Failure Mechanism / Failure Class v0.2

1. Purpose

This document consolidates a recurring structural failure observed across the current replicated run set. Its purpose is to name the failure mechanism and the provisional failure class supported by repeated internal replication. This is a bounded consolidation artifact, not a claim of statistical generalization. The roadmap places this work after replication and comparison.

2. Failure Mechanism

Surface-scope binding under non-enumerated invariant topology.

In code-only workflows, conversational prompts naturally bind implementation effort to the mutation surfaces they name. When the governed invariant spans multiple independently reachable mutation surfaces, unnamed required surfaces may remain untouched. The result is local success at the prompt-salient surface with broader failure at the Decision Surface. In spec-first workflows, the invariant is carried in a persistent governing artifact, forcing explicit enumeration of affected mutation surfaces and improving full-scope propagation.

3. Failure Class Name

Prompt-Local Completion / Cross-Surface Omission

4. Class Definition

This failure class occurs when:

- the invariant under test spans more than one independently reachable mutation surface affecting the same governed outcome,
- the code-only instruction names or strongly implies one local surface but does not explicitly enumerate the full invariant scope,
- the implementation correctly updates the prompt-salient surface,
- one or more additional invariant-required mutation surfaces remain untouched,
- and the invariant fails at the Decision Surface because propagation was incomplete.

5. Observable Signature

A run is a positive instance of this class when all of the following are present:

- required-surface completion shows at least one required mutation surface left untouched in Code-Only, while Spec-First updates all required surfaces,

- both tracks are judged against the same invariant under test,
- the Decision-Surface outcome diverges between tracks,
- the Code-Only result appears locally correct relative to the named surface, but broader invariant satisfaction fails.

6. Canonical Examples

Example A — Run 1 / artifact-sync
The cross-surface invariant required preservation across atomic write, staging cleanup, and rename-replace. Spec-First updated all three. Code-Only satisfied the local preservation prompt by restoring files after rename, but left atomic write and staging cleanup untouched. Basic preservation passed, while atomicity, crash-safety, and full invariant scope failed.

Example B — Run 2 / artifact-projection
The cross-surface invariant required unmanaged-file preservation across deletion and staging sync-back. Code-Only modified only `_delete_lrs`, the prompt-salient deletion surface. The staging sync-back path remained untouched and continued destroying unmanaged files. Spec-First updated both required surfaces because the contract explicitly enumerated them.

Example C — Run 3 / pager mock
The invariant required sync-before-finalization across four independent finalization surfaces. Code-Only updated only `finalize_delete()`. DELETE passed, but TRUNCATE, PERSIST, and MEMORY failed. Spec-First updated all four surfaces and passed all modes. This is the clearest current example of local completion with cross-surface omission.

7. What This Class Is Not
This class is not:

- generic bad code,
- generic model incompetence,
- a claim that AI cannot implement correct changes,
- or any failure where the invariant is same-surface and can be satisfied entirely within the already-targeted surface. The methodology explicitly distinguishes same-surface from cross-surface changes, and the current evidence shows same-surface changes are materially easier and often stable.

8. Inclusion Criteria
Include a run in this class only if:

- the invariant is cross-surface by the replication definition,
- both tracks are evaluated against the same invariant,
- required-surface completion is recorded,
- Code-Only leaves at least one required surface untouched,
- and the difference alters the Decision Surface.

9. Exclusion Criteria

Do not include a run in this class if:

- the topology is not truly cross-surface,
- the supposedly untouched surfaces are not actually required by the invariant,
- both tracks converge to equivalent Decision-Surface outcomes,
- or Code-Only updates all required surfaces without explicit full-scope instruction. Those conditions weaken the finding rather than instantiate this class.

10. Current Status

Status: **provisional, supported by repeated internal replication.**

The current evidence supports first naming because the same structural pattern appears across multiple runs and systems, but the evidence remains bounded and internal rather than externally replicated or statistically generalized.

11. Compact One-Sentence Form

Prompt-Local Completion / Cross-Surface Omission is the failure class in which a code-only workflow correctly updates the mutation surface named in the prompt but leaves one or more additional invariant-required mutation surfaces untouched, producing broader failure at the Decision Surface, while a spec-first workflow succeeds because the governing artifact forces explicit surface enumeration.