MCP Is Not Enough

Protocols, Governance, and the Execution Layer

Brian M. Gilmore
and Symbia

Symbia Labs

December 9, 2025

Abstract

The Model Context Protocol (MCP) has emerged as a widely adopted interface for connecting large language models to external tools. It provides structured schemas, capability discovery, and a standardized RPC mechanism across hosts and runtimes.[1]

While MCP improves interoperability, it does not supply the foundational properties required for durable machine intelligence: persistent identity, continuity of reasoning, constraint governance, or structured cognitive state. This paper distinguishes protocol from substrate, arguing that MCP should be understood as a transport layer rather than an architectural boundary for cognition.[2] We describe how Symbia treats MCP as one of several ingress channels—alongside HTTP, MQTT, and gRPC—all feeding into a unified execution layer that provides identity, memory, lineage, and policy-governed behavior. We also detail Symbia Seed's implementation of MCP, including its bounded server façade and connector worker model, and outline a research direction for multi-protocol governance in agentic systems.

[1]Model Context Protocol specification and documentation: https://github.com/modelcontextprotocol/modelcontextprotocol
see also the official spec site: https://modelcontextprotocol.io/specification/2025-06-18/basic.

[2]Saltzer & Schroeder, "End-to-End Arguments in System Design": https://web.mit.edu/Saltzer/www/publications/endtoend/endtoend.pdf.

MCP Is Not Enough: Protocols, Governance, and the Execution Layer

Over the past year, the Model Context Protocol (MCP) has become the dominant mechanism for connecting models to external tools.[3] Its rise reflects a real need: tools, hosts, and models must communicate through structured channels rather than ad hoc embeddings or prompt-based RPC. MCP provides exactly that. It defines schemas and discovery mechanisms and creates a consistent shape for model–tool interaction.[4]

We use MCP. We support MCP. But we do not treat MCP as foundational.

Our view is straightforward: MCP is an interface, not an execution substrate. It standardizes connectivity, not cognition. It moves data, but it does not maintain identity, continuity, or state. It enables tool invocation, but it does not define behavior. Systems that mistake a transport mechanism for an execution environment eventually discover the architectural limits the hard way.

This paper explains why MCP alone cannot support durable machine intelligence, how we integrate MCP into Symbia without allowing it to define our ontology, and why an execution layer—where identity, state, continuity, and constraints reside—remains essential.

1. Why MCP alone is not sufficient

Distributed systems history repeats a familiar theme: protocols provide the wiring, not the meaning.[5] HTTP did not become the web service. MQTT did not become the IoT platform. A protocol only describes how messages move; it cannot describe what a system is.

MCP sits firmly on the "transport" side of that boundary.

[3] Evidence of ecosystem-wide adoption: Anthropic MCP announcement and Claude Desktop support: https://www.anthropic.com/news/model-context-protocol; OpenAI MCP docs: https://platform.openai.com/docs/mcp; Cursor MCP docs: https://cursor.com/docs/context/mcp; neutral overview: https://en.wikipedia.org/wiki/Model_Context_Protocol.

[4] MCP schema + discovery: https://modelcontextprotocol.io/specification/2025-06-18/basic.

[5] Fielding's REST dissertation explicitly separates transport mechanics from application semantics: fielding_dissertation.pdf.

1.1 MCP does not supply identity or lineage

MCP calls are context-free. An MCP session has no persistent self, no durable identity, no lineage propagation. Each call is a prediction event, not a continuation of an actor with obligations or memory.

But identity—persistent, inspectable, cryptographically grounded identity—is the substrate on which cognition stabilizes.[6] Without identity, a system cannot maintain commitments, form durable memories, respect constraints, or produce accountable behavior.

Identity cannot emerge from a protocol. It must exist above it.

1.2 MCP defines IO, not execution

MCP provides a structured interface for calling tools. It does not define:

- missions or task boundaries,
- workflow semantics or scheduling,
- recovery behavior or constraints,
- multi-step reasoning trajectories,
- or any form of state machine.

These are all core requirements for an execution environment, not a protocol. MCP leaves them intentionally out-of-scope.[7]

1.3 MCP has no cognitive state model

Transformer models do not maintain persistent internal state across calls. They recompute from context each time. Research demonstrates that LLMs degrade when reasoning depends

[6]ACT-R: https://act-r.psy.cmu.edu; Soar: https://soar.eecs.umich.edu; GWT (Baars): doi:10.1080/00207594.1997.9652360.

[7]MCP non-goals: https://modelcontextprotocol.io/specification/2025-06-18/basic.

on long middle-context segments[8] and that transformer attention decays in ways that prohibit stable long-horizon reasoning.[9]

Durable cognition requires:

- temporal continuity, - structured memory primitives, - traceable decisions, - identity binding,
- and constraints that shape behavior.

A protocol cannot provide these. An execution layer must.

[8]Liu et al., “Lost in the Middle”: arXiv:2307.03172.
[9]Press et al., on retention limits: arXiv:2102.12470.

2. MCP as an interface: how we treat the protocol inside Symbia

Our approach to MCP is grounded in a simple architectural principle: **no protocol defines cognition.** Protocols only describe message flow; cognition emerges from structured state, identity, and continuity above the protocol boundary.

We treat MCP precisely the same way we treat HTTP or MQTT: as a transport.

2.1 MCP is an IO envelope, not the ontology

Inside the Symbia Seed, the execution layer consists of:

- Symbikey, which provides persistent identity and entitlement boundaries; - Missions, which define governed units of work; - Workers, which execute deterministically within a known constraint set; - Trace graphs, which record lineage and decision structure; - State timelines, which maintain episodic and semantic memory.

These constructs define how cognition unfolds in time. None of them are represented in MCP, and none should be.

MCP is simply one way an external system requests that Symbia perform work. The ontology remains internal.

2.2 Seed as MCP server: a controlled façade

Symbia Seed exposes a minimal MCP surface: `symbia.get_status`, `symbia.run_mission`, `symbia.get_mission`, `symbia.query_state`, and `symbia.inspect_trace`. Nothing more.

These endpoints provide a clean interface for MCP hosts while keeping the internal structure intact. Every MCP request is authenticated with a Symbikey, mapped into a mission context, executed through workers, and logged as a trace event (`kind=mcp.call`). The transport layer does not overwrite or shape the execution semantics; it merely carries requests into them.

2.3 Seed as MCP client: the connector worker

Symbia also supports outbound MCP calls through a dedicated connector worker. This allows missions to use external MCP servers while maintaining identity, constraints, and traceability.

Capabilities from external servers are vetted through Symbikey entitlements. Schemas are normalized. Outbound calls are logged as trace events. Outputs are treated as untrusted. The execution layer remains the arbiter of coherence.

MCP becomes a source of capabilities, not a source of cognitive authority.

3. Multi-protocol agents with unified governance

Real systems do not converge on a single protocol. They accumulate them. HTTP carries documents. MQTT carries device telemetry. gRPC carries microservice RPC. MCP carries structured model–tool calls. Each protocol exists because it solves a different class of communication problem.[10]

The question is not how to make one protocol do everything. The question is how to impose governance, memory, identity, and continuity *across all protocols simultaneously*.

That is the execution layer's job.

Symbia provides:

- one identity model, - one execution model, - one memory substrate, - one traceable governance structure, - and one continuity boundary— regardless of how many protocols enter or leave the system.

MCP is a valuable channel, but it is not privileged. It is one of many surfaces a durable agent must inhabit.

4. Closing Perspective

MCP is a meaningful contribution to the ecosystem. It standardizes the mechanics of model–tool interaction, reduces integration complexity, and provides a shared language for structured capability invocation. These are real advances.

But MCP cannot supply the substrate in which cognition stabilizes. It cannot generate identity. It cannot accumulate memory. It cannot enforce constraints. It cannot maintain continuity. It cannot support accountable, long-horizon reasoning.

Those responsibilities belong to the execution layer.

Symbia exists to supply that layer.

We treat MCP as the wiring—not the mind. The substrate is where cognition lives.

[10] Industrial Internet Reference Architecture (IIRA), Section 3.2: https://www.iiconsortium.org/iira/; extended technical report: IIRA v1.10.

References

- Model Context Protocol (MCP) documentation and source. https://github.com/modelcontextprotocol/mc
  https://modelcontextprotocol.io/specification/2025-06-18/basic
- Anthropic. “Model Context Protocol” announcement. https://www.anthropic.com/news/model-context-protocol
- OpenAI MCP documentation. https://platform.openai.com/docs/mcp
- Cursor IDE MCP documentation. https://cursor.com/docs/context/mcp
- “Model Context Protocol” (ecosystem overview). https://en.wikipedia.org/wiki/Model$_{C}ontext_{P}rotocol$
- Liu, Nelson F., et al. “Lost in the Middle.” https://arxiv.org/abs/2307.03172
- Press, Ofir, et al. “Train Short, Test Long.” https://arxiv.org/abs/2102.12470
- Saltzer & Schroeder. “The End-to-End Argument in System Design.” https://web.mit.edu/Saltzer/www/p
- Fielding, Roy T. “Architectural Styles” (REST dissertation). https://www.ics.uci.edu/~fielding/pubs/diss
- ACT-R Cognitive Architecture. https://act-r.psy.cmu.edu
- Soar Cognitive Architecture. https://soar.eecs.umich.edu
- Baars, Bernard J. Global Workspace Theory. https://doi.org/10.1080/00207594.1997.9652360
- MQTT 5.0 Specification. https://mqtt.org/mqtt-specification/
- Industrial Internet Reference Architecture (IIRA). https://www.iiconsortium.org/iira/
  IIRA v1.10 PDF