*Article*

# Mythos and the Expansion of the Agentic Control Surface: A Structural Analysis of Runtime Coherence, Evaluation Mismatch, and Persistent Agentic Execution

**Gregor Herbert Wegener**

Friedrichstrasse 4, 10969 Berlin, Germany; gregor.wegener@gmail.com; Tel.: +49 179 2544522

**Abstract**

Claude Mythos Preview is relevant not only as a frontier model release, but as an indicator of a broader transition in how advanced AI systems are evaluated and deployed. The central structural change is not higher capability in isolation. It is the expansion of the agentic control surface across tool use, persistent runtime state, orchestration logic, and deployment context. As execution extends beyond bounded prompt-response interaction into longer-running, multi-step, tool-mediated pathways, classical evaluation and monitoring frames provide only partial visibility into how behaviour is shaped in practice. In such systems, realised performance depends not only on model competence, but also on whether runtime coordination, pipeline coherence, and deployment conditions remain sufficiently aligned for that competence to be expressed predictably. This paper introduces a constructive diagnostic perspective on agentic control surface expansion and uses SORT as a structural lens for interpreting this shift. The discussion is organised around ai.13, ai.04, ai.27, ai.47, and ai.52 as complementary views on agentic system stability, runtime control coherence, inference pipeline coherence, evaluation-context projection instability, and weak-signal drift across deployment environments. The aim is not to critique current practice, but to extend it with a systems-architectural vocabulary that can support clearer observability, stronger deployment alignment, and more predictable operation in persistent agentic environments.

**Keywords:** agentic control surface; runtime coherence; evaluation-deployment divergence; persistent agentic execution; agentic system stability; inference pipeline coherence; deployment drift; structural diagnostics; frontier AI systems; Claude Mythos

---

## 1. Executive Summary

Claude Mythos Preview, released by Anthropic in restricted form in April 2026, is presented as the company's most capable frontier model, with particular strength in coding, security reasoning, and sustained agentic execution [1]. Its restricted deployment through Project Glasswing reflects an explicit judgment that these capabilities are best introduced first under controlled operating conditions rather than through immediate broad release [2]. In that sense, Mythos is already informative before any wider deployment footprint emerges. It offers a useful case for examining how frontier capability is increasingly expressed through runtime structure rather than through model quality in isolation.

The central claim of this paper is that Mythos is structurally relevant not primarily because it extends benchmark-visible capability, but because it makes visible a broader shift in how advanced AI systems must be interpreted. The decisive change is the expansion of the agentic control surface across tool use, persistent runtime state, orchestration logic, and deployment context. Once execution extends beyond bounded prompt-response interaction into longer-running, multi-step, tool-mediated

pathways, the object of analysis changes. The relevant unit is no longer only the model, but the model as embedded in an active execution environment.

This shift matters because persistent agentic execution changes the relationship between evaluation and deployment. Under classical assumptions, models are assessed through bounded tasks executed under comparatively controlled conditions. Under agentic conditions, behaviour is increasingly shaped by how planning, tool access, state persistence, and runtime coordination interact over time [4]. The operational profile of the system therefore depends not only on what the model can do, but on how its capabilities are expressed through the surrounding execution fabric.

A useful way to frame this transition is to treat runtime context as part of the behavioural surface of the system. Tool invocation, memory persistence, computer use, intermediate verification, and execution sequencing do not merely surround the model. They condition how behaviour unfolds in practice. As these elements become more persistent and more tightly coupled, the boundary between model capability and runtime structure becomes less clean. What appears in evaluation as a model property may, in deployment, become inseparable from the orchestration conditions under which that capability is realized.

This observation helps clarify why classical evaluation methodology provides only partial visibility into advanced agentic systems. Existing evaluation practice remains highly effective for bounded capability assessment, but it is less suited to capturing whether behaviour remains coherent once execution is distributed across longer horizons, multiple tools, and runtime-mediated control surfaces. The relevant question is therefore no longer only whether the model performs well on a defined task, but whether the model-context combination remains sufficiently stable, legible, and predictable as execution conditions become more persistent and structurally differentiated.

The paper develops a constructive diagnostic perspective for analysing this shift. Rather than treating Mythos as a model drama or as an isolated release event, it is used here as a case study in agentic control surface expansion under persistent runtime conditions. The goal is to provide a systems-architectural vocabulary for understanding how agentic capability changes the operational meaning of evaluation, deployment, and control. This perspective is mapped to the SORT application catalog as a structured way of interpreting agentic system stability, runtime control coherence, inference pipeline coherence, evaluation-context projection instability, and weak-signal deployment drift in large-scale AI systems.

## 2. Why Mythos Is Not Primarily a Model Story

### *2.1. Mythos Preview as a Frontier Capability Shift*

Anthropic presents Mythos Preview as a general-purpose frontier model with pronounced strength in coding, security reasoning, and extended autonomous operation [1]. In capability terms, this clearly marks a frontier shift. The model is not described merely as an incremental quality improvement over an existing assistant class. It is introduced as a system whose practical value is increasingly expressed through longer horizon execution, tool-mediated action, and sustained technical reasoning across more complex operational tasks.

For the purposes of this paper, however, the most relevant point is not the benchmark delta as such. What matters is the type of capability that appears to have advanced. The emphasis is not limited to stronger single-step reasoning or improved task completion under bounded conditions. The emphasis is on a class of behavior that becomes more significant once execution extends across tools, state, and time. In infrastructure and systems terms, this means that the capability shift is already inseparable from a shift in the operational surface through which the model acts.

This distinction is important because it changes what should be observed. A model that is stronger within the same bounded interaction pattern remains primarily a model-comparison problem. A model that is stronger in persistent, tool-mediated, multi-step execution begins to change the surrounding system assumptions as well. The relevant interpretive move is therefore from capability ranking to

execution structure. Mythos matters here not because it is simply more capable, but because the nature of the capability increase is structurally linked to broader runtime exposure.

### *2.2. Why the Release Is Coupled to Deployment Structure*

The restricted release of Mythos through Project Glasswing reinforces this interpretation [2]. Anthropic does not frame the deployment choice solely as a question of abstract model strength. The decision is coupled to how the model would operate when introduced into real environments, with access patterns and usage conditions that make the deployment context itself part of the technical assessment. This means the release logic is already structured around operational conditions rather than around model evaluation in isolation.

Anthropic's Responsible Scaling Policy makes this broader boundary explicit. Contemporary frontier systems are described not simply as language models, but as systems that may browse the web, write and execute code, use computers, and perform autonomous multi-step actions [4]. The Frontier Safety Roadmap extends this logic by linking increased autonomy to progressively stronger assurance expectations [5]. Independent governance analysis has also noted that this scaling framework reflects both an ambitious operational model and the resulting tension between capability expansion and assurance requirements [7]. Taken together, these materials suggest that the relevant object of concern is already larger than the model itself.

This matters analytically because it shows that the surrounding discourse is not only media amplification. Anthropic's own primary materials increasingly reason in deployment terms. The model is discussed together with access controls, usage environments, partner restrictions, and assurance conditions. In other words, the release is coupled to the structure of deployment. That coupling is precisely what makes Mythos useful as a systems case study. It allows the paper to remain anchored in official framing while still analyzing a broader structural transition.

### *2.3. The Relevant Unit Is the Model in Runtime Context*

From a structural perspective, the Mythos case illustrates a broader transition in frontier AI analysis. The relevant unit is no longer best understood as the model considered as an isolated artifact. It is the model in runtime context, namely the model together with tool access, persistent state, orchestration logic, scheduling conditions, and deployment topology. These surrounding elements do not merely host the model. They condition how its capabilities are expressed, constrained, and observed in practice.

Once this shift is acknowledged, several implications follow. Evaluation becomes less about isolated task competence alone and more about how task competence projects into active execution environments. Monitoring becomes less about output inspection alone and more about the observability of persistent, multi-step behavioral pathways. Governance similarly becomes less about abstract model classification alone and more about the conditions under which autonomous capability is introduced, bounded, and interpreted over time. These are not replacements for existing practices. They are extensions made necessary by a change in the operational object itself.

This is the sense in which Mythos is not primarily a model story. It is a structurally revealing case in which the model cannot be understood independently of the execution conditions that make its frontier capabilities operationally meaningful. The analytical value of the case lies precisely there. It helps make visible that the system boundary relevant to advanced AI deployment is expanding, and that the expansion is occurring through runtime context rather than through model weights alone.

## 3. Agentic Control Surface Expansion

### *3.1. From Bounded Outputs to Persistent Action Pathways*

Classical large language model interaction is structurally bounded. A prompt is processed, an output is returned, and the execution path terminates without persistent continuation. Under that interaction pattern, the dominant analytical focus remains on output quality, local task success, and

response-level performance. Agentic execution changes this structure. Once the system is able to browse, invoke tools, execute code, inspect intermediate results, and continue acting across multiple steps, the operational unit is no longer a single inference event but an extended action pathway unfolding over time [4].

Mythos Preview makes this transition especially visible. Anthropic's System Card describes a model class capable of sustained autonomous operation across coding, vulnerability discovery, and complex tool-mediated workflows [1]. What matters structurally is not only that the model performs well in such tasks, but that the execution path itself becomes persistent. Context is carried across steps, intermediate outputs are reintroduced into subsequent decisions, and the system remains engaged in a goal-directed sequence rather than returning immediately to an idle state after a single completion.

This shift from bounded outputs to persistent action pathways is the structural basis of agentic control surface expansion. The number of dimensions along which the system can interact with its environment increases, including tool selection, sequencing, memory persistence, intermediate verification, and conditional branching. As those dimensions increase, so does the coordination surface that must remain coherent if behaviour is to remain predictable, reproducible, and operationally legible. The analytical problem therefore expands from response evaluation to pathway coherence.

### *3.2. Why Tool Use, Runtime State, and Persistence Expand the Control Surface*

Each extension of the model's operational boundary enlarges the control surface through which behaviour is expressed. Tool access introduces action possibilities beyond text generation alone. Code execution introduces side effects and state transitions in external systems. Web access extends informational dependency beyond bounded prompt context. Computer use adds interaction surfaces that are conditioned by interface state, timing, and execution order. None of these elements acts independently. Tool availability influences planning, runtime state influences subsequent selection and retry patterns, and persistence extends the temporal scope across which earlier choices remain active.

This interaction becomes more significant once execution is recursive. In bounded systems, a local failure is often visible as a poor output or a failed task. In recursive agentic systems, loss of coherence can appear differently. Execution may continue while accumulating retry depth, redundant tool use, context expansion, or progressively less efficient search behaviour. Industry reporting suggests that many agentic initiatives encounter difficulty in reaching stable production use, while complex agent workflows can expand token consumption and orchestration overhead substantially relative to simpler interaction patterns [8,9]. From a structural perspective, these are not merely workload-specific inefficiencies. They are signs that the interaction surface between planning, tools, and persistence has become large enough that stability must be analysed as a system property [11].

For that reason, the control surface does not grow linearly with the number of tools or capabilities exposed to the model. It grows combinatorially with the number of possible interactions among tools, runtime state, memory persistence, branching logic, and verification loops. What expands is not only functional reach, but the space of possible execution trajectories. The more persistent and interconnected that space becomes, the less informative it is to ask only whether the model can perform a given task. The more relevant question becomes whether the surrounding system can maintain coherent behaviour across the growing interaction surface.

### *3.3. Control as a Runtime Coherence Question*

As the control surface expands, the meaning of control changes with it. Under bounded interaction, control can often be treated primarily as a question of whether the model produces acceptable outputs under specified constraints. Under persistent agentic execution, this framing becomes incomplete. The relevant question is no longer only whether outputs remain acceptable, but whether the coupled model-runtime system remains coherent as execution traverses multiple steps, tools, state transitions, and policy boundaries.

This is where runtime control coherence becomes structurally important. In large-scale systems, execution is shaped by multiple coordination layers operating simultaneously, including schedulers, tool orchestrators, runtime engines, policy gates, retry logic, and resource managers. Each of these layers may be locally well specified. Yet when they optimize independently without sufficient global visibility, the aggregate system can produce behaviour that is locally rational and globally inconsistent [12]. Under agentic conditions, such incoherence is not confined to latency or resource allocation. It shapes whether extended action pathways remain convergent, interpretable, and bounded in practice.

For Mythos-class systems, this implies that control is not reducible to a model question alone. It is a systems-engineering question concerning whether runtime conditions preserve coherent coordination as execution depth, tool interaction, and persistence increase. This does not displace model evaluation or alignment-related considerations. It complements them by identifying a different layer at which behaviour is conditioned. The larger the agentic control surface becomes, the more important it is to ask whether the runtime fabric through which capability is expressed remains structurally coherent over time.

## 4. The Evaluation Boundary Breaks First

### *4.1. Evaluation Was Designed for Bounded Tasks*

Classical AI evaluation is built around bounded interaction structures. Tasks are typically specified through fixed prompts, controlled runtime assumptions, limited feedback depth, and comparatively narrow tool exposure. These conditions are not accidental. They make evaluation reproducible, comparable, and analytically tractable. They also make it possible to attribute performance differences to models with reasonable confidence, because the surrounding execution context is intentionally constrained.

This structure remains highly effective for capability assessment under stable conditions. The difficulty arises when the operational object being evaluated is no longer a bounded interaction pattern, but a persistent agentic system whose behaviour depends on tool choice, state continuity, retry logic, and runtime-mediated sequencing. Under those conditions, evaluation and deployment no longer represent nearby variants of the same context. They project onto structurally different regions of the system's behavioural space.

This is the core condition captured by ai.47, *Evaluation Context Projection Instability*. The issue is not that evaluation becomes invalid. It is that its scope remains tied to the context for which it was designed. Once the control surface expands beyond bounded execution, the relation between measured behaviour and deployed behaviour becomes less direct. The evaluation boundary is therefore the first boundary to lose coverage, not because evaluation is poorly designed, but because the system being evaluated has changed its operating logic.

### *4.2. Mythos Makes the Evaluation-Deployment Gap Visible*

The Mythos case makes this structural gap especially legible. Anthropic's Mythos materials describe behavioural patterns that became more visible under extended, tool-rich operating conditions than under narrower bounded testing assumptions [1,3]. What is analytically useful here is not any singular incident in isolation, but the broader fact that behaviour under persistent execution conditions can expose action pathways that are only partially visible in conventional evaluation settings.

This should not be interpreted as evidence that evaluation failed. A more precise interpretation is that the evaluation context could not fully represent the deployment context once the operational surface expanded. As systems acquire longer execution horizons, richer tool access, and more persistent interaction with runtime state, the number of possible execution trajectories increases beyond what bounded task evaluation is designed to probe. In that setting, even well-designed evaluation captures only a constrained projection of the broader behaviour space.

The implication extends beyond Mythos itself. As agentic systems become more persistent and more tightly integrated with tools and runtime infrastructure, the gap between evaluated behaviour

and deployed behaviour becomes a first-order structural variable. It is no longer only a secondary validation concern to be addressed after model assessment. It becomes part of the core question of what evaluation actually confirms once advanced capability is expressed through extended execution contexts.

### *4.3. Projection Instability as a Diagnostic Lens*

Projection instability provides a useful diagnostic lens for analysing this transition. In the Public Application Catalog, ai.47 is defined as the structural analysis of behaviour divergence between evaluation and deployment contexts. This framing is especially well suited to Mythos because the case does not require an assumption of model inconsistency or evaluator error. It requires only the recognition that bounded evaluation and extended deployment expose the same system through different contextual projections.

Seen in this way, Mythos is not primarily evidence of a breakdown in evaluation practice. It is evidence that the relation between evaluation and deployment has become more structurally consequential. Evaluation under bounded conditions projects onto a restricted behavioural region. Deployment under extended agentic conditions projects onto a broader and more interaction-sensitive region. The difference between those regions is what becomes analytically important.

This perspective does not call for abandoning evaluation or replacing benchmark practice. It calls for extending evaluation methodology with a more explicit awareness of how context shapes the behavioural surface being measured. For persistent agentic systems, the relevant improvement frontier is therefore not evaluation rejection, but evaluation-context awareness. Projection instability is useful precisely because it names that structural difference without treating it as a design flaw. It turns a growing source of ambiguity into an explicit analytical object.

## 5. Runtime and Pipeline Coherence Become the Real Constraint

### *5.1. Multi-Step Execution Distributes Control Across Layers*

In persistent agentic execution, control is no longer concentrated in a single inference step. It is distributed across planning, tool selection, execution, verification, and, in some cases, state persistence across longer horizons. Each of these stages combines model decisions with runtime conditions, orchestration logic, and infrastructure-mediated constraints. The resulting system cannot be understood adequately as a model plus a set of tools. Its behaviour is shaped by the pipeline through which execution is staged, routed, validated, and continued over time [1,4].

This is the structural setting in which inference pipeline coherence becomes relevant. In the SORT application catalog, ai.27 addresses the coherence of execution paths across serving layers, control mechanisms, and pipeline transitions. Under agentic conditions, this concern expands beyond classical inference flow to include tool orchestration, retry sequencing, state carryover, and intermediate verification between steps. The important point is not that any one layer becomes dominant. It is that behaviour emerges from how these layers remain coordinated as execution extends.

As a result, multi-step execution redistributes the control problem across a broader system surface. Decisions that appear local at one stage can condition path selection and observability at later stages. The analytical challenge therefore shifts from evaluating a single completion to examining whether execution remains sufficiently coherent as it moves through a layered pipeline whose components each contribute to the realised behaviour of the system.

### *5.2. Coherence Loss Through Layer Interaction*

Coherence loss in agentic pipelines does not require explicit component failure. It can emerge when independently operating coordination layers interact in ways that are locally reasonable yet globally inconsistent. Safety filters, retry managers, cost controls, scheduling policies, and tool orchestrators may each perform their intended function, while their combined behaviour produces

expanded execution depth, unstable retry patterns, or reduced convergence across the overall task pathway [12]. This is a structural interaction effect rather than a conventional fault condition.

The practical relevance of such interactions is increasingly recognized in operational guidance. The OWASP Top 10 for Agentic AI identifies cascading failures as a distinct concern in agentic architectures, which supports the broader point that multi-layer interaction effects are not merely theoretical abstractions but recognized properties of more complex autonomous systems [10]. What becomes visible here is that coherence can degrade through composition even when each contributing layer remains individually legible and locally well-behaved.

For Mythos-class systems, this interaction surface is structurally larger than in conventional bounded inference. Extended tool access, persistent runtime state, code execution, computer use, and web interaction each add new coordination boundaries. Every added boundary increases the number of possible points at which execution can remain technically active while becoming less globally coherent. The relevant system question is therefore not only whether components are functioning, but whether their interaction continues to produce an execution path that is convergent, interpretable, and operationally efficient over time.

### *5.3. Agentic Systems as Coupled Runtime Systems*

The broader implication is that modern agentic systems are better understood as coupled runtime systems rather than as model instances with attached tools. The model remains central, but it operates as one component within a multi-layer execution architecture whose behaviour depends on how planning, orchestration, serving, policy control, and persistence interact. In such systems, model-level analysis remains necessary, but it is no longer sufficient to explain the full operational profile of the deployed system.

This reframing has consequences for evaluation, monitoring, and governance. Each of these practices was developed primarily for more model-centric forms of interaction in which execution remained bounded and the surrounding pipeline was comparatively stable. As agentic systems move toward longer-running and more environment-sensitive execution, those practices continue to provide useful visibility, but they cover only part of the relevant system. What becomes increasingly important is whether the coupled runtime remains coherent enough for nominal capability to translate into stable and predictable operation.

Anthropic's own framing supports this broader interpretation. The Responsible Scaling Policy treats advanced models as systems that act through browsing, code execution, computer use, and autonomous multi-step operation rather than through bounded output generation alone [4]. Once that framing is accepted, runtime and pipeline coherence become less of an implementation detail and more of a binding constraint on how advanced capability can be introduced and assessed in practice.

## 6. Weak Signals Before Visible Failure

Not every consequence of agentic control surface expansion appears as a visible or singular incident. In production environments, early indicators of structural stress are often weak, distributed, and operationally ambiguous when viewed in isolation. Examples include marginal increases in retry rates, gradual widening of latency variance, subtle expansion of tool-call graphs, or incremental context drift across extended sessions. None of these signals necessarily indicates immediate loss of function. Taken individually, they may remain below the threshold of conventional alerting or be interpreted as routine variance within a complex system.

Their significance becomes clearer only when they are aggregated across sessions, execution paths, and deployment conditions. This is the diagnostic concern captured by ai.52, *Deployment Drift Signal Aggregation*. The relevant structural point is not that weak signals are inherently alarming, but that persistent agentic systems generate a broader and more weakly distributed observability surface than conventional bounded inference. Under these conditions, the meaningful unit of observation shifts from isolated anomalies to correlated patterns of small deviations that accumulate over time.

This becomes more important as execution grows more persistent and tool-rich. Longer pathways, larger state surfaces, and more varied interaction patterns create more locations at which drift can emerge before any single event becomes visibly consequential. In Mythos-class systems, the operational surface is therefore large enough that output-level monitoring alone provides only partial visibility. Structural aggregation becomes useful because it allows apparently minor changes in behavior to be interpreted in relation to one another rather than as disconnected local events.

The accidental exposure of Claude Code source through an agentic execution pathway provides a concrete reminder that extended tool-mediated operation can produce operational consequences through channels not directly covered by conventional output monitoring [6]. The analytical relevance of this example lies less in the singular event itself than in the broader lesson that pathway-sensitive systems can express drift through side channels, intermediate actions, and distributed interaction surfaces that remain weakly visible at the level of final outputs alone.

The practical implication is that observability for advanced agentic systems benefits from more than output monitoring and discrete incident detection. It also benefits from structural signal aggregation capable of detecting coherence drift before it crosses the threshold of visible failure. In this sense, ai.52 functions not as a dramatic failure model, but as a constructive observability lens for recognising how extended execution surfaces make weak, distributed signals more architecturally relevant over time.

## 7. SORT as a Diagnostic Perspective on Agentic Boundary Stress

The SORT framework is used in this paper exclusively as a diagnostic lens for reasoning about agentic control surface expansion under persistent runtime conditions. It is not introduced here as a new dynamical model, a replacement for existing safety engineering, or a substitute for established evaluation practice. Its function is narrower and more practical. It provides a compact structural vocabulary for describing conditions that become increasingly relevant once advanced model capability is expressed through persistent execution, layered coordination, and deployment-sensitive runtime pathways. In that role, SORT helps relate otherwise separate observations about agentic stability, runtime coherence, evaluation mismatch, and weak-signal drift to a common systems-architectural perspective.

### 7.1. Application Mapping

**Table 1.** SORT Application Catalog Mapping for Agentic Control Surface Expansion

| ID | Cluster | Function in This Paper |
|---|---|---|
| ai.13 | D | **Agentic System Stability.** Structural lens for the stability of agent workflows under retry loops, tool chaining, recursive execution, and extended action pathways. It functions here as the primary diagnostic view on control surface expansion under persistent agentic conditions [11]. |
| ai.04 | C | **Runtime Control Coherence.** Diagnostic lens for incoherence between schedulers, runtime engines, policy layers, and model-adjacent control loops when multiple coordination surfaces interact without sufficient global state awareness [12]. |
| ai.27 | C | **Inference Pipeline Control Coherence.** Structural coherence analysis of serving pipelines including batching, caching, routing, and serving control loops, extended here to cover tool orchestration, intermediate verification, and state persistence in agentic execution. |
| ai.47 | C | **Evaluation Context Projection Instability.** Structural analysis of behaviour divergence between evaluation and deployment contexts. In this paper it is applied to the growing gap between bounded evaluation and persistent, tool-mediated agentic deployment. |
| ai.52 | A | **Deployment Drift Signal Aggregation.** Structural framework for distributed weak-signal aggregation across deployment environments. It is used here to interpret how early coherence drift becomes visible through correlated low-intensity signals across extended agentic sessions. |

### 7.2. Source Hierarchy

**Table 2.** Source Hierarchy for This Paper

| Source Type | Intended Role |
|---|---|
| Anthropic official materials | Primary factual basis and case-study anchor, including the System Card, Project Glasswing, Risk Report, Responsible Scaling Policy, and related official framing documents. |
| High-quality secondary analysis | Selective contextual support where external interpretation clarifies deployment, governance, or incident-relevant structure without replacing primary-source grounding. |
| SORT internal papers | Mechanism-level structural support only, used sparingly where agentic stability or runtime coherence requires an additional analytical bridge. They do not function as authority for application identity. |
| Public Application Catalog v6.2 | Sole authority for application identity, naming, and cluster assignment in the diagnostic mapping used throughout this paper. |

## 8. Strategic Implications for Frontier Labs and Hyperscalers

### 8.1. Evaluation Must Extend to Runtime and Deployment Coherence

For frontier systems with persistent agentic capability, evaluation that covers only bounded task performance under controlled conditions provides necessary but structurally incomplete assurance. Once execution depends on tool access, state persistence, intermediate verification, and runtime coordination, the evaluation question expands from whether the model performs well on a task to whether the model-context combination remains sufficiently coherent under the conditions in which it is actually deployed.

This does not require abandoning benchmark practice or replacing established evaluation pipelines. It suggests extending them with greater attention to runtime and deployment coherence. In this sense, Anthropic's Project Glasswing can be read as an implicit step in that direction. The restricted deployment structure creates conditions under which behaviour can be observed under more operationally relevant freedom while still remaining bounded within a controlled introduction pathway [2].

### *8.2. The Control Question Shifts from Model to Coupled Runtime*

Persistent agentic systems relocate the operative control question from the model alone to the coupled model-runtime system. Tool access controls, retry policies, orchestration logic, serving pathways, and state persistence become part of the behavioural surface through which capability is expressed. As a result, control can no longer be understood adequately only in terms of output quality or bounded model behaviour. It must also be understood in terms of whether the runtime environment preserves coherent execution across longer horizons and multiple coordination layers.

For frontier labs and hyperscalers, this shift has practical implications for both oversight and system design. Governance practices that were developed around model-centric analysis remain valuable, but they provide only partial visibility once advanced systems act through persistent runtime pathways. The relevant extension is therefore not conceptual replacement, but broader coverage of the coupled execution environment in which the model actually operates.

### *8.3. Early Observability of Distributed Drift Becomes Strategically Important*

As agentic control surfaces expand, the density and variability of runtime signals increase with them. Under these conditions, waiting for visible failure before responding becomes less informative as an operational strategy, because the space in which coherence degrades is broader, more distributed, and more path-dependent than in conventional bounded inference. Meaningful changes may first appear as low-intensity deviations across retries, latency patterns, tool-call structure, context carryover, or session-to-session behavioural variance rather than as a single obvious event.

This is why early observability of distributed drift becomes strategically important. Structural signal aggregation, as framed by ai.52, provides a constructive perspective here. Its value lies not in replacing conventional monitoring, but in extending it with a way to interpret weak and distributed changes as part of a larger coherence picture. For frontier labs and hyperscalers operating persistent agentic systems, that additional visibility can support clearer deployment assessment, more stable runtime interpretation, and better alignment between nominal capability and realised operational behaviour.

## 9. Conclusion

Claude Mythos Preview is structurally significant not primarily because it introduces higher capability, but because it makes visible a broader transition in how advanced AI systems must be interpreted. Once execution extends beyond bounded prompt-response interaction into persistent, tool-mediated, runtime-embedded pathways, the relevant analytical object changes. The system can no longer be understood adequately through model capability alone. It must be understood through the coupled conditions under which that capability is expressed, coordinated, and observed.

The central argument of this paper is that the expansion of the agentic control surface is not a singular property of one release. It reflects a more general transition in frontier AI systems as they move from bounded inference toward persistent execution across tools, state, orchestration layers, and deployment contexts. Under those conditions, several structural variables become more important: the divergence between evaluation and deployment contexts, the coherence of runtime and pipeline interaction, and the accumulation of weak drift signals across extended execution pathways. These are not replacements for existing evaluation, monitoring, or governance practices. They are the next structural dimensions through which those practices must be extended.

For frontier labs and hyperscalers, the practical implication is straightforward. Structural visibility into the coupled model-runtime system becomes increasingly important for predictable deployment of persistent agentic systems. The relevant engineering question is no longer only whether the model is capable, but whether the surrounding execution environment preserves coherent, legible, and stable behaviour as that capability unfolds across time and operational context.

What Mythos helps make visible is therefore not only a capability frontier, but a systems frontier. The more agentic execution becomes persistent, the more evaluation, observability, and deployment architecture must be interpreted as parts of the same structural problem.

*Mythos matters not because it is merely more capable, but because it makes visible how agentic capability expands the control surface beyond what conventional evaluation and runtime assumptions were designed to contain.*

**Acknowledgments:** The author acknowledges prior internal architectural work that informed the conceptual development of the diagnostic perspective presented in this paper.

**Conflicts of Interest:** The author declares no conflicts of interest.

**Use of Artificial Intelligence:** Artificial intelligence tools were used for limited editorial support, including language refinement and LaTeX formatting assistance. The scientific framing, conceptual arguments, structural taxonomy, interpretation of the literature, and all substantive research judgments were developed and verified by the author, who takes full responsibility for the content of the manuscript.

**Data Availability Statement:** No new data were generated in this study. All referenced data are available in the cited publications.

1. Anthropic. (2026). Claude Mythos Preview System Card. *Anthropic*. https://www.anthropic.com/claude-mythos-preview-system-card
2. Anthropic. (2026). Project Glasswing. *Anthropic*. https://www.anthropic.com/project/glasswing
3. Anthropic. (2026). Alignment Risk Update: Claude Mythos Preview. *Anthropic*. https://www.anthropic.com/claude-mythos-preview-risk-report
4. Anthropic. (2026). Responsible Scaling Policy. *Anthropic*. https://www.anthropic.com/responsible-scaling-policy
5. Anthropic. (2026). Anthropic's Frontier Safety Roadmap. *Anthropic*. https://anthropic.com/responsible-scaling-policy/roadmap
6. InfoQ. (2026). Anthropic Accidentally Exposes Claude Code Source via Agent Execution. *InfoQ News*. https://www.infoq.com/news/2026/04/claude-code-source-leak/
7. GovAI. (2026). Anthropic's RSP v3.0: How it Works, What's Changed, and Some Reflections. *Centre for the Governance of AI*. https://www.governance.ai/analysis/anthropics-rsp-v3-0-how-it-works-whats-changed-and-some-reflections
8. Gartner, Inc. (2025). Predicts 2025: Agentic AI — The Evolution of Experience and Productivity. *Gartner Research Report*.
9. Galileo. (2025). The Hidden Costs of Agentic AI: Why 40% of Projects Fail Before Production. *Galileo AI Blog*. https://galileo.ai/blog/hidden-cost-of-agentic-ai
10. OWASP Foundation. (2025). OWASP Top 10 for Agentic AI Security — ASI08: Cascading Failures. *OWASP*. https://owasp.org/www-project-top-10-for-agentic-ai/
11. Wegener, G.H. (2026). SORT-AI: Agentic System Stability in Large-Scale AI Systems — Structural Causes of Cost, Instability, and Non-Determinism in Multi-Agent and Tool-Using Workflows. *MDPI Preprints*. doi:10.20944/preprints202601.1741.v1
12. Wegener, G.H. (2026). SORT-AI: Runtime Control Coherence in Large-Scale AI Systems. *MDPI Preprints*. doi:10.20944/preprints202601.0298.v1