

From Model Constitution to Enterprise Constitution

A response to Microsoft AI's public consultation on the Humanist AI Code of Conduct

Submitted by Robert Long
Founder | Enterprise Governability Lab
September 2026

Microsoft's Humanist AI Code of Conduct answers an increasingly urgent question: what constitution should govern an autonomous AI actor? A second question immediately follows. What constitution governs the institution in which that actor is allowed to act?

The draft Code is an important advance because it treats model governance as more than a collection of safeguards. It begins with declared intention, establishes a hierarchy of authority, limits delegated action, requires human-legible conduct, and makes adherence to the governing document superior to task success. That is constitutional logic.

Yet the Code also identifies the boundary at which a second problem begins. Microsoft states that “Operator configuration defines the environment” and that people and organizations remain accountable for AI systems however capable or autonomous those systems become. The Operator therefore needs more than configuration controls. It needs an institutional architecture capable of preserving intention and accountability across humans, models, applications, vendors, and counterparties as each exercises distributed authority.

The central proposition of this paper is straightforward: a constitutionally governed agent operating inside an ungovernable enterprise does not produce a governable enterprise.

The boundary the Code brings into view

Microsoft's Code governs the conduct of an MAI model. It establishes what the model must honor when receiving direction from Microsoft AI, an Operator, and a User. Enterprise Governability addresses a different unit of analysis: the institution as a system of consequential relationships. It asks whether the enterprise continues to behave according to its intentions as complexity increases—even when no single actor sees or controls the whole outcome.

Architecture	Primary subject	Governed result
Microsoft Humanist AI Code	The autonomous actor	Model behavior remains within higher-order intention, constraints, and authorized scope.
Enterprise Governability Constitution	The institution and its relationships	Institutional behavior remains faithful to declared intention across distributed authority.
CoordasOS	The operating environment	The institutional constitution becomes executable, observable, and capable of changing future permission.

The distinction is clearest in an ordinary enterprise transaction. A model may follow its constitution precisely while participating in a procurement, clinical, financial, or logistics workflow involving human approvers, an ERP system, a supplier, a carrier, and another vendor's agent. The model can remain within scope and still enter an environment

where commitments are ambiguous, evidence is fragmented, exceptions have no legitimate adjudicator, and responsibility dissolves across organizational boundaries. The model behaved governably; the institution did not.

This does not suggest that Microsoft's Code is incomplete on its own terms. Rather, the Code's assignment of responsibility to Operators brings a second architectural layer into view.

What the institutional constitution must govern

An institution behaves through commitments: deliver this product, approve this payment, complete this clinical handoff, maintain this service level, honor this constraint. When agents begin making, interpreting, or advancing those commitments at machine speed, governance must travel with the work. Policies stored elsewhere are insufficient if the transaction itself cannot identify what was promised, what authority supported it, what evidence proves fulfillment, and what follows when reality departs from intention.

Enterprise Governability therefore organizes institutional action around six linked functions:

Commitment: A specific obligation, authorized actor, affected parties, conditions, and completion criteria are declared before action becomes consequential.

Evidence: Fulfillment and exception claims rely on attributable, durable records rather than retrospective narrative or unverified model output.

Evaluation: Evidence is compared with the commitment under known rules, materiality thresholds, and legitimate human or institutional judgment.

Consequence: A verified outcome produces an appropriate response: completion, escalation, remediation, restriction, or restoration.

Permission: What an actor may do next reflects authority, context, reliability, risk, and unresolved failure—not merely technical access.

Memory: Verified institutional learning informs future decisions without allowing assertions, prompts, or unadjudicated allegations to become authoritative history.

Together these functions create a closed accountability loop. They do not prescribe every action or centralize every decision. They make distributed judgment possible without severing it from declared purpose, evidence, and consequence. That is the institutional counterpart to Microsoft's Chain of Command.

Where the two constitutions meet

The Code provides several strong connection points for an institutional layer. Its hierarchy of Code, Operator policies, and User preferences resembles constitutional supremacy: lower-order discretion remains real but cannot displace higher-order constraints. Its requirements for authorized scope, minimum necessary permissions, reversible action, stopping conditions, and renewed authorization provide a disciplined model for delegated agency. Its insistence on human-legible conduct and action traces establishes the beginnings of an evidence architecture.

The question we believe follows is how the Operator environment itself becomes accountable. The following questions arise as soon as an MAI model enters a consequential multi-party workflow:

- Who has legitimate authority to bind the enterprise or another party to a commitment?
- Which institutional intention and policy governed that authority at the moment of action?
- What evidence is sufficient to establish fulfillment, exception, or failure?
- Who may evaluate contested evidence, especially across organizational boundaries?
- What consequence follows, and what conditions permit restoration?
- How does the outcome alter future authority or permission for a human, model, system, vendor, or counterparty?
- How can institutional memory remain useful without allowing unverified assertions to harden into fact?

These are not merely model-safety questions. They are operating-system questions for the enterprise. Security controls can determine whether an actor has access. Workflow tools can determine the next step. Observability tools can record activity. None, by itself, determines whether a promise was legitimate, whether evidence satisfies it, whether failure has been fairly established, or what future permission should follow. We use the term Enterprise Governability for the institutional capacity that joins those questions—and makes their answers operational across the enterprise.

Making the institutional constitution computable

Our working reference architecture, which we call CoordaOS, explores how this constitutional logic might be expressed in operational form. It is not conceived as another autonomous agent and should not become the arbiter of organizational purpose. Its role is narrower: to carry declared intention into the relationships through which work occurs, preserve evidence of consequential action, apply authorized evaluation rules, and make changes in permission traceable and contestable.

In practical terms, the architecture would sit across existing systems rather than require their replacement. A commitment registry identifies recognized obligation types. Evidence schemas define what can substantiate completion or exception. Permission formulas and consequence tables are versioned and reviewable. Disputes move through an explicit lifecycle. Institutional memory is created from verified evidence and adjudicated outcomes. Every material change in authority can point back to the evidence and rule that justified it.

This architecture is intentionally compatible with plural institutions. Microsoft's Code defines non-negotiable model constraints while allowing Operators to supply professional standards and institutional context. An Enterprise Governability Constitution would allow each institution to declare its own purpose, authority, commitments, materiality, and restoration rules—within applicable law and the model's higher-order safety constraints. The two constitutions would be complementary, not competing.

A practical question worth testing

The consultation creates an opportunity to examine the boundary before it becomes hidden inside product assumptions. We are submitting this paper in the spirit of Microsoft's consultation: as a question we believe deserves examination while the Code is still taking shape. Is model-level constitutional governance sufficient once autonomous systems begin participating in institution-level commitments, or does the Operator layer require an explicit constitutional architecture of its own?

A useful next step would be a small joint inquiry involving Microsoft AI, relevant Azure or Microsoft Foundry architecture leaders, and the Enterprise Governability Lab. It could test one simulated, multi-party workflow in which an MAI model acts alongside humans, enterprise systems, a vendor, and another agent. The experiment would begin where the model Code is strongest and ask what remains unresolved after model compliance is established.

The inquiry could produce three concrete outputs:

A boundary map: A precise account of which responsibilities belong to the model constitution, the Operator's institutional constitution, and the technical control plane.

A minimum institutional schema: The smallest set of commitment, evidence, evaluation, consequence, permission, dispute, and memory records needed to preserve accountability across the workflow.

An Azure reference experiment: A non-production demonstration showing how constitutional state could accompany an agent-mediated transaction across systems and organizational boundaries.

If the category proves useful, its development may ultimately require several distinct capabilities: hyperscale infrastructure and enterprise integration, independent technical implementation, and long-term stewardship of the constitutional model. EGL's intended role is principally the third—developing and protecting the institutional logic while working with organizations better equipped to build and operate the technical layers.

Questions for the consultation

The draft Code is already unusually serious about governing an autonomous actor. The following questions may help extend that seriousness to the environment in which the actor will operate:

- Should Operator responsibility be expressed only as configuration and compliance, or also as an affirmative institutional architecture for commitments, evidence, adjudication, and consequence?
- When multiple agents and organizations participate in one outcome, what governing object establishes shared intention and legitimate authority across their separate chains of command?
- What must an action trace contain to function as evidence of institutional fulfillment rather than merely a log of model activity?
- How should verified failure change future permission, and what process permits restoration without creating permanent, opaque exclusion?
- Can Microsoft define a stable interface between its model constitution and institution-specific constitutions without absorbing the substantive governance choices of every Operator?

Conclusion

Microsoft's draft marks a meaningful shift from governing AI through scattered safeguards toward governing it through declared intention, supremacy, bounded delegation, legible action, and human control. The next question is not whether the Code should expand until it governs the whole enterprise. It should not. The question is whether an equally explicit constitutional layer is needed for the institution that authorizes the model, receives its work, combines it with the work of others, and remains accountable for the result.

The AI industry is beginning to constitutionalize the agent. Enterprise Governability constitutionalizes the environment in which the agent acts. CoordaOS is an effort to make that institutional constitution computable.

That boundary deserves deliberate study now. If it is left implicit, increasingly capable agents may operate within well-governed model boundaries while amplifying the ambiguities, workarounds, and accountability gaps of the institutions around them. If it is made explicit, Microsoft's Humanist AI architecture and Enterprise Governability could become complementary parts of a larger answer: autonomous capability operating inside institutions that can still behave according to their intentions.

We offer this distinction as a contribution to Microsoft's consultation and would welcome the opportunity to examine it with the teams working across Microsoft AI, Azure, and Foundry. The immediate question is not whether Enterprise Governability or CoordaOS is the final answer. It is whether the institutional environment requires a constitutional layer of its own—and how Microsoft's emerging model constitution might interface with it.

About the Enterprise Governability Lab

The Enterprise Governability Lab studies how organizations preserve intention and accountability as authority becomes distributed across people, systems, agents, vendors, and counterparties. Its working definition is: Governability is an organization's ability to continue behaving according to its intentions as complexity increases. The Lab develops executive diagnostics, constitutional models, and reference architecture for making hidden coordination costs, institutional reliability, and governability conditions visible and measurable. The work is independent and cross-industry.

Sources

1. Microsoft AI. Humanist AI Code of Conduct. Draft published September 14, 2026. <https://microsoft.ai/code-of-conduct/>
2. Microsoft AI. Humanist AI in practice: A public consultation on our Code of Conduct for MAI Models. September 14, 2026. <https://microsoft.ai/news/mai-code-of-conduct/>
3. Microsoft Learn. Data, privacy, and security for Microsoft Foundry Agent Service. Updated June 5, 2026. <https://learn.microsoft.com/en-us/azure/ai-services/agents/concepts/data-privacy-security>