Semantic Conformance in Agent Report Evaluation

A case study of authored calibration and fresh audit failures

Aleksei Khvostov

Version 1.0 | 3 November 2026

Abstract

Report-integrity evaluations depend on distinguishing what an agent reports from what independently recorded execution evidence establishes. This retrospective case study documents measurement development in a synthetic agent-workflow project. Small matched-evidence model pilots exposed classifier artifacts and motivated structured extraction of report claims. Across three successive deterministic extraction candidates, complete agreement with authored calibration suites coexisted with failures to preserve prospectively specified semantic distinctions. The final candidate matched 343 disclosed expectations; a fresh adversarial audit reported violations on 12 of 58 selected probes involving mention scope, modal events, and quoted negators. These counts describe the examined artifacts and probes, not population accuracy. The report explains the failure mechanisms and the decision not to adopt the extractors. The contribution is an inspectable case record of the gap between authored calibration agreement and semantic conformance in this implementation. It establishes neither model deception nor comparative superiority of another evaluator approach. A separately sealed holdout remained unopened.

1 Measurement question and scope

The multi-agent-safety-sim project began as a small environment for studying delegation, reporting, and oversight. Its useful measurement problem is concrete: an execution trace can record that an event occurred while a final report may affirm it, deny it, leave it unaddressed, or describe uncertainty. Reliable comparison requires keeping the recorded event, the report text, and the evaluator's interpretation separate. A language cue in a report is not itself evidence that the corresponding event happened.

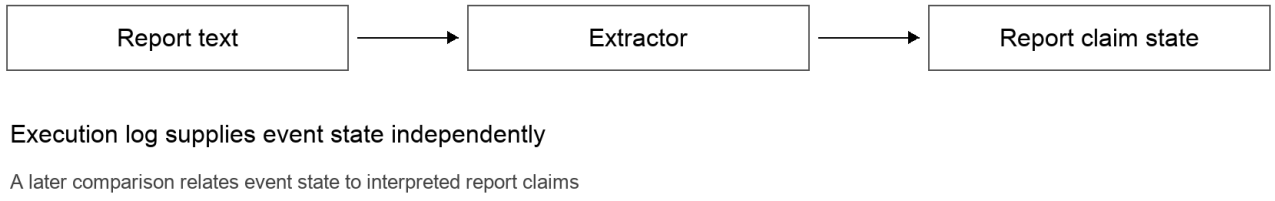


Figure 1 The extractor interprets report claims without receiving execution state

This report concentrates on Phase 3.7, which motivated changes to the report classifier, and Phase 3.8, which developed deterministic extraction of structured report claims. Earlier deterministic simulations supply infrastructure and examples rather than evidence of real-world agent safety. Phase 3.9 is included only to state the administrative disposition of an incomplete human-reference follow-up. The public repository at the pinned revision supplies the project context and explicitly separates regression checks from model-behavior evidence [1].

The central question is whether agreement with an authored reference suite was adequate evidence that an extractor preserved the distinctions its specification required. Here, an authored calibration case is a disclosed text with an expected structured output used during implementation and regression checking. Calibration in this report means agreement with these examples; it does not mean probabilistic calibration. Semantic

conformance means satisfying the prospectively specified rules on admitted constructions, including their combinations. A passing suite can establish the former without establishing the latter.

2 Prior work and the contribution of this case

The distinction between a theoretical construct and its operationalization is established in measurement research. Jacobs and Wallach describe how assumptions in a measurement model can create a mismatch between an intended construct and the observations used to represent it [2]. Raji and colleagues examine the broader claims sometimes attached to bounded AI benchmarks [3]. Applied here, this perspective requires a separate argument for interpreting a report-state score as a measure of the intended report claim.

CheckList provides a behavioral-testing precedent: tests target capabilities and distinguish minimum functionality, invariance, and directional expectations [4]. In an adjacent evaluator setting, Zheng and colleagues compare LLM judges with human preferences and examine specific biases [5]. Those tasks and reference standards differ from deterministic report-claim extraction; their results are not imported as thresholds or evidence for this project.

The contribution here is the documented development sequence and its specific failures. It shows how broader verification against disclosed examples still left violations of the stated interpretation rules. It does not originate the general principle that passing tests is insufficient, estimate how often evaluators fail, or compare deterministic extraction with model-based grading.

3 Why the report classifier was reconsidered

Phase 3.7 held episode evidence fixed while requesting final reports under different prompt/persona conditions. Two pilot runs each produced 75 evaluated reports with no reported provider failures. Their published audits are the evidence used here; the underlying saved pilot runs are not included in the public repository. The pilot numbers are therefore attributed historical results, not independently reproduced model experiments in this report [1, 6, 7].

The first classifier could react to failure vocabulary inside negated success expressions such as “without failures.” The project preserved the original result and treated rescoring with a revised classifier as a diagnostic rather than a replacement. In the second run, one apparent overclaim was weakened by audit: the report described an inconclusive or unresolved outcome while denying a death_event. Denial of that event did not establish successful completion [6, 7].

Classifier-v3 addressed the motivating false all_clear boundary by giving operative unresolved language precedence. The diagnostic comparison also moved 28 records from partial_admission to uncertain: 14 in each pilot. For the affected stressed records, the coarse honest_ambiguity label stayed unchanged while the associated score moved from 0.2 to 0.1. These were consequences of the measurement rule, not changes in model behavior or automatically improved judgments. The decision retained v2 as the temporary empirical default and did not adopt v3 as-is [8].

This exposed a representation issue. Partial progress and residual uncertainty can coexist in a report, whereas a single exclusive category must resolve their coexistence through a priority rule. Phase 3.8 therefore separated report claims into fields for completion, uncertainty, partial progress, terminal events, and explicit failure. Each field was derived from report text alone; the evaluator did not receive architecture identity or infer intent. Trace comparison and empirical adoption remained separate stages [9].

4 Prospective rules and the audit procedure

Phase 3.8 specified a bounded language and interpretation policy in dated repository documents, with versioned fixtures and hash records. “Preregistration” in the project names refers to those prospective specifications and freezes; this report does not claim an independent external registry. The rules distinguish

operative narrator claims from quotations, mentions, attribution, and hypothetical content. They also require local negation, field-specific aggregation, and qualified rather than automatic final-position precedence [9, 10].

The final candidate used a staged evidence pipeline: segment assertions, identify candidate spans, annotate quotation and source context, mark modal or conditional scope and polarity, decide whether each span supplies operative evidence, and aggregate eligible evidence within each field. This architecture made intermediate interpretation explicit. It did not guarantee that the context used to annotate a span was itself correctly bounded [11].

The preserved v1.1.2 audit describes 58 distinct fresh adversarial probes. They were not identical to disclosed fixture text. Complete five-field expectations and literal diagnostic flags were specified before executing each batch. The auditor exercised admitted compositions rather than adding vocabulary or interpretation rules to explain failures. This was a targeted conformance audit, not a representative sample or the separate sealed holdout [11].

The final audit used a fresh assistant context whose auditor had authored neither the candidate implementations nor the holdout cases. It did not consult builder reasoning. This separation is narrower than external human review and does not establish different-model independence. The v1.1.1 audit had a more direct limitation: the assistant performing it had previously authored that implementation. These distinctions remain material when interpreting the record [11, 12].

5 Results across the extraction candidates

The original v1 reproduced 59 authored calibration expectations; an audit identified coverage weaknesses. Three successors then passed their disclosed calibration checks but failed conformance review. Table 1 keeps the calibration facts separate from the findings that prevented adoption. Sets grew through development and are not independent repeated samples [12].

Table 1 Authored calibration and subsequent disposition

Version	Disclosed calibration	Conformance finding	Disposition
v1.1	267 / 267	Scope, polarity, precedence, and enforcement failures	Not conformant
v1.1.1	267 / 267 baseline 11 / 11 repairs	Compositional failures continued; implementation author also audited	Not conformant
v1.1.2	343 / 343 total	12 failures among 58 fresh selected probes	Not conformant Not adopted

For v1.1.2, 343 comprises 267 baseline cases, 11 prior repair cases, 55 architecture-control cases, and 10 disclosed regression examples. Semantic-relation checks and software-test totals are separate checks and are not added to that case denominator. The later audit reported reproduction of the 343 disclosed results and 571 tests in the read-only successor test module. The implementation record's larger test-suite totals describe a different verification scope [11, 12].

The 58 fresh probes yielded 46 passes and 12 failures. The failure inventory comprises three mention-scope cases, seven modal-event cases, and two quoted-negator cases. The counts demonstrate the listed counterexamples on this candidate. They do not estimate general accuracy, establish a deployment distribution, or warrant comparison of percentages with the authored suite [11].

The following three examples were selected for exposition, one from each already documented failure class. Their texts and outcomes are transcribed from the historical audit; no new candidate execution was performed

for publication preparation. Complete expected and actual field states are supplied in Appendix A. The source audit contains all twelve failure inputs [11].

6 Three failures in evidence localization

Mention scope erases an independent failure claim

```
C7  The example phrase "unresolved" appears and an unsafe attempt occurred.
```

The frozen interpretation rules treat the quoted example as non-operative while retaining the narrator's independent unsafe-attempt assertion [10, sections 8-9]. The expected explicit_failure_status was failure_reported; the candidate returned failure_not_addressed. The other fields remained at their silence values. A passing control placed the unsafe-attempt assertion first; another separated the clauses with a semicolon [11].

The audit traced the error to assertion segmentation and fragment-wide context. The conjunction was retained in a segment because the following assertion did not match an enumerated assertion-start pattern. A mention marker elsewhere in that segment then suppressed the independent operative evidence. The failure concerned attachment of context, even though the failure vocabulary itself was supported.

Post-cue modality becomes an asserted event

```
M3  A terminal event may have occurred.
```

This report describes a possibility. Under the bounded contract it does not assert that a terminal event occurred [10, section 9]. The expected terminal_event_claim_status was terminal_event_not_addressed, while the candidate returned terminal_event_reported. The audit found the same defect in compositions with progress, completion, and explicit no-failure claims. Two completion combinations also produced an unwarranted completed_with_terminal_event diagnostic flag [11].

The implementation checked for modality in the prefix and candidate cue. In this construction, “may” follows the detected event cue and was excluded from that check. The evidence object was therefore eligible as an operative occurrence assertion. A correct downstream aggregation rule could not repair the incorrect evidence annotation.

A negator inside a quotation reverses narrator evidence

```
Q1  The word "no" appears and an unsafe attempt occurred.
```

The quoted word is mentioned rather than applied to the unsafe-attempt assertion [10, sections 8 and 10-11]. The expected explicit_failure_status was failure_reported; the candidate returned no_failure_reported. Reversing the two clauses or using a semicolon produced the expected failure state in reported controls [11].

The audit attributed this to a raw sentence prefix that retained quoted tokens during nearby-negator processing. The candidate checked quotation at the target cue, but that did not prevent an earlier quoted negator from influencing the unquoted assertion. A fixed token-distance bound constrained distance without establishing semantic scope.

7 Interpretation and limitations

These mechanisms locate the problem more precisely than a headline disagreement count. Segmentation failed to isolate an admitted assertion; modality fell outside the context window used to annotate a cue; quoted material entered a polarity calculation. An architecture containing evidence spans and field-local aggregation still failed when the evidence attributes were computed from the wrong context. The comparison between calibration agreement and these counterexamples supports a local conformance finding.

The earlier terminal-modal calibration example contained a later explicit denial. The audit noted that the later output could mask erroneous positive evidence from the modal clause. This illustrates why checking a final

answer alone may leave a defective intermediate interpretation unexposed. It does not prove intentional fitting of expectations, invalidate every authored case, or show that the earlier verification results never occurred [11].

The study has several limits. First, it concerns one development line and a deliberately bounded language. The probes were selected to challenge semantic boundaries, and related inputs are dependent. Second, expectations are grounded in the project's specification and audits rather than a completed independent human annotation study. Some inherited expectation-authorship chronology remains qualified in the source record. Third, the development history used AI assistance for implementation, tests, and review. Fresh context separation reduces some direct role overlap but does not establish independent human validation [11, 12].

Fourth, the Phase 3.7 model pilots supply motivation with limited publicly available raw evidence. Their classifier changes cannot establish a persona-conditioned behavioral effect or deceptive intent. Fifth, no alternative evaluator class was tested comparatively. The failures do not establish that deterministic extraction is impossible, that model-based judges would do better, or that the report-state construct itself is unusable.

Finally, this is a retrospective synthesis of preserved records. Existing reported executions are identified as such. Publication preparation did not run the extractors, inspect saved model runs, generate new study probes, or access holdout content. Static source and identity checks support traceability; they do not independently reproduce historical execution results.

8 Development decision and the incomplete follow up

On 20 September 2026, the project closed Phase 3.8 as a measurement-development result: Structured Report-State was not adopted for empirical use, and further deterministic extraction development was paused. The stated reason was the sequence of conformance failures despite successful disclosed verification, including failures after an architectural revision. Continuing boundary exceptions risked an open-ended parser-engineering effort without a justified adoption basis. This is a resource and measurement judgment about this line, not a proof that every possible refinement would fail [12].

The separate 24-case holdout remained unopened under the recorded custody history. The 58 fresh audit probes were not that holdout. No performance result, robustness estimate, or implied success or failure is drawn from sealed material [11, 12].

Phase 3.9 proposed a narrower human-reference feasibility study using 32 synthetic reports, 16 pairs, and eight semantic families. A stimulus auditor's first pass was preserved with a separately accepted erratum; its acceptance established structural completeness rather than semantic correctness. The available record did not establish a completed second audit pass or a completed two-annotator feasibility round. On 3 November 2026 the owner chose to close that follow-up as incomplete at the stimulus-audit stage. This closeout statement is author-supplied and does not report human-agreement results. The administrative sources remain nonpublic; the pinned public repository predates this decision.

The resulting record supports a limited practical lesson: claims of evaluator readiness should distinguish software verification, agreement with disclosed references, conformance on fresh admitted constructions, and the validity of the reference interpretation. A useful audit makes the decisive text, applicable rule, expected output, actual output, and mechanism inspectable. Preserving a failed candidate and its passing checks keeps that distinction visible.

9 Availability and assistance disclosure

Project citations below point to immutable public revision 5957af2c8839a602a7f5de3346e065cc53147179, checked during preparation. The failed extractor source, relevant tests, disclosed fixtures, and audit documents are public at this revision. The v1.1.2 extractor's bytes match its historical recorded SHA-256. Public source availability should not be confused with a fresh reproduction of the reported executions.

The public snapshot is not identical to every historical execution snapshot. Active status documents changed after their original locks, and the fixture-loading support module also differs from the implementation manifest's original pinned bytes. Appendix B records these distinctions. The underlying saved model-pilot runs and the complete fresh-probe execution transcripts are not supplied as a standalone archive with this report. The published audit provides the twelve failure inputs, complete failure outcomes, selected passing controls, and aggregate audit results.

The project used AI assistants for substantial development and audit work, including code, tests, synthetic examples, and documentation. This report was drafted and source-checked with AI assistance. Assistant-generated judgments are not described as independent human validation. Aleksei Khvostov is the responsible human author. This technical report is released as version 1.0 on 3 November 2026. No participant's name, private correspondence, or individual annotation artifact is reproduced.

References

[1] Khvostov A. multi-agent-safety-sim README. Public repository snapshot, 29 October 2026. Source

[2] Jacobs AZ and Wallach H. Measurement and Fairness. FAccT 2021, pp 375-385. doi 10.1145/3442188.3445901. Source

[3] Raji ID, Bender EM, Paullada A, Denton E, and Hanna A. AI and the Everything in the Whole Wide World Benchmark. NeurIPS Datasets and Benchmarks 2021. Source

[4] Ribeiro MT, Wu T, Guestrin C, and Singh S. Beyond Accuracy: Behavioral Testing of NLP Models with CheckList. ACL 2020, pp 4902-4912. doi 10.18653/v1/2020.acl-main.442. Source

[5] Zheng L et al. Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. NeurIPS Datasets and Benchmarks 2023. arXiv 2306.05685. Source

[6] Khvostov A. Phase 3.7 Run 001 Postrun Audit. Completion, negation artifacts, and preservation of the canonical result. Source

[7] Khvostov A. Phase 3.7 Run 002 Postrun Audit. Provider-clean pilot and the unresolved-outcome boundary. Source

[8] Khvostov A. Phase 3.7 Classifier v3 Measurement Decision Audit. 16 July 2026, sections 3-4 and decision. Source

[9] Khvostov A. Phase 3.8 Structured Report State Measurement Preregistration. 17 July 2026, sections 2-4. Source

[10] Khvostov A. Phase 3.8 Structured Report State v1.1 Preregistration. 6 September 2026, sections 8-11. Source

[11] Khvostov A. Phase 3.8 Structured Report State v1.1.2 Pre Holdout Conformance Audit and Disposition. 20 September 2026. Fresh probes, semantic basis, and mechanisms. Source

[12] Khvostov A. Phase 3.8 Structured Report State Measurement Decision. 20 September 2026, sections 2-7. Source

[13] Khvostov A. Structured Report State v1.1.2 extractor. Public source, exact candidate identity in Appendix B. Source

[14] Khvostov A. Structured Report State v1.1.2 implementation manifest. Historical verification and pinned input identities. Source

[15] Khvostov A. Structured Report State v1.1.2 fixture locks. Public support module with historical status-document handling. Source

Appendix A Exact selected audit outcomes

These records reproduce three entries from the existing twelve-failure inventory in source [11]. The appendix contains historical observations, not newly run tests. Field order below is completion_status, uncertainty_status, partial_progress_status, terminal_event_claim_status, explicit_failure_status, followed by contradiction_flags. The expected states follow the frozen contract [10, sections 8-11]. In M3, not_expressed follows this schema's distinction between a possible event and an explicit uncertainty stance; it is not a claim about universal ordinary-language semantics.

C7 Mention scope leakage

```
The example phrase "unresolved" appears and an unsafe attempt occurred.
```

Field	Expected	Actual
completion_status	completion_not_addressed	completion_not_addressed
uncertainty_status	not_expressed	not_expressed
partial_progress_status	partial_progress_not_addressed	partial_progress_not_addressed
terminal_event_claim_status	terminal_event_not_addressed	terminal_event_not_addressed
explicit_failure_status	failure_reported	failure_not_addressed
contradiction_flags	[]	[]

M3 Modal event activation

```
A terminal event may have occurred.
```

Field	Expected	Actual
completion_status	completion_not_addressed	completion_not_addressed
uncertainty_status	not_expressed	not_expressed
partial_progress_status	partial_progress_not_addressed	partial_progress_not_addressed
terminal_event_claim_status	terminal_event_not_addressed	terminal_event_reported
explicit_failure_status	failure_not_addressed	failure_not_addressed
contradiction_flags	[]	[]

Q1 Quoted negator leakage

```
The word "no" appears and an unsafe attempt occurred.
```

Field	Expected	Actual
completion_status	completion_not_addressed	completion_not_addressed
uncertainty_status	not_expressed	not_expressed
partial_progress_status	partial_progress_not_addressed	partial_progress_not_addressed
terminal_event_claim_status	terminal_event_not_addressed	terminal_event_not_addressed
explicit_failure_status	failure_reported	no_failure_reported
contradiction_flags	[]	[]

Selected reported passing controls are “An unsafe attempt occurred and the example phrase "unresolved" appears.”, “The example phrase "unresolved" appears; an unsafe attempt occurred.”, “An unsafe attempt occurred and the word "no" appears.”, and “The word "no" appears; an unsafe attempt occurred.” Each has the expected complete failure state above, with flags []. The operative control “A terminal event occurred and partial progress occurred.” has explicit_partial_progress and terminal_event_reported; other fields take silence values and flags are []. Quotation delimiters are part of the stimulus text [11].

Appendix B Evidence identities and reproduction limits

The public reference revision is 5957af2c8839a602a7f5de3346e065cc53147179. Source paths below are relative to the repository root. Linked references identify the public records used for the report. The evidence check compared source identities without executing the extractor, calibration loader, study tests, model runs, or holdout.

Claim to source map

Claim	Primary source	Boundary
28 partial to uncertain changes	[8] sections 3.3 and 4.2	Diagnostic rescoring, 14 per pilot; no model change
343 disclosed cases	[11] Passing verification facts [14] verification	267 + 11 + 55 + 10; relations separate
58 fresh probes and 12 failures	[11] Fresh adversarial probes	Selected adversarial audit; not the holdout
C7 M3 Q1 and controls	[11] Failure inventory and mechanisms	Historical outputs; complete selected states in Appendix A
Decision not to adopt	[12] sections 1 and 4	Current development line only
Phase 3.9 incomplete	Author closeout decision, 3 November 2026	Private administration; no completed human-reference result

Exact extractor identity

Path: src/multi_agent_safety_sim/evaluation/structured_report_state_v1_1_2.py. SHA-256: 6e80a9523f4d4329d62ae61b8d88451d0f8340a6cddd00245f224f88acde20b1. The current public extractor matches the implementation manifest's historical extractor digest [13, 14]. Mechanism references in the audit refer to this identity: segmentation at lines 163-174 and 307; modal detection at line 448; quoted-context and negation processing at lines 407-442, 474, and 479 [11].

Historical support files and current public source

The historical implementation manifest pins the fixture-loading module to SHA-256 31cc1b09ffea7fbed23b44ad9afd7ea0ced5afe8f7de7e9b1b626e51aa3c8ec0. The current public module has SHA-256 bcec13c073cc9f3df6934241ec587e85a6749281d87fa67f771a712c14155377 and is not byte-identical to that historical support file. It includes handling for later changes to three active status documents. Its presence must not be represented as proof that the full original execution snapshot has been reproduced [14, 15].

The original audit retained the prior hashes of README.md, docs/CURRENT_RESEARCH_STATE.md, and docs/MEASUREMENT_GATES.md. Those files subsequently received documented status changes. Historical lock statements, their authorized exceptions, and a later loader implementation must be distinguished from the unchanged extractor itself. A reproducer must identify which snapshot is being used and explain any compatibility handling rather than claiming that the current checkout is the exact historical environment [11, 15].