LLM Workflow Release Ops

Phil Stevens
philipstevens.github.io

Ongoing reliability and release discipline for in-production LLM workflows: eval-gated changes, drift detection, safer rollouts, and fast rollback.

Best Fit

YOU HAVE

- A workflow already in production (or about to ship).
- Regular changes: prompts, model versions, docs/index, tool behavior, policies.
- Incidents that are expensive to debug or repeat unexpectedly.

YOU WANT

- Predictable releases with evidence-based ship/no-ship decisions.
- Regression tests that grow with incidents (no repeat failures).
- Monitoring tied to rollback triggers (not vanity dashboards).

Operating Model

CADENCE	WHAT HAPPENS	WHAT YOU GET
WEEKLY	• Review production signals (latency/cost/refusal/safety/tool errors) • Quality sampling on a small rotating set (human rubric) • Triage drift indicators and prioritize fixes	• Weekly stability memo (what changed, what's trending, what to fix) • PR-ready recommendations (gates, prompts, retrieval thresholds)
PER RELEASE	• Run eval gates on proposed changes (prompt/model/retrieval/docs/tools) • Analyze deltas vs baseline and must-pass set results • Define rollout plan and rollback triggers for that release	• Release gate report (pass/fail + deltas + risks) • Rollout checklist + monitoring window plan
MONTHLY	• Update test set from incidents and new product surface area • Review gate thresholds and ratchet where stable • Calibrate alerts (reduce noise; catch real regressions)	• Monthly drift report + test-set maintenance summary • Threshold ratcheting plan

What's Included

Release Gating

- Gate runs and analysis for every change that can shift behavior.
- Must-pass maintenance (new cases added when incidents occur).
- Approval-ready "ship / hold" recommendation with evidence.

Drift & Regression Control

- Signals for retrieval health, refusal drift, safety events, tool failures.
- Investigations that connect drift to root cause (prompt, index, tool, model).
- Small-to-medium fixes: prompts/config/routing/retrieval thresholds.

Operational Rigor

- **Version control discipline:** prompt/config hashes, model identifiers, retrieval/index versions.
- **Trace requirements:** enough to replay and audit outputs without guesswork.
- **Rollback readiness:** explicit triggers and tested procedures.

Minimum Prerequisites

PREREQUISITE	MINIMUM	REASON
Eval harness	Runnable gate suite with must-pass subset	Retainer is about preventing regressions over time.
Traceability	Version identifiers in trace/logs	Without this, incident response becomes guesswork.
Release path	Canary/shadow/ramp capability (even manual)	Allows safe rollout and rapid rollback.

If you don't have these yet, the first month often starts with a short Build & Harden sprint to establish them.

Success Criteria

- ✓ Regressions are caught by gates or monitoring before users report them.
- ✓ Releases are repeatable: same process, same evidence, clear approvers.
- ✓ Operational metrics stay within agreed bounds despite upstream change.
- ✓ Every incident produces a new regression test (no repeat failures).

Boundaries

- Building brand-new workflows from scratch (handled via Audit + Build & Harden).
- Large re-architecture or platform build work.
- Large labeling efforts or full-time QA programs.

Next Step

Review one workflow's current gates and monitoring, then define the first month's reliability plan. If you're pre-prod, start with an Audit or Build & Harden sprint.

Book an intro call:

calendly.com/philipstevens4/intro

Email: philipstevens4@gmail.com
Web: philipstevens.github.io

Scan to book a call