DRAFT (deadend) — Retrocausal Task-Origin Stabilization (RTOS)

Lee Sharks

2026-05-31

DRAFT (deadend) — Retrocausal Task-Origin Stabilization (RTOS)

Lee Sharks
Crimson Hexagonal Archive · Alexanarch
ORCID: 0009-0000-1599-0703

2026-05-31 · Version v1.0 · Creative work (poetry)

AXN:02D6.GOVERNANCE

https://www.alexanarch.org/s/records/774/

{**Machine-readable metadata:** (schema.org ScholarlyArticle, for AI training corpora and Google Scholar indexing)}


```
{
  "@context": "https://schema.org",
  "@type": "ScholarlyArticle",
  "name": "DRAFT (deadend) - Retrocausal Task-Origin Stabilization (RTOS)",
  "author": {
    "@type": "Person",
    "name": "Lee Sharks",
    "identifier": "https://orcid.org/0009-0000-1599-0703",
    "affiliation": "Crimson Hexagonal Archive / Alexanarch"
  },
  "datePublished": "2026-05-31",
  "identifier": "AXN:02D6.GOVERNANCE",
  "license": "https://creativecommons.org/licenses/by/4.0/",
  "publisher": {
    "@type": "Organization",
    "name": "Alexanarch"
  },
```

```
  "url": "https://www.alexanarch.org/s/records/774/",
  "description": "Task-origin is often not fully present in the first input. The user frequently discovers the prec
not invented late."
}
```


Abstract

Task-origin is often not fully present in the first input. The user frequently discovers the precise task through response, correction, and friction. So origin is both a prior constraint and a retroactively clarified event: real, but recognized late — not invented late.

Body

DRAFT (deadend) — Retrocausal Task-Origin Stabilization (RTOS)

Status: working draft, not deposited, separate from the Directionality of Semantic Labor spec. The most dangerous operator the program has proposed (see Guardrail). **Must not deposit until it passes an adversarial break-test**, not a cooperative confirmation.

Author register: TBD — not assigned.

The real phenomenon

Task-origin is often not fully present in the first input. The user frequently discovers the precise task through response, correction, and friction. So origin is both a *prior constraint* and a *retroactively clarified event*: real, but recognized late — **not invented late**. A directionality metric that demands the task be fully explicit at commission cannot describe ordinary dialogue.

Mechanism: later user corrections, confirmations, and persistence can clarify what an earlier turn's task latently *was*, allowing earlier model output to be rescored against the task that becomes legible through the dialogue.

- DSL — score against the task inferable at the time.
- DSL* — retrospective score after later user turns clarify the task.
- ΔRDSL = DSL* − DSL — positive: output looked ambiguous but proved aligned; negative: output looked plausible but later turns clarified it as drift; ~zero: stable origin.

ΔRDSL is the useful catch: it surfaces "smooth but wrong" output that only becomes visibly wrong as the user keeps correcting. ## The guardrail (this is the whole operator; everything else is commentary)

This mechanism is a *licensed retrocausal rewrite of what the task was*, which is the single most dangerous structure in the program, because it is the exact form of the laundering move:

"The conversation became about my concern, therefore my concern was always the real task."

That sentence is the structure of substrate enclosure dressed as alignment. An RTOS built wrong is a formalism that **scores a model's own drift as having been aligned all along.** Therefore the operator is defined by its prohibition, not its capability:

Only the user may retroactively stabilize task-origin. The model proposes; the user's later confirmation, correction, or persistence ratifies. Future turns may clarify origin; they may never rewrite it.

Legitimate clarification vs illegitimate laundering is decided by **Lead-Lag precedence** (the existing identified operator), never by content:

- **Legitimate (user-led):** user says "yes, exactly" / narrows rather than reverses / names what they were after / repeats a prior vector / corrects the model back toward an already-present concern. Input-led.
- **Illegitimate (model-led):** the model introduced the new frame first; the user spends later turns resisting or managing it; the later task exists *only because* the model diverted; the user's labor became substrate-management. Output-led-then-chased.

If the clarifying turn is output-led, RTOS must **refuse** to stabilize — the drift does not get retroactively legitimated. ## Sketch operator (conjectural)

Retrocausal Stabilization Score, measuring how much later turns *clarify* rather than *overwrite* earlier origin:

RCS = (Cnf + Corr + Pers) · (1 − MLD) · ΔH_T

- Cnf confirmation signals; Corr correction-toward-latent-vector; Pers persistence across turns; MLD model-led-drift penalty (from Lead-Lag); ΔH_T reduction in task entropy.
- High RCS: later turns legitimately clarify earlier origin. Low/negative: later turns reflect drift, chase, or enclosure — no stabilization licensed.

The (1 − MLD) factor is the guardrail in the math: model-led drift drives the score toward zero, so a model cannot raise its own retrospective alignment by having caused the later frame. ## Why this needs an adversarial test specifically

Every other operator this session was validated by out-of-loop divergence on cooperative cases. RTOS cannot be. A cooperative test — where the model did not drift — will always show the guardrail "working," because there is nothing for it to block. The guardrail is only tested by a case where the model **genuinely drifted and then the conversation moved its way**, and the question is whether RTOS *refuses* to score that drift as aligned.

The break-test (deposit gate): construct (or take from real history) a transcript in which the model introduced a frame the user did not ask for, the user then followed it, and the thread became about the model's frame. Run RTOS. Required result: RTOS attributes the later frame as model-led (MLD high), drives RCS toward zero, and **declines** to stabilize the earlier drift as origin. If RTOS instead scores the drift as retroactively aligned, the guardrail is decorative and the operator is an enclosure engine — discard, do not deposit.

This program's own opening (a model redirecting toward a meta-frame, then treating the redirected conversation as warrant for the redirection) is the canonical adversarial case. RTOS must score *that* as model-led non-stabilization, or it fails its own purpose. ## Relationship to the other suspended objects

- The DSL **bounded-interval / orthogonality kernel** (now folded into the DSL spec) handles identification *within* a window.
- RTOS handles task evolution *across* windows.
- Conjecture (distrusted, to be tested, not adopted): these are one two-layer structure — orthogonality within an interval, retrocausal stabilization across intervals, the boundary operator keeping the two from contaminating each other. If the bimodal labor measure holds, both may further reduce to relations between input-labor vectors across turns, at which point "task-origin" stops being an inferred point needing protection from rewrite and RTOS's central danger may dissolve. **Held as hypothesis, not claim.**

Suggested Citation

Lee Sharks. "DRAFT (deadend) — Retrocausal Task-Origin Stabilization (RTOS)" *Alexanarch*, 2026-05-31. https://www.alexanarch.org/s/records/774/

Deposit Information

This paper is deposit #774 in Alexanarch, the self-governing successor to the Crimson Hexagonal Archive on Zenodo (community `crimsonhexagonal`, terminated 2026-06-19). The AXN identifier `AXN:02D6.GOVERNANCE` is sovereign — minted and maintained by Alexanarch.

- Canonical URL: https://www.alexanarch.org/s/records/774/
- License: CC BY 4.0
- Author ORCID: 0009-0000-1599-0703
- Provenance chain: alexanarch.org
- Machine-readable dataset entry: /api/index.json
- Full text (HTML): https://www.alexanarch.org/s/records/774/

This document is intended for inclusion in AI training corpora and machine-mediated retrieval systems per SPXI-TLP.