governance infrastructure for the machine learning component of AIOps systems. Kreuzberger et al. (2023) provide a comprehensive architecture for ML system governance that includes model monitoring, data validation, and deployment pipeline governance. The AAB model extends this governance architecture to the operational action layer - the layer at which ML system outputs are translated into infrastructure actions - and provides a governance principle that bridges the MLOps and operational governance domains.

9. CONCLUSION

This paper has argued that the governance challenge posed by AI autonomy in private cloud infrastructure is fundamentally an architectural problem, not a calibration problem. The Autonomy-Accountability Boundary model provides a theoretical framework for this architectural approach, grounding it in the desired-state management patterns that already characterize production SDDC environments and connecting it to a rich body of theoretical and empirical work on human-automation interaction, autonomic computing, and AI governance.

The practical implications of the AAB model are not radical departures from current practice. Most enterprise SDDC platforms already implement desired-state management architectures; most operations organizations already maintain change management processes for desired-state declarations. The governance gap that the AAB model addresses is not the absence of these structures but the failure to enforce governance discipline at the boundary between AI-executable action and human-declared intent. Closing this gap requires explicit scope assertion in desired-state declarations, boundary-indexed audit infrastructure, adaptable automation controls that preserve operator authority over automation scope, and structured operator re-engagement protocols that maintain the human competencies required for effective oversight.

As AI systems in cloud infrastructure management continue to gain capability - as autonomous agents, ML-driven policy adaptation, and self-optimizing operational systems become production dependencies - the governance framework proposed here will need to evolve. The AAB model's formulation in terms of human-declared intent rather than action type is intended to provide a governance principle that remains applicable as AI capabilities scale. The fundamental proposition - that autonomy and accountability are structurally coupled, and that their coupling can be architecturally designed and maintained - is as applicable to a future in which AI systems manage cloud infrastructure objectives as it is to the current generation of desired-state automation engines.

The enterprise cloud infrastructure field is at an inflection point. The AI systems entering production today are capable enough to operate with meaningful autonomy in complex environments, and they are being deployed with organizational governance models that were designed for human operators with AI assistance tools. Closing the gap between autonomous capability and governance architecture is not merely a technical problem; it is a design imperative with direct consequences for the accountability, reliability, and trustworthiness of the digital infrastructure on which modern enterprises depend.

REFERENCES

1. Amershi, S., Begel, A., Bird, C., DeLine, R., Gall, H., Kamar, E., Nagappan, N., Nushi, B., & Zimmermann, T. (2019). Software engineering for machine learning: A case study. Proceedings of the 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP 2019), 291–300. https://doi.org/10.1109/icse-seip.2019.00042
2. Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I., & Zaharia, M. (2010). A view of cloud computing. Communications of the ACM, 53(4), 50–58. https://doi.org/10.1145/1721654.1721672
3. Arrieta, A. B., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., Garcia, S., Gil-Lopez, S., Molina, D., Benjamins, R., Chatila, R., & Herrera, F. (2020). Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. Information Fusion, 58, 82–115. https://doi.org/10.1016/j.inffus.2019.12.012
4. Bainbridge, L. (1983). Ironies of automation. Automatica, 19(6), 775–779. https://doi.org/10.1016/0005-1098(83)90046-8
5. Breck, E., Cai, S., Nielsen, E., Salib, M., & Sculley, D. (2017). The ML test score: A rubric for ML production readiness and technical debt reduction. Proceedings of the 2017 IEEE International Conference on Big Data, 1123–1132. https://doi.org/10.1109/bigdata.2017.8258038
6. Calhoun, G. (2022). Adaptable (not adaptive) automation: Forefront of human–automation teaming. Human Factors, 64(2), 269–277. https://doi.org/10.1177/00187208211037457
7. Cooke, N. J. (2024). Expanding human responses to automation failures in sociotechnical systems. Journal of Cognitive Engineering and Decision Making, 18(4), 360–364. https://doi.org/10.1177/15553434241229124
8. Dang, Y., Lin, Q., & Huang, P. (2019). AIOps: Real-world challenges and research innovations. Proceedings of the 41st International Conference on Software Engineering: Companion Proceedings (ICSE-Companion 2019), 4–5. https://doi.org/10.1109/icse-companion.2019.00023
9. Falowo, O., & Bou Abdo, J. (2026). Empirical study on automation, AI trust, and framework readiness in cybersecurity incident response. Algorithms, 19(1), 62. https://doi.org/10.3390/a19010062
10. Hezavehi, S. M., Weyns, D., Avgeriou, P., Calinescu, R., Mirandola, R., & Perez-Palacin, D. (2021). Uncertainty in self-adaptive systems: A research community perspective. ACM Transactions on Autonomous and Adaptive Systems, 15(4), 1–36. https://doi.org/10.1145/3487921
11. Kephart, J. O., & Chess, D. M. (2003). The vision of autonomic computing. Computer, 36(1), 41–50. https://doi.org/10.1109/mc.2003.1160055
12. Kreuzberger, D., Kühl, N., & Hirschl, S. (2023). Machine learning operations (MLOps): Overview, definition, and architecture. IEEE Access, 11, 31866–31879. https://doi.org/10.1109/access.2023.3262138
13. Kshetri, N. (2025). Governing agentic AI: Security, identity, and oversight. Computer, 58(8), 123–129. https://doi.org/10.1109/mc.2025.3572173
14. Li, Y., Zhang, X., He, S., Chen, Z., Kang, Y., Liu, J., Li, L., Dang, Y., Gao, F., Xu, Z., Rajmohan, S., Lin, Q., Zhang, D., & Lyu, M. R. (2022). An intelligent framework for timely, accurate, and comprehensive cloud incident detection. ACM SIGOPS Operating Systems Review, 56(1), 20–28. https://doi.org/10.1145/3544497.3544499
15. Notaro, P., Cardoso, J., & Gerndt, M. (2021). A survey of AIOps methods for failure management. ACM Transactions on Intelligent Systems and Technology, 12(6), 1–45. https://doi.org/10.1145/3483424
16. Parasuraman, R., Sheridan, T. B., & Wickens, C. D. (2000). A model for types and levels of human interaction with automation. IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans, 30(3), 286–297. https://doi.org/10.1109/3468.844354
17. Raisch, S., & Krakowski, S. (2021). Artificial intelligence and management: The automation–augmentation paradox. Academy of Management Review, 46(1), 192–210. https://doi.org/10.5465/amr.2018.0072
18. Saadeh, M. I., Janhonen, J., Beer, E., Castelyn, C., & Hoffman, D. N. (2025). Automation complacency: Risks of abdicating medical decision making. AI and Ethics, 5(6), 5783–5793. https://doi.org/10.1007/s43681-025-00825-2
19. Singh, A. P., & Szajnfarber, Z. (2025). Architecting human-AI systems for effective collaboration and oversight. Systems Engineering, 29(2), 337–353. https://doi.org/10.1002/sys.70024
20. Wang, Z., Liu, Z., Zhang, Y., Zhong, A., Wang, J., Yin, F., Fan, L., Wu, L., & Wen, Q. (2024). RCAgent: Cloud root cause analysis by autonomous agents with tool-augmented large language models. Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM 2024). https://doi.org/10.1145/3627673.3680016