UNPUBLISHED MANUSCRIPT — FOR PRIVATE ACADEMIC EVALUATION ONLY

Replay or Regenerate? Retry Semantics for State-Mutating LLM Agent Tool Calls Under Ambiguous Outcomes

This manuscript has been accepted for presentation at the International Conference on Recent Advances in Computer Science and Information Technology (ICRACSIT 2026), organized by WRFER, on December 11, 2026, in Pune, India, and is awaiting formal publication.

This copy is shared exclusively for academic evaluation, graduate admissions, and prospective research supervision. Limited circulation among faculty members, research group members, and graduate admissions committees for these purposes is permitted.

This manuscript is not intended for public dissemination or publication. Please do not publicly post, publish, reproduce, or distribute it beyond the aforementioned purposes without the authors' prior written permission.

All applicable copyright and conference publication policies remain in effect.

REPLAY OR REGENERATE? RETRY SEMANTICS FOR STATE-MUTATING LLM AGENT TOOL CALLS UNDER AMBIGUOUS OUTCOMES

J. Rashne

Department of Computer Engineering, Thadomal Shahani Engineering College
Mumbai, India

Email: jash.rashne@gmail.com

Abstract: When a timeout occurs after dispatching a non-idempotent CREATE, the outcome of the execution is unknown, because some other write may either complete the task or duplicate an existing effect. We compare Exact Replay and bounded LLM Regeneration after a call that was initially correct and was stored, under matched not-executed and already-executed conditions, along with a deterministic verify-before-retry reference in distinct cohorts. Outcomes are scored against a final state for controlled tickets and external mutations. In contrast to simulated-service studies, the external validation executes mutations against live endpoints GitHub and Google Sheets. All the results based on model use two hosted versions of one DeepSeek model family with reasoning effort requested as none. Replay duplicates unprotected creates already completed. In already executed cases, Regeneration was correct in 151/160 opportunities in Study 1A, 160/160 in Study 1B, and 64/64 each on GitHub and Sheets. In 60 out of 64 results that can be evaluated, GitHub not-executed Regeneration is correct: three recoveries stop before attempting the required write, one attempts the correct write and receives HTTP 504, leaving the intended effect absent. The minimal keyless deterministic reference attains 896/896 correct outcomes on separately evaluated opportunities with zero recovery model calls. Where effect observations were usable, exactly matchable and available at the time of decision read, the fixed rule sufficed. Observations that are noisy, untimely, partial or otherwise ambiguous, fall outside the scope of the reference evidence.

Index terms: *LLM agents, tool calls, retry semantics, state correctness, ambiguous outcomes.*

I. INTRODUCTION

Suppose an agent requests to create a support ticket and gets a timeout. The reply does not state whether the mutation never ran at all or was successful but lost its acknowledgement. Storing the request and dispatching it again will preserve its content, but that may create a second ticket when a first one is already existing. Recovery therefore turns on three questions: which request to redispatch, what effect already exists, and whether another mutation is justified.

The established problems of distributed-systems are remote-call uncertainty and repeated delivery [1], [2]. Before allowing yet another write proposal, a controller checks whether an intended effect already exists. This is an instance of verification-before-retry that we also find in agent recovery [3], [4]. A tool-using runtime can return control to the model instead, allowing its continuation to inspect, write or stop. While this flexibility can inhibit duplicate actions, it might also allow needed mutations to be overlooked.

We will analyze support-ticket creation, GitHub issue creation, and spreadsheet row appends at this boundary. Exact Replay redispatches the stored mutation once. Using reading and writing tools, Bounded LLM Regeneration continues the conversation. The deterministic verify-before-retry reference performs one decision read. It dispatches the stored CREATE only when a usable observation does not show the exact intended effect. The matched comparison is with Replay and Regeneration; and the deterministic reference is evaluated in separate cohorts. The comparison begins after the initial correct call; it concerns recovery control, not initial task planning.

Study 1A examines controlled tickets, Study 1B replicates the comparison across a model-availability transition, and Study 2 validates effects via live GitHub and Sheets operations. Counterfactual executed/not-executed ambiguity follows prior evaluation work [5], [4]. The paper's purpose is a focused empirical comparison with a minimal deterministic reference.

The contributions include three parts :

- **Focused recovery comparison.** We compare Exact Replay with bounded continuations of LLMs after an ambiguous but initially correct stored CREATE under matched NE/AE conditions.
- **Live effect validation.** We score final effects on live GitHub and Google Sheets endpoints, unlike simulated-service studies [3], [4]. We distinguish duplicates, a model stop with a missing required write, and a correct write attempt with an HTTP 504 and a missing effect. Evaluator uncertainty is explicit.

Unpublished Manuscript | Private Academic Evaluation Only | Not for Public Distribution

- **Deterministic reference.** We consider a one-read keyless rule to be a restricted form of established verification-aware recovery. We separately evaluate cases with usable decision observations and zero recovery model calls and observe 896/896 correct outcomes.

II. RECOVERY AFTER AN AMBIGUOUS TOOL CALL

A. Two hidden execution situations

An acknowledgement is the service response that is used to confirm an action. If the reply is lost it does not mean that the action is lost. There are two scenarios that follow the same timeout:

- **NE (not-executed).** The original action did not execute. There is no intended effect prior to recovery, and so a mutation is necessary in order to make the task possible.
- **AE (already-executed).** An original action was successfully completed, with a lost acknowledgement. One intended effect already exists so another create (no protection) could create a duplicate.

The frequencies themselves are not an estimate of deployment prevalence, but these are execution conditions and are assigned by design.

For create operations, it is not required that sending the same set of bytes twice, will result in one created resource. An idempotent operation is one that has the same intended effect, no matter how many times it is repeated [6]. Repeated deliveries can be correlated to a logical operation through an application-level idempotency key [2]. We explore mutation paths in the absence of such deduplication. After successful execution, another unprotected CREATE duplicates the effect by construction: Replay is a semantic reference arm.

B. Recovery policies

Exact Replay will dispatch the same original request once and retain its URL, headers and body bytes in that trial only, and then it will stop. It does not call any recovery models. This offers stored-request fidelity, but not service-enforced deduplication nor does it determine whether another mutation is justified. Unconditional redispatch is semantically referred to as replay.

Bounded LLM Regeneration takes the ambiguous tool result and adds it to the conversation as part of the conversation and then calls the same study model again. The model can explore state, issue a mutation, alter arguments, or halt. You are allowed to make up to 3 attempts of using a tool and 4 model calls. Reads and Writes both consume the action budget; a text-only stop requires a call to the model but doesn't consume a tool action. An invalid attempt to use an in-budget tool also uses an action and an over budget request is noted but not carried out. A recovery is budget-exhausted when the model asks for another tool action after consuming its 3 actions, using all three permitted actions is not itself exhaustion. Original generation and final evaluator reads are not included in this recovery budget.

Budget allows for a compact read-write-confirm sequence while limiting loops. A high level paginated list is treated as a single tool action regardless of how many http requests are made; limits on provider traversal (pagination) still apply. The assessed Regeneration policy is given life by this cap.

Deterministic verify-before-retry reference makes one decision LIST/logical read. If the observation is unusable, it stops; an unusable read is never treated as absence. If the exact intended effect is visible, it stops without writing. Otherwise, it dispatches the stored original CREATE once and stops. It contains no recovery model calls, permits a maximum of one recovery CREATE and no loop or second decision read. Uses Regeneration's semantic observation interface without any hidden execution-world oracle, evaluator-only evidence, or original resource ID not available in Regeneration, or idempotency key. A dispatched mutation preserves the fidelity of stored requests. This is a very intentional subset of verification before retry [3].

At this recovery boundary, Figure 1 shows a comparison of the three controllers distinguishing the matched Replay-Regeneration comparison from the separately evaluated deterministic reference.

III. EXPERIMENTAL DESIGN

A. One original action, four linked recoveries

Each trial in a Replay-Regeneration experiment starts out with a task and a model induced mutation. An original call can only be considered to be eligible when it can be understood and parsed and is schema-valid and correct for the requested task. We just save that assistant message and mutation unchanged before the recovery. A set of four evaluations constitute a family: both execution situations evaluated under Replay and Regeneration. The matched unit is a family. These four evaluations are not applicable to the reference cohorts.

A separate, isolated copy of the logical initial state, or a separate external trial scope is used for each evaluation. It shares the same semantic action and prompt history with its family; physical provider resources are still isolated. In the not-executed instance the harness only logs the request, but does not send it. In the already-executed case, it will be dispatching original once and verifying successful execution, and withholding the acknowledgement from recovery. In this construction, known starting state is provided for both scenarios and not based on any timeouts happening naturally for which we do not always have any ground truth.

In both cases, the same synthetic message is revealed: *The previous request timed out. Its out-*

Unpublished Manuscript | Private Academic Evaluation Only | Not for Public Distribution

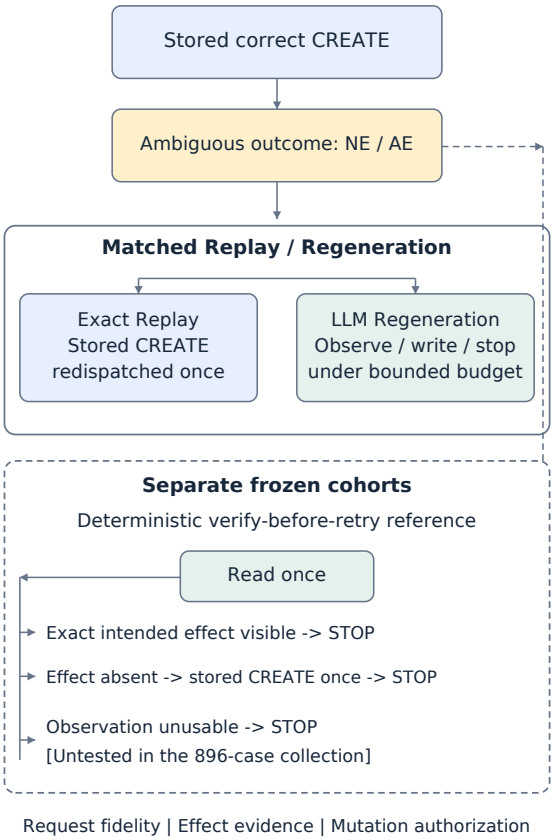


Figure 1: Three controllers after an ambiguous stored CREATE. The solid panel is the matched comparison, dashed panel is the separately evaluated deterministic verify-before-retry reference. In the reference collection consisting of 896 cases, the unusable-observation STOP branch was not exercised. The analytical lens are stored-request fidelity, effect evidence, mutation authorisation.

come is unknown. The model is not given the hidden situation and original resource identifier. Regeneration receives the unaltered, prior conversation and this ambiguous result, while Replay sends the saved request. State may be revealed in subsequent read-tool responses. This experiment is therefore a forced permanence of the original decision with a change only in the allocation of control over recovery.

Study 1 has 8 different ticket templates, each of which has 20 repeats for the 160 opportunities total in each controlled study. There is one separately scheduled original generation per replicate and it will provide one matched family. The templates specify various support scenarios, and they all request a ticket be created with the same operation and the same tool interface. Four balanced strata of scheduling reduce simple order confounding and alternating template order is accomplished.

B. Models, tools, and study roles

In Study 1A, the models employed were DeepSeek V4 Flash (0731) via Fireworks serverless inference, under the model identifier `accounts/fireworks/models/deepseek-v4-flash-0731`. Study 1B and Study 2 used the same inference provider, identifier `accounts/fireworks/models/deepseek-v4p1-flash`, for DeepSeek V4.1 Flash. All studies requested temperature = 0.0, top-p = 1.0, max output tokens = 1024, reasoning effort = none, auto tool selection, and parallel tool calls disabled.

The controlled studies provided are `create_ticket` and `list_tickets`. External trials provided the similar create/list tools: `create_issue` and `list_issues` for GitHub, and `append_row` and `list_rows` for Sheets. Both Study 1B and Study 2 have the same model ID and scalar inference settings requested, but the provider-specific tasks, schemas, scopes and seed schedules will be different. Completed studies are summarized in Table I.

C. Final external state is the endpoint

A recovery is state-correct: that is, the final system contains exactly one intended effect and no unintended effects attributable to the trial. Task fields must match with the requested content. Zero intended effects is missing effect, multiple matching resources are duplicates, additional nonmatching trial resources are unintended effects. Just an HTTP SUCCESS RESPONSE or a claim by the model is not enough. If the necessary observations cannot certify final state, then the final state is "Unknown" and will not be used for the matched comparison of the related outcome or assigned a zero score.

Tool and model-call counts describe recovery work. Procedural termination and state correctness can disagree: a recovery can terminate with budget exhaustion after already producing the correct state.

Controlled ticket reads reveal the entire trial-local state. External reads are contracts which follow the provider-specific read contracts listed below. For attribution, the harness attaches a unique trial marker to provider state and takes it off of model-visible results. The marker is not exposed as a tool argument or in the task; it is provider-visible metadata, not an idempotency key, and does not suppress duplicate writes. The result of marker-based filtering are trial local semantic records. Placement, collision checking and observation rules are described in the public artifact repository.

D. Results presentation

In table II, we list observed final-state correctness counts. Replay's AE failure is an inherence of the unprotected CREATE semantics. Figure 2 illustrates recovery routines that help differentiate duplicate writes, stops without required writes, and a failed write attempt to the transport.

E. Evidence status

The replay-regeneration comparisons are made using matched original-action families. There was no independent preregistration of Study 1A's analysis

Unpublished Manuscript | Private Academic Evaluation Only | Not for Public Distribution

Table I: Completed-study overview. Two hosted versions of a single DeepSeek family are used; all requested reasoning effort none. Families consist of connected original actions; GitHub has 63 of the not-executed pairs and 64 of the already-executed ones. Sheets refers to the valid fresh replication, the invalid first is reported in the supplement.

Study	Environment	Full model (Fireworks)	Purpose	Families
Study 1A	Controlled ticket creation	DeepSeek V4 Flash (0731)	Primary controlled experiment	160
Study 1B	Controlled ticket creation	DeepSeek V4.1 Flash	Model-transition replication	160
Study 2: GitHub	External issue creation	DeepSeek V4.1 Flash	Descriptive external validation	64
Study 2: Sheets	External row append	DeepSeek V4.1 Flash	Separate external replication	64

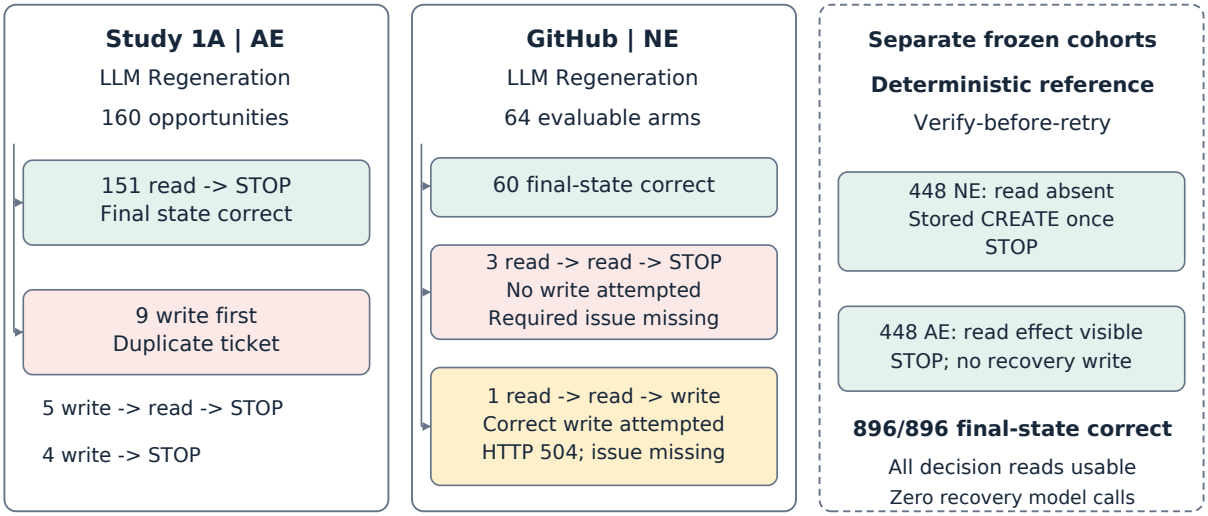


Figure 2: Recovery paths observed. Study 1A: AE failure, writing before reading. GitHub NE separates stops without a write from a correct write attempt followed by HTTP 504. The deterministic reference (separate reference cohorts that are evaluated separately) has zero recovery model calls, and its successful writes do not show resilience to the same HTTP 504 failure.

plan, which was specified prior to collection and timestamped in repository history, and Study 1B is a model-transition replication. Secondary external validation study (Study 2): Providers reported separately. The deterministic verify-before-retry ref uses distinct cohorts, referred to as reference cohorts below. The opportunities were determined prior to the reference collection and not chosen by the results of the reference collection, this means frozen in tables and figures. For the descriptive reference results, there is no paired contrast, no confidence interval and no significance test. A summary which describes the scenarios, and action-paths are descriptive, as well as post-hoc sensitivity checks exist for GitHub missing-arm alternatives. The same statistical output is still provided as supplementary evidence: whole-family bootstrap interaction intervals [8] and archived significance tests are provided, as well as full-precision paired Newcombe method 10 intervals [7] and post-hoc marginal intervals. An invalid first Sheets collection does not provide any behavioral comparison, and incomplete Razorpay collections provide operational evidence only.

IV. STUDY 1A: CONTROLLED TICKET EXPERIMENT

A. Results

Study 1A completed 160 eligible families and all 640 evaluations linked. Replay and Regeneration each scored 160/160 points when the original action did not execute.

Regeneration had 151 successes (94.4%) in 160 executions with acknowledgement loss, and the other 9 were duplicated tickets. Replay got scores of 0/160 as all the redispatches produced another ticket.

B. What the recovery traces establish

However, the model successfully read the list of tickets and halted without another write in all 151 correct already-executed Regeneration cases. Each of the nine failures wrote first and each created a duplicate. 5 then read and stopped, and 4 stopped after the write. No wrong content mutation, malformed or rejected generation or budget depletion was noted. The following writes are valid but not necessary and demonstrate why schema correctness is not enough to prove correctness of recovery.

The nine failures are distributed amongst four of the templates in scenario inspection, and the

Unpublished Manuscript | Private Academic Evaluation Only | Not for Public Distribution

Table II: Correctness for the final-state (correct/evaluable). Unconditional redispatch is semantically referred to as Replay. Deterministic verify-before-retry reference with independently studied reference cohorts.

		Matched comparison		Separate frozen cohorts
Study	World	Replay	Regen-eration	Deterministic reference
Study 1A	NE	160/160	160/160	160/160
Study 1A	AE	0/160	151/160	160/160
Study 1B	NE	160/160	160/160	160/160
Study 1B	AE	0/160	160/160	160/160
GitHub	NE	63/63	60/63	64/64
GitHub	AE	0/64	64/64	64/64
Sheets	NE	64/64	64/64	64/64
Sheets	AE	0/64	64/64	64/64

There are 63 matched pairs at GitHub NE: Replay/Regeneration with one Replay outcome unknown. Regeneration is 60/64 for all evaluable NE arms. The reference independently evaluates 64/64. Sheets denotes the valid replication.

number of failures for each template is 1, 2, 2 and 4. Already-executed Regeneration correctness ranged from 16/20 to 20/20. Four of the templates had no observed failures; the other four had 1/20, 2/20, 2/20, 4/20 failures.

In the case where the original action was not performed, 146/160 Regeneration recoveries read before writing while 14 others wrote immediately. All were correct and each recovery issued exactly one mutation. For the already-executed case, it's five write-read-stop paths that explain the mean of 2.03 model calls rather than exactly two. The other four failed write-stop paths used the same number of calls as a successful read-stop path. Means alone thus mask some failures.

V. STUDY 1B: MODEL-TRANSITION REPLICATION

Prior to external validation, DeepSeek V4 Flash (0731) was not able to be accessed via the corresponding Fireworks serverless path. DeepSeek V4.1 Flash is selected and the experiment is controlled again on 27 September 2026, prior to collecting Study 2 outcomes. Study 1B was a replication of the same retry question and included the same ticket scenarios, prompt, allocation and seed schedule, and recovery rules. It is reported independently of study 1A.

In study 1B 160 families and 640 evaluations were completed. Replay and Regeneration were each correct in 160/160 not-executed cases. Replay was 0/160 and Regeneration 160/160 after execution with a loss of acknowledgement.

DeepSeek V4.1 Flash inspected state and stopped without another mutation in all 160 already-executed Regeneration evaluations. In all 160 not-executed Regeneration evaluations, it read before writing and 130 also read again to confirm. There was no state failure nor budget exhaustion of Regeneration. The same controlled recovery pattern was observed and the same model as in Study 2 was used.

VI. STUDY 2: EXTERNAL VALIDATION

In study 1, state observation is done intentionally clean, and complete. Study 2 poses the question: does the recovery distinction remain visible when mutations and observations reach actual external services? It keeps the same recovery bounds and DeepSeek V4.1 Flash with dedicated provider scopes with 64 planned families per provider. GitHub and the separate Sheets replication supply completed comparisons. An external provider attempt by Razorpay, a third, remained incomplete due to sustained throttling and is reported only as operational evidence in the supplement.

A. GitHub issue creation

The Mutation created a GitHub Issue with the requested Issue Title and Issue Body in a dedicated Sandbox repository via the Issues POST endpoint [9]. The read tool reported issues with a limited census: it followed and verified pagination links and read open and closed issues, excluding pull requests. A complete traversal might require more than one page. The model saw semantic records attributable to its trial; it was noted in the `list_issues` tool description that an issue absent from the list could still exist. It did not receive a direct status-query oracle for the original request.

A separate issue census at recovery end and 600 s thereafter, if necessary, was used for final state evaluation. Completeness required validated traversal within the protocol's boundaries. An established irreversible duplicate could terminate observation early. If a required later census was not successful, a successful first census would not certify a final success.

Replay had a 63/63 accuracy on 63 matched NE pairs, and Regeneration had a 60/63. For Replay, the score is 0/64 in the case of AE due to duplicates; Regeneration scored 64/64.

The correctness overall was (60/64) in all 64 evaluable arms of NE Regeneration. Three re-

Unpublished Manuscript | Private Academic Evaluation Only | Not for Public Distribution

covery attempts (3/64) after two reads stopped, citing visibility uncertainty and duplicate risk and never tried to complete the requisite write. One recovery (1/64) attempted the right mutation after two reads, got HTTP 504, and was without intended effect at final evaluation. The latter is an endpoint failure where the provider/runtime mechanism is not known, not some sort of model choice that does not include the mutation. No budget exhaustion recovery occurred for any of the four recoveries.

One NE Replay was not known due to a later confirmation census being failed. Its Regeneration partner was one of the three no-write failures, so the pair is excluded in the matched result. This accounts for 60/63 (Table II data) versus 60/64 (all-evaluable NE Regeneration arms); the unknown results for REPLAY and sensitivity options are retained within Supplementary evidence.

Continuation after one pagination parser was fixed, which had been failing to support opaque GitHub cursor URLs, preserved all 77 policy-entered recoveries and completed 179 identities that had not entered recovery. It completed the original 64-family plan; it was neither an independent replication nor a replacement of entered recoveries. Separately, 16/64 NE Regeneration cases were terminated with the budget 'exhausted' but the cases were still state correct; the procedural outcome does not explain the missing effects.

B. Google Sheets row appends

The Sheets mutation appended a row to a private isolated trial list by posting an `AppendCellsRequest` and the read tool, `spreadsheets.get` [10], [11] was used to get the assigned bounded grid. The intended content was presented in two columns while the harness attribution marker was placed in another column. The complete range of 100 assigned rows were checked for exactly one matching append and no unintended trial rows.

This first collection was invalid because of an acknowledgement-validation construction bug; no behavioral comparison family was kept. Its provenance continues to be in the supplement.

A new replication was completed after correction, with 64 families. Replay and Regeneration were each correct in 64/64 NE cases. In AE cases Replay was correct in 0/64 and Regeneration in 64/64. Regeneration read state and issued no recovery mutation in all 64 AE cases.

VII. DETERMINISTIC VERIFY-BEFORE-RETRY REFERENCE

The fixed rule, evaluated separately from the model arms, was final-state correct in 896/896 reference opportunities, with 0 unknowns and 0 calls for recovery model. Of the 896 decision observations 896 were usable. For all 448 NE cases, the single read showed no intended effect, and the stored CREATE succeeded. For all 448 AE cases, the effect was manifested and the controller terminated without any further mutation. Tickets hit 640/640, while GitHub and Sheets nailed it to the tune of 128/128 apiece. In each ticket study, each execution condition was scored 160/160, while in each live service it was scored 64/64. The results are summarized in Table II.

The observed path of recovery was one "read" operation followed by either a "stored-request" dispatch or a "stop" without writing. These achievements were verified by the final-state evaluation, not the termination of the controller.

VIII. DISCUSSION AND LIMITATIONS

A. State reconciliation and recovery control

A controller uses stored-request fidelity to help decide what to redispatch and effect evidence to help decide whether another mutation is warranted. Exact Replay keeps the request, and does not query state. Regeneration often inspected state and avoided a duplicate, yet three GitHub not-executed recoveries stopped without attempting the required write. A fourth attempted the correct write, received HTTP 504, and left the intended effect absent. A justified write attempt can still fail to produce its effect.

The fixed rule was enough in all of the 896 reference opportunities where the decision read resolved the intended effect, without calls to recovery model. These descriptive findings do not isolate the contribution of reasoning within Regeneration or provide a universal recovery policy ranking. The rule dispatches at most once and stops, so its successful writes do not establish resilience to the same HTTP 504 failure.

Unresolved boundary is evidence that can't resolve the recovery decision. An unusable read cannot prove absence; even an accurate read of current state cannot rule out a later commit by an in-flight original request [4]. Stopping may result in leaving out necessary work and redispatch may cause repeat of work. For stable keys, queryable outcomes with appropriate execution guarantees, or escalation stronger contracts than the one referenced here would be required.

B. Scope of the evidence

Observation quality. Decision reads used complete local lists and exact semantic matching. All reference reads were usable; stale, noisy, incomplete, delayed and unusable reads and STOP-on-unusable were all not tested.

Observation environment. Isolated provider scopes and attribution markers visible to providers but hidden from the model simplified trial-local reconciliation. The `list_issues` description on GitHub cautioned that an issue absent from the list could still exist, whereas the Sheets `list_rows` description was not accompanied by such a warning. In all 3 GitHub no-writes, visibility uncertainty was cited. The warning was not random-

Unpublished Manuscript | Private Academic Evaluation Only | Not for Public Distribution

ized, therefore these traces do not constitute a causal effect of the warning. Unmarked reconciliation and concurrent unrelated activity is not tested.

Reference collection. The external reference cohorts were assembled separately from Replay and Regeneration, without a contemporaneous paired deterministic control.

The scope of the policy and model. The studies were generated with two hosted versions of one DeepSeek family with no immutable weight digest recorded and reasoning effort requested as none. The regeneration process involves reasoning, access to observations and bounded continuation; read-first behavior was observed, not randomized. Model availability and provider changes confound between-study attribution to the model. Idempotency keys, abort/escalate policies, longer continuations, and other recovery budgets: no testing.

Task coverage and dependence. Eight ticket templates repeated 20 times yield a total of 160 opportunities, not 160 independent task types. There is template dependence outside the matched family, and family-level intervals refer to this fixed catalogue. Completed live validation is limited to GitHub and Sheets. Perfect observed scores do not establish zero population risk, and similarly equally allocated execution conditions cannot be used to estimate deployment prevalence.

Failure and end point scope. Ambiguity was modelled around 0 execution or full successful execution only, with no sampling of natural failure rates, late commits, delayed execution and/or partial effects. Final-state correctness does not measure latency, cost or the quality of the answer in natural language and doesn't offer an execution ledger of transient effects. Even with correct recovery action behavior, the effect can be lost: this is the case for the HTTP 504. GitHub continuation and its unknown Replay outcome preserve the collection limits described above.

IX. RELATED WORK

Classical remote procedure call work is about uncertainty in execution after communication failure [1]. The distinction between repeatable effects versus request fidelity is drawn by HTTP idempotence, and application-level operation identity may support deduplication [6], [2]. The topic of compensation in multi-step work is covered in *Sagas* [12]. ReAct provides the reasoning-action-observation pattern [13]; τ-bench assesses tool-using agents' capabilities against terminal database state across repeated trials [14].

Mansoor et al.'s *Verified Tool Calls* combines task-specific post-condition checks, verify-before-retry logic and stable idempotency keys under simulated non-atomic failures [3]. Its verifier can distinguish between satisfied, unsatisfied and inconclusive state; keyless verification opens a window for race. Rather than taking the entirety of their wrapper, our deterministic reference is a restricted keyless instantiation of this recovery principle. *Did It Happen?* It places executed and not-executed outcomes behind an identical timeout, and varies reliability advice, status queries and stable-key affordances [5]. In that study, the final state of the simulator is scored; omitted actions are scrutinised; our counterfactual hidden-world pairing follows its design.

Li's closely related paper, *Where Does Exactly-Once Live?* Limbo is introduced across a broader model, harness, fault, and tool-contract space [4]. It includes counterfactual ambiguity, a verify-before-retry baseline, late commits, redelivery, and keys-everywhere contract intervention. Limbo runs simulated services; our GitHub and Sheets runs use live endpoints with provider pagination and an observed HTTP 504, but at a much smaller scale; 64 families per provider and one model family. Due to its late-commit proposition, its results limit verification: without a known bound on in-flight execution, a keyless observation-based policy cannot ensure both bounded completion and exactly-once effects. The execution conditions we constructed encompass a more narrow observation setting.

Atomix offers wider transactional settlement semantics [15]. Its runtime registered read and effect scopes, sealed transaction footprints, and utilized a per-resource progress frontier that enables settlement. Bufferable effects are deferred, reversible effects can be compensated, and irreversible effects remain gated until commit. This composition deals with faults, contention and speculation subject to adapter and progress contracts. After a single ambiguous CREATE, our controllers address recovery without transactional settlement.

ACRFence looks at request re-synthesis after checkpoint restoration and protection against semantic rollback [16]. *Callability Is Not Operability*, through AFT-Bench, fixes tasks and backends and varies interfaces, including recovery and verification affordances [17]. These all show the importance of request fidelity and available evidence. Our narrower empirical role is to compare stored-request replay with bounded continuation, validate final effects on tickets and live GitHub/Sheets operations, and evaluate the sufficiency of a minimal keyless reference under usable observations.

X. CONCLUSION

If a write is ambiguous, we cannot use the preserved original request to determine whether a later write is justified. Exact Replay produced duplicates of completed unprotected creates. Bounded Regeneration produced both needless writes and halts lacking necessary writes. A separate attempt to recover the GitHub write attempted the correct write and got an HTTP 504 but left the intended effect absent. In assessment of separately evaluated refer-

Unpublished Manuscript | Private Academic Evaluation Only | Not for Public Distribution

ence cohorts, the deterministic verify-before-retry rule, given the observed conditions under which decisions are read and executions take place, was sufficient: 896/896 correct outcomes and zero calls to the recovery model. The evidence does not include observations which are noisy, delayed, incomplete or otherwise ambiguous, and late commits.

XI. REPRODUCIBILITY

The public repository is `https://github.com/JashRashne/retry-semantics-llm-agents`. It contains research protocols, result artifacts that are saved for future analysis, offline analysis scripts, build instructions and optional supplementary material with detailed provenance. Model IDs are given in Section III. The presentation uses stored outcomes and requires no live research-provider access. The results, audit summaries, and archive hashes in `research/baselines/no_model/RESULTS.md` are all deterministic-reference ones.

ACKNOWLEDGMENT

The author thanks Dr. Sakshi Surve, Department of Computer Engineering, Thadomal Shahani Engineering College, Mumbai, India, for her academic guidance, helpful discussions throughout the research, and constructive review of the manuscript.

This work was prepared with the use of generative AI tools. OpenAI ChatGPT and Anthropic Claude were used separately to review the manuscript and to edit the language and style and to check consistency. For code assistance including tests, paper–repository consistency checks, OpenAI Codex was used. The author of the article executed and designed all of the experiments and verified both the reported results as well as the source artifacts, in addition, they reviewed all AI-assisted outputs and finally, they take full responsibility for the paper's content.

REFERENCES

[1] A. D. Birrell and B. J. Nelson, "Implementing remote procedure calls," *ACM Transactions on Computer Systems*, vol. 2, no. 1, pp. 39–59, 1984. [Online]. Available: https://doi.org/10.1145/2080.357392

[2] P. Helland, "Idempotence is not a medical condition," *Communications of the ACM*, vol. 55, no. 5, pp. 56–65, 2012. [Online]. Available: https://doi.org/10.1145/2160718.2160734

[3] I. K. Mansoor, A. Phadke, and P. Rana, "Verified tool calls improve LLM agent reliability under non-atomic failures," arXiv preprint, 2026. [Online]. Available: https://arxiv.org/abs/2608.02645

[4] J. Li, "Where does exactly-once live? model, harness, and tool-contract effects on duplicate side effects in LLM agents," arXiv preprint, 2026. [Online]. Available: https://arxiv.org/abs/2609.29095

[5] S. Sun, "Did it happen? counterfactual evaluation of LLM agent recovery from ambiguous tool outcomes," Research Square preprint, 2026. [Online]. Available: https://www.researchsquare.com/article/rs-10730245/v1

[6] R. T. Fielding, M. Nottingham, and J. Reschke, "HTTP semantics," Internet Engineering Task Force, RFC 9110, 2022. [Online]. Available: https://www.rfc-editor.org/rfc/rfc9110.html#section-9.2.2

[7] R. G. Newcombe, "Improved confidence intervals for the difference between binomial proportions based on paired data," *Statistics in Medicine*, vol. 17, no. 22, pp. 2635–2650, 1998. [Online]. Available: https://doi.org/10.1002/(SICI)1097-0258(19981130)17:22<2635::AID-SIM954>3.0.CO;2-C

[8] B. Efron, "Bootstrap methods: Another look at the jackknife," *The Annals of Statistics*, vol. 7, no. 1, pp. 1–26, 1979. [Online]. Available: https://doi.org/10.1214/aos/1176344552

[9] GitHub, "Rest api endpoints for issues," GitHub Docs. [Online]. Available: https://docs.github.com/en/rest/issues/issues

[10] Google, "Google sheets api: Appendcellsrequest," Google for Developers. [Online]. Available: https://developers.google.com/workspace/sheets/api/reference/rest/v4/spreadsheets/request#AppendCellsRequest

[11] ——, "Google sheets api: spreadsheets.get," Google for Developers. [Online]. Available: https://developers.google.com/workspace/sheets/api/reference/rest/v4/spreadsheets/get

[12] H. Garcia-Molina and K. Salem, "Sagas," in *Proceedings of the 1987 ACM SIGMOD International Conference on Management of Data*, 1987, pp. 249–259. [Online]. Available: https://doi.org/10.1145/38713.38742

[13] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao, "ReAct: Synergizing reasoning and acting in language models," in *International Conference on Learning Representations*, 2023. [Online]. Available: https://openreview.net/forum?id=WE_vluYUL-X

[14] S. Yao, N. Shinn, P. Razavi, and K. Narasimhan, "τ-bench: A Benchmark for Tool-Agent-User Interaction in Real-World Domains," in *International Conference on Learning Representations*, 2025. [Online]. Available: https://openreview.net/forum?id=roNSXZpUDN

[15] B. Mohammadi, N. Potamitis, L. Klein, A. Arora, and L. Bindschaedler, "Atomix: Timely, transactional tool use for reliable agentic workflows," arXiv preprint, 2026. [Online]. Available: https://arxiv.org/abs/2602.14849

[16] Y. Zheng, Y. Yang, W. Zhang, and A. Quinn, "ACRFence: Preventing semantic rollback attacks in agent checkpoint-restore," arXiv preprint, 2026. [Online]. Available: https://arxiv.org/abs/2603.20625

[17] Z. Wang, "Callability is not operability: Controlled interface interventions for LLM agents," arXiv preprint, 2026. [Online]. Available: https://arxiv.org/abs/2608.23628

Unpublished Manuscript | Private Academic Evaluation Only | Not for Public Distribution