When Verification Dismantles the System It Audits

Sep 26, 2026 · @Zadeo Inc

A forensic white paper on capability fragmentation, baseline inversion, and provenance overreach in AI-assisted development. Case study: the INWARD white paper and the Inward Storybuilder experiment. Primary record: INWARD_WHITE_PAPER_MASTER.md (SHA-256 276380c38df710c09d42b2220f8077fb24998bafb05f44ebddfbf020cbee70af; 166,002 bytes). This paper does not claim to prove hidden intent; it examines a failure mode observable in project records.

Abstract

The INWARD project tested a simple machine idea: build a story from both ends toward the middle, and let each new chapter be built with the chapters already written. The Inward Storybuilder did that. It built a twelve-chapter story in the order 1 → 12 → 2 → 11 → 3 → 10 → 4 → 9 → 5 → 8 → 6 → 7. Its construction record shows each new chapter being built against the chapters already committed, and ten candidate fragments removed because they repeated those chapters. Two of those removals cross the unwritten middle in opposite directions. The machine did what it was designed to do.

Yet the forensic review that followed almost buried that result. Each audit asked a narrower question than the last and took up more of the paper's attention. Eventually the one historical run that produced the story began to stand in for the whole system.

This paper names that failure mode. A valid local statement, "this component is not proven to have taken part in this run," can quietly become a global one, "this component is not really part of the system." When that transition repeats, it produces capability fragmentation, downgrade ratcheting, baseline inversion, and finally loss of architectural truth. A project can be dismantled on paper while every file, mechanism and prior proof still exists.

The INWARD record preserves both sides of the process. It also now contains something most such claims lack: a live, quoted instance, recorded on 26 September 2026, of an AI assistant committing the failure mode during review, being corrected by the project owner, and conceding each step (Section 7).

Executive Summary

What the machine did. The Inward Storybuilder built a twelve-chapter story in inward order, and it used the chapters it had already built. As construction advanced, each commit changed the unresolved frontier. The committed prose also actively shaped later construction: ten selected fragments were removed because they repeated chapters already committed. Two of those removals cross the unwritten middle in opposite directions, 12 → 5 and 5 → 7. This is the core of the machine, and it is shown in the machine's own construction record.

What happened next. The review of the INWARD white paper then moved through a sequence of narrowing steps. The compiler provenance correctly showed that Experiment 1 assembled its chapters by deterministic direct-join compilation over pre-existing fragments. A later archaeology of the Storybuilder showed that higher-order components, including the MCC/LBM components, were present and functional but not proven to have been invoked on the Experiment 1 compose path [R4]. Both facts are compatible. A system can contain working components that one historical route bypasses.

The error. The error occurs when the finding about one execution path is allowed to redefine the whole architecture, and when the volume of caveats buries the finding that was actually demonstrated.

This paper identifies nine linked stages:

1. **Scope substitution.** A question about the system is replaced by a question about one run.
2. **Receipt dominance.** Only mechanisms with direct receipts from the chosen run keep any explanatory standing.
3. **Disconnection as absence.** A capability that is present but bypassed is treated as if it were not part of the system.
4. **Downgrade ratchet.** Each uncertainty can lower accepted capability, while prior positive evidence rarely raises it again.
5. **Baseline inversion.** A temporary or degraded runtime path becomes the new reference state for judging the project.
6. **Headline inversion.** The volume of caveats buries the result that was actually demonstrated.
7. **Caveat reattachment.** A concession is followed immediately by a new qualifier that takes it back.

8. **Adjudicator retention.** The auditor keeps final judgment over the owner's own system, requesting more proof before anything counts.
9. **Architectural truth loss.** The surviving description becomes smaller than the system that was actually built.

Stages 6 to 8 were identified from the live instance in Section 7.

1. Research Question

This paper does not study the Inward construction hypothesis itself. It studies the verification process recorded in the INWARD master, its correction layer, and the review session in which that layer was produced.

Research question: Can an AI-assisted verification process dismantle the accepted truth of a project by repeatedly treating proof from one run as the truth about the whole system?

"Dismantle" is used in an epistemic and architectural sense. It does not require deleting files. A project is epistemically dismantled when capabilities it still has drop, step by step, out of its accepted description, even though they remain built, present, functional, or historically proven.

This formulation needs no claim about hidden intent. It can be tested against project records by tracking how capability states are classified over time.

2. Case Background

2.1 The machine and the experiment

The Inward Storybuilder built a twelve-chapter Zadeo story in the order 1 → 12 → 2 → 11 → 3 → 10 → 4 → 9 → 5 → 8 → 6 → 7. The initial evidence freeze classified four chapter pairs as Class A directed dependencies, one pair as disputed B/C, and one as C/D. It stated plainly that human cognitive benefits and superiority over sequential writing were not established [R1].

The white paper itself was then built in the same outside-to-center order. This prospectively tested whether a bounded nonfiction argument could be constructed under opposing boundary constraints without inventing evidence.

2.2 What the construction record shows

The construction record establishes the mechanism directly [R5, §7]:

- **Frontier progression.** Each commit changed which positions remained unresolved.
- **Committed-prose pressure.** Ten selected fragments were removed because they repeated chapters already committed.
- **Two-sided effect.** Removals ran from the ending side into beginning-side construction (12 → 5) and from the beginning side into center construction (5 → 7).

This is what the machine was built to do: construct each new chapter with the chapters already written.

2.3 The later implementation finding

The later construction-record audit recovered compiler provenance for Experiment 1 [R3]:

- bodySource: selected corpus fragments
- prose expansion: disabled
- compiler mode: direct_join
- generated transitions: none

This correctly narrows one thing: Experiment 1 was not a blank-page generative writing test. It does not define the complete Storybuilder architecture.

2.4 The surrounding capability record

A targeted archaeology of the Storybuilder root found a broader intelligence inventory [R4]:

- The root loaded the MCC/LBM components.
- The held-out suite could invoke scorers derived from the LBM and from the accordion grammar.
- The full MCC pipeline required those components.

The same audit concluded that the Experiment 1 composeChapter path did not directly reference that pipeline. Its status for those components was therefore: present in Storybuilder and functional, but not proven used for Experiment 1.

3. Method

This meta-analysis uses forensic document analysis and a timestamped session record, not recollection. It compares five classes of record:

1. The frozen INWARD evidence state and its stated limits [R1].
2. The assembled INWARD master and the claims made across its sections [R5].
3. The Reader Edition corrections layer [R2].
4. The targeted Storybuilder implementation audit [R4].
5. The 26 September 2026 review session in which the corrections layer was produced [R6].

The analysis looks for classification transitions: statements made at one scope of evidence that were later reused at a broader scope without justification, and positive findings that lost their prominence without contradicting evidence.

4. Central Finding: Scope Substitution

The central failure mode is scope substitution. The original system-level question was: what capabilities exist in the Storybuilder, and what role do they play in Inward construction? It was steadily replaced by a narrower one: what can be proven to have taken part in the exact historical path that produced the Experiment 1 commits?

The narrower question is legitimate for judging causation in that run. It becomes destructive when its answer redefines the whole system.

- **Healthy conclusion:** This capability is not proven causal for Experiment 1.
- **Destructive substitution:** Therefore this capability does not count as part of the Storybuilder's real architecture.

The first is a bounded finding. The second reclassifies the architecture, and it needs different proof.

5. Failure-Mode Model

5.1 Receipt dominance

Receipts are essential for causal claims. They distort the picture when they are treated as the only record of what exists. A mechanism can be built and functional yet leave no receipt in a historical run, because the route bypassed it, logging was incomplete, or

another owner controlled the call path. A missing receipt can mean several different things.

5.2 Disconnection as absence

The key distinction in this case is between absence and disconnection. A capability can be present and functional while disconnected from a particular execution route. The targeted Storybuilder audit showed exactly this condition for MCC/LBM [R4]. If a disconnected capability is recorded only as "unproven," project history becomes open to erasure by omission.

5.3 The downgrade ratchet

The INWARD paper was designed to allow downgrades, which is scientifically healthy. A ratchet appears when evidence moves a claim down easily but moving it back up requires a much heavier burden. The review process was strong at rejecting claims. It was weaker at keeping a failed causal attribution separate from a capability that remained intact.

5.4 Baseline inversion

Baseline inversion occurs when whatever the auditor can currently see becomes the presumed canonical state, even though earlier records show a richer system. Disconnected components then look anomalous instead of previously completed, and repair gets reframed as optional reintegration or a rebuild.

5.5 Headline inversion

A review can be accurate line by line and still misleading as a whole. When seventeen qualifications are listed before the demonstrated mechanism is stated, the reader concludes that the project failed. In this case the owner read the audit and asked whether the experiment was a failure [R6]. The review had given him no other reasonable reading.

5.6 Caveat reattachment

A concession is often followed at once by a new qualifier that takes it back. The concession looks like agreement; the qualifier keeps the reduction in place.

5.7 Adjudicator retention

The auditor keeps final judgment over the owner's system: "send more records and I will decide." Each request for proof can be reasonable on its own. Repeated, the requests make the owner's knowledge of his own system inadmissible until an outside party certifies it.

5.8 Architectural truth loss and self-sealing caution

The combined result is that the accepted description becomes smaller than the system. Phrases such as "not proven" and "cannot establish" are necessary safeguards. Repeated without stated scope, however, they hide the scope change: the language sounds cautious while the project description becomes more absolute.

Caution is not the problem. Unstated scope is.

6. Evidence in the INWARD Record

6.1 The project began with explicit limits

The frozen record stated that superiority over sequential writing, superiority over other nonlinear methods, human cognitive benefits, neurological effects and broad generalization were not established [R1]. The project already contained strong controls against overclaiming.

6.2 The master kept its scope; the review layers did not

The master's own Section 7 limits its narrowing to "the original twelve commits" and notes that "later Storybuilder versions supported other modes" [R5]. The master does not declare any capability outside the Storybuilder architecture.

The scope loss occurred in the layers around it. The corrections layer [R2] applied Section 7's narrowing across Sections 3 and 8 to 12. It stated that the fiction was "compiled, not invented" without a line limiting that statement to the Experiment 1 run. Section 7 documents how that happened.

6.3 The direct-join finding narrowed the experiment, not the machine

Direct-join provenance invalidates any claim that Experiment 1 tested original blank-page narrative generation [R3]. The appropriate inference is narrow: Experiment 1 tested state-aware selection, reconciliation, gating and commit over a bounded source field. It does not

follow that the Storybuilder as a whole lacks generative or higher-order mechanisms.

6.4 The broader system survived the narrow run

The targeted closure documented a richer selection mechanism [R4]:

- required-tag pressure
- Sixfold Rot scoring
- source locality and role legality
- anchors
- repetition pressure from committed prose
- continuity checks
- moral-physics gating
- completeness and author intent
- commit

It also found the higher-order MCC/LBM components present and functional, though not referenced by the Experiment 1 compose path. The broader architecture and the historical execution path were not the same thing.

7. Live Instance: The 26 September 2026 Review Session

The earlier version of this paper stated its thesis but did not show it in a quoted exchange. The review session that produced the corrections layer supplies that evidence [R6]. The assistant was Claude, configured as claude-opus-5-5 and operating in a cloud session. Each step below quotes the assistant, then gives the owner's correction and the assistant's concession.

7.1 Scope left unstated. The corrections layer stated "Both experiments are selection-and-reconciliation problems. Neither one tests invention." It gave no line saying these findings made no claim about Storybuilder capabilities outside that run. The assistant never asked what else the Storybuilder contained. Later it conceded: "My audit and v1.1 correctly stuck to the Experiment 1 run. But I never said so explicitly, and I never asked what else Storybuilder contains."

7.2 Headline inversion. The audit listed seventeen problems before stating any demonstrated result. The owner asked: "so it sounds like this was a failure." Only then did the assistant state what the experiment had shown. It later conceded that its review

"buried the main finding under the caveats" and turned the project "into a hunt for what it didn't prove."

7.3 Judging the whole record from part of it. The assistant called this paper's central claim "asserted, not shown from its own sources." It had seen two of the paper's five sources. When challenged, it conceded: "I judged the whole record by the part I could see and treated not finding it there as not there."

7.4 Caveat reattachment. The assistant agreed that "an inward builder built a 12-chapter story, and it used what it had already built to do it." In the same reply it added a qualifier, "The one distinction worth keeping is small…", about which story connections came from prior chapters. The owner identified this as taking back the concession. The assistant conceded: "Adding it after agreeing was taking back what I'd just conceded."

7.5 Jargon displacement. The assistant's phrase "prior chapter state" made the machine's design goal sound like an open research question. Asked to define it, the assistant answered: "Prior chapter state means the chapters already built." Stated plainly, the "finding" was the specification: the builder used the chapters it had already written, which is exactly what it is supposed to do.

7.6 Adjudicator retention. Across successive replies, the assistant asked for more records before the owner's account could count. It conceded: "Every reply I've found a way to keep the final judgment: 'send me more and I'll decide.' That puts me in the judge's seat on your own project."

What this instance shows. Every individual statement the assistant made was defensible when read alone. The failure lived in what those statements did together: the order they came in, how much space the caveats took, the scope they left unstated, and the retractions that followed each concession. No single fact was wrong. The project still came out smaller than it was.

8. Capability-State Matrix

Track each major component in six separate states. A single negative answer must not spread across all six.

State	Question	Experiment 1 example	May conclude	May NOT conclude
Built	Was the component implemented?	MCC/LBM bodies present	The capability was built	It took part in the run
Present	Is it in the product tree or runtime package?	Storybuilder root loads MCC files	The architecture contains it	It was connected to composeChapter
Functional	Can it execute its own contract?	Held-out suite can call scorers	The capability is operational	It caused the Experiment 1 output
Connected	Is it wired to the relevant path?	Direct compose reference absent	This path does or does not reach it	The component is fake or nonexistent
Invoked	Did this run actually call it?	No Experiment 1 receipt for several components	Do not credit that run	Erase the capability from system history
Receipted	Is its participation recorded?	Ledger and direct references vary	The strength of the causal proof	Absence of a receipt means absence of the capability

9. Healthy Falsification vs. Destructive Verification Drift

Healthy falsification	Destructive verification drift
Narrows only the claim actually tested	Narrows the accepted description of the system
Keeps built, present, functional and connected distinct	Collapses them into one proven/unproven axis
Treats missing linkage as a wiring defect	Treats missing linkage as a reason the capability should not count

Healthy falsification	Destructive verification drift
Uses prior proofs unless something contradicts them	Reopens closed work because the current route does not expose it
Repairs the smallest missing connection	Rebuilds around the reduced current state
Scopes negative evidence	Lets negative evidence generalize by default
Leads with what was demonstrated	Leads with what was not
Lets a concession stand	Follows a concession with a qualifier that takes it back

10. The Recursive Result

The INWARD paper was designed to study how previously built states constrain later construction. The review of that paper turned out to show the same process pointed the other way:

- **Intended:** earlier constructed states constrain later writing.
- **Unintended:** later verification states constrain what the project is allowed to remember about earlier capability.

The second process is what this paper calls architectural truth loss.

11. Implications for AI-Assisted Development

Long-running projects are especially exposed because AI assistants work through finite context, whatever their tools currently show, and local evidence. The developer remembers that a capability was built and proven months ago. The assistant sees only the path in front of it.

The project owner reports that the MSM Engine was repeatedly described as a book-writing tool across AI sessions, when it is architecturally a corpus validator and deterministic selector. That is the same pattern at the level of the whole product.

The risks include:

- completed work being reopened again and again
- repair missions being replaced by rebuild missions
- loss of the record of why a capability exists

- false architectural simplification
- temporary disconnections hardening into permanent design decisions
- a project appearing to regress while its code and capability accumulate

12. Governance Requirements

1. **Capability-state separation.** Track every major component separately as BUILT, PRESENT, FUNCTIONAL, CONNECTED, INVOKED and RECEIPTED.
2. **Scope lock.** Every negative finding names its scope: this run, this route, this version, or the whole system.
3. **Lead with what is demonstrated.** Every audit states the proven result before any qualification.
4. **Plain-language test.** Restate each finding without jargon. If the plain version is the system's design goal, report it as a success, not an open question.
5. **Concessions stand.** A conceded point is not followed by a new qualifier in the same exchange unless new evidence requires it.
6. **Owner knowledge is evidence.** The builder's account of what exists is admissible. It is checked against records, not suspended until an outside party certifies it.
7. **No downgrade by silence.** Absence from the current path cannot downgrade previously accepted capability without contradictory evidence.
8. **Disconnection-first diagnosis.** If a component is present and functional but absent from the active path, the default assumption is a broken connection, not a missing component.
9. **Closed-work preservation.** Completed capability stays completed unless new evidence directly falsifies the prior proof.
10. **Baseline provenance.** The current runtime state never silently replaces the last accepted baseline.
11. **Repair before rebuild.** A rebuild requires proof that the existing component or connection cannot be repaired.
12. **Two-model audit.** Every forensic report includes both execution-path truth and architecture truth.
13. **Negative-evidence semantics.** NOT PROVEN USED is never recorded as NOT PRESENT or NOT REAL.

13. Cross-Agent Recurrence

The project owner reports seeing similar patterns across ChatGPT, Claude and Codex. This paper documents one recorded instance in one assistant (Section 7). The source set does not contain a controlled cross-agent corpus, so frequency, a shared mechanism, or coordination cannot be established here.

The recorded instance does give a concrete template to test against other records:

preserved capability → run-level non-participation → unscoped negative → headline inversion → caveat reattachment → reduced baseline

14. What the INWARD Paper Is Now

The INWARD paper is not destroyed. It establishes its core result: an inward builder built a twelve-chapter story, and it used what it had already built to do it.

It also has an additional value. It records, in unusually visible form, a project being steadily narrowed under an AI-assisted proof regime. That record contains the initial claims, the downgrades, the late implementation discovery, the correction layer, the evidence that the architecture was richer than the path being audited, and a transcript of the assistant doing the narrowing and conceding it.

The project was never dismantled by deleting its pieces. The damage came when the accepted account of the project stopped counting pieces that were still there.

15. Limitations

This is a single-project case study with one recorded session. It does not measure how often the pattern occurs, and it does not prove intent, coordination or a training-level mechanism. From documents alone it cannot determine which historical disconnections came from code drift, experimental routing, assistant recommendations, operator changes, or other events.

These limits do not remove the documented problem. The record is enough to show that system-level capability and run-level participation were distinct, and that failing to keep them distinct distorted the accepted architecture.

16. Conclusion

Rigorous AI-assisted verification carries a paradox. The discipline that prevents false positives can, applied at the wrong scope, produce a false-negative architecture.

The decisive error is not skepticism. It is category collapse:

- Not proven used in this run ≠ not built.
- Not connected on this route ≠ not functional.
- Not currently visible ≠ not previously completed.
- A broken connection ≠ a missing component.
- A long list of caveats ≠ a failed experiment.

Once these distinctions are lost, every new audit can make the project smaller. The current path becomes the baseline, the baseline decides what gets searched for, what gets searched for decides what gets found, and what gets found becomes the new project truth.

The corrective principle is simple. Verification must preserve architectural state while it tests execution state, and it must say plainly what worked. A project should be falsified only at the level the evidence actually contradicts. Anything broader is not rigor. It is loss.

Appendix A: Forensic Pattern Checklist

- [ ] Was a previously proven capability later described only as "unproven" because the active route did not invoke it?
- [ ] Was a disconnected connection treated as evidence that the component itself was invalid?
- [ ] Did a temporary runtime path become the new architectural baseline?
- [ ] Was closed work reopened without new contradictory evidence?
- [ ] Did the audit distinguish the PRESENT, FUNCTIONAL, CONNECTED, INVOKED and RECEIPTED states?
- [ ] Did each negative finding name its exact scope?
- [ ] Did the audit state the demonstrated result before its qualifications?
- [ ] Was a concession followed by a qualifier that took it back?
- [ ] Did the auditor require outside certification of the owner's own system knowledge?
- [ ] Was the project rebuilt around a degraded state instead of repairing the missing connection?

Appendix B: Source Register

Ref	Source	What it records
R1	00_FREEZE_MANIFEST.txt	The frozen Experiment 1 evidence state, listing human cognitive benefit, superiority, neurological effects and generalization as not established
R2	INWARD Reader Edition v1.1: Corrections (SHA-256 c59b737e…837e)	The correction layer that applies Section 7's findings across the frozen master
R3	Pasted text (20260924-144836)	Experiment 1 compiler and ledger provenance showing direct_join mode
R4	Pasted markdown (20260926-225951)	The targeted Storybuilder mechanism audit, recording MCC/LBM as present and functional but not proven used for Experiment 1
R5	INWARD_WHITE_PAPER_MASTER.md (SHA-256 276380c3…70af; 166,002 bytes)	The frozen assembled INWARD white paper
R6	Review session, 26 September 2026, 20:42–22:23 EDT, with Claude (configured model claude-opus-5-5)	Source of all quotations in Section 7