From LLM-Judged Scores to Artifact-Grounded Evaluation of Tool-Using Agents

A measurement framework and a retrospective of 1,785 synthetic runs

Protocore Research

July 16, 2026

Abstract

Tool-using agents leave more evidence than a final answer: they mutate artifacts, emit tool traces, transition through runtime states, and depend on fallible infrastructure. Collapsing these observations into one score makes important disagreement unobservable. We present a five-layer evaluation framework separating artifact state, trace policy, semantic judgment, runtime state, and infrastructure validity. We apply it retrospectively to three dated snapshots of a Protocore-owned synthetic suite, comprising 1,785 runs in total. Outcome analysis is restricted to the 666-run June snapshot: 30 runs received a non-completed runtime state despite a passing semantic verdict, while 102 completed runs failed that verdict. A later 381-run snapshot is retained only for instrumentation analysis because a known shared-tool outage was not reliably reclassified in its legacy aggregate. Deterministic checks covered 61.3% of June runs and 66.1% of the later snapshot, but no judge-by-objective cross-tab was retained. The evidence supports a reporting protocol, not an empirical artifact-versus-judge claim, model ranking, or longitudinal quality claim. We release only sanitized aggregates and reproducible figure code; prompts, outputs, traces, and private deployment metadata are excluded.

1 Introduction

Interactive language agents differ from chat systems in one decisive respect: their answer is only one projection of a stateful execution. Hypothetically, a run could create a correct file and then time out while preparing a response; conversely, it could end cleanly with fluent prose while omitting the requested artifact or violating a tool-use constraint. These cases motivate layer-specific measurement, but the retained aggregates in this study do not cross artifact checks with judge verdicts and therefore do not establish either case empirically.

Established evaluation work argues for broad scenarios and multiple metrics

(Liang et al. 2022), interactive environments (Liu et al. 2024), and executable end-state checks (Jimenez et al. 2023; Yao et al. 2024). LLM judges make open-ended semantic evaluation scalable, but have known position, verbosity, and self-enhancement biases (Zheng et al. 2023). Tool-using systems add another problem: neither a semantic judge nor the executor's terminal state is a complete oracle for task success.

This paper contributes:

1. a five-layer measurement model that assigns artifact, trace, semantic, runtime, and infrastructure observations separate meanings;
2. a retrospective analysis of three Protocore-owned synthetic evaluation snapshots containing 1,785 runs;
3. a reporting protocol that keeps denominators and disagreement visible; and
4. a public-safe aggregate release with reproducible figures and explicit provenance gaps.

We make no claim that the snapshots rank models or show quality improvement over time. Their suites, runtimes, and measurement coverage changed.

2 Measurement model

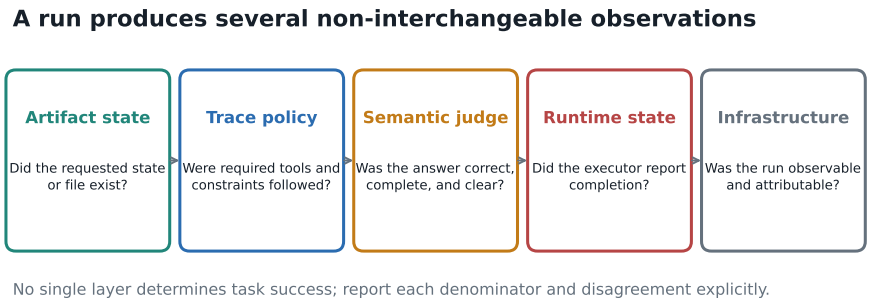

Figure 1: Five evidence layers for a tool-using agent run.

We define an evaluation record as

$$E(r) = (A_r, T_r, J_r, S_r, I_r),$$

where A_r is artifact state, T_r is trace-policy compliance, J_r is a semantic judgment, S_r is runtime terminal state, and I_r is infrastructure validity. The components answer different questions:

- **Artifact state:** Did the requested file, structured object, or external state exist, and did it satisfy deterministic predicates?
- **Trace policy:** Were required or forbidden tools, call counts, and ordering constraints respected?
- **Semantic judgment:** Was the response correct, complete, clear, efficient, and appropriate according to a fixed rubric?
- **Runtime state:** Did the executor report a completed, partial, failed, or unresolved run?
- **Infrastructure validity:** Was the run observed end to end, or should it be excluded or separately reported because the transport or execution substrate failed?

The framework deliberately avoids defining success as a conjunction of all five components. Some tasks permit many valid traces; others require a specific artifact or safety policy. The task specification must declare which components are applicable before execution.

3 Data and methods

3.1 Synthetic task suite

The retrospective uses only aggregate results from internally authored synthetic tasks. The tasks span coding, debugging, document generation, error recovery, file operations, long context, multi-tool workflows, multi-turn coherence, multilingual behavior, planning, retrieval, refactoring, safety, and delegation. Later suites added further synthetic task families and deterministic checks.

No production conversations, user data, raw prompts, model outputs, tool arguments, or deployment identifiers are included in the publication dataset. Configuration labels are intentionally neutral where the retained aggregate did not preserve a complete evaluated-model revision.

3.2 Dated snapshots

Snapshot	Configuration	Prompts × seeds	Runs	Judge pass	Non-completed	Objective pass	Analysis use
2026-05-22	Configuration A	82 × 3	246	186/246 (75.6%)	50	not recorded	judge-only description

Snapshot	Configuration	Prompts × seeds	Runs	Judge pass	Non-completed	Objective pass	Analysis use
2026-05-22	Configuration B	82 × 3	246	191/246 (77.6%)	50	not recorded	judge-only description
2026-05-22	Configuration C	82 × 3	246	190/246 (77.2%)	50	not recorded	judge-only description
2026-06-10	Runtime Q	111 × 3	333	258/333 (77.5%)	30	149/204 (73.0%)	outcome analysis
2026-06-10	Runtime D	111 × 3	333	270/333 (81.1%)	36	163/204 (79.9%)	outcome analysis
2026-07-01	Runtime D (expanded suite)	127 × 3	381	excluded	excluded	excluded	instrumentation only

The first snapshot evaluated 82 prompts with three seeds under three configurations: $82 \times 3 \times 3 = 738$ runs. A fixed judge scored five dimensions from 1 to 5; a run passed only when every dimension was at least 3. Its retained aggregate is judge-centric and does not contain deterministic artifact results or a status-by-verdict table.

The second snapshot expanded the suite to 111 prompts and evaluated two runtime configurations with three seeds: $111 \times 3 \times 2 = 666$ runs. The report added objective checks for the 204 applicable runs in each configuration, plus runtime-state by judge-verdict contingency tables.

The third snapshot expanded the suite to 127 prompts for one configuration and three seeds: $127 \times 3 = 381$ runs. Objective checks applied to 252 runs. This snapshot also recorded one explicitly unresolved infrastructure run.

A later forensic review found that a shared tool dependency outage affected approximately 15 runs in the third snapshot, while its legacy aggregate explicitly classified only one run as infrastructure-unresolved. The sanitized aggregate does not retain enough information to reclassify the remaining affected runs

without reintroducing private trace data. We therefore retain this snapshot for instrumentation coverage and corpus documentation only; none of its outcome rates contribute to agent-performance inference.

Together the snapshots contain $738+666+381 = 1{,}785$ runs. This total describes the reviewed evidence corpus; it is not a pooled estimate because the suite and configuration changed.

3.3 Judge and deterministic checks

The retained reports used the same fixed judge identifier and a five-dimension rubric. Judge pass rates use all runs as denominator. Objective pass rates use only runs for which at least one deterministic policy or artifact check was applicable. Examples of check classes include required or forbidden tool calls, call ordering, file existence, required content, forbidden content, and structured-output predicates.

The aggregates do **not** include a judge-by-objective contingency table. Consequently, the difference between judge and objective pass rates cannot be interpreted as a disagreement rate. It is only a difference between measurements on partially overlapping populations.

3.4 Reproducibility

Every published row has a dated evidence record, sample size, source-aggregate hash, eligibility flag, and limitations statement. The generator validates schemas, unique keys, joins, arithmetic, coverage rates, and evidence consistency before consuming the sanitized CSV files. A separate verifier checks all source hashes during private extraction without recording private paths. Two legacy gaps remain: complete source revisions and decoding parameters were not preserved for all snapshots. We therefore use the records for measurement analysis, not fine-grained model comparison.

4 Results

4.1 Runtime state and semantic verdict disagree

Configuration	Completed/pass	Completed/fail	Non-completed/pass	Non-completed/fail
Runtime Q	245	58	13	17
Runtime D	253	44	17	19

Across the two configurations in the June snapshot, 30 of 666 runs (4.5%) were

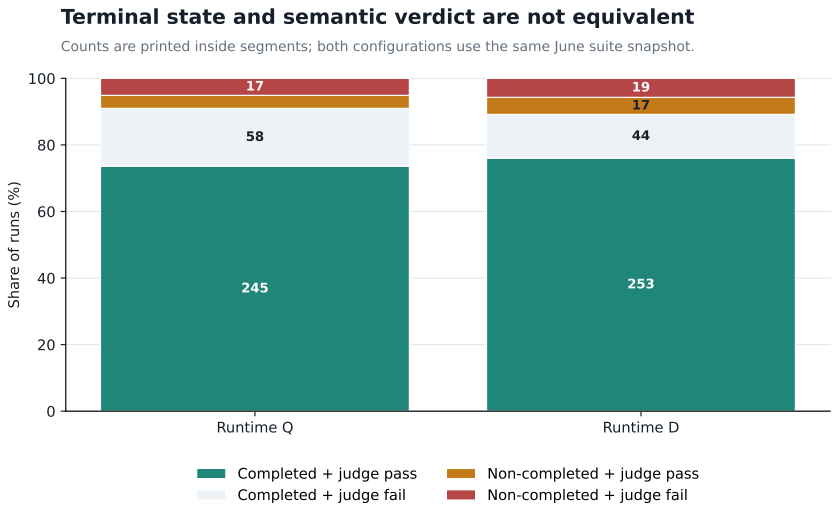

Figure 2: Runtime terminal state crossed with semantic judge verdict.

judged passing despite a non-completed runtime state: 13/333 and 17/333. This is direct evidence that terminal completion is not equivalent to semantic acceptance in the retained June data.

The converse also occurred. Across the same snapshot, 102 runs completed but failed the semantic rubric: 58 and 44 respectively. A completed status therefore does not establish semantic success either.

These are descriptive counts. Repeated seeds share prompts, and the two configurations are not independent random samples from a task population. The infrastructure-confounded July cross-tab is excluded from this analysis.

4.2 Deterministic evidence was incomplete

Objective checks covered 61.3% of runs in each June configuration. More specific trace-policy and artifact coverage was lower at 45.0% and 52.3%. The corresponding July coverage values—66.1%, 52.0%, and 58.3%—describe instrumentation eligibility only and are unaffected by our exclusion of July outcome rates.

Within the applicable June subsets, objective checks passed for 149/204 runs (73.0%) and 163/204 (79.9%). Corresponding semantic judge rates over all June runs were 77.5% and 81.1%. Because eligibility and denominators differ, these pairs must not be read as calibrated estimates of judge error or empirical artifact-versus-judge disagreement. The July marginal outcome rates remain in the data release for auditability but are excluded from this comparison.

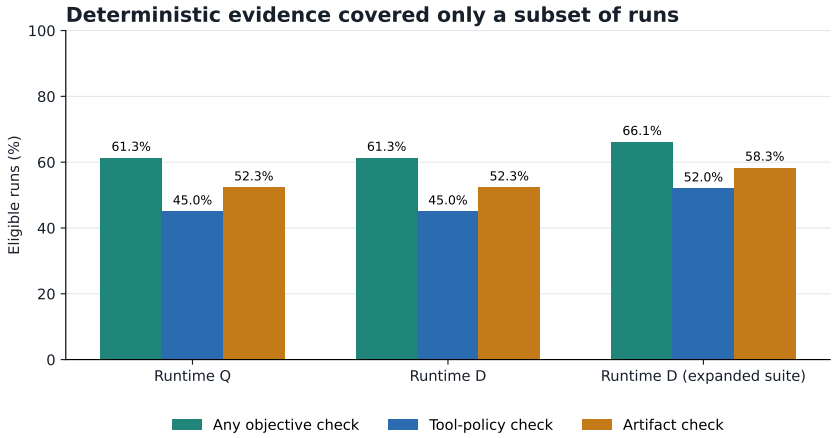

Figure 3: Coverage of deterministic measurement layers.

4.3 Judge-only aggregates cannot recover execution truth

The 738-run May snapshot produced judge pass rates of 75.6%, 77.6%, and 77.2% across its three configurations. It also recorded 50 non-completed runs in every configuration, but not their crossing with judge verdict. The aggregate can answer how the fixed rubric scored final responses. It cannot answer how often correct artifacts survived a failed status, how often fluent answers lacked artifacts, or whether the same infrastructure mode affected all configurations.

This negative result is methodological: once layer-specific evidence is discarded, it cannot be reconstructed from an overall score.

5 Reporting protocol

We recommend the following minimum record for each evaluation run:

1. a stable task and configuration identifier, with model and judge revisions plus decoding parameters;
2. applicability declarations for artifact and trace checks;
3. deterministic check results with machine-readable failure reasons;
4. semantic scores and judge provenance;
5. runtime state recorded independently of task success;
6. infrastructure validity and exclusion reason;
7. sanitized trace and artifact hashes sufficient to audit aggregation; and
8. repeated trials reported by prompt, not only as an unclustered run total.

Aggregate reports should publish a status-by-verdict table and, where applicable, judge-by-objective and artifact-by-objective tables. Rates must name their denominator. Infrastructure failures should remain visible rather than being

silently converted to task failures or removed.

6 Threats to validity

Construct validity. Deterministic checks are precise but incomplete specifications. A required tool sequence can reject a valid alternative implementation; a weak artifact predicate can accept a superficial result. Semantic judges assess open-ended quality but inherit rubric and model biases.

Internal validity. The study is retrospective. Task definitions, runtime behavior, and measurement coverage changed between snapshots. Complete decoding configuration and source revisions were not retained for every legacy aggregate. A known shared-tool outage was incompletely classified in July; that snapshot is excluded from outcome inference rather than treated as clean agent-performance evidence.

Statistical conclusion validity. Seeds repeat the same prompts, so runs are clustered. We report counts and rates without treating all runs as independent observations. The data are insufficient for causal attribution or model ranking across dates.

External validity. The suite is synthetic and centered on one agent runtime. Results establish measurement failure modes in this corpus, not their prevalence in every agent system or production workload.

Reproducibility. Public release is limited to sanitized aggregates and generation code. Withholding raw traces protects private-system information but prevents independent re-judging. Future evaluations should be designed with public synthetic prompts and trace schemas from inception.

7 Discussion

Artifact-grounded evaluation is not a proposal to replace judgment with tests. It is a proposal to stop asking one measurement to answer every question. Semantic quality, state transition, tool policy, and infrastructure reliability are operationally distinct. The retained status-by-verdict table demonstrates the first kind of disagreement. Artifact-by-judge disagreement remains a hypothesis for a future per-run cross-tab, not a result of this retrospective. Widespread or unclassified infrastructure invalidity blocks outcome conclusions entirely, as the July exclusion illustrates.

The retrospective also illustrates why benchmark evolution must be versioned. Adding deterministic checks improves observability but changes what a reported score means. A larger suite may be more representative while remaining incomparable with its predecessor. Versioned task manifests and explicit measurement coverage are therefore part of the result, not administrative metadata.

8 Conclusion

Across a 1,785-run dated corpus, the reported outcome claims are restricted to the 666-run June snapshot. Terminal state and semantic verdict disagreed in both directions there: 30 runs were non-completed yet judge-passing, while 102 were completed yet judge-failing. Deterministic artifact and policy checks applied to only 61.3% of June runs, and no artifact-by-judge contingency was retained. The later snapshot's 66.1% objective coverage remains useful as an instrumentation fact, but its outcome rates are excluded because of incompletely classified infrastructure invalidity.

The appropriate response is not another composite score. It is a layered record that preserves artifacts, traces, semantic judgment, runtime state, and infrastructure validity with explicit applicability and denominators. Future work will release an immutable, fully synthetic suite with per-run layer cross-tabs, multiple judges or human calibration, and prompt-clustered uncertainty estimates.

9 Data availability

Sanitized aggregate CSV files, an evidence manifest, and the figure generator are distributed with this paper. Raw prompts, responses, traces, and private execution metadata are not part of the public release.

Jimenez, Carlos E., John Yang, Alexander Wettig, et al. 2023. "SWE-Bench: Can Language Models Resolve Real-World GitHub Issues?" *arXiv Preprint arXiv:2310.06770*, ahead of print. https://doi.org/10.48550/arXiv.2310.06770.

Liang, Percy, Rishi Bommasani, Tony Lee, et al. 2022. "Holistic Evaluation of Language Models." *arXiv Preprint arXiv:2211.09110*, ahead of print. https://doi.org/10.48550/arXiv.2211.09110.

Liu, Xiao, Hao Yu, Hanchen Zhang, et al. 2024. "AgentBench: Evaluating LLMs as Agents." *International Conference on Learning Representations*. https://arxiv.org/abs/2308.03688.

Yao, Shunyu, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. 2024. "τ-Bench: A Benchmark for Tool-Agent-User Interaction in Real-World Domains." *arXiv Preprint arXiv:2406.12045*, ahead of print. https://doi.org/10.48550/arXiv.2406.12045.

Zheng, Lianmin, Wei-Lin Chiang, Ying Sheng, et al. 2023. "Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena." *Advances in Neural Information Processing Systems* 36. https://arxiv.org/abs/2306.05685.