WHAT CAN WE INFER FROM AGENT TRACES? OBSERVATION–CLAIM IDENTIFIABILITY FOR ACTION-ROUTING EVALUATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Agent evaluation requires checking not only recorded outputs, but also whether their interpretation preserves the intended question and whether the evidence determines its answer. We introduce an executable source-to-claim contract that distinguishes a changed question, insufficient evidence, and a conclusion requiring revalidation after its supporting inputs change. The workflow checks proposed representation corrections against the original question, acquires evidence from a declared catalogue, and revalidates the resulting answer. We evaluate it on source-linked queries using retained navigation, text-interaction, and hosted tool-use traces, with recorder-conformance checks on 10,800 execution records. A controlled asynchronous execution case shows that identical client-timeout projections can conceal different later transaction outcomes; acquiring transaction evidence resolves the ambiguity. On the tested contracts, direct SMT and finite enumeration reach the same conditional conclusions as the integrated workflow. In a controlled three-layout study of 36 retained records, selective access reads fewer, equal, or more bytes than direct access to the required evidence; a minimal directly available field can be cheaper than a selectively acquired ledger. The contribution is a verifiable path from an evaluation question to evidence and a qualified answer. Guarantees remain conditional on the supplied semantics, admissible execution space, and evidence catalogue.

1 INTRODUCTION

Agent evaluation pipelines project execution histories into state-changing trajectories, successful-call lists or normalized errors. Such projections may not answer another question about dispatch or side effects. Figure 1 shows the distinction using native OpenTelemetry spans and SQLite transactions: identical client-timeout projections accompany different later outcomes. Saved transaction primitives determine each sample; the contract checks whether observations exclude other target values. Sections 5 and 7 examine related transformations of retained Agent records.

A second problem precedes missing evidence: the representation itself may change the target. If raw call counts are 3 and 2, both map to the bin "2+". Equality of those bins is a valid question, but it is not equality of raw counts. Correct inference inside that abstraction cannot justify an unverified transition back to the original question. This motivates a workflow that checks both the source-to-target interface and the sufficiency of observations.

We address three questions. RQ1: can a semantic contract distinguish a changed question from missing evidence? RQ2: can source-linked acquisition settle the original question and support revalidation after change? RQ3: under equal access and guarantees, when are simpler methods sufficient? Our contributions are an executable native-to-claim interface, a diagnosis–acquisition–revalidation workflow, and measured applicability limits. A single contract connects three actions to the original target: correct its representation, obtain missing evidence, or revalidate an invalidated receipt. The author supplies the target and admissibility premises; the tool checks their formal consequences and exposes the source-linked evidence path. Established inference and solver techniques implement this evaluation interface.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

2 A SOURCE-TO-CLAIM CONTRACT

Let $\mathcal{T}_N$ be a nonempty finite space of declared native-interface states or execution pairs, $\mathcal{T}_A$ an abstract space, α an adapter, g_N the native target and g_A the abstract target. An obtainable field has a decoder d_j, an abstract expression o_j when applicable, a source path and a nonnegative declared acquisition weight. "Native" identifies the interface vocabulary, not a proof that every valuation is physically reachable. The intended target, bounds, causal or program constraints and source interpretation remain explicit author-supplied premises.

The guard checks three obligations:

$$\forall t \in \mathcal{T}_N : \quad \alpha(t) \in \mathcal{T}_A, \tag{1}$$

$$g_N(t) = g_A(\alpha(t)), \tag{2}$$

$$d_j(t) = o_j(\alpha(t)). \tag{3}$$

The first checks coverage of the declared native domain by the abstract representation, including relational constraints. It does not prove that the native domain covers the real system. The second preserves the question rather than merely its data type. The third checks consistency between the declared decoder and abstract observation. Units are stated and conversions encoded in expressions; the tool does not infer units or intent from prose. Hash-checked retrieval and source tests validate selected decoder executions, not arbitrary future Python behavior.

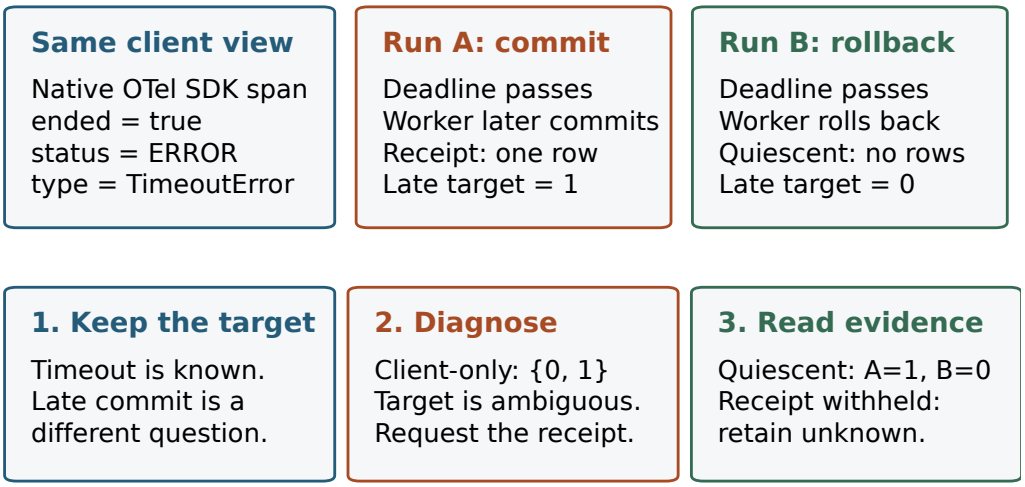


Figure 1: A non-count investigation with an external recorder. Runs A and B are actual controlled SDK executions, not invented span records. Their named client-status projection agrees while the late-commit target differs. Transaction evidence settles each case; client evidence alone preserves ambiguity. This does not assert equality of complete spans or a defect in OpenTelemetry.

2.1 THREE QUESTIONS WITH DIFFERENT QUANTIFIERS

An observation O identifies g globally precisely when the target is constant on every observation fiber:

$$\forall t, t' \in \mathcal{T} : \; O(t) = O(t') \Rightarrow g(t) = g(t').$$

This fiber-constancy criterion is shared with finite-policy identifiability and claim-relative evidence sufficiency (Luo et al., 2026; Zhu & Chang, 2026). For an acquired observation o, case identification requires constancy only on the nonempty set $\{t : O(t) = o\}$. Direct sample calculation evaluates the native target using sufficient saved fields and asserts nothing about unobserved worlds. Empty fibers indicate inconsistent contracts or observations and must not produce vacuous certificates. Sharp finite-space extrema describe logical ambiguity, not sampling uncertainty.

Global design covers every admissible fiber; case investigation stops at a singleton target-value set; sample calculation uses sufficient recorded primitives. We report their costs and coverage separately.

Table 1: A complete temporal contract for Figure 1. Symbols abbreviate the released interface; premises are supplied from the fixture, not inferred by the solver.

Object	Concrete definition and evidence
Question	Did a commit occur after client-span end and before worker quiescence?
Variables	$t = (u, c, r, w)$: timeout, commit presence, order and worker-terminal status; $u, c, w \in \{0, 1\}$, $r \in \{-1, 0, 1\}$.
Constraints	$c \Leftrightarrow (r \neq 0)$; $r = 1 \Rightarrow u$; $\neg u \Rightarrow w$. The native target is $g_N(t) = \mathbf{1}[c \wedge r = 1]$. The correct abstract interface is the identity on these variables.
Observation	Named span projection (ended, status, exception type) supplies u. Actual A: $(1, 1, 1, 1)$; B: $(1, 0, 0, 1)$. Both have $u = 1$, but targets 1 and 0.
Acquisition	Span u: 0 units; terminal receipt w: 1; transaction packet (c, r): 3. Without it A/B remain ambiguous; acquiring it determines both targets.
Validity	One database per episode, no other writers, joined worker, and commit interval strictly on one side of span end. Overlap or missing quiescence withholds a source-derived answer.

For $u = 1$, the two realized worlds witness bounds $[0, 1]$. For $u = 0$, the fixture's second constraint excludes a late commit. This latter deduction is conditional on the fixture, not a law about arbitrary ended spans. A separate target $g_N = u$ is already identified. Section 5 tests the source decoder and Appendix E gives the acquisition paths.

2.2 Conditional guarantees and their boundary

If Equations 1–3 hold and g_A is identified by the abstract observations, then g_N is identified by the corresponding native observations. Equal native observations imply equal mapped observations; coverage makes the two images admissible; abstract identification and target preservation imply equal native targets. This is a composition argument, not a claimed new theorem. The converse need not hold when the abstract space contains unreachable extra states. Our implementation therefore checks native fibers directly after the semantic obligations instead of requiring abstract identification as a necessary condition.

Observational refinement cannot enlarge a current ambiguity set; expanding the admissible space cannot remove existing target values at a shared observation. Neither proves a premise true. Appendix G locates premise sources and shows how dropping the fixture implication $r = 1 \Rightarrow u$ changes no-timeout bounds from $[0, 0]$ to $[0, 1]$. A solver witness is symbolic: an observed failure requires telemetry, and an executable counterexample requires realization or replay.

3 Executable diagnosis and evidence acquisition

The symbolic implementation uses Z3 (de Moura & Bjørner, 2008) to seek a native valuation violating each preservation obligation. It then searches for a pair satisfying admissibility, equal observations and unequal targets. SAT returns a witness; UNSAT establishes the corresponding conditional statement; unsupported syntax, empty spaces, timeout or solver UNKNOWN withhold certification. Supported expressions use finite Boolean/integer domains and explicit total operations. A returned UNSAT query is hash-bound and replayable, but replay shares the compiler and is not a separately verified proof of compiler correctness.

The source-to-claim interface records the exact question, target, domain, constraints, source identity, decoder, acquisition conditions and units. It distinguishes an absent acquisition path, a field never recorded, an access restriction and a solver refusal. A certificate must have a concrete acquisition path; a free oracle containing the desired target answer is not a valid observation. The implementation retains proposal, dispatch, binding, body entry, resource access, commit, completion and final-record distinctions even when only some stages were instrumented.

Algorithm 1: source-linked evidence workflow.

1. Validate the contract and source catalogue; verify acquired receipts and preserve unavailable fields.
2. Check coverage, native-target preservation and decoder commutation. Return a violated obligation and witness, or retain an unsupported status.
3. For global design, test all-fiber identification. Select an obtainable repair and reverify it. For a case, use the fixed cost/name acquisition order until the nonempty current fiber has a singleton *target-value set*, or evidence is exhausted. An empty fiber reports inconsistency.
4. Separately compute the saved-sample target from recorded primitives, without substituting source-derived deductions for absent telemetry.
5. Report assumptions, acquisition receipts, witness/proof status and separate selection weights, payload bytes and computation.

The global planner uses witness-guided weighted hitting sets and cost-ordered exact subsets. Reiter (1987) characterizes diagnoses as inclusion-minimal hitting sets of conflicts. Bittner et al. (2012) study minimum-cost sensor selection with counterexample-guided refinement. Luo et al. (2026) enumerate exact minimum-cardinality identifying supports via hitting sets for a finite policy class. Every identifying set must distinguish accumulated unequal-target pairs. An optimal candidate passing the universal verifier attains the catalogue-relative lower bound (Appendix A). Adapter replacements retain the native target, domain and constraints; the extension neither synthesizes arbitrary programs nor improves a completed exact optimum.

Case acquisition uses declared order, never unacquired values, with the same stopping rule for every backend. Its condition is $F(o) \neq \varnothing$ and $|\{g(t) : t \in F(o)\}| = 1$, not $|F(o)| = 1$. A regression case retains two possible executions with the same target; all three backends correctly stop. Other controls retain multi-valued and empty fibers.

4 Evidence design and correctness

We retain three types of evidence without treating them as interchangeable. The controlled set contains 48 development contracts and 12 subsequently frozen structural compositions. A source-derived count stress set contains 144 contracts: three query structures over 48 task clusters, with 16 tasks from each of Navigation, TextWorldExpress and the isolated hosted panel. Its raw-equality/clipped-count mismatch is deliberately injected. These are not 144 independent research questions or naturally discovered defects. Finally, all eight prior native applications are integrated unchanged, preserving their original targets, constraints and evidence prices. They comprise four task clusters and six source records; no new task or model is introduced.

These earlier comparison inputs and code were hash-frozen before their method comparison; dependencies and timing receipts are retained. For the recorder extension below, development controls and source records were known before the extension freeze. Complete-record aggregation, selected-record consequences and randomized access measurements followed that freeze. Both are retrospective tool validation, not prospective environment evaluations. Identity adapters in the eight unchanged applications are preservation controls; the recorder extension supplies nonidentity transformations without recasting controlled question misuse as a discovered benchmark defect.

4.1 Equal-guarantee and bounded baselines

The direct SMT baseline uses a separately written expression compiler and explicit preservation and paired-world assertions. It imports neither the tested guard nor its compiler; it has no automatic evidence planner. The finite Python reference evaluates worlds and fibers without Z3. These are separate code paths produced with the same assistant, not independent human implementations. All three consume the same public contracts without stored sample values. A bounded property checker tests fixed extrema and seeded random assignments under a 256-candidate budget. Passing this check never produces a universal certificate. Complete obtainable native logs form the saved-sample baseline.

All three conditional backends agree on 212/212 contracts. Semantic witnesses are checked by the finite evaluator; the guard's preservation queries are replayed. Seven new integration tests in-

Table 2: Conditional conclusions agree. Mismatch counts include repeated controlled structures, not independently discovered scientific errors.

Contract cohort	Mismatch	Identified	Insufficient
Development controls (48)	8	24	16
Held-out structures (12)	3	6	3
Count-adapter stress (144)	48	96	0
Unchanged applications (8)	0	6	2
Total (212)	59	132	21

clude empty spaces, unsupported types, incomplete maps, inconsistent observations and signed raw targets, alongside the earlier 27 semantic tests. Shared assumptions and compiler trust remain limitations despite agreement.

The stronger property checker finds all 59 mismatches: no detection advantage is established. A bounded search without a counterexample does not certify unsearched states; exhaustive enumeration of the same finite domain does give the same conditional guarantee. In the count stress set, only nine of 48 deliberately mismatched interfaces change the saved sample answer; the other samples happen to agree. Full native logs correctly compute all 144 saved targets. Agreement on a sample and preservation of the declared question are different checks.

5 NONIDENTITY RECORDING: DIAGNOSIS, REPAIR AND REVALIDATION

5.1 SOURCE PREDICATES AND UNCHANGED QUESTIONS

Pinned recorder syntax trees expose Navigation pose changes, TextWorldExpress visible-state changes, hosted successful-call filtering, and the error literal shared by pre-dispatch rejection and caught invalid arguments. These are four interfaces in three transformation families from our instrumentation around public environments. Exhaustive Boolean checks agree with separate literal predicates; ordered reconstruction matches every retained final record. The error branch is checked only when its original reason is eligible. Source hashes bind the hosted recorder to the original execution plan.

Table 3: Recorder conformance on retained records. The last column is a counterfactual *question misuse*: records whose projected count would disagree with dispatch count. It is not a defect rate or a new paired-runtime counterexample count.

Interface	Records / tasks	Map errors	Proxy differs
Navigation pose projection	6,000 / 300	0	229
TextWorldExpress state projection	2,400 / 200	0	2,327
Hosted successful-call filter	2,400 / 100	0	966
Hosted error normalization	16 / 4	0	8

The normalizer contributes 18 eligible event slots in 16 records from four tasks, overlapping the hosted filter data. The three unique source files contain 6,000 Navigation, 2,400 TextWorldExpress and 2,400 hosted records: 10,800 executions, not 13,200 independent records. No malformed record was dropped. These full-file checks establish conformance to selected local paths, not correctness of arbitrary Python or completeness of the native admissible space.

For each interface we fix four questions/stages (Figure 2). Its intended projected-output count is preserved. Reusing that representation to answer dispatch count violates target preservation. Retaining a dispatch variable repairs the representation but does not make an unavailable ledger observable. Obtaining the source-linked ledger then identifies the same dispatch target. The last three stages share the native target, domain and constraints; a repair cannot substitute the easier output-count question. The intended first stage is a separate positive-control question, not a repair achieved by changing the estimand.

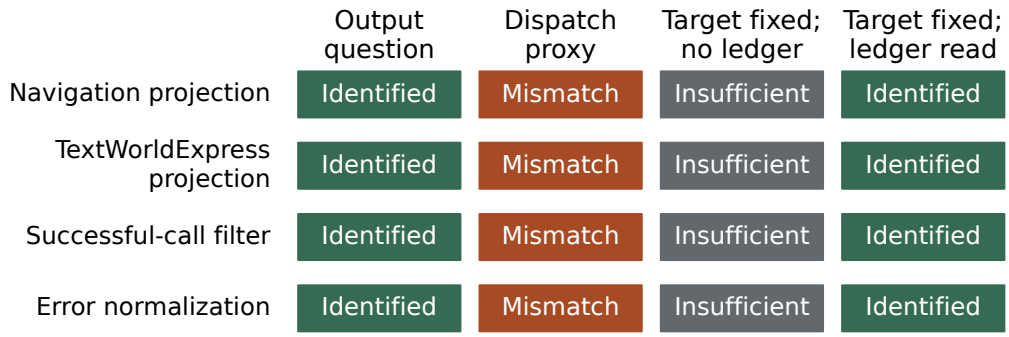


Figure 2: All three conditional backends agree on 16 contracts: four interfaces crossed with four stages, not 16 independent applications. The native dispatch question stays fixed through columns 2–4. Column 1 checks the different, intended output-count question.

The four mismatched contracts return finite-verified witnesses and target-preserving repairs; a 256-candidate property checker also detects all four. Additive-count preservation extends from events to finite sums under the declared source meanings.

5.2 What happens when the evidence changes?

After issuing an identifying receipt, nine controlled changes per interface test unchanged reuse, serialization order, target change, decoder change, removed source path, revoked permission, altered source bytes, unsupported syntax and restoration. All 36 controls match their expected categories across three backends (108 results), with no invalid receipt reuse. A changed hash withholds reuse, not validation of new semantics. A separate seven-stage migration on two copied native episodes exercises actual file reads: refreshing identity still refuses the old decoder; a declared decoder repair restores the unchanged target (Appendix H). Both studies use shared integrity policies, not guard-specific detection advantages or natural incident rates.

5.3 An external recording interface and a temporal target

In our pinned fixture, ending the client span does not cancel the submitted worker (OpenTelemetry Authors, n.d.b). A timeout from `Future.result(timeout=...)` ends that wait without itself cancelling the callable (Python Software Foundation, n.d.). Table 1 instantiates this distinction. SDK 1.36.0 and the fixture are frozen before nine retained executions: three repeats each of late commit, timeout then rollback, and on-time commit. Native SDK JSON and episode-local SQLite evidence validate the temporal target.

Three late commits and three rollbacks share the timeout projection. An empty database at the timeout instant does not distinguish them. Transaction evidence supplies the quiescent query and commit interval. The decoder refuses absent quiescence or interval overlap rather than inventing the physical commit instant.

All three backends settle 9/9 cases with obtainable transaction evidence. With that receipt deliberately withheld from every method's catalogue, 6/9 remain ambiguous. The separate client-timeout question is identified in 9/9. Complete obtainable evidence and a script with the same semantics are also correct. The workflow distinguishes a wrong timeout proxy, a preserved but under-observed temporal question, and an answer verified against the source. These repeated controlled checks establish the interface's operation, not an accuracy estimate over independent tasks.

5.4 Native case meanings and mechanism evidence

The eight earlier source-linked applications retain their original questions: six obtainable catalogues identify globally, seven current observations identify their case, and six targets are directly com-

putable from saved primitives. All conditional backends agree. Two cases need no additional fields, and complete hidden-state change remains unknown. The difference between seven and six is a source-rule deduction about invalid binding, not invented body-entry telemetry (Appendix B).

On one successful hosted Planner pair, both systems emitted 15 calls, but Basic dispatched 15 and Checked 14. Their raw ledgers settle the difference at +1. For the invalid `ls(a=true,path=...)` request, the trace records dispatch and a normalized error, not body entry. The pinned signature accepts only `a`; the dispatcher's `fn(**args)` therefore fails binding before entry. This source-rule deduction identifies that case, not the whole catalogue, and is not direct telemetry (Appendix B).

Fresh-fixture replay of those exact arguments raises TypeError without profiled body/helper entry or state change. This is post-collection mechanism diagnosis, not a replacement closed-loop execution. Controlled body-failure, post-commit failure and synchronized timeout-then-completion tests separately validate the stage distinctions.

Other applications retain charged-but-omitted steps, positive support/success controls and unknown complete-state change, with source identities and obtainable primitives cross-checked (Appendix B).

6 When are simple methods sufficient?

6.1 Shared lazy access rather than selection proxies

The access experiment selects the first record for each of the first 12 lexicographic task IDs per environment: 36 records from 36 source tasks, without selecting on outcomes. Its question concerns the first attempt by ordinal. Both selective and full-native policies use the same lazy JSON reader, integrity check, parser and cache. Selective investigation first acquires the projected predicate, then the dispatch primitive only if its current fiber is ambiguous. The strong full-native baseline reads only the necessary dispatch primitive, not redundant bundles. Unacquired values never choose a field. All three backends share this policy and final verifier; source primitives are used to grade answers only after acquisition.

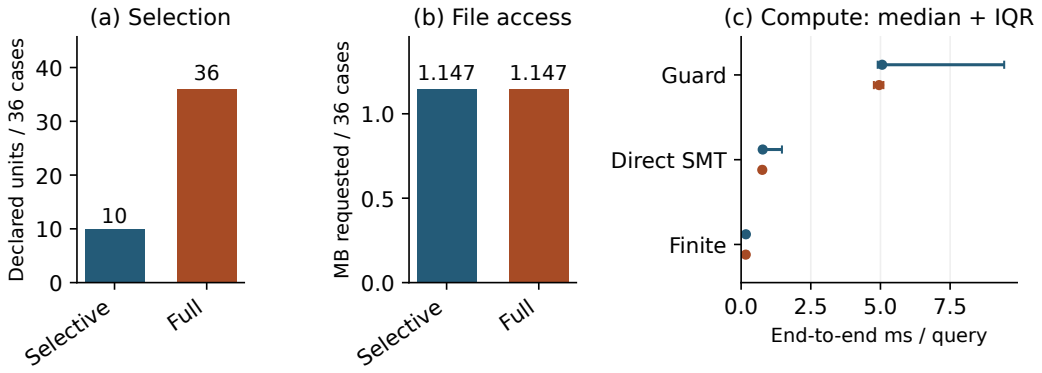


Figure 3: Selective access (blue) versus necessary full-native access (orange), on the same 36 records. Declared selection cost falls from 36 to 10, but requested bytes are identical. Timing intervals are empirical quartiles over 36 records and five repeats per policy/backend, not population confidence intervals. All 36 cases are correctly settled by every method.

The projected field has declared weight zero and the additional ledger weight one. Selection spends 10 versus 36 units, but both policies request 1,146,582 bytes and open 36 files: a cheap field still requires its JSON record. Common source scan, integrity checks and packet preparation take 2.164 s separately. Five randomized repeats give 1,080 measurements, not independent queries. Median selective/full-native milliseconds are 5.061/4.951 for the guard, 0.775/0.759 for direct SMT and 0.171/0.166 for enumeration. Bytes are process requests, not physical disk traffic; OS cache is not flushed.

A controlled materialization of the same 36 records tests access granularity (Appendix H). Whole-record bytes remain equal. With two minimal fields, selective/full bytes are 6,771/5,232; with a separate existing ledger and derived projection they are 213,470/404,548. Thus selection can cost more, and the small direct field is cheaper than either ledger policy. Setup is reported separately; every backend shares the same access strategy. Batches of 1/5/20 instances of the same query allow equal result caching, not additional independent questions or a speed guarantee.

6.2 Retained resource comparisons and applicability

The 212-contract experiment uses five randomized repeats with equal preloaded access. All 60 subprocess batches complete: 3,180 timings, with 1 s solver-call and 60 s parent limits. Direct SMT is faster than the guard in every cohort. Enumeration wins on small spaces; symbolic methods win on the larger two-count domain. Appendix C reports timings and operating-system peak RSS including native solver allocations. These are implementation-specific tradeoffs.

On the eight applications, the fixed policy and deduplicated full-native access both cost 9 units and settle seven cases. The original 18-unit all-catalogue comparison includes redundant bundles; it is retained as provenance, not the headline baseline. The deduplication sensitivity was post-comparison, using only expressions and costs, not hidden values.

The prior 432-scenario analysis gives identical certified global optima for both exact methods. Table 4 identifies integration outputs: source-linked obligations, verified acquisition and stale-receipt revalidation, not improved solver answers. The layout comparison locates conditional read savings and losses; it establishes neither universal speed nor original collection-cost savings.

Table 4: Choosing a method under the same semantics and access. These are outputs and responsibilities, not an exclusive capability or performance ranking.

Method	Required input / valid output	Evidence planning and validation boundary
Native-log script	Saved primitives and target; sample value.	User supplies retrieval and meaning. No all-fiber statement follows.
Property checks	Property and searched states; witness or bounded absence.	No automatic acquisition. Exhausting the same finite domain permits the same conditional proof.
Direct SMT / enumeration	Same contract; witness, conditional verdict or refusal.	Same case order, access and verifier. Global repair needs separate search.
Integrated workflow	Target, adapter, catalogue; obligation, evidence request and verified repair.	Orchestrates acquisition and revalidation. Same answers; no measured detection or speed advantage.

7 Public execution context

The retained hosted panel uses 100 BFCL tasks (Patil et al., 2025) through Sophnet (n.d.). Its frozen plan records the historical endpoint, exact provider IDs (DeepSeek-V4-Flash and glm-5.3-flash) and settings. Direct/Planner proposers, Basic/Checked runtimes and three replicate identifiers yield 2,400 executions. Fresh processes preserve within-episode state and prevent cross-episode carryover. Both capability gates pass. The prior shared-state indicator affected 191 records across 14 tasks, not 191 scoring errors. Our artifact retains the complete isolated panel separately from the old panel.

Table 5: Capability is checked on distinct tasks, not API returns. The frozen gates require at least 30 full-success tasks per fixed model/configuration and 30 jointly successful tasks in a focal runtime comparison. Ranges do not pool configurations.

Model	Full-success runs	Tasks/config.	Joint tasks D / P
DeepSeek-V4-Flash	736/1200	74–78	78 / 74
glm-5.3-flash	684/1200	65–72	65 / 72

In the isolated both-success strata, DeepSeek Direct has zero attempt counterexamples among 189 pairs, and Planner one among 179, from one task. GLM has none among 170 and 172 pairs. All four focal comparisons have equal proposals and final records. The replicates are not provider-guaranteed random seeds; exact-history response caching defines a specified response-coupled runtime comparison. Both-success strata are outcome-conditioned descriptions, not causal effects. Zero observed counterexamples do not certify general equivalence. The recorder-query counts above neither replace these sparse paired results nor establish cross-model counterexample prevalence.

The prospective TextWorldExpress 1.1.0 panel (Jansen & Côté, 2023) retains attempt differences in 43/261 memory-policy and 126/192 reactive-policy jointly successful pairs. Navigation uses BabyAI/MiniGrid (Chevalier-Boisvert et al., 2019; 2023). Our instrumentation supplies the state-changing, all-completed, all-invoked and proposal-preserving projections. These environments do not prove every symbolic witness reachable. Source-linked audits and mechanism replay provide the additional evidence, without increasing the model count or establishing counterexample prevalence.

8 Related work: shared foundations, different contracts

Blackwell (1953) compares experiments, while Clarkson & Schneider (2010) formalize properties of sets of traces. These supply established observation concepts. Cost-based diagnosability synthesis (Bittner et al., 2012) and conflict-based diagnosis (Reiter, 1987) precede our evidence selection. Alive2 validates LLVM IR transformations by bounded refinement checking (Lopes et al., 2021). We apply per-transformation validation to supplied evaluation adapters over declared finite domains, linked to obtainable fields and global, case and sample questions.

AgentBoard evaluates multi-turn trajectories (Ma et al., 2024); ClawProBench evaluates process evidence and specifies a runtime adaptation contract (Xiao, 2026). Our adapter comparison concerns executable preservation obligations over a declared finite domain. ToolEmu studies agent risk (Ruan et al., 2024); LEDGER links claims to actions and artifacts (Kim et al., 2026); OpenTelemetry specifies agent spans in development conventions (OpenTelemetry Authors, n.d.a). These systems have different objectives; lacking our interface is not a failure on their tasks. Runtime verification also permits inconclusive verdicts while continuations remain unresolved (Leucker & Schallhart, 2009).

Two contemporaneous preprints closely overlap. Beyond Local Accuracy combines finite-policy identifiability, collision witnesses and exact minimum-support synthesis (Luo et al., 2026); those operations are shared foundations, not our distinguishing claim. ClaimReceipt combines claim-relative sufficiency with typed transaction evidence and a committed experiment universe (Zhu & Chang, 2026). It is more than log recomputation: coverage of expected runs is an additional question that our source hashes alone do not answer. Our focus is checking that a supplied adapter still asks the native question, then linking permitted acquisition and revalidation to that unchanged target. This is a difference in the implemented evaluation contract, not evidence that either work cannot be extended. Appendix F compares inputs, outputs and guarantees without a performance ranking.

9 Scope, adoption and conclusion

The trust boundary includes author-supplied targets, bounds, source interpretation and admissibility. A correct solver cannot repair a false premise, and backend agreement does not independently validate it. Recorder misuse queries and external-SDK schedules are controlled author-prepared applications, not independent user studies or prevalence samples. Their successful checks do not establish broad distributed-system transfer, arbitrary-program correctness or deployment safety.

Use a script when sufficient saved primitives answer a sample question, and enumeration for a small complete domain. The integrated interface targets researchers maintaining changing adapters or evidence catalogues who need semantic checks, acquisition and revalidation. Supplying a correct contract remains their responsibility. Adoption and productivity gains are unmeasured; selection weights, bytes, computation and setup effort remain distinct.

The workflow links the original question to evidence and a qualified answer, distinguishing changed targets, insufficient observations and invalidated receipts while preserving cases where complete logs or simpler solvers suffice.

Reproducibility Statement

The artifact preserves the preceding contracts and native executions, plus a source-located premise ledger, 40 logical sensitivity queries, 9,720 access-layout timing batches and a two-episode source-migration application. These reuse existing records. One offline command verifies retained results, source identities, acquisitions and displays; a minimal case command demonstrates ambiguity and receipt-based resolution. Primary dependencies and the observed transitive install are pinned. Installation was performed in a new isolated environment; source extraction and reconstruction are verified separately. Optional remeasurement writes a new result, never replacing retained timings. Automated reconstruction is not independent human reproduction.

AI Use Statement

Codex/ChatGPT was used extensively for literature retrieval and synthesis; research ideation and the development of conceptual and formal frameworks; hypothesis and experimental-design development; assistance in formulating mathematical claims and proof exposition; generation of controlled synthetic contracts, test cases, and fixture inputs; implementation, testing, and debugging of adapters, solvers, reference code, and analysis code; experiment orchestration; data processing, statistical summaries, and result interpretation; and drafting and polishing manuscript text, figures, and LaTeX. Synthetic inputs are distinguished in the artifact from retained execution records and observed hosted-model responses.

The sole human author participated throughout problem formulation, research ideation, the development of conceptual and formal frameworks, hypothesis and experimental-design development, methodological decision-making, interpretation, manuscript preparation, and review, and retains responsibility for the final text, claims, code, data, analyses, and submission. Verification procedures include source-linked provenance, literal and symbolic comparisons, reconstruction of reported results, and rendered-output inspection. Separate code paths produced with the same assistant are treated as automated cross-checks only; they are not presented as independent human verification or third-party replication. The access and maintenance extension reuses retained records and introduces no new hosted-agent evaluation calls.

References

Benjamin Bittner, Marco Bozzano, Alessandro Cimatti, and Xavier Olive. Symbolic synthesis of observability requirements for diagnosability. *Proceedings of the AAAI Conference on Artificial Intelligence*, 26(1):712–718, 2012. doi: 10.1609/aaai.v26i1.8225.

David Blackwell. Equivalent comparisons of experiments. *The Annals of Mathematical Statistics*, 24(2):265–272, 1953. doi: 10.1214/aoms/1177729032.

Maxime Chevalier-Boisvert, Dzmitry Bahdanau, Salem Lahlou, Lucas Willems, Chitwan Saharia, Thien Huu Nguyen, and Yoshua Bengio. BabyAI: A platform to study the sample efficiency of grounded language learning. In *International Conference on Learning Representations*, 2019. URL https://arxiv.org/abs/1810.08272v4.

Maxime Chevalier-Boisvert, Bolun Dai, Mark Towers, Rodrigo de Lazcano, Lucas Willems, Salem Lahlou, Suman Pal, Pablo Samuel Castro, and Jordan Terry. Minigrid & Miniworld: Modular & customizable reinforcement learning environments for goal-oriented tasks. In *Advances in Neural Information Processing Systems*, volume 36, pp. 73383–73394, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/e8916198466e8ef218a2185a491b49fa-Abstract.html. Datasets and Benchmarks Track.

Michael R. Clarkson and Fred B. Schneider. Hyperproperties. *Journal of Computer Security*, 18 (6):1157–1210, 2010. doi: 10.3233/JCS-2009-0393.

Leonardo de Moura and Nikolaj Bjørner. Z3: An efficient SMT solver. In *Tools and Algorithms for the Construction and Analysis of Systems*, volume 4963 of *Lecture Notes in Computer Science*, pp. 337–340. Springer, 2008. doi: 10.1007/978-3-540-78800-3_24.

Peter Jansen and Marc-Alexandre Côté. TextWorldExpress: Simulating text games at one million steps per second. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pp. 169–177. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.eacl-demo.20. URL https://aclanthology.org/2023.eacl-demo.20/.

Daehong Kim, Haichao Miao, and Shusen Liu. LEDGER: Claim-to-evidence trace graphs for auditing LLM agents. arXiv:2608.18398v1, 2026. URL https://arxiv.org/abs/2608.18398v1. Preprint.

Martin Leucker and Christian Schallhart. A brief account of runtime verification. *Journal of Logic and Algebraic Programming*, 78(5):293–303, 2009. doi: 10.1016/j.jlap.2008.08.004.

Nuno P. Lopes, Juneyoung Lee, Chung-Kil Hur, Zhengyang Liu, and John Regehr. Alive2: Bounded translation validation for LLVM. In *Proceedings of the 42nd ACM SIGPLAN International Conference on Programming Language Design and Implementation*, pp. 65–79, 2021. doi: 10.1145/3453483.3454030.

Junhao Luo, Ning Huang, Ziqi Sha, Wenxuan Tang, and Wei Deng. Beyond local accuracy: A protocol-level identifiability audit for controlled LLM reasoning evaluation. arXiv:2608.13326v1, 2026. URL https://arxiv.org/abs/2608.13326v1. Preprint.

Chang Ma, Junlei Zhang, Zhihao Zhu, Cheng Yang, Yujiu Yang, Yaohui Jin, Zhenzhong Lan, Lingpeng Kong, and Junxian He. AgentBoard: An analytical evaluation board of multi-turn LLM agents. In *Advances in Neural Information Processing Systems*, volume 37, pp. 74325–74362, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/877b40688e330a0e2a3fc24084208dfa-Abstract-Datasets_and_Benchmarks_Track.html. Datasets and Benchmarks Track.

OpenTelemetry Authors. Semantic conventions for GenAI agent and framework spans, n.d.a. URL https://github.com/open-telemetry/semantic-conventions-genai/blob/b06f7a2c840ceeacd8478acd3443e697ce390f96/docs/gen-ai/gen-ai-agent-spans.md. Development specification, commit b06f7a2c840ceeacd8478acd3443e697ce390f96. Accessed September 10, 2026.

OpenTelemetry Authors. Tracing API: Span end and lifetime, n.d.b. URL https://opentelemetry.io/docs/specs/otel/trace/api/. Stable specification except where marked otherwise; Python SDK 1.36.0 pinned in the artifact. Accessed September 12, 2026.

Shishir G. Patil, Huanzhi Mao, Fanjia Yan, Charlie Cheng-Jie Ji, Vishnu Suresh, Ion Stoica, and Joseph E. Gonzalez. The Berkeley Function Calling Leaderboard (BFCL): From tool use to agentic evaluation of large language models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 48371–48392. PMLR, 2025. URL https://proceedings.mlr.press/v267/patil25a.html.

Python Software Foundation. concurrent.futures: Future result, timeout and cancellation, n.d. URL https://docs.python.org/3.12/library/concurrent.futures.html. Python 3.12 documentation; execution used CPython 3.12.14. Accessed September 12, 2026.

Raymond Reiter. A theory of diagnosis from first principles. *Artificial Intelligence*, 32(1):57–95, 1987. doi: 10.1016/0004-3702(87)90062-2.

Yangjun Ruan, Honghua Dong, Andrew Wang, Silviu Pitis, Yongchao Zhou, Jimmy Ba, Yann Dubois, Chris J. Maddison, and Tatsunori Hashimoto. Identifying the risks of LM agents with an LM-emulated sandbox. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2309.15817v2.

Sophnet. Chat completions API documentation, n.d. URL https://www.sophnet.com/docs/component/API.html. Accessed September 10, 2026. Historical experiment endpoint and request settings are retained separately in the frozen artifact.

YuanHang Xiao. ClawProBench: Trace-aware evaluation of declared agent configurations with runtime coverage and frozen holdouts. arXiv:2608.22510v1, 2026. URL https://arxiv.org/pdf/2608.22510v1. Preprint; title as printed on the consulted version-1 PDF.

Peiying Zhu and Sidi Chang. ClaimReceipt: Verifying evidence sufficiency and coverage in agent evaluations. arXiv:2609.01992v1, 2026. URL https://arxiv.org/abs/2609.01992v1. Preprint.

A PROOF OBLIGATIONS AND CERTIFICATE SCOPE

For the fiber criterion, a factorization $g = h \circ O$ implies target constancy for equal observations. Conversely, define $h(o)$ as the common target on each nonempty fiber; this is well-defined. In the finite nonempty setting, attainable minimum and maximum values give sharp logical bounds. None of these arguments estimates a population event rate.

For native lifting, take t, t' with equal decoded observations. Commutation gives equal abstract observations of $\alpha(t), \alpha(t')$. Coverage makes these images members of the declared abstract space. Abstract identification gives equal abstract targets, and preservation gives equal native targets. Additional unreachable abstract worlds can prevent abstract identification without preventing native identification; the implementation's direct native query avoids treating the sufficient lifting condition as necessary.

For weighted repair, every identifying set hits every accumulated unequal-target witness pair. The minimum hitting-set weight is a lower bound on all feasible identifying sets. If a minimum-weight candidate passes the universal verifier, it attains the bound. This requires a correct compiler, a completed optimality result and true catalogue assumptions. Adapter replacements are checked against the original native target and constraints; the finite list of offered replacements is not the set of all possible program repairs.

UNSAT-query replay binds the formula to the declared contract and regenerated obligation. It checks that a submitted unrelated UNSAT formula cannot be substituted for the intended query. It does not verify the solver implementation or source decoder program. SAT witnesses are evaluated by the separate finite interpreter. Historical telemetry, source-rule deductions, instrumented replay and purely abstract valuations retain separate evidence labels.

B SOURCE APPLICATION CATALOGUE AND OPERATIONAL LIMITS

Table A1: Native case outcomes. A dash denotes missing direct telemetry, not a zero.

Question	Sample	Case	Global catalogue
Basic minus Checked dispatch	1	1	Identifying
Body entry after invalid binding	–	0	Not identifying
Dispatch units on two failed calls	2	2	Identifying
Charged navigation steps omitted	1	1	Identifying
Recorded grammar-support certificate	1	1	Identifying
Work with unchanged visible projection	16	16	Identifying
Recorded task success without failure	1	1	Identifying
Unrecorded complete-state change	–	–	Not identifying

The source capsule contains six exact records and 20 required source texts, including original hosted responses used to verify emitted calls and pinned binding/dispatcher code. Whole-file hashes are verified against the accessible original execution files during preparation. A small capsule alone cannot reconstruct or verify omitted whole-file bytes. Every extractor lists its location and semantic rule. The existing exact questions and costs are preserved, not reconstructed from favorable method outputs.

In the binding case, body entry requires both submission and successful binding under the declared code semantics. The observed invalid keyword falsifies the binding premise, so body entry is conditionally false. The original recorder lacks the body event itself. In the complete-state case, no analogous rule or saved primitive determines hidden state from the visible projection; the answer stays unknown.

The full-primitive-log sensitivity was added after the initial all-catalogue totals were observed. It removes redundant composite bundles and exact-expression aliases using only catalogue expressions and prices. It is a post-comparison fairness check, not a new preregistered primary experiment. Both its cost tie and the original all-catalogue totals remain in the artifact. No claim of original collection, annotation or API-bill savings follows from selection among already saved fields.

C REPEATED COMPUTATION AND MEMORY

Table A2: On-demand access: totals per 36 source records and milliseconds per query. Quartiles pool record/repeat measurements; they are not confidence intervals.

Backend / policy	Units	Requested bytes	Median ms	Q1 / Q3
guard / conditional	10	1,146,582	5.061	[4.902, 9.440]
guard / full native	36	1,146,582	4.951	[4.764, 5.106]
direct smt / conditional	10	1,146,582	0.775	[0.709, 1.465]
direct smt / full native	36	1,146,582	0.759	[0.684, 0.831]
finite / conditional	10	1,146,582	0.171	[0.125, 0.235]
finite / full native	36	1,146,582	0.166	[0.128, 0.233]

The extension's 14 development tests cover extracted-predicate equivalence, target-preserving repair, all contract stages, stale receipts, source mutation, unavailable fields, empty fibers, unsupported syntax and lazy access integrity. They precede the extension freeze. Complete-file conformance and randomized access results follow it. A receipt binds the canonical contract, source identity and backend result; all backends share the same validity policy. A changed source identity requires a new validation, even if its decoded sample value happens to be unchanged. Restoration is a positive control, not proof that arbitrary future changes are safe.

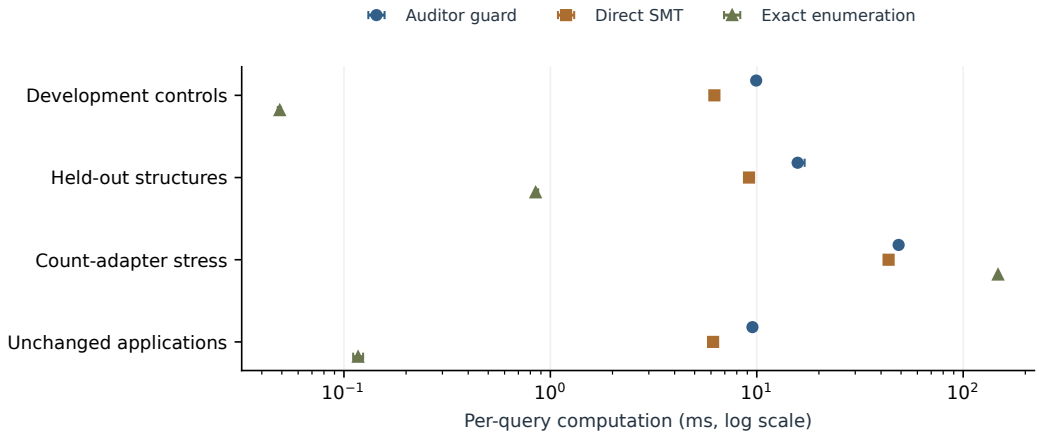


Figure A1: Retained baseline computation: median of five repeat-level medians, with empirical interquartile ranges. Cohorts are different structures, not an ordered scaling curve.

Table A3: Milliseconds: median of five within-repeat medians, with empirical first/third quartiles. No population confidence interval is implied.

Query cohort	Auditor guard	Direct SMT	Exact enumeration
Development controls	9.93 [9.93, 10.10]	6.23 [6.11, 6.24]	0.05 [0.05, 0.05]
Held-out structures	15.77 [15.53, 17.05]	9.17 [9.10, 9.23]	0.85 [0.84, 0.87]
Count-adapter stress	48.67 [48.46, 48.69]	43.52 [43.24, 43.63]	147.54 [147.52, 148.08]
Unchanged applications	9.53 [9.32, 9.78]	6.14 [5.90, 6.18]	0.12 [0.11, 0.12]

The 212 contracts comprise 48 development specifications, 12 structural compositions, 144 count-stress queries and eight unchanged applications. Three backends and five repeats give 3,180 query measurements in 60 subprocess batches. All backends receive the same parsed input access; no source-value metadata is supplied to a solver. Code imports and batch initialization are measured separately. Native solver-call limits are 1 s and the parent batch limit is 60 s. Process peak RSS is obtained from the operating system and includes native solver allocations. The released raw measurements permit alternative summaries without turning repeat variability into independent-task uncertainty.

Table A4: Peak process RSS in MiB / median batch startup in ms. RSS is the maximum over five batches, not additional per-query memory.

Query cohort	Auditor guard	Direct SMT	Exact enumeration
Development controls	89.0 / 46.8	82.1 / 47.7	48.2 / 48.6
Held-out structures	88.8 / 46.3	81.4 / 47.2	48.2 / 47.8
Count-adapter stress	94.0 / 46.4	76.9 / 46.7	48.4 / 46.9
Unchanged applications	90.0 / 47.1	82.7 / 48.2	48.3 / 46.9

The direct SMT comparator intentionally has no repair planner but performs the same native semantic and global identification obligations. During actual case acquisition, all conditional backends use the same fixed cost/name order and stop on a nonempty fiber with a singleton target-value set; the finite reference checks each step. Additional guard features are assessed as interface outputs, not as proof of superior solving speed.

D PUBLIC EXECUTION, RECORDING MAPS AND SUPPORT

Hosted runtime comparisons are paired under specified exact-history response reuse. Replicate IDs do not guarantee independent provider randomness. All-task results remain available beside outcome-conditioned both-success descriptions. Attempt disagreement is defined on ordered timing-free execution records, not merely successful final calls. Final projections must be named. A native normalized argument error does not itself identify function-body execution or side effects.

Output-support coverage asks whether a model's supported routes intersect a valid-route set. A grammar certificate can establish nominal support without establishing useful emission probability. If route samples are conditionally independent and identically distributed given task x, with identified valid-route mass $SM(x)$, then $BSC_M = \mathbb{E}_x[1 - (1 - SM(x))^M]$. Without those assumptions this formula is not a general repeated-Agent success law. Hosted finite emissions do not identify theoretical LLM support; unavailable support remains unknown.

The retained earlier global catalogue contains 432 task–structure–cost scenarios, not 432 new tasks; both exact repair backends resolve 336 at the same certified optimum. Its source provenance and timing revisions belong to the prior artifact, and are not pooled with the new unchanged applications or repeated semantic measurements. Earlier contaminated and later isolated hosted runs are separate. Successful re-extraction of an old record does not repair its causal provenance.

E EXTERNAL SDK APPLICATION: SEMANTIC INTERFACE AND EVIDENCE

The client span is produced by OpenTelemetry Python SDK 1.36.0, not by our final-route recorder. Its native JSON is preserved, including identifiers, status, timestamps and exception attributes. The exception stack trace is explicitly not collected by the fixture's privacy policy, preventing local paths from entering the anonymous artifact. The primary projection uses only ended/status/exception type; complete spans and latencies are not asserted equal. CPython Future and SDK source files, versions and hashes are included with their licenses.

The author defines the question and the bounded interface. Native variables are timeout, commit presence, order relative to span end, and worker-terminal status. The order decoder compares a commit begin/return interval with the SDK end timestamp. No commit maps to zero, an entirely earlier interval to -1, and an entirely later interval to 1. Overlapping intervals, timestamp disagreement, unsupported formats or absent quiescence cause refusal. This encoding preserves the declared strict temporal predicate when its decoder preconditions hold; it is not a proof about arbitrary clocks or distributed transactions.

The catalogue requests the client span, worker-terminal receipt and transaction evidence, in that fixed order. Declared weights are 0, 1 and 3; the transaction packet exposes commit presence and order with component weights 1 and 2. A shared reader performs identity checks and permits identical caching for every backend. These are declared access preferences, not monetary or optimal adaptive acquisition costs. Withholding is an explicit experimental catalogue restriction; the retained source is not falsely described as never recorded. A separate full-evidence validation checks the SQLite

database itself. Original timestamps, raw SDK records and transaction receipts are retained; the published targets can be recomputed without another model call.

The freeze precedes retained execution and method comparison but follows specification design and nine development boundary tests. Standard timeout semantics were already known. Three schedules repeated three times provide nine execution checks, not nine independent applications. No task was selected for a favorable measured result, and no superiority or prevalence gate was imposed. The original hosted panel, its sparse natural paired counterexamples and its response-coupling limitations remain unchanged.

F Related contracts and shared guarantees

Table A5: Comparison of documented contracts, not a benchmark of relative performance. Each guarantee assumes the supplied semantics; absent integration is not an impossibility claim.

Approach	Input / assumptions	Output, guarantee and evidence repair
Diagnosability synthesis (Bittner et al., 2012)	Transition model, faults and available sensors	Diagnosability and sensor selection by cost, minimality and delay. Distinguishing executions and selection already precede this work.
Translation validation (Lopes et al., 2021)	LLVM IR source/target semantics and supported bounds	Bounded refinement check or counterexample; motivates per-transformation validation, not our acquisition catalogue.
Beyond Local Accuracy (Luo et al., 2026)	Frozen finite policy class, estimand and observation support	Conditional identifiability, collision witnesses and exact minimum support. Our additional interface checks a supplied native-to-abstract target map against source-linked fields.
ClaimReceipt (Zhu & Chang, 2026)	Typed receipts, claim semantics and committed experiment universe	Claim-sufficiency and coverage verdicts, with selective verification. Our hashes bind available sources but do not prove complete ingress coverage.
This workflow	Native target, adapter, admissibility and obtainable-field catalogue	Preservation obligations, global/case/sample separation, verified acquisition and receipt revalidation. Solver, compiler and source premises remain trusted.

The comparisons identify shared mathematics and different implemented interfaces. They do not establish superiority over an alternative implementation supplied with the same obligations, nor substitute for checking the scientific meaning of a contract.

G PREMISE PROVENANCE AND TARGET-FIXED SENSITIVITY

The source ledger distinguishes a definition, two fixture-scheduling premises, and five source interpretation or decoder boundaries. Each entry supplies source identity, line anchors, a check type and its limit. Locating a branch is not proof of arbitrary Python execution, and a receipt assertion is not independent evidence of its truth. In particular, the episode-local database/no-other-writers interpretation and clock comparability remain scientific premises even when a decoder checks an exclusivity flag or rejects overlapping intervals.

Table A6: Temporal premises and the scope of available checks. The complete ledger includes source anchors and hashes.

Premise	Basis	What the check does not prove
$c \Leftrightarrow r \neq 0$	Definition in transaction decoder	Physical meaning after removing the definition
$r = 1 \Rightarrow u$	Worker release after span end; successful wait path	A universal law about SDK spans or clocks
$\neg u \Rightarrow w$	Successful Future wait in the fixture	Termination of arbitrary spawned work
No other writers	Episode-local database and single worker	Whole-system exclusivity from a Boolean flag
Commit ordering	Timestamp agreement, interval and quiescence guards	A precise commit instant or distributed-clock validity
Source identity	Recorded file hash matches retrieved bytes	Truth, completeness or semantic correctness

All eight subsets of the three printed constraints are evaluated for five fixed observation settings, giving 40 logical queries and 120 backend evaluations. A separately written literal calculation over the 24 raw valuations checks every target set. Under all three constraints, eight valuations are admissible. Dropping $r = 1 \Rightarrow u$ changes the no-timeout target set from $\{0\}$ to $\{0, 1\}$, whereas dropping only $\neg u \Rightarrow w$ does not change that answer. A committed-late receipt or a rollback receipt still settles its respective target under every constraint subset. All nine retained native executions satisfy the original constraints.

The target is held fixed throughout. Removing $c \Leftrightarrow r \neq 0$ admits contradictory encodings and is explicitly an encoding stress control. Removing a fixture restriction expands a declared space; it does not establish that the added symbolic witness is physically reachable. No sensitivity result is a new natural execution, probability estimate or human approval of a premise.

H ACCESS LAYOUTS AND SOURCE-LEVEL MAINTENANCE

H.1 DESIGN AND FAIRNESS

The extension reuses the exact 36 source records already selected without outcome filtering. The original records and their earlier outcomes were known. Before comparison, the protocol, measurement code, layouts and maintenance snapshots were hash-frozen. This is retrospective controlled tool validation, not new held-out environment evidence. No superiority or nonzero-counterexample gate was imposed.

Three materializations expose the same first-ordinal dispatch target: the unmodified whole record; two separate minimal projected/dispatch fields; and a projected field plus the existing complete attempt ledger. There is no padding or injected delay. Derived views retain source identity and are cross-checked against original primitives. A common materialization reads 1,146,582 source bytes, writes 1,566,990 bytes across all layouts and takes approximately 0.045 s in the retained preparation run. It is distinct from the older full-source scan and is not original collection cost.

Each policy/backend receives identical manifests, hashing, parser, file/value caches and result-cache permission. The selective order is projection then ledger if ambiguous; full-native access reads only the necessary raw field available in that layout. Source primitives grade answers after acquisition, never select a field. Cold means an empty application cache, not a flushed OS cache. Query setup

uses preloaded contracts and manifests; measured first-query time is acquisition plus verification, not a process launch or source-materialization time.

Table A7: Requested bytes and file opens per 36 records. Lower/tied/higher compares selective against full-native within a layout. All three backends give the same counts and 36/36 correct case answers.

Layout	Selective bytes	Full bytes	Lower / tied / higher	Opens S/F
Whole record	1,146,582	1,146,582	0/36/0	36/36
Minimal fields	6,771	5,232	0/0/36	46/36
Ledger + view	213,470	404,548	26/0/10	46/36

The separate-ledger layout saves bytes on 26 tasks and reads more on ten; minimal-field selection reads more on all 36. The direct minimal field remains cheaper than selective ledger access. These are layout-conditioned policy effects, not a unique solver capability, measured physical disk traffic, remote-system evidence or a reduction in the original model bill.

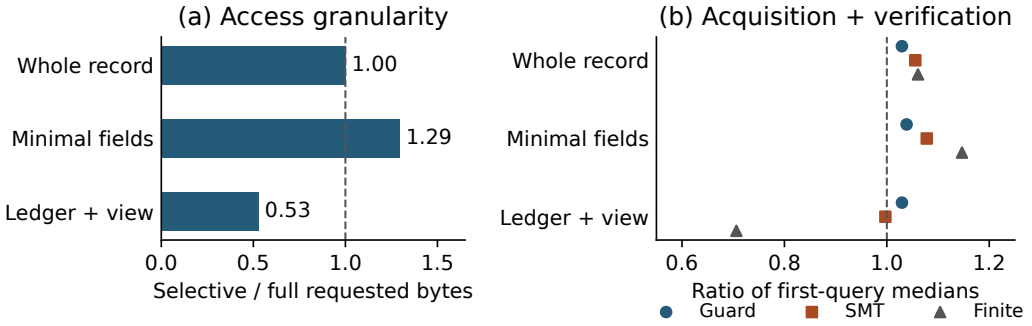


Figure A2: Controlled layout sensitivity. One denotes the full-native comparator within the same layout/backend. Left: exact byte ratios. Right: descriptive ratios of first-query timing medians, not confidence intervals. A lower byte ratio does not imply a faster guard. Absolute values and empirical quartiles accompany the figure.

Table A8: Acquisition plus verification, milliseconds: median [first, third quartile], pooling 36 records and five repeats in the one-query batches. Quartiles describe measurement spread, not population uncertainty.

Layout	Backend	Selective ms [Q1,Q3]	Full ms [Q1,Q3]
Whole record	Guard	4.929 [4.716,9.132]	4.788 [4.642,4.931]
Whole record	Direct SMT	0.749 [0.689,1.411]	0.710 [0.658,0.773]
Whole record	Finite	0.147 [0.116,0.191]	0.139 [0.106,0.185]
Minimal fields	Guard	4.874 [4.716,8.988]	4.693 [4.544,4.840]
Minimal fields	Direct SMT	0.672 [0.622,1.308]	0.623 [0.598,0.654]
Minimal fields	Finite	0.068 [0.055,0.096]	0.059 [0.051,0.067]
Ledger + view	Guard	4.900 [4.677,9.138]	4.759 [4.595,4.846]
Ledger + view	Direct SMT	0.672 [0.633,1.354]	0.674 [0.638,0.716]
Ledger + view	Finite	0.069 [0.055,0.137]	0.097 [0.082,0.129]

Three backends, two policies, three layouts, five randomized repeats and batches of 1/5/20 instances of the *same query* yield 9,720 timed batches, not that many independent problems. All answers are correct. Each batch begins with fresh application caches; subsequent identical questions can reuse a verified answer under the same frozen source epoch. After the first answer, both policies require zero further payload bytes. This is a cache-positive control, not evidence about 20 different questions or sources changing between requests. Fresh source validation is required outside that unchanged-epoch assumption. Raw cached lookup timings are retained without making precision-sensitive speed claims.

H.2 Actual source migration with an unchanged target

Two copied episodes, late commit and timeout rollback, undergo seven stages: original; transaction-key migration with stale identity; refreshed identity with old decoder; explicit decoder repair; permission revocation; restored source with incompatible decoder; and complete restoration. The migration renames the saved row field without adding telemetry. The declared v2 decoder restores the original meaning; the target, domain and constraints remain fixed. Every stage reads actual copied files, not only modified symbolic expressions.

Each conditional backend matches all 14 expected episode-stage results (42 evaluations); the full-source sample script matches 14 corresponding sample/refusal results. Stale identity yields source refusal, refreshed identity still yields schema refusal, repaired decoding restores the answer, and revoked permission withholds the transaction source. No stale receipt is accepted. Shared integrity and decoding explain agreement: this is a maintenance workflow check, not extra detection accuracy or measured human productivity. Repair semantics are supplied by the author/implementation, not automatically inferred from an arbitrary renamed field. Original telemetry and source files remain untouched.