How Much Coordination Gain Is Real? A Paired Noise-Floor Protocol for Multi-Agent LLM Benchmarks

Alibek Kaliyev
alibek.kaliyev@utexas.edu
The University of Texas at Austin
Austin, Texas, USA

Artem Maryanskyy
artem.maryanskyy@uber.com
Uber
San Francisco, California, USA

Abstract

Multi-agent LLM coordination papers report small benchmark deltas as evidence that one architecture beats another. A prior question: how much paired trial-0 disagreement do two protocols produce on the same model and benchmark when their API inputs are *configuration-equivalent* (matched by code inspection plus a SHA-256 byte audit), short of full identity-replay? On Claude Haiku 4.5 against τ^2-bench retail, the clean configuration-equivalent contrast (no_coord vs. intercept, both inert at trial 0) gives signed paired gaps of +10 pp and 0 pp across two n=100 seeds; pooled across both, +5 pp with Wilson CI $[-2, +12]$, not significant. The largest single-seed contrast (+18 pp pull-vs-intercept, p_{corr}=0.012) did not reproduce at the second seed (−3 pp, p_{corr}=1.0); no trial-0 contrast is significant after Bonferroni at either seed or pooled. The envelope of observed paired gaps spans $[-3, +18]$ pp across two seeds, with pooled upper Wilson CI $\lesssim$ 15 pp. **Seven of ten recent multi-agent coordination architectures report headline effects below this local floor, and one more sits inside the envelope**; whether they survive a same-model paired replication is, by construction, untested in their original settings.

We define *coordination-active pass*^{k}, pass^k restricted to trials where the coordination mechanism is logically active, as the minimum reporting protocol, with sample-size targets and runtime hooks in the body. Measurements run on **ET-MCP**, a task-scoped negative-knowledge store conformant with MCP 2026-07-28, used as a substrate to isolate reader-side choices, not as a contribution. On Haiku 4.5 the candidate readers (pull, intercept) do not improve trial-1 recovery; we give a preliminary diagnosis of failure modes with refinements on existing production hook surfaces [1, 2, 16, 23].

1 Introduction

Two LLM agents work in parallel on the same multi-step itinerary. The first attempts to book a flight, discovers no inventory, abandons the path, and moves on. The second, started moments later by the same orchestrator, attempts the same booking through the same API and re-discovers the same dead end. The first agent's failure lived only in its private scratchpad. This is a structural property of how multi-agent LLM systems (MAS) currently coordinate [9, 12]. Recent architectures address it with coordination channels (structured handoffs, shared memory, server-side state) and report small benchmark gains as evidence one design beats another.

Why we wrote this paper. We initially set out to evaluate two such designs (an agent-side *pull* channel and a framework-side *intercept* channel) against a no-coordination baseline on τ^2-bench retail. The head-to-head shows no detectable effect at the available power: the coordination-active subsets contain n=8–17 informative pairs after ties, far short of what a medium-sized improvement would require. Unpacking that non-result forced a question we should have asked at the start: *what is the run-to-run variance between protocols whose API inputs are equivalent at trial* 0*?* A single number is enough to invert the read of much of the recent coordination literature. The rest of the paper answers that question and traces what it implies for our own and others' architectural claims.

ET-MCP as substrate. Measuring variance under matched payload required a representative coordination substrate, so we built one. **ET-MCP** is an Ephemeral Trace store of typed negative-knowledge events exposed via the Model Context Protocol (MCP) 2026-07-28 [20]: task-scoped, append-mostly, with a TF-IDF (term-frequency / inverse-document-frequency) ranker over FAILED_PATH-typed events. Two reader-side variants ride on the same store. *Pull* makes the agent the active reader: it calls `trace.query` when peer state might inform the next decision. *Intercept* makes the framework the active reader: it consults the store transparently on every tool call and prepends a `[PEER-WARNING]` block when (name, arguments) match a peer event. Intercept rides on the same hook surface already deployed for blocking, validation, retries, and caching in Strands [2], AgentCore Policy [1], LangChain [16], and the Microsoft Agent Framework [18].

The result that reframed the paper. Running this substrate against its no-coordination baseline yielded no detectable effect at the available power. At τ^2-bench retail, Haiku 4.5, T=0, n=100 tasks, pull, intercept, and no_coord all reach trial-1 success 0.54 (paired sign tests fail to reject, p=1.0 for all three contrasts; §6.3.2). Decomposed by trial, no_coord and intercept are *configuration-equivalent* at trial 0: their requests to the API are equivalent by code inspection (system prompt, tool list, sampling parameters, message-array construction; see §6.3.1 for what we did and did not verify). This is the only genuinely clean contrast in the trio; the paired sign-test disagreement is 21/11/68 at seed-1 and 13/13/74 at seed-2 (+10 pp and 0 pp signed gaps; pooled +5 pp, Wilson CI $[-2, +12]$, Bonferroni p_{corr}=0.711, not significant). The largest single-seed paired contrast was pull-vs-intercept at seed-1 (27/9/64, +18 pp, p_{corr}=0.012), but it did not reproduce at seed-2 (18/21/61, −3 pp, p_{corr}=1.0); the pooled gap is +7.5 pp with Wilson upper CI ∼15 pp. We characterize the configuration-equivalent envelope on this benchmark, model, and harness as a paired-gap distribution whose observed range across two seeds is $[-3, +18]$ pp and whose pooled upper Wilson CI is $\lesssim$ 15 pp—no single-seed Bonferroni significance survives a second seed. The floor is local, not universal; cross-model and cross-domain probes (Haiku airline n=30, Sonnet retail n=30) confirm it does not transfer.

Coordination-active pass^k. To separate run-to-run variance from mechanism effect we define $\text{PASS}^k_{\text{active}}$: marginal pass^k restricted to tasks where the peer-coordination store was non-empty at trial start. Marginal pass^k answers "should I deploy the whole system?" Active pass^k answers "did coordination help when it had an opportunity to." The two diverge whenever empty-store trials carry unrelated configuration perturbations, which is the situation we encounter. Three runtime alarms operationalize the metric on Amazon Bedrock AgentCore and SageMaker Model Monitor [23] (§6.5).

Contributions. Four contributions track the threads above:

- **(C1) Measurement protocol.** A same-model, paired, configuration-equivalent variance procedure for coordination-architecture claims on state-validated MAS benchmarks. The floor splits into an *identity-replay* component (byte-identical payloads, which we do not measure) and a *configuration-equivalent* component (same intended setup, code-inspection-verified, which we do) (§6.3.1).
- **(C2) Finding.** On τ^2-bench retail with Haiku 4.5 across two n=100 seeds, the clean configuration-equivalent paired-gap pooled CI is $[-2, +12]$ pp (no_coord vs. intercept); no single-seed contrast is significant after Bonferroni once a second seed is measured. The observed envelope spans $[-3, +18]$ pp with pooled upper Wilson CI $\lesssim$ 15 pp. Seven of ten recent multi-agent coordination architectures (Table 4) report headline gains below this envelope and one more sits inside it; each falls below or inside the envelope measured here.
- **(C3) Metric.** $\text{PASS}^k_{\text{active}}$ (coordination-active pass^k): pass^k conditioned on a non-empty coordination store. Sample-size guidance and three runtime instrumentation alarms ship the metric as a pre-deployment release gate (§6.5).
- **(C4) Failure-mode diagnosis (preliminary).** The trial-level data surface three candidate failure modes of naive negative-knowledge coordination (writer mis-attribution; per-match injection noise; brittle keying), with matched architectural refinements on existing production hook surfaces (§7). Single-judge audit and small-n validation leave these as hypotheses for confirmatory work.

ET-MCP itself and the harness mitigating a documented litellm [6, 7] bug class appear as supporting artifacts rather than claimed contributions.

2 Background and Motivation

Coordination failure is measured, not anecdotal. The Multi-Agent System Taxonomy (MAST) [9] attributes 36.94% of failures in 1,600+ production agent traces to inter-agent coordination breakdowns, the single largest category, and CodeDelegator [12] documents context pollution scaling with handoff payload size. Coordination is therefore a measurable target, not an anecdotal one; the reliability of those measurements is the load-bearing question.

ET-MCP as measurement substrate. We use ET-MCP, a task-scoped negative-event store exposed through MCP 2026-07-28 [20] primitives, as the substrate under test. The architectural details are secondary to this paper's claim; what matters is that ET-MCP is an MCP-conformant coordination layer that can be toggled on and off in paired runs, isolating a coordination-active arm against a matched baseline. The remainder of the paper is about what that paired comparison reveals about variance, not about ET-MCP itself.

3 Related Work

Prior coordination systems. CA-MCP [15] is the nearest prior system by name and substrate: server-to-server, positive plan state, single-trial; the authors explicitly disclaim cross-trial use. Terrarium [22] is a second MCP-based system with a pull-style interface, but targets attack-surface analysis. MPAC [26] proposes Lamport-clock conflict resolution for the multi-principal case. Reflexion and ExpeL [29, 36] share the negative-knowledge intuition but inject via prompt context and are single-agent or cross-task-transfer. Long-lived cross-task memory (MemGPT [25], G-Memory [34]) operates at task-spanning rather than within-task granularity. AgentVerse and related orchestration substrates surveyed in [9] round out the prior-systems comparison revisited in Table 4. Framework-level hooks (Strands [2], AgentCore Policy [1], LangChain middleware [16], Microsoft Agent Framework [18], AutoGen [32], MetaGPT [14]) carry blocking, validation, retries, and caching but no coordination semantics, and any of them could host the same measurement protocol.

Measurement methodology: the actual gap. A concurrent thread measures the reliability of agent benchmarks. Yu et al. [33] frame the multi-agent case as lacking remote direct memory access. Variance-decomposition work spans a 12-metric framework [27], an ICC framework [21], Anthropic-side infrastructure noise [3], aggregate pass@1 variance [8], systematic agent reliability [13], and input-noise robustness [30]. **None of these reports paired configuration-equivalent replication on a state-validated multi-agent coordination benchmark, and none conditions on a coordination-active sub-event.** Bedi et al. [5] (KDD '25, same workshop series) document a component-vs-system paradox in clinical multi-agent settings but treat the system as a black box. Cuadron et al.'s τ^2-bench-verified [11] fixes tasks while ours fixes transport; the two compose. AgentTrace's post-hoc causal graphs [31] and STATE-Bench [19]'s memory-agnostic harness are complementary rather than competitive. The central claim of this paper, that coordination effect sizes should be reported relative to a same-model paired coordination-active variance gate, is what fills that gap. Ou et al. [24] compare orchestrator-mediated vs. shared-notebook coordination on TravelPlanner but neither pair at matched payload nor report a coordination-active variance floor, and we revisit them in Table 4 as one of the recent coordination papers whose reported effects straddle the paired noise envelope measured here on Haiku 4.5 / τ^2-bench retail (pooled upper CI $\lesssim$ 15 pp).

4 ET-MCP Architecture

ET-MCP is the substrate that makes the measurement protocol of §6 possible; it is not the contribution of this paper. This section covers only what the measurement story needs: the minimal tool surface, the failure-only writer policy, and the pull-vs-intercept reader-side distinction that §6 compares.

4.1 Tool surface and lifecycle

The substrate is structured around one trace namespace per `task_id`, served over the stateless MCP 2026-07-28 transport [20]. Agents see three operations: `trace.write` (append a typed event), `trace.query` (TF-IDF ranking over the typed payload, returning top-k peer events with one-line summaries), and `trace.cas` (compare-and-swap for the rare double-commit race). The task lifecycle (`init`, `complete`, TTL sweep) is owned by the orchestrator through a separate channel that participating agents cannot reach, so no agent can tear down a peer's namespace. The TF-IDF ranker is a single replaceable component.

4.2 Writer policy

Events carry an envelope (`event_id`, `task_id`, `event_type`, `agent_id`, `timestamp`, `version`, `payload`). Five typed event categories cover the negative-outcome cases mapped to the MAST failure taxonomy [9]: failed paths, constraint violations, abandoned approaches, irreversible intermediate decisions, and tool errors. The default `failure_only` client-side selection policy writes only on these negative outcomes. The τ^2 retail evaluation specializes this to a single FAILED_TRIAL_ACTION type keyed on the trial's terminal reward.

4.3 Two reader-side architectures: pull and intercept

Figure 1 contrasts the two designs. **Pull** places the agent as the *active reader*: it calls `trace.query` when it estimates peer events might inform the next decision, paying decision overhead, round-trip latency, and an enlarged tool surface for that control.

Intercept relocates the reader into the framework. The orchestrator intercepts every tool call, and if the call's (name, arguments) appears in the task-scoped store as a peer FAILED_PATH or FAILED_TRIAL_ACTION event, it prepends a `[PEER-WARNING]` block to the response. Intercept removes the agent-side decision, the LLM round-trip, and any agent code changes; the tool surface and prompt are unchanged. The hook surface intercept rides on is already a production primitive in Strands [2], AgentCore Policy [1], LangChain middleware [16], and Microsoft Agent Framework [18], so intercept is an additive extension to existing runtimes, not a new framework.

Both architectures share the same event taxonomy, failure-only writer policy, task-scoped lifecycle, and MCP transport, and are configuration-equivalent at trial 0: pull's augmenter is a no-op when the store is empty, and the intercept hook is pass-through on no argument match. The full audit of this equivalence is in §6.3.1. The next section instantiates both in the reference implementation that produces the measurements of §6.

5 Implementation

To produce clean measurements, we built an Anthropic-native τ^2-bench harness (~700 lines) that replaces only the conversation-state construction layer; τ^2's domain database, tools, task definitions, and DB-hash reward are unchanged, and no model, sampling, or scoring code is touched, so the harness rewrite cannot itself contribute to the trial-0 disagreement of §6.3.1. This bypasses a documented bug class in the stock τ^2-bench/litellm pipeline [6, 7] that corrupts $\geq 30\%$ of Haiku 4.5 trials, enabling zero-infrastructure-error sweeps. The harness and the `et_mcp` substrate of §4 ship together in the open-source repository.

6 Evaluation

6.1 Benchmarks and conditions

τ^2-bench retail [4] is the primary benchmark for the reader-side architecture head-to-head. It scores state-validated pass^k on a customer-service tool-using domain. We run n=100 tasks (seed 42), with Haiku 4.5 as both assistant and user simulator, a 20-turn budget, and 2 trials per task. Cross-model and cross-domain probes (§6.4) use Sonnet 4.5 retail and τ^2-bench airline at n=30 each.

6.2 Metrics and statistical methodology

Completion rate is the fraction of trials whose final output passes the benchmark scorer. **Pass^k** on τ^2-bench follows the unbiased estimator of Chen et al.: the marginal success rate at k=1, and the both-trials-succeed rate at k=2. Two protocols are **configuration-equivalent** at a given trial when code inspection confirms identical request-side augmenter behavior at that trial (system prompt prefix, tool-response modifications, request headers, sampling parameters). This is necessary but not sufficient for byte-identical wire payloads (§6.3.1).

Coordination-active pass^k (PASS$^k_{\text{active}}$) restricts pass^k to tasks where the peer-coordination store is non-empty at each of trials $1\ldots k$, the structurally-defined coordination-active sub-event. It is causally pre-specifiable from the protocol semantics: when the store is empty the coordination mechanism is inactive, so any cross-protocol comparison on store-empty trials measures noise rather than architecture. For k=2 in our setting this is equivalently the trial-1 success rate on tasks where trial 0 failed. Each contrast uses its own paired subset (tasks where both compared protocols failed trial 0), so per-row rates are not jointly comparable across contrasts.

Effect size throughout is Cliff's δ with the Romano et al. (2006) magnitude thresholds. We apply paired sign tests (binary pass^k on τ^2; the equivalent McNemar test agrees and is reported in Table 3). The τ^2 head-to-head and coordination-active contrasts are reported outside the Bonferroni correction families as the pre-specified primary tests; all multiplicity corrections in this paper are Bonferroni with m=3. All trials are logged to JSONL artifacts under `eval/results/`; analysis is reproducible via the `analysis` CLI in the repository.

Pre-registration. The Haiku n=100 retail head-to-head is the pre-specified primary test: protocols, K=3 writer, seed 42, and T=0 were frozen before the run (K=3 was pilot-tuned on a prior n=30 sweep—§6.6). PASS$^k_{\text{active}}$ is causally pre-specifiable but was applied to already-frozen data. Sonnet, airline, M1, and P2 are exploratory; Sonnet $n\geq 250$ retail is pre-registered as confirmatory.

6.3 Results

6.3.1 Request-equivalence: what the trial-0 floor measures. The trial-0 paired gap decomposes into four layered sources; naming them keeps the headline numbers honest. The first is a byte-identical first-request audit (E1); the second is within-protocol API stochasticity at ≤ 3 pp (E2); the third is between-protocol harness

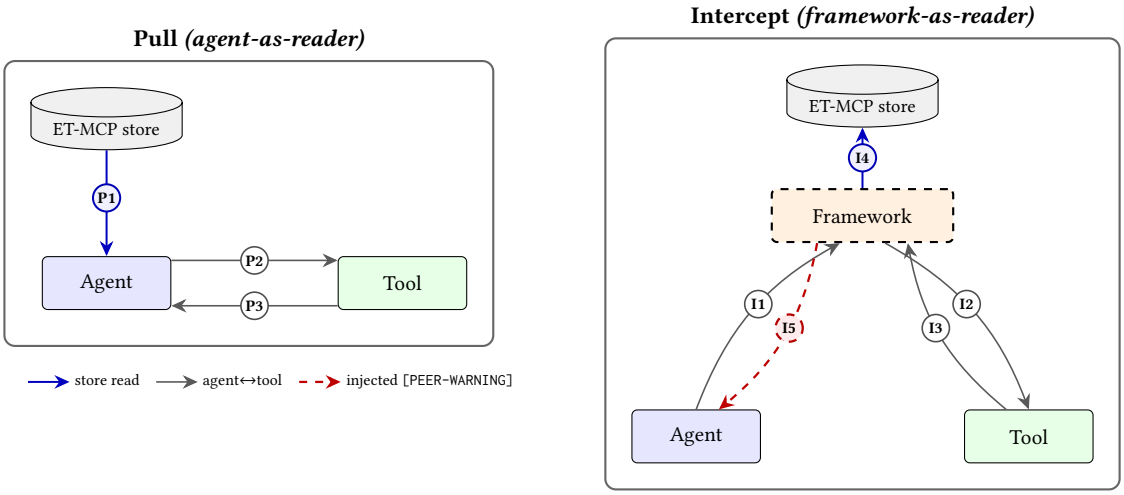

Figure 1: Reader-side architecture contrast. The two designs share the event store, writer policy, and MCP transport; they differ only in who reads and when the payload is presented. Pull (left): P1 store → agent (curated summary at trial start); P2 agent → tool (call); P3 tool → agent (raw response). Intercept (right): I1 agent → framework (call); I2 framework → tool (forward); I3 tool → framework (raw); I4 framework → store (consult); I5 framework → agent (response, with `[PEER-WARNING]` prepended on argument match). Pull presents the curated payload once; intercept injects raw warnings per matching call.

drift once the user-simulator emits its first sampled output (E3); the fourth is a hypothesized pull-specific structural perturbation (E4), which the second seed did not confirm. Strictly speaking, what we measure at trial 0 is multi-turn paired variance between two configuration-equivalent harness runs; we label it a *coordination noise floor* because it is the gate any coordination claim on this benchmark must clear, not because the variance itself is coordination-specific. One structural consequence is worth making explicit: the floor estimate is invariant to writer parameters such as K by construction. The writer fires only at the close of a trial with reward < 1, so no writer setting can perturb a trial-0 payload; K shapes trial-1 coordination *content*, and its effects are diagnosed separately (M1–M3, §6.3.2).

At trial 0 the store is empty, so reader hooks are no-ops. The three arms share the agent system prompt (`AGENT_SYSTEM_TEMPLATE`; pull's augmenter returns input unchanged when `warnings_block()` is empty), the protocol-independent tool list (no `trace.query` injection), user-simulator prompt and priming, sampling parameters (`T=0`, `max_tokens=4096`, same `claude-haiku-4-5`), and a pass-through intercept hook. Two SHA-256 audits confirm E1: an offline reconstruction of `client.messages.create` kwargs (40/40 cells byte-identical: n=10 tasks × 4 first-request payload components, each hashed across all 3 protocols) and a live wire-byte audit (30/30 user-simulator opens byte-identical on 10 headline tasks).

The Messages API exposes no `seed`, so T=0 requests are not bit-deterministic; a paired n=30 within-protocol replicate (L3) bounds E2 at ≤ 3 pp. Once the user-simulator emits its first sampled output, it becomes the agent's first message and downstream wire bytes diverge across runs (within *or* between protocol); that drift is E3. Pull's seed-1 p_{corr}=0.012 initially looked like a small real structural perturbation in pull's augmenter (E4); the second seed did not reproduce it (§6.3.2), so E4 is now a candidate-not-confirmed source rather than an established one. E1–E3 together (plus E4 as a hypothesis) form the *configuration-equivalent* envelope, and the coordination-active metric is designed to condition away E2–E4. The head-to-head below measures any remaining effect against it.

6.3.2 Reader-side architecture head-to-head: pull vs. intercept on τ^2-bench retail. What this shows. On the coordination-active subset (trial 1 of tasks where trial 0 failed), pull and intercept tie and both directionally underperform the no-coordination baseline; with n=8–17 informative pairs after ties, the subset is underpowered for a medium-sized effect, so we read this as no detectable effect at this power rather than a clean null. The trial-0 paired gaps anchor an empirical noise floor that any architectural claim must clear. Pooled across two n=100 seeds, the clean configuration-equivalent contrast (no_coord vs. intercept) gives +5 pp (CI [−2, +12], not significant); the largest contrast (pull vs. intercept) pools to +7.5 pp with upper Wilson CI ≈15 pp; and the largest single-seed gap (+18 pp, seed-1, p_{corr}=0.012) did not reproduce at seed-2.

τ^2-bench retail [4] reports state-validated pass^k on customer-service conversations (§6.1). All runs use the ~700-line Anthropic-native harness (§5). A pilot n=30 smoke on the stock τ^2-bench/litellm pipeline observed ≥ 30% of Haiku 4.5 trials terminating with malformed `tool_use/tool_result` sequences, silent at the harness layer, consistent with the documented bug class [6, 7]. Our harness observed zero infra errors across 600 trials.

Protocols. The three protocols share an identical writer side. At the close of any trial with reward < 1, the final k=3 tool calls are written as FAILED_TRIAL_ACTION events in the task-scoped store. The τ^2 harness uses this simplified single event type instead of the full five-category taxonomy of §4.2 because retail trials rarely surface explicit tool errors mid-trial; the failure signal is the trial's terminal reward. (k=3 is a causal-attribution heuristic, pilot-tuned

Table 1: τ^2-bench retail headline results ($n = 100$ tasks $\times$ 2 trials per protocol, 200 trials per cell, Claude Haiku 4.5; zero infra errors across 600 trials). All three protocols collapse to the same trial-1 success rate (0.540): coordination does not move the metric once the store is populated.

Protocol	succ. rate	$\text{pass}^1_{\text{Chen}}$	$\text{pass}^2_{\text{Chen}}$	Trial-1 succ.
`no_coord`	0.540	0.540	0.380	0.540
`pull`	0.580	0.580	0.440	0.540
`intercept`	0.490	0.490	0.330	0.540

Marginal cell metrics for the three protocols. $\text{pass}^k_{\text{Chen}}$ follows the Chen et al. unbiased estimator (marginal success rate at k=1; both-trials-succeed rate at k=2). The *trial-1 succ. rate* column reports the success rate on trial 1 only, where the trace store contains peer events from trial 0 and coordination is logically active.

on a prior n=30 sweep; the n=100 run confirms this configuration rather than pre-registering k=3 as optimal.)

The three protocols differ only in the reader interface. `no_coord` leaves the store unused. `pull` (agent-as-reader) prepends a curated peer-warnings summary to the agent's system prompt at trial start. `intercept` (framework-as-reader) prepends a `[PEER-WARNING]` block to the tool *response* when the call's (name, arguments) match a FAILED_TRIAL_ACTION event. Any pass^k difference between pull and intercept under matched payload is therefore attributable to the reader-side architecture, and we observe none at trial 1.

Setup and headline. 100 tasks (seed 42) $\times$ 3 protocols $\times$ 2 trials gives 600 trials in total. Both the assistant and the simulated user are `claude-haiku-4-5` at T=0, with a 20-turn budget per trial. The task-scoped store is shared across the 2 trials within a protocol cell. Figure 2 summarizes the result. Table 1 reports cell-level metrics; Table 2 reports the coordination-active paired tests.

Coordination-active head-to-head: no detectable effect at this power. The trial-1 outcome on the trial-0-failed subset was pre-specified as the architectural test (P4; §6.2): on those tasks the store contains peer events, so the coordination mechanism is active. Pull and intercept tie at trial-1 success rate 0.28 on the 29-task both-failed subset (paired sign test $W/L/T = 5/5/19$, $p = 1.0$). Each directionally underperforms `no_coord` on its own trial-0-failed subset (pull 0.143 vs. 0.286, n=28; intercept 0.20 vs. 0.343, n=35), but neither reaches significance at this subset size. After ties, the informative pairs number 10/8/17, well below the power needed to detect a medium-sized effect; "null" would overclaim, and we read this as no detectable effect at the available power rather than evidence that the mechanisms are inert. The unconditional trial-1 test over all 100 tasks returns identical ties. The takeaway is that the naive deployed-today coordination forms (last-K writers paired with either curated reading or per-match injection) do not transfer peer state into pass^k at this scale and model, and the directional evidence is at least consistent with mild interference rather than help; both readings would require a larger paired sample to separate.

Trial-0 decomposition: an empirical noise-floor estimate. At trial 0 the store is empty under every protocol, so the coordination mechanism is inactive. Of the three pairwise contrasts, only no_coord-vs-intercept is genuinely configuration-equivalent: code inspection (see `protocols.py`) confirms that both arms reduce to pure no-ops on the empty store. Pull's reader still re-enters its augmenter on every trial-0 turn (which returns input unchanged when `warnings_block()` is empty), so its contrast with the other two arms is not configuration-equivalent in the same sense, so its contrasts must be read separately from the clean one. Paired sign tests at seed-1 (n=100, Bonferroni m=3; the second seed follows below): no_coord-vs-intercept 21/11/68 (p_{corr}=0.330); pull-vs-no_coord 18/10/72 (p_{corr}=0.555); pull-vs-intercept 27/9/64 (p_{corr}=0.012). Wilson 95% CIs on the signed gaps are [−2, +16] pp, [−1, +19] pp, and [+6, +26] pp respectively: the first two cross zero, the third does not—at this seed.

Table 2: Coordination-active pass^k test (P4, defined in §6.2; §7): paired sign tests on trial-1 outcomes restricted to tasks where *both* compared protocols had a trial-0 failure, so both stores contain peer events and the coordination mechanism is necessarily active. Subset n varies by contrast. All three pairwise comparisons fail to reject on this subset; with 10/8/17 informative pairs after ties the test is underpowered for a medium-sized effect, so this is no detectable effect at the available power rather than a clean null. The recovery-rate ordering against the `no_coord` baseline shows both coordination architectures slightly underperforming no coordination at recovering from prior failure (W/L/T = wins/losses/ties).

Comparison	subset n	t1 succ. rate	W/L/T	sign p
pull vs. intercept	29	0.28 / 0.28	5 / 5 / 19	1.00
pull vs. no_coord	28	0.14 / 0.29	2 / 6 / 20	0.29
intercept vs. no_coord	35	0.20 / 0.34	6 / 11 / 18	0.33

Each contrast uses its own paired subset; rows are not cross-comparable. The high tie counts (19, 20, 18) leave informative samples of 10, 8, 17 tasks, so these sign tests are near-powerless and the directional recovery deficits are hypothesis-generating only. Unconditional trial-1 paired sign tests over all n=100 tasks return identical ties (pull-vs-intercept 13/13/74; pull-vs-no_coord 19/19/62; intercept-vs-no_coord 20/20/60; all p=1.0). This table conditions on trial-0 failure; the pass^2 column in Table 1 counts both-trials-success unconditionally over all 100 tasks. The two are arithmetically consistent: pull's pass^2=0.44 equals 0.71 $\times$ 62/100, where 0.71 is pull's conditional trial-1 success rate given trial-0 success.

Reproducibility: a second seed. We re-ran the full n=100 retail sweep at a second seed (`seed42`→`seed-2`, same task IDs, same harness, fresh API draws). Trial-1 rates remain effectively flat (0.520/0.540/0.510 for no_coord/pull/intercept); the head-to-head still shows no detectable effect. Trial-0 paired sign tests at seed-2: no_coord-vs-intercept 13/13/74 (0 pp signed gap, p_{corr}=1.0); pull-vs-no_coord 15/18/67 (−3 pp, p_{corr}=1.0); **pull-vs-intercept** 18/21/61 **(−3 pp, p_{corr}=1.0)**. The seed-1 +18 pp pull-vs-intercept significance does not reproduce; the sign even flips. We correspondingly retract E4 (pull-specific structural perturbation) as an established finding—a single seed's p_{corr}=0.012 on a contrast that pools to 45/30/125 (+7.5 pp, p_{corr}=0.316) over both seeds is exactly the artifact a paired noise-floor protocol should catch, and ours did.

Pooled across both seeds (n=200): no_coord-vs-intercept 34/24/142 (+5 pp, CI [−2, +12] pp, p_{corr}=0.711); pull-vs-no_coord 33/28/139 (+2.5 pp, p_{corr}=1.0); pull-vs-intercept 45/30/125 (+7.5 pp, CI [−1, +15] pp, p_{corr}=0.316). *No paired contrast is significant after Bonferroni at any single seed or pooled.* The clean-contrast pooled signed gap is $\approx$ 5 pp with Wilson upper bound $\sim$ 12 pp; the largest pooled

upper bound across any contrast is ∼ 15 pp (pull-vs-intercept). The across-seed observed range of signed gaps is [−3, +18] pp, with the upper end anchored on a single seed that did not replicate. We therefore characterize the configuration-equivalent floor as a paired-disagreement envelope whose pooled upper CI is $\lesssim$ 15 pp; calling it "10–18 pp" from one seed alone overclaimed the precision. The motivation for P4 (coordination-active pass^k, §6.2) stands: even a ∼15 pp envelope swamps the small headline gains recent coordination architectures report.

Mechanism diagnosis: three failure modes. (M1) Writer misattribution. The causally-relevant call is often upstream; last-K records downstream symptoms. A single-judge Haiku 4.5 LLM-judge audit on 30 analyzable trial-0 failures returns 77% UPSTREAM (direction-of-magnitude motivation, not a confirmed effect; see L4 in §6.6). *(M2) Per-match injection noise.* The intercept store accumulates $\bar{n}$=2.77 events per trial (max 8), and each matching call fires a separate warning. *(M3) Brittle keying.* Literal-tuple payloads do not transfer: `cancel_order(#W123)` does not match `cancel_order(#W456)` even when both target shipped orders. §7 discusses refinements matched to M1–M3.

Table 3: Minimum sample size (tasks per arm) to detect a candidate pass^k improvement Δ on τ^2-bench retail at α=0.05, power 0.80, baseline p_1=0.50. "Indep." is the two-proportion z-test for independent arms. "Paired" is the McNemar test calibrated on the seed-1 trial-0 discordance rate (p_d=0.32, `no_coord` vs. intercept; pooled across both seeds p_d=0.29, which barely moves the column). Read-out: τ^2-bench retail's public 114 tasks at one seed detect only Δ≥20 pp at the independent-arm rate (n=93), or Δ≥15 pp paired (n=110).

Δ (pp)	Indep. n	Paired n	τ^2-retail seeds*
5	1565	1003	impractical
10	388	249	4 seeds / 3 paired
15	**170**	**110**	**2 seeds / 1 paired**
18	117	76	2 seeds / 1 paired
20	93	61	1 seed / 1 paired
25	58	38	1 seed / 1 paired

* Number of full 114-task seeds required to reach n; "paired" counts the paired condition.

6.4 Cross-model and cross-domain probes

Two n=30 probes test whether the underpowered non-result and the floor are Haiku-retail-specific; Figure 3 summarizes the trial-0 gaps. The probes illustrate the metric rather than making coordination claims at this scale.

Sonnet 4.5 retail n=30. Trial-1 rates: `no_coord` 0.500, pull 0.733, intercept 0.733; raw pull-vs-no_coord p=0.039 does not survive Bonferroni m=3. The trial-0 gap magnitudes span 3.3–20 pp (overlapping the Haiku-retail observations, with the largest magnitude slightly above their upper edge) but with the direction flipped: pull starts −16.7 pp *behind* at trial 0, so its +23 pp marginal gain partly recovers an unrelated trial-0 deficit. The coordination-active metric makes this direction-flip visible; marginal pass[1] obscures it. Sonnet n≥250 is pre-registered as confirmatory.

Haiku airline n=30. Trial-1 rates: `no_coord` 0.500, pull 0.533, intercept 0.567 (all paired p>0.69); trial-0 gaps collapse to ≈ 0 pp, so the floor is domain-dependent. On the coordination-active subset (n=14), the direction matches retail (`no_coord` 0.357, intercept 0.286, pull 0.214): the naive last-K scheme does not help and plausibly impairs recovery.

6.5 From noise floor to deployment threshold

Converted to a release-time decision rule, the Haiku-retail envelope (pooled upper Wilson CI ≈15 pp, §6.3.2) becomes a *minimum detectable effect (MDE)* for sample-size planning within that regime. Because the floor varies by domain (Haiku airline ≈ 0 pp) and capability tier (Sonnet retail magnitudes 3.3–20 pp), the MDE table is regime-specific, and transport to other model/domain pairs requires re-running the paired protocol. Applying a two-proportion z-test at α=0.05, power 0.80, baseline p_1=0.50, Table 3 gives the required sample sizes.

What transfers, and a cheap probe. Re-running the full paired protocol per (model, domain) pair is the gold standard but not the entry cost. The floor's observed variation tracks two identifiable drivers: how much of the task set sits near the model's decision boundary (Haiku airline collapses to 0 pp gaps because most trial-0 outcomes are ties; Sonnet retail shifts which tasks are marginal, flipping gap direction), and the user-simulator's sampled-output entropy, which seeds the E3 drift. Both are estimable from a paired n=30 clean-contrast probe (baseline vs. an inert-hooked baseline) before any full sweep: our own n=30 probes are what surfaced the airline collapse and the Sonnet direction flip. The practical rule we propose: report a same-model paired probe alongside any coordination claim, and escalate to the full n≥100 protocol only when the claimed effect is within ∼2× the probe's upper CI.

Motivation only: literature gap-size context. Table 4 sets ten recent multi-agent coordination contributions next to the Haiku-retail pooled envelope: seven headline gains fall below the clean-contrast pooled upper CI (12 pp) and one more sits inside the 12–15 pp band up to the largest pooled upper CI. The ten are recent (2023–2026) multi-agent coordination architecture papers we found citing MAST [9] or CodeDelegator [12] and reporting a single-number headline architectural delta on a named benchmark; this is illustrative rather than systematic. The point is to motivate reporting same-model paired floors alongside future coordination claims, not to adjudicate the underlying systems in their original (different model, task, metric, n) settings, which would require per-row paired replication. We acknowledge the tension head-on: this paper argues against single-number cross-setting comparisons, and Table 4 is precisely that juxtaposition; we include it as a motivating prompt because the gap-size visibility outweighs the methodological cost, not because the comparison itself adjudicates anything.

Coordination-active pass^k is the minimum pre-deployment measurement gate that any coordination claim on τ^2-bench (and the broader τ-bench family of state-validated benchmarks) should pass; Table 3 ships the release-gate.

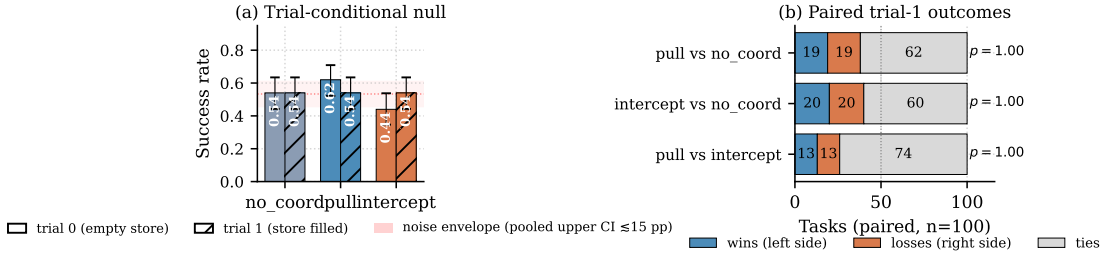

Figure 2: Headline visual (seed-1 of two n=100 seeds; §6.3.2 reports the second seed and pooled tests). The coordination-active head-to-head shows no detectable effect at the available power; the trial-0 paired gap is the noise-floor estimate. (a) Coordination-active head-to-head. Success rate by protocol at trial 0 (empty store, coordination inert) and trial 1 (store filled). `no_coord` and intercept are configuration-equivalent at trial 0 by code inspection; their +10 pp signed paired gap at this seed (Bonferroni p_{corr}=0.330, CI $[-2, +16]$ crosses zero) pools to +5 pp (CI $[-2, +12]$) across both seeds and is the clean empirical noise floor. Pull's +18 pp paired gap against intercept at this seed (p_{corr}=0.012) did not reproduce at seed-2 (-3 pp, p_{corr}=1.0; pooled +7.5 pp, upper CI ≈15 pp), so we read it as a single-seed draw from the envelope rather than an established pull-specific perturbation. At trial 1, where coordination is logically active, all three protocols collapse to 0.54. (b) Paired trial-1 outcomes over n=100 tasks. Ties dominate; all three sign tests fail to reject the no-difference null (p=1.0). Informative pairs after ties on the coordination-active subsets of Table 2 are 10/8/17, so this is no detectable effect at the available power rather than a clean null.

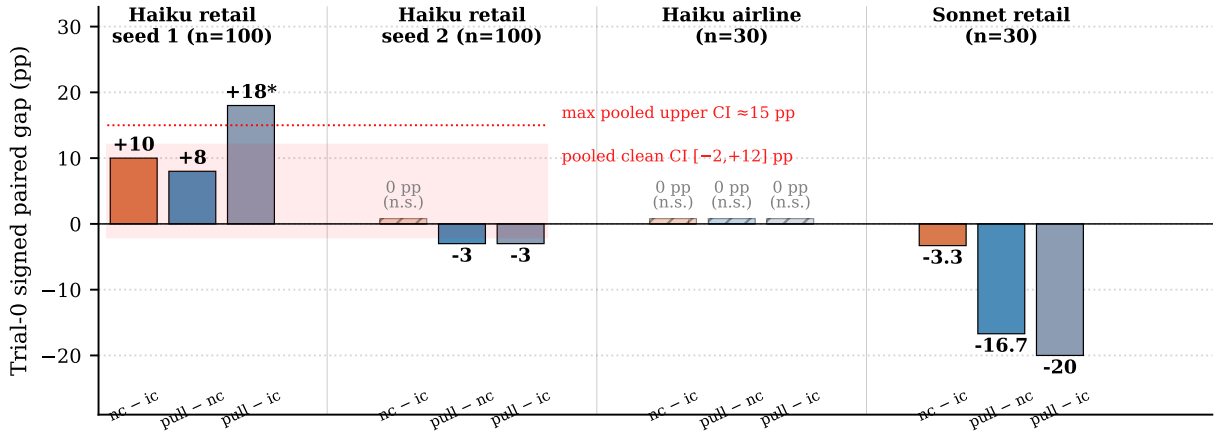

Figure 3: Trial-0 paired-sign-test signed gaps across seed, model, domain, and contrast. The noise floor is a measurement output of the coordination-active pass^k protocol, not a universal constant. Haiku retail: the seed-1 gaps (+10/+8/+18 pp) collapse to 0/−3/−3 pp at seed-2; the red band marks the pooled clean-contrast CI $[-2, +12]$ pp and the dotted line the largest pooled upper Wilson CI ≈15 pp. The starred seed-1 +18 pp bar is the non-replicating single-seed result retracted in §6.3.2. The floor is domain- and capability-dependent: Haiku airline at n=30 shows 0 pp gaps across all three contrasts, while Sonnet retail at n=30 spans magnitudes 3.3–20 pp with the direction flipped—pull starts trial-0 *behind* `no_coord` on Sonnet rather than ahead—so the 20 pp magnitude there sits slightly above the upper edge of the Haiku-retail observations.

Monitoring schema. Three runtime rules for AgentCore / SageMaker Model Monitor [23] deployments: (R1) CUSUM on coordination-active pass^k over a golden-task set, alarm at $\pm 1.5\sigma_{\text{floor}}$; (R2) rolling-$N$ 95th-percentile injection-event count, alarm at $\geq$ 50% above release baseline (right-skew defeats Gaussian 2σ); (R3) coordination-key collision rate, alarm at $> 0.5\%$.

Relation to release-gate and reliability work. The Anthropic infrastructure-noise study [3] reports 6 pp configuration gaps on SWE-Bench from CPU/RAM in a single-agent setting; Mustahsan et al. [21] propose an intraclass correlation coefficient (ICC) framework on FRAMES/GAIA; CI/CD quality gates [17] and behavioral drift monitoring [28] cover adjacent ground. Ours is the first MDE-grounded release-gate keyed on a coordination-architecture noise floor with store-conditional restriction on a state-validated coordination benchmark.

Table 4: Motivation examples for why same-model paired floors matter; not a verdict on the underlying systems. Each row's effect is shown alongside our Haiku-retail pooled envelope purely to motivate the protocol; the comparison is heterogeneous (different model, task, metric, n) and any individual row may be entirely real in its original setting. "vs. floor" marks whether the published gap is below the clean-contrast pooled upper CI of 12 pp (↓), inside the 12–15 pp band up to the largest pooled upper CI (=), or above 15 pp (↑); the floor itself varies by model and domain (§6.4). Read ↓ as *unresolved against a same-setting paired floor*, never as *refuted*.

Paper	Bench / Task	Effect	vs. floor
CA-MCP [15]	TravelPlanner BERTScore	+1.2 pts	↓
Terrarium [22]	Meeting Scheduling	1.0–2.7 pp	↓
AgentVerse [10]	HumanEval (Solo vs. Group)	+1.8 pp	↓
AutoGen [32]	MathChat over PoT/PS	∼+6 pp	↓
MetaGPT [14]	SoftwareDev quality	< 10 pp	↓
Silo-Bench [35]	GPT-OSS P2P vs. BP	+6 pp	↓
Reflexion [29]	HumanEval	+11 pp	↓
Ou et al. [24]	TravelPlanner orch.+notebook	+13.5 pp	=
Reflexion [29]	ALFWorld	+22 pp	↑
Ou et al. [24]	TravelPlanner combined	+17.5 pp	↑

6.6 Limitations

Six limitations bound the claims above. **L1 (identity-replay):** request-equivalence is verified by code inspection plus a 40/40 SHA-256 payload-hash audit on a separate n=10 retail sample (10 tasks × 4 first-request payload components, each byte-identical across all three protocols; §6.3.1); the headline n=100 sweep was not itself byte-audited, but the audit shows the request-assembly invariant holds. The Messages API exposes no seed, so the pooled envelope (upper CI ≈15 pp) upper-bounds the residual server-side stochasticity floor. **L2 (writer pilot-tuning):** the writer (K=3, single FAILED_TRIAL_ACTION type) was tuned on a prior n=30 sweep; $\text{PASS}^k_{\text{active}}$ is pre-specifiable in principle but was applied to already-collected data here. **L3 (within-protocol replicate + second seed).** A paired n=30 within-protocol audit gives trial-0 signed gaps of 0–3.3 pp across all three protocols (p=1.0), bounding pure API stochasticity at ≤ 3 pp. A second n=100 headline seed (§6.3.2) confirms that no single-seed paired contrast is significant once we measure twice: pooled across both seeds, all three trial-0 contrasts land at signed gaps of 2.5–7.5 pp with Wilson upper CIs ≤ 15 pp and Bonferroni $p_{\text{corr}} > 0.3$. Pull's seed-1 +18 pp gap against intercept was not reproduced (−3 pp at seed-2), so we retract E4 (pull-specific perturbation) as an established source. **L4 (single-judge M1):** the 77% upstream rate uses a same-model-class judge with no Cohen's κ or independent replication; M1 is a hypothesis for confirmatory work. **L5 (cross-model/-domain n=30):** Sonnet retail and Haiku airline show regime-specificity rather than coordination claims; Sonnet n≥250 is pre-registered. The n=100 headline's coordination-active subsets are tight (pull vs. intercept paired, n=29, 5/5/19), so the architectural head-to-head is an underpowered non-result rather than a clean null, and is weaker evidence than the measurement contribution. **L6 (trial-0 inertness):** the protocol assumes peer-coordination flows reader-side only; writer-side anticipation effects are out of scope. **Scope.** Haiku 4.5 retail carries the n=100 headline; outcome conversion is not detectable at the available power at this model and scale.

7 Forward Work

The coordination-active null (§6.3.2) isolated three failure modes of naive coordination, each with a matched refinement on existing production hook surfaces (Strands [2], AgentCore Policy [1], LangChain middleware [16]). **P1: causally-attributed writers** address *M1 writer mis-attribution*: an LLM-judge consumes the trajectory and reward and re-weights which trial calls get written, replacing the recency-only last-K heuristic. **P2: selective intercept** addresses *M2 per-match injection noise*: a score gate fires only when a peer event is relevant, novel, and decision-proximal (preliminary signal below). **P3: predicate constraint extraction** addresses *M3 brittle literal keying*: an extractor turns the trajectory into a typed predicate (e.g. `tool=cancel_order` with `order.status=shipped`) so intercept fires on the predicate, not the literal argument tuple. **P4: coordination-active pass^k** (§6.2) is the release-time gate any refinement should clear. P1 and P3 each require an LLM-judge component not validated here. A shared trace store also opens security surfaces (poisoned writes, cross-principal exfiltration); the prototype assumes a trusted single-tenant deployment.

P0: oracle positive control (implemented). The protocol shows naive coordination does not clear the floor but has not yet demonstrated that it *would* detect a real gain—a measurement-sensitivity gap a reviewer correctly flagged. We have implemented an `oracle` arm in the released harness: it rides pull's exact activation semantics (inert at trial 0, so the floor measurement is undisturbed) but, once a prior trial has failed, injects the task's *golden action sequence* from the benchmark's evaluation criteria—perfect coordination content on the same surface. Coordination-active pass^k against `no_coord` on the oracle arm upper-bounds what any reader-side mechanism could show on this benchmark. Run at the headline configuration (same 100 seed-42 tasks, Haiku 4.5, T=0), the oracle confirms trial-0 inertness (−3 pp paired gap vs. `no_coord`, inside the envelope) and lifts recovery on its trial-0-failed subset from 0.35 to 0.43 (trial-1 marginal 0.60 vs. 0.54)—but the paired coordination-active contrast is 8/6/21 (W/L/T) at n=35, p=0.79: *even ground-truth guidance does not separate at this n*. The direction is right, so the metric reads real gains; the binding constraint is the subset itself: 57% of oracle trial-1 attempts fail with the golden action sequence in the system prompt, i.e. most coordination-active failures on this benchmark are not knowledge-limited (user-simulator dynamics, policy-confirmation flows). This sharpens the negative reading of §6.3.2: no reader-side mechanism could have certified a gain here, because a perfect-content oracle cannot. Certifying coordination requires benchmarks whose coordination-active subsets are knowledge-limited by construction—or the pre-registered Sonnet n≥250 scale.

P2 (selective intercept), preliminary signal. A score-gated intercept that fires only when a peer event is relevant, novel, and decision-proximal beat vanilla intercept on the coordination-active

subset at Haiku 4.5 retail n=30 (+15.8 pp, paired sign 2/7/21, p=0.18; hyperparameters not pre-registered). Directionally consistent and underpowered; Sonnet at $n \geq 250$ is the natural confirmatory test.

8 Conclusion

The observed trial-0 paired-gap envelope on τ^2-bench retail spans $[-3, +18]$ pp across all three contrasts and two n=100 Haiku 4.5 seeds; the clean configuration-equivalent contrast (no_coord vs. intercept) gives +10 and 0 pp, pooling to +5 pp (CI $[-2, +12]$), with a pooled upper Wilson CI of ∼15 pp on the largest between-protocol contrast; no single-seed Bonferroni significance survives a second seed. ET-MCP, the task-scoped trace-store substrate, made the matched-payload comparison possible; the measurement discipline, not the substrate, is what we expect to travel. Notably, our own seed-1 pull-vs-intercept p_{corr}=0.012 did not reproduce at seed-2—a worked example of the protocol catching the single-seed artifact it is designed to catch.

Seven of ten recent coordination architectures report headline gains below that envelope and one more sits inside it. The coordination-active head-to-head here is an underpowered non-result, not a clean null: the informative pairs after ties are n=8–17, well short of what a medium-sized effect would require, so it reframes the question rather than retiring it. The negative reading applies to *naive* coordination under a measurement-honest protocol, not to coordination as a research program. P1–P3 (§7) are the mechanisms we expect to clear the floor, and coordination-active pass^k is the gate they should be judged against. Reporting a same-model paired floor alongside any multi-agent delta is the practice the field can adopt now, with the protocol here as a starting template.

References

[1] Amazon Web Services. 2025. Amazon Bedrock AgentCore Policy: Real-Time Tool-Call Boundary Enforcement. Product announcement. https://aws.amazon.com/about-aws/whats-new/2025/12/amazon-bedrock-agentcore-policy-evaluations-preview/

[2] Amazon Web Services. 2025. Strands Agents SDK: Hooks API for Agent Lifecycle Interception. Open-source SDK documentation. https://strandsagents.com/docs/user-guide/concepts/agents/hooks/

[3] Anthropic Engineering. 2026. Quantifying Infrastructure Noise in Agentic Coding Evals. Engineering blog post. https://www.anthropic.com/engineering/infrastructure-noise

[4] Victor Barres, Honghua Dong, Soham Ray, Xujie Jia, and Shunyu Yao. 2025. τ^2-bench: Evaluating Conversational Agents in a Dual-Control Environment. arXiv:2506.07982 [cs.CL] https://arxiv.org/abs/2506.07982

[5] Suhana Bedi, Y. Mlauzi, J. Shin, Sanmi Koyejo, and Nigam H. Shah. 2025. The Optimization Paradox in Clinical AI Multi-Agent Systems. In *KDD 2025 Workshop on Agentic and Generative AI Evaluation*. arXiv:2506.06574 [cs.AI] https://arxiv.org/abs/2506.06574

[6] BerriAI litellm community. 2025. Anthropic message history validation error after consecutive tool calls. GitHub issue. https://github.com/BerriAI/litellm/issues/15322

[7] BerriAI litellm community. 2025. Anthropic provider produces malformed `tool_use`/`tool_result` sequences on multi-turn tool calls. GitHub issue. https://github.com/BerriAI/litellm/issues/12404

[8] Bjarni Haukur Bjarnason, André Silva, and Martin Monperrus. 2026. On Randomness in Agentic Evals. arXiv:2602.07150 [cs.AI] https://arxiv.org/abs/2602.07150

[9] Mert Cemri et al. 2025. MAST: A Taxonomy of Multi-Agent System Failures from 1,600+ Production Traces. In *Advances in Neural Information Processing Systems (NeurIPS)*.

[10] Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chen-Ming Chan, Heyang Yu, Yaxi Lu, Yi-Hsin Hung, Chen Qian, Yujia Qin, Xin Cong, Ruobing Xie, Zhiyuan Liu, Maosong Sun, and Jie Zhou. 2024. AgentVerse: Facilitating Multi-Agent Collaboration and Exploring Emergent Behaviors. In *ICLR*. arXiv:2308.10848 [cs.CL] https://arxiv.org/abs/2308.10848

[11] Alejandro Cuadron and Amazon AGI. 2025. τ^2-Bench-Verified: Corrected Task Definitions for Tool-Agent Evaluation. GitHub release. https://github.com/amazon-agi/tau2-bench-verified

[12] Tianxiang Fei, Cheng Chen, Yue Pan, Mao Zheng, and Mingyang Song. 2026. CodeDelegator: Mitigating Context Pollution via Role Separation in Code-as-Action Agents. arXiv:2601.14914 [cs.CL] https://arxiv.org/abs/2601.14914

[13] Aayush Gupta. 2026. ReliabilityBench: Evaluating LLM Agent Reliability Under Production-Like Stress Conditions. arXiv:2601.06112 [cs.AI] https://arxiv.org/abs/2601.06112

[14] Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, et al. 2024. MetaGPT: Meta Programming for a Multi-Agent Collaborative Framework. In *ICLR*. arXiv:2308.00352 [cs.AI] https://arxiv.org/abs/2308.00352

[15] Meenakshi Amulya Jayanti and X. Y. Han. 2026. Enhancing Model Context Protocol (MCP) with Context-Aware Server Collaboration. arXiv:2601.11595 [cs.DC] https://arxiv.org/abs/2601.11595

[16] LangChain Team. 2026. LangChain Middleware: `wrap_tool_call` and `after_model` Hooks. Framework documentation. https://docs.langchain.com/oss/python/langchain/middleware/custom

[17] Alexandre Cristóvão Maiorano. 2026. Automated Self-Testing as a Quality Gate: Evidence-Driven Release Management for LLM Applications. arXiv:2603.15676 [cs.SE] https://arxiv.org/abs/2603.15676

[18] Microsoft. 2026. Microsoft Agent Framework: Function Middleware with Shared State. Framework documentation. https://learn.microsoft.com/en-us/agent-framework/agents/middleware/shared-state

[19] Microsoft Research. 2026. STATE-Bench: A Memory-Agnostic Benchmark for Stateful Agent Evaluation. Project page and blog post. https://opensource.microsoft.com/blog/2026/05/19/introducing-state-bench-a-benchmark-for-ai-agent-memory/

[20] Model Context Protocol Working Group. 2026. Model Context Protocol Specification, Release Candidate 2026-07-28. https://modelcontextprotocol.io/specification/2026-07-28/

[21] Zairah Mustahsan, Abel Lim, Megna Anand, Saahil Jain, and Bryan McCann. 2025. Stochasticity in Agentic Evaluations: Quantifying Inconsistency with Intraclass Correlation. arXiv:2512.06710 [cs.LG] https://arxiv.org/abs/2512.06710

[22] Mason Nakamura, Abhinav Kumar, Saaduddin Mahmud, Sahar Abdelnabi, Shlomo Zilberstein, and Eugene Bagdasarian. 2025. Terrarium: Revisiting the Blackboard for Multi-Agent Safety, Privacy, and Security Studies. arXiv:2510.14312 [cs.MA] https://arxiv.org/abs/2510.14312

[23] David Nigenda, Zohar Karnin, Muhammad Bilal Zafar, Raghu Ramesha, Alan Tan, Michele Donini, and Krishnaram Kenthapadi. 2022. Amazon SageMaker Model Monitor: A System for Real-Time Insights into Deployed Machine Learning Models. In *KDD*. https://www.amazon.science/publications/amazon-sagemaker-model-monitor-a-system-for-real-time-insights-into-deployed-machine-learning-models

[24] Long Ou, Saujas Vaduguru, and Daniel Fried. 2025. Analyzing Information Sharing and Coordination in Multi-Agent Planning. arXiv:2508.12981 [cs.CL] https://arxiv.org/abs/2508.12981

[25] Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. 2024. MemGPT: Towards LLMs as Operating Systems. In *COLM*.

[26] Kaiyang Qian, Xinmin Fang, and Zhengxiong Li. 2026. MPAC: A Multi-Principal Agent Coordination Protocol for Interoperable Multi-Agent Collaboration. arXiv:2604.09744 [cs.MA] https://arxiv.org/abs/2604.09744

[27] Stephan Rabanser, Sayash Kapoor, Peter Kirgis, Kangheng Liu, Saiteja Utpala, and Arvind Narayanan. 2026. Towards a Science of AI Agent Reliability. arXiv:2602.16666 [cs.LG] https://arxiv.org/abs/2602.16666

[28] Abhishek Rath. 2026. Agent Drift: Quantifying Behavioral Degradation in Multi-Agent LLM Systems Over Extended Interactions. arXiv:2601.04170 [cs.MA] https://arxiv.org/abs/2601.04170

[29] Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language Agents with Verbal Reinforcement Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*. arXiv:2303.11366 [cs.AI] https://arxiv.org/abs/2303.11366

[30] Ruipeng Wang, Yuxin Chen, Yukai Wang, Chang Wu, Junfeng Fang, Xiaodong Cai, Qi Gu, Hui Su, An Zhang, Xiang Wang, Xunliang Cai, and Tat-Seng Chua. 2026. AgentNoiseBench: Benchmarking Robustness of Tool-Using LLM Agents Under Noisy Condition. arXiv:2602.11348 [cs.AI] https://arxiv.org/abs/2602.11348

[31] Zhaohui Geoffrey Wang. 2026. AgentTrace: Causal Graph Tracing for Root Cause Analysis in Deployed Multi-Agent Systems. In *ICLR 2026 Workshop on Agents in the Wild*. arXiv:2603.14688 [cs.MA]

[32] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Jiale Liu, Ahmed Awadallah, Ryen W. White, Doug Burger, and Chi Wang. 2023. AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation Framework. In *ICLR 2024 LLM Agents Workshop*. arXiv:2308.08155 [cs.AI] https://arxiv.org/abs/2308.08155

[33] Zhongming Yu, Naicheng Yu, Hejia Zhang, Wentao Ni, Mingrui Yin, Jiaying Yang, Yujie Zhao, and Jishen Zhao. 2026. Multi-Agent Memory from a Computer Architecture Perspective: Visions and Challenges Ahead. arXiv:2603.10062 [cs.AR] https://arxiv.org/abs/2603.10062

[34] Guibin Zhang, Muxin Fu, Guancheng Wan, Miao Yu, Kun Wang, and Shuicheng Yan. 2025. G-Memory: Tracing Hierarchical Memory for Multi-Agent Systems. arXiv:2506.07398 [cs.MA] https://arxiv.org/abs/2506.07398

[35] Yuzhe Zhang, Feiran Liu, Yi Shan, Xinyi Huang, Xin Yang, Yueqi Zhu, Xuxin Cheng, Cao Liu, Ke Zeng, Terry Jingchen Zhang, and Wenyuan Jiang. 2026. Silo-Bench: A Scalable Environment for Evaluating Distributed Coordination in Multi-Agent LLM Systems. arXiv:2603.01045 [cs.MA] https://arxiv.org/abs/2603.01045

[36] Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. 2024. ExpeL: LLM Agents Are Experiential Learners. In *AAAI Conference on Artificial Intelligence*. arXiv:2308.10144 [cs.AI] https://arxiv.org/abs/2308.10144