Why Scaling Models Is Not Enough

The Case for Organizational Depth in Agent Architecture

By Hadi Nayebi

Hadosh Academy — Agents Series, Essay 2 (Companion White Paper)

March 2026 (Revised)

Abstract

This paper argues that achieving long-horizon, self-improving artificial agency requires increasing organizational depth — the number of interacting, persistent subsystems — at least as much as it requires increasing model capacity. We draw on complex systems theory, modular design principles, and biological information theory to propose a framework for evaluating agent architectures. Every major claim is labeled: **[REF]** for externally supported claims with citations, **[SPECULATIVE]** for reasonable extrapolations beyond current evidence. Unlabeled content represents proposals, experimental designs, definitions, or analogies. A references section provides full citations.

1. Introduction

This white paper accompanies the blog essay “We Could Have Had AGI By Now.” The blog makes the case in accessible terms. This paper provides the analytical framework, labeled claims, falsifiable predictions, and a practical experiment for builders who want to test the hypothesis in their own work.

The central question is straightforward: what kind of complexity does long-horizon agency require? We argue that the field's dominant strategy — scaling model parameters and training data — addresses one kind of complexity (internal function complexity) while largely neglecting another (organizational system complexity). Both are necessary. Neither alone is sufficient.

2. Two Kinds of Complexity

When people say "complex system," they often conflate two fundamentally different meanings.

Internal complexity (function complexity)

[REF] A trained transformer is a high-dimensional, nonlinear function exhibiting emergent behavior at scale (Wei et al., 2022). Scaling laws describe predictable relationships between model size, data, compute, and loss (Kaplan et al., 2020; Hoffmann et al., 2022). These are genuine achievements in internal function complexity — the model can approximate an enormous range of computations.

Organizational complexity (system complexity)

[REF] Herbert Simon's foundational analysis of complex systems distinguishes between monolithic complexity and hierarchical, nearly-decomposable complexity (Simon, 1962). In nearly-decomposable systems, subsystems interact strongly within their own boundaries and weakly across boundaries. This structure enables independent modification of parts without destabilizing the whole — a property Simon identified as essential for the evolution and persistence of complex systems.

[REF] Stuart Kauffman's work on self-organization extends this: complex adaptive systems — from cells to ecosystems — organize through interacting subsystems with differentiated roles, regulatory feedback, and multi-timescale dynamics (Kauffman, 1993). Complexity that persists is not merely complicated; it is structured.

[SPECULATIVE] A modern LLM possesses the first kind of complexity in abundance but the second kind only weakly. A single model has no durable identity outside its context window. Its "state" is the prompt. It

cannot selectively update one skill without retraining the whole. It lacks the nearly-decomposable structure that Simon identified as the hallmark of evolvable complex systems.

3. Why “Turing Complete” Misses the Point

[REF] Neural networks can emulate arbitrary computation in principle — this follows from universal approximation theorems (Hornik et al., 1989; Cybenko, 1989). But “in principle” is not the engineering target. The target is a practical substrate for systems that:

- Keep working after the session ends
- Improve from experience without retraining
- Remain coherent across months
- Can be inspected, updated, and governed by their operators

[REF] This parallels a well-understood principle in software engineering: monolithic systems can theoretically do anything modular systems can, but modular architectures are preferred because they are easier to maintain, test, extend, and evolve (Baldwin & Clark, 2000; Parnas, 1972). The advantage is not theoretical capability — it is practical evolvability.

[SPECULATIVE] A single monolithic policy can simulate many modules internally, but that simulated modularity is hard to inspect, hard to independently update, and hard to evolve without interference. This is why evolvable systems — biological and engineered — consistently develop modular structure as they grow in complexity.

4. Information Metabolism: From Context to Knowledge

Consider an agent at time *t*. The model has a context window — its working state. Now ask: where did that state come from?

Generated state: tokens produced by the model in the current session.

Injected state: content inserted by architecture — persistent files, regulatory mechanisms, policy constraints, tool results, retrieved memory, schemas, logs.

[SPECULATIVE] A model-centric worldview pushes toward maximizing generated state: let the model derive everything from scratch. An architecture-centric worldview pushes toward increasing injected state: keep durable structure outside the model; treat the model as one cognitive module among several.

Injected state enables persistence across sessions, selective retrieval, deterministic regulation, and localized upgrades. Generated state alone creates a closed loop. Closed loops can be clever, but they tend toward instability under long horizons.

4.1 Injected Context Ratio

[SPECULATIVE] If you want one practical metric to track whether a system is building organizational structure or conducting a monologue, use a rough injected context ratio:

> **Injected ratio = (tokens from files, tools, regulatory mechanisms, memory) / (total context tokens)**

A low ratio means the system mostly regenerates state internally; long-horizon behavior depends on fragile self-consistency. A higher ratio means the system depends on stable external anchors; behavior becomes auditable and stabilizable. Over time, an architecture-centric system should show more injected context from durable structures, less repeated re-derivation of the same constraints, and fewer failures from forgotten steps.

4.2 The Central Dogma of Agent Information

[REF] In molecular biology, Crick's central dogma describes the directional flow of genetic information: DNA → RNA → Protein (Crick, 1970). DNA stores the blueprint. RNA transcribes it into working form. Proteins execute the functions. The information flows in a defined direction, with regulatory mechanisms controlling what gets expressed and when.

[SPECULATIVE] An analogous information flow operates in well-structured agent architectures:

> **Persistent files → Context window → Actions → Persistent files**

Persistent files (instruction files, memory stores, skill libraries) serve as the agent's hereditary material — encoding identity, rules, and accumulated knowledge. The context window is the active workspace

where this hereditary information is transcribed alongside immediate inputs. The model's output — its actions and reasoning — is the functional expression. And critically, the cycle completes: experience from actions flows back into persistent files through consolidation processes, analogous to how epigenetic modifications update the regulatory landscape of a genome without rewriting the genome itself.

[SPECULATIVE] This is what we call **information metabolism**: the continuous cycle of absorbing tokens into durable memory, recalling them when needed, and refining the persistent structures that govern future behavior. A well-metabolizing system grows more capable over time without retraining its core engine. A poorly-metabolizing system loses information between sessions and must regenerate context from scratch — the equivalent of an organism that cannot form long-term memories.

5. The Random Walk Problem in Agent Behavior

A bare agent has an action space: reply, reason, call a tool, request permission, delegate, compact context, stop. With minimal organizational structure, the model chooses among these actions probabilistically. Over time, the resulting action chain is sensitive to small context variations. The same task can produce different sequences on different runs.

[REF] This is not a failure of the model. It is a well-understood property of stochastic processes: without regulatory constraints, probabilistic systems exhibit high variance over long horizons. Organizational theory addresses this through processes, checklists, roles, gates, and enforcement — reducing variance without eliminating adaptability (Simon, 1947; March & Simon, 1958).

[SPECULATIVE] In agent architectures, regulatory mechanisms — lifecycle interception, instruction files, validation gates — serve the same function: they reduce the variance of probabilistic behavior to tolerable levels. They are the earliest form of organizational regulation applied to AI agents.

6. The Core Hypothesis

[SPECULATIVE] To reach long-horizon, self-improving competence, systems must increase organizational depth — the number of interacting persistent subsystems — at least as much as they increase model capacity.

Organizational depth includes:

- Persistent memory stores with defined schemas
- Regulatory mechanisms that intercept lifecycle events
- Obligation-tracking systems that enforce completion
- Verification and rollback mechanisms
- Multi-timescale learning loops (active work vs. consolidation)
- Modular capability libraries

[SPECULATIVE] If this hypothesis holds, then model scaling alone can keep raising ceiling performance on isolated benchmarks, but it will not produce the stability and evolvability that sustained, real-world labor replacement requires.

7. What Makes Systems Evolvable

[REF] Drawing on Simon (1962), Baldwin & Clark (2000), and biological systems theory (Bonner, 1988), we identify seven properties of organizationally complex systems:

Property	In Organizations	In Agent Architecture
Hierarchy	Subsystems inside subsystems	Nested instruction layers and scoped directories
Boundaries	Clear interfaces, scoped context	Compartmentalized files and scoped memory
Persistence	State survives across interactions	Durable stores (knowledge, logs, obligation records)
Regulation	Automatic reflexes, compliance	Lifecycle interception at defined events
Multi-timescale	Fast actions, slow restructuring	Active work loops vs. consolidation loops
Local modification	Improve one part, keep rest stable	Edit one capability module without affecting others
Selection		

Property	In Organizations	In Agent Architecture
	Keep what works, retire what fails	Measurable evaluation and promotion/retirement

[SPECULATIVE] A transformer can be internally complex, but it does not naturally expose these organizational levers. A monolithic model has one primary way to change: adjust weights (via training) or adjust prompt (via context). Both are global interventions. Neither is a local, modular edit with stable boundaries — the kind of modification that Simon identified as essential for evolving complexity.

8. The Case for External Structure

[REF] Even if a sufficiently large model could internalize the equivalent of regulatory mechanisms, memory routing, and policy enforcement, external organizational structure retains practical advantages identified in modular design theory (Baldwin & Clark, 2000; Parnas, 1972):

- **Replaceability:** swap the engine without rewriting identity — analogous to how organisms can adapt to dietary changes without rewriting their genome
- **Local change:** modify one regulatory rule without retraining the entire system
- **Inspectability:** read the policy; audit the decision — the organizational structure is transparent
- **Safety:** enforce constraints at the regulatory layer, independent of the model's probabilistic outputs
- **Incremental growth:** add a new capability module without destabilizing existing ones

These are not aesthetic preferences. They are engineering properties of evolvable systems.

[SPECULATIVE] The economic argument reinforces this: organizational improvements are typically cheap, deterministic, and compositional. A verification checklist costs almost nothing. A resource-tracking rule costs almost nothing. A permission gate costs almost nothing. If these prevent even a small fraction of failures, they dominate marginal gains from model scaling in many real deployments — particularly those requiring sustained operation over weeks or months.

9. A Taxonomy of Organizational Roles

Instead of prescribing specific implementations, we define a minimum set of roles a system must perform to exhibit organizational depth. The specific mechanisms are less important than the differentiation itself:

Role	Function	Biological Analogy
Scheduler	Manage obligations, priorities, resources	Metabolic regulation — controlling which processes run when
Planner	Decompose objectives into manageable units	Executive function — breaking goals into actionable steps
Executor	Interface with external tools and environments	Motor system — translating intention into action
Verifier	Validate outputs against criteria	Immune system — detecting and rejecting malformed outputs
Memory Manager	Store, retrieve, and scope information	Hippocampal system — encoding and retrieving experiences
Consolidator	Distill experience into durable forms	Sleep consolidation — converting short-term traces into long-term structure
Critic	Detect recurring failure patterns	Pain signaling — identifying what is not working
Capability Builder	Create new workflows from experience	Neuroplasticity — forming new circuits from repeated patterns
Watchdog	Detect runaway behavior and resource exhaustion	Homeostatic regulation — maintaining stable operating parameters

[SPECULATIVE] An agent that performs all of these roles through a single undifferentiated prompt is functionally less organized than one that separates them into distinct, inspectable components — regardless

of the specific technology used to implement the separation. The argument is for differentiation as a principle, not for any particular implementation of it.

10. Architecture Patterns

One pattern that embodies these principles is a system of **tracked obligations**:

- Obligations exist as structured, persistent objects
- Each obligation has: an objective, success criteria, a resource budget, a status, and an observation log
- Regulatory mechanisms prevent the system from abandoning active obligations
- An obligation cannot be marked complete until its observation log has been distilled into durable memory

This pattern forces the system to behave like an organization: commitments cannot be ignored, experience cannot be left unprocessed, and progress has a verifiable lifecycle.

[SPECULATIVE] The same pattern can be expressed in different formalisms: as tracked jobs, as intentions composed of behaviors, as goals in a planning language. The representation varies. The organizational principle — that obligations have lifecycles and completion requires consolidation — is the invariant.

10.1 Governance Without Freezing the System

If the architecture can modify itself, how do you prevent runaway drift? Separate the ability to propose changes from the ability to deploy them.

[REF] This mirrors the two-phase commit pattern in distributed systems (Gray, 1978): propose changes to a staging area, evaluate them against defined criteria, and promote them only if they improve outcomes without violating constraints. This does not require an immutable constitution. It requires a deployment gate.

[SPECULATIVE] In agent architectures, this translates to a system where self-modifications (new capabilities, updated rules, restructured memory) are staged, tested against verification criteria, and promoted only if they demonstrate improvement — analogous to how biological mutations undergo selection before being propagated.

11. The Biological Lens: Agents as Cognitive Organ Systems

Analogies are useful when they illuminate structure, dangerous when they pretend to be proofs. We use biological parallels as a lens, not a claim of equivalence.

11.1 From Molecules to Organ Systems

[REF] Early life likely involved molecules that both stored information and performed catalytic functions — the RNA World hypothesis (Gilbert, 1986). Over evolutionary time, biology separated these concerns: DNA for storage and heredity, RNA for transcription, proteins for execution, membranes for compartmentalization, signaling molecules for regulation, and eventually specialized tissues and organ systems (Bonner, 1988).

[SPECULATIVE] The lesson is not that any particular molecule was insufficient. The lesson is that higher-order capability required new organizational layers — each layer introducing a new class of mechanism:

- **Hereditary material** — persistent information stores encoding identity and accumulated knowledge (analogous to DNA and chromatin)
- **Regulatory mechanisms** — systems that control what gets expressed and when, providing deterministic governance over probabilistic processes (analogous to restriction enzymes, promoters, and epigenetic marks)
- **Compartmentalization** — boundaries that scope information and prevent interference between subsystems (analogous to cell membranes and organ boundaries)
- **Metabolic cycles** — continuous processes that convert raw inputs into structured, usable forms (analogous to metabolic pathways)
- **Cognitive memory tissue** — persistent structures that accumulate and consolidate experience over time (analogous to long-term potentiation and synaptic remodeling)
- **Consolidation and remodeling** — periodic restructuring that converts short-term experience into durable upgrades (analogous to sleep-dependent memory consolidation)

[SPECULATIVE] Note what this framing does NOT prescribe: it does not name specific implementations. It identifies classes of mechanisms that complex adaptive systems appear to require. Just as biology

benefits from having restriction enzymes as a class — sequence-specific regulatory controls — without requiring any particular restriction enzyme (EcoRI is one of thousands), agent architectures benefit from having regulatory lifecycle mechanisms as a class, without requiring any particular implementation.

11.2 The Regulatory Landscape

[REF] As of early 2026, multiple independently developed CLI agent platforms have converged on lifecycle interception mechanisms — systems that intercept and regulate key events in the agent's operation (tool use, session start/stop, context compaction, delegation). This convergence across independently developed platforms suggests an emergent recognition of the need for regulatory layers.

[SPECULATIVE] The convergence itself is evidence for the organizational depth hypothesis: independent development teams, solving practical reliability problems, arrive at architecturally similar solutions. This pattern — convergent evolution toward regulatory mechanisms — mirrors how biological lineages independently evolve similar functional structures (eyes evolved independently over 40 times across species; wings evolved independently in insects, birds, and bats).

[REF] Not all platforms have adopted lifecycle interception. Some have explicitly declined community proposals for such mechanisms, maintaining a model-centric approach that relies on the model's internal coherence rather than external regulation. This divergence provides a natural experiment: platforms with regulatory mechanisms versus those without, operating on the same underlying model capabilities.

12. Falsifiable Predictions

[SPECULATIVE] The following predictions are testable consequences of the organizational depth hypothesis. If they fail, the hypothesis is weakened.

Prediction A: A smaller model with strong organizational depth will outperform a larger model with weak organization on long-horizon tasks (measured over weeks, not minutes), at lower total cost.

Prediction B: Systems that externalize memory and verification into persistent structures will show measurable improvement over time without retraining the core model — demonstrated by decreasing error rates and increasing first-attempt success rates across sessions.

Prediction C: The injected context ratio (Section 4.1) will correlate negatively with cross-run variance — systems with higher injected ratios will produce more consistent behavior across independent runs of the same task.

These are measurable. You can run two systems for months: one that mostly regenerates state within the model, one that persists state and uses regulatory mechanisms to enforce lifecycles. Compare outcomes.

13. A Practical Experiment for Builders

If you want to test the organizational depth hypothesis in your own work, here is a controlled experiment:

Setup

Pick a long-horizon project — something that takes weeks, not minutes. A multi-phase research project, a software build with iterative requirements, a content production pipeline with multiple deliverables. Run it twice:

Version A (Minimal Structure): Use an agent with minimal organizational scaffolding. Rely on conversational interaction and ad-hoc prompting. No persistent obligation tracking, no lifecycle regulation beyond defaults, no consolidation phases, no compartmentalized memory. The model generates and regenerates its own context each session.

Version B (Organizational Depth): Use the same agent (same underlying model) with explicit organizational layers: persistent instruction files defining operational phases; regulatory mechanisms enforcing lifecycle constraints; a persistent obligation-tracking system; compartmentalized memory and knowledge files; and scheduled consolidation phases where traces are analyzed and distilled into durable upgrades.

Metrics to Track

Metric	What It Measures	Expected Result
Retries per milestone	Reliability of first-attempt execution	Version B: fewer retries
Missed requirements	Completeness of output	Version B: fewer misses
Cost/time per milestone	Efficiency over long horizons	Version B: lower after initial setup
Context restatements	How often you re-explain the same thing	Version B: near zero after week 1
Cross-run variance	Consistency of approach across sessions	Version B: lower variance
Injected context ratio	Architecture maturity (see Section 4.1)	Version B: increasing over time

Expected Outcome

In most real projects, Version B wins even with a weaker model. The initial setup cost is higher, but it amortizes as the agent accumulates structured experience. By week two or three, Version B should require significantly fewer human interventions per milestone, show more consistent output quality, and cost less per unit of useful output.

If Version A wins — that is, if the unstructured approach is cheaper, more reliable, and more consistent over the full project duration — then the organizational depth hypothesis is weakened for that class of task. Report the result either way. The point is measurement, not advocacy.

14. Common Objections

"A huge model with long context solves memory." [REF] Long context reduces some retrieval friction, but it does not create selection, boundaries, or modular upgrades. It expands the working space — a bigger desk. It does not create organizational structure — filing cabinets, policies, or quality controls. **[SPECULATIVE]** The distinction between workspace size and organizational depth is fundamental to the hypothesis.

“Mixture-of-Experts models are modular already.” [REF] MoE is internal specialization within the weight space (Shazeer et al., 2017). It improves efficiency and can enable some functional differentiation. **[SPECULATIVE]** But it is not externally governed — the operator cannot inspect, modify, or selectively update individual experts. The modularity exists inside the model, where it serves computational efficiency rather than organizational evolvability.

“External structure is just scaffolding.” Scaffolding implies temporary support that gets removed. **[SPECULATIVE]** The more realistic view is that organizational structure is not scaffolding — it is skeleton. The structures that enable the organism to function are not removed once the organism matures. They become more sophisticated.

“Bigger models will internalize architecture.” [SPECULATIVE] Even if internalization is possible in principle, it sacrifices the properties that make external organization valuable: inspectability (you cannot read the internalized policy), replaceability (you cannot swap the engine without retraining), local modification (you cannot edit one internalized rule), and cost efficiency (you are paying model-scale compute for what deterministic software can provide cheaply).

15. The Economic Dimension

[SPECULATIVE] If the cost of cognitive delegation continues to fall along current trend lines — and the trajectory from 2023 to 2026 suggests it will — then the economic implications extend beyond individual productivity.

[SPECULATIVE] For roughly $100/month in compute (a figure consistent with current pricing trends for LLM API access combined with modest local infrastructure), an individual can operate a structured agent system capable of handling a significant fraction of the cognitive labor involved in running a small operation: compliance, scheduling, communication, research, bookkeeping, document generation, and project management.

[SPECULATIVE] This represents a potential structural shift in how economic value is created. The historical barrier to entrepreneurship was not intelligence but infrastructure — the accountants, lawyers, assistants, and specialists that only organizations could afford. If agent architectures can absorb much of this organizational overhead, the minimum viable team for many kinds of economic activity approaches one person augmented by their cognitive infrastructure.

[SPECULATIVE] The corollary is less optimistic: individuals and organizations that do not develop organizational depth in their agent systems may find themselves at a compounding disadvantage. Not because they lack access to AI — the models are increasingly commoditized — but because they lack the organizational structures that convert raw model capability into reliable, persistent, self-improving professional output.

16. Summary

[REF] Model scaling increases internal function complexity. This is well-characterized and valuable (Kaplan et al., 2020).

[SPECULATIVE] Architecture scaling increases organizational complexity. If the target is long-horizon, self-improving, professionally capable agency, organizational complexity is not optional. It is the substrate.

[SPECULATIVE] The practical direction: build persistent information stores with defined schemas; regulate lifecycle events with deterministic mechanisms; track obligations to ensure completion; run consolidation cycles that convert experience into durable knowledge; and treat the model as a powerful engine within a larger cognitive organ system — not as the system itself.

The engine can be replaced. The organizational structure — the cognitive organ system you grow around it — is what persists, evolves, and compounds.

References

1. Baldwin, C. Y., & Clark, K. B. (2000). *Design Rules: The Power of Modularity.* MIT Press.

2. Bonner, J. T. (1988). *The Evolution of Complexity by Means of Natural Selection.* Princeton University Press.

3. Brooks, R. A. (1991). Intelligence Without Representation. *Artificial Intelligence*, 47(1-3), 139-159.

4. Crick, F. (1970). Central Dogma of Molecular Biology. *Nature*, 227, 561-563.

5. Cybenko, G. (1989). Approximation by Superpositions of a Sigmoidal Function. *Mathematics of Control, Signals, and Systems*, 2(4), 303-314.

6. Gilbert, W. (1986). Origin of Life: The RNA World. *Nature*, 319, 618.

7. Gray, J. (1978). Notes on Data Base Operating Systems. In *Operating Systems: An Advanced Course.* Springer.

8. Hoffmann, J., et al. (2022). Training Compute-Optimal Large Language Models. *arXiv:2203.15556.*

9. Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer Feedforward Networks are Universal Approximators. *Neural Networks*, 2(5), 359-366.

10. Kaplan, J., et al. (2020). Scaling Laws for Neural Language Models. *arXiv:2001.08361.*

11. Kauffman, S. A. (1993). *The Origins of Order: Self-Organization and Selection in Evolution.* Oxford University Press.

12. March, J. G., & Simon, H. A. (1958). *Organizations.* Wiley.

13. Parnas, D. L. (1972). On the Criteria To Be Used in Decomposing Systems into Modules. *Communications of the ACM*, 15(12), 1053-1058.

14. Shazeer, N., et al. (2017). Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer. *arXiv:1701.06538.*

15. Simon, H. A. (1947). *Administrative Behavior.* Macmillan.

16. Simon, H. A. (1962). The Architecture of Complexity. *Proceedings of the American Philosophical Society*, 106(6), 467-482.

17. Wei, J., et al. (2022). Emergent Abilities of Large Language Models. *arXiv:2206.07682.*

Hadosh Academy — Agents Series

- Essay 1: “LLMs Are Not the Agents”
- Essay 2: “We Could Have Had AGI By Now” (blog) + “Why Scaling Models Is Not Enough” (this white paper)
- Essay 3: “Your Brain Was Never Built for This”
- Essay 4: “The Words They Assume You Know”

Resources: - hadi-nayebi.github.io - skool.com/claude-agents-engineering