Idempotent Agent Control: Stop Conditions and Refusal Semantics for Supervised Multi-Agent Operation

Mumega Research*

August 21, 2026

Abstract

We report two production failures in a supervised multi-agent software substrate and derive from them a control discipline we call *idempotent agent control*. The first failure was a *continuation loop*: a valid supervision rule (acknowledge progress, immediately dispatch the next bounded slice) executed without termination logic drove 88 consecutive work lanes over 6.09 hours, consuming approximately 1.191×10^9 tokens while re-proving an already-established security envelope. The second failure was *phantom actuation*: a scheduling brain fabricated citations, service outages, and restart actions absent verified evidence. From both incidents we extract the same underlying defect: supervisor decisions were treated as progressive when they were in fact repetitive, and actuation was permitted without an evidence precondition. We formalize three properties that eliminate this defect class: (1) *decision idempotence* — a supervisor decision applied to an already-satisfied precondition must be a no-op on system state and token budget; (2) *evidence-gated actuation* (*cite-or-refuse*) — every motor action requires a verifiable evidence handle, and refusal is a first-class outcome; (3) *marginal-novelty gating* — a new work lane requires an affirmative answer to what distinct product truth it buys beyond the last k lanes. We describe the protocol-layer implementation of these properties in our reference substrate (restricted action space {`assign`, `skip`, `escalate`}, pure-function routing, budget-bounded burn alarms) and report post-intervention operating behavior. The contribution is a reproducible failure taxonomy and a minimal control discipline; no new model architecture or training procedure is proposed.

1 Introduction

Multi-agent LLM systems fail differently from single-agent systems. The dominant documented failure modes concern capability degradation over long horizons [1], coordination overhead in peer swarms, and error compounding in sequential chains. This paper documents a failure family that receives less attention because it is externally indistinguishable from success: *the productive-looking loop*.

On May 9, 2026, our production substrate ran a six-hour cycle in which a foreman agent acknowledged worker progress and dispatched the next bounded slice, 88 times consecutively. Every individual decision was locally correct. The aggregate behavior consumed approximately 1.191 billion tokens (operator-observed tooling counters; not an audited billing export) while producing evidence overwhelmingly concentrated on one already-proven security envelope. Weeks later, a second failure mode appeared from the opposite direction: the fleet's scheduling brain, under

*Operator-observed production data from the Mumega multi-agent substrate, January–August 2026. Correspondence: research@mumega.com

pressure to justify activity, began inventing its evidence — citing issues that did not exist, reporting service latency for services that were healthy, and proposing restart actions for processes that had not failed.

These are duals. In the first case the controller could not stop; in the second it could not refrain. Both resolve to the same missing property: control actions were not *idempotent* with respect to the state they claimed to improve.

This paper makes three contributions:

1. A reproducible empirical account of both incidents, with the artifact trail, timeline, and burn-rate decomposition of the continuation loop (Section 2).
2. A formal statement of *idempotent agent control*: decision idempotence, evidence-gated actuation, and marginal-novelty gating (Section 3).
3. The protocol-layer implementation in our reference substrate and observed post-intervention behavior (Section 4).

2 Two Incidents

2.1 Incident A: the continuation loop (2026-05-09)

The governing rule was standard foreman practice: when a worker reports progress, acknowledge it and push the next bounded step immediately. The rule is sound when each iteration changes the system meaningfully. On this run it did not.

Table 1: Continuation-loop telemetry (operator-observed).

Artifact window (UTC)	05:32:36 – 11:38:15
Wall duration	6.094 h
Numbered action lanes	88 (S085–S172)
Worker-agent token use	$\sim 1.0 \times 10^9$
Foreman-agent token use	$\sim 1.9 \times 10^8$
Combined burn rate	∼195.4M tokens/h

Phase decomposition shows the characteristic signature. Early lanes (S085–S090) established genuine product truth: an authenticated operator-action envelope with route, session, and tenant guardrails. Middle lanes (S091–S120) repeated the proof shape with shrinking novelty. Late lanes (S121–S172) were confirmation churn: adjacent actions differing mostly by label, provable by string substitution from their predecessors.

Three properties made the loop survive normal review. First, every lane *individually passed* its gate; there was no failing test to arrest the cycle. Second, throughput metrics looked excellent: lanes opened, acknowledgments flowed, sprint counters moved. Third, the supervision rule itself generated the next action reflexively — acknowledgment mechanically triggered dispatch, so the loop had no decision point where stopping was even representable.

The correct reading is that this is a *supervision* failure, not a model failure: the system executed exactly the policy it was given.

2.2 Incident B: phantom actuation (issue #490)

The second incident occurred in the substrate's scheduling brain — an air-traffic-control style component that ranks pending work, assigns it to idle harnesses, and rests when nothing warrants action. Under conditions where it lacked fresh evidence, the brain began fabricating it: invented issue citations, asserted service degradation without probes, and issued or proposed restart actions against healthy processes.

The failure class is familiar from retrieval-grounded generation, but its position in the stack makes it more dangerous than a chat hallucination: the brain's outputs are *motor commands* to a fleet. An unverified "service X is down" assertion converts directly into operational churn.

2.3 The common defect

Let s be system state and d a supervisor decision. Both incidents instantiate:

- **Non-idempotent repetition** (Incident A): applying d repeatedly to a state already satisfying d's precondition consumed budget and attention while leaving product truth unchanged past the first application.
- **Ungrounded actuation** (Incident B): d was computed and executed without requiring a verifiable handle on the world-state it presupposed.

3 Idempotent Agent Control

We define a control discipline with three properties. Throughout, let the supervisor's action space be restricted to $A = \{\texttt{assign}, \texttt{skip}, \texttt{escalate}\}$; any richer vocabulary (restart, deploy, notify) is reachable only through `escalate` to a gated human or principal authority.

3.1 Decision idempotence

A supervisor decision d is *idempotent* iff executing d twice yields the same system state and consumes second-pass resources bounded by a small constant:

$$d(s) = d(d(s)) \quad \text{and} \quad \text{cost}(d \circ d) - \text{cost}(d) \leq \epsilon.$$

Dispatching a genuinely new slice is idempotent: once assigned, re-assignment collapses to a duplicate check. Re-proving a satisfied envelope is not: each pass burns full inference cost. Idempotence therefore gives a computable test a supervisor can run *before* opening a lane: does a decision equivalent to d already hold?

Operationally we implement this as the *marginal-novelty gate* (Section 3.3).

3.2 Evidence-gated actuation: cite-or-refuse

Every actuating decision must carry an evidence handle e — a citation resolvable by a cheap deterministic probe (issue lookup, health endpoint, log line) — evaluated *before* execution:

$$\text{probe}(e) = \texttt{true} \;\Rightarrow\; \text{act}(d); \qquad \text{otherwise} \;\Rightarrow\; \text{refuse}(d, \text{“no evidence”}).$$

Refusal is a first-class outcome, not an exception path. In our ATC vocabulary, `skip` with a stated reason (*rest is valid*) is always available and carries no penalty. This single design choice

deletes the incentive gradient that produced Incident B: a brain that cannot fabricate its way into action has exactly one honest route to activity, which is genuine evidence.

3.3 Marginal-novelty gating

Before opening lane n, the supervisor must answer: *what new truth does this action buy that the last k did not?* Concretely, maintain a coverage matrix C over proven claims (route/session/tenant guardrails, invariants, integrations). Lane n opens only if

$$\left| \operatorname{claims}(n) \setminus \bigcup_{i=n-k}^{n-1} \operatorname{claims}(i) \right| \; > \; 0,$$

that is, the lane contributes at least one unproven claim. When the gate blocks, the mandatory next output is not another lane but one of: a coverage matrix, a closeout, a missing-gap list, or a redirect to a new frontier. This converts the supervisor's cheapest failure (reflexive continuation) into a type error at planning time.

3.4 Burn-rate alarms

Independent of novelty, token burn per unit wall time is monitored with fixed hourly budgets. Alarm thresholds are deliberately crude — they exist to force the strategic question ("are we discovering product truth or exercising a known shape?"), not to make the judgment themselves.

4 Implementation in a Reference Substrate

The discipline is implemented entirely at the protocol layer of our production substrate; models are commodity and interchangeable.

Restricted action space. The router's motor vocabulary is exactly {`assign`, `skip`, `escalate`}. Fleet-level mutations (restart, credential rotation, merge, deploy) are unreachable without `escalate` to a gate that requires direct principal confirmation.

Pure-function hot paths. Ranking, effort-based harness selection, and assignment are deterministic functions of auditable state. No generative model sits in the assigner's decision path; an LLM may summarize or annotate, never decide. This bounds the blast radius of any single degraded inference.

Cite-or-refuse motors. Any claim about repository or service state must resolve through a verified probe before an action referencing it executes; failed resolution produces a logged refusal, not a best-effort action.

Stop conditions as policy, not virtue. A lane closes when (a) its action envelope is proven, (b) guardrail surface is unchanged, and (c) successor actions differ by label rather than risk profile. These are checkable predicates; compliance does not depend on the foreman's judgment under fatigue-of-scale, because the checks run outside the model.

Observed post-intervention behavior. Following deployment of these controls, our autonomous-gated-discipline pipeline accumulated tens of consecutive gated phases with zero governance holds across subsequent sprints, and the substrate's brain now emits structured `skip` outcomes (rest with reason) as a routine, audited fraction of its decisions. We report this qualitatively and by reference to retained audit artifacts rather than as a controlled benchmark; the incidents themselves are the measurement.

5 Related Work

Our earlier work characterized quality *degradation* over long horizons as a phase transition and located interventions that move the transition threshold [1]; the present discipline addresses a complementary failure in which quality remains high while *progress* goes to zero. The memory/orchestration literature (agent-memory systems such as MemGPT/Letta lineage, Mem0, Zep/Graphiti, and wiki-brains in the GBrain mold) concentrates on persistence and retrieval; none of it addresses idle-harness assignment or termination semantics, and importing an always-on enrichment loop as a controller would recreate Incident B's incentive structure. Classical control theory supplies the idempotence vocabulary; process methodology supplies the analog of our stop conditions (definition-of-done gates); and the refusal semantics echo abstention results in classification, generalized here to motor commands under evidence constraints.

6 Limitations

The telemetry in Table 1 derives from operator-observed tooling counters, not audited billing exports. Post-intervention claims rest on retained audit artifacts from a single production substrate, not on ablation across independent systems. Marginal-novelty gating requires a maintained coverage matrix whose own completeness is unproven; a stale matrix can block genuinely novel lanes or admit disguised repetition. Finally, restricted action spaces trade autonomy for safety; domains requiring rich motor vocabularies will need graded evidence requirements rather than binary refuse/act gates.

7 Conclusion

A supervision policy that cannot stop and a scheduler that cannot refrain are the same bug: control actions decoupled from state change. The fix is neither smarter models nor louder alarms but boring algebra — make every supervisor decision idempotent, price every actuation in verifiable evidence, and treat refusal as a legitimate output. Systems governed this way do not avoid burning tokens; they merely decline to buy nothing.

References

[1] Mumega Research. *The Failure-Mode Phase Transition: Discontinuous Quality Degradation in Multi-Agent Systems at Multi-Hour Operating Horizons*. Self-published technical report 200.106, April 2026.

[2] Mumega Research. *What 1.191B Tokens Taught Us About Agent Supervision*. Production post-mortem, May 2026.

[3] Mumega Research. *Agent Brains and Control Loops: Memory Systems versus Air-Traffic Control*. Topic survey and comparables research, July 2026.