REDEVOPS · RUNTIME PILOT · V0.2.X

Turn one repeated operational workflow into a measurable learning system.

Without replacing the model, the framework or the systems of record. We freeze what you already run, wrap it in a governed decision loop, and measure whether the **same frozen model** decides better after learning from **verified outcomes**.

- Measured — published, held-out, reproducible
- Planned · hypothesis — under evaluation
- Architectural claim — mechanism



Alex Rybachek · Miami, FL · redevops@redevops.io

THE PRODUCTION PROBLEM

Every serious AI app rebuilds the same layer — badly, every time.

Push an agent past the demo and you find yourself hand-building the same undifferentiated plumbing around the model. It is not the reasoning that breaks in production; it is **everything the model does not own**.

WHAT EVERY TEAM REBUILDS AROUND THE MODEL

`context` `memory` `routing` `execution` `retries` `verification` `approvals` `governance` `replay` `observability`

WHAT ALMOST NO ONE ADDS — AND WHAT TURNS A WORKFLOW INTO A LEARNING SYSTEM

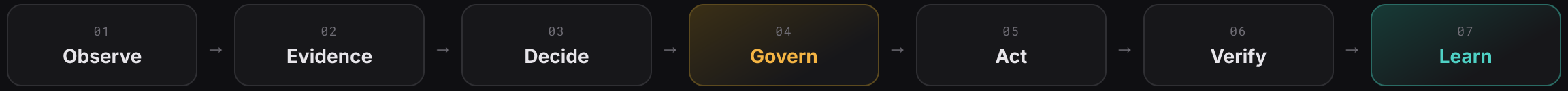
`decision` `verified outcome` `Experience`

A workflow that runs is not a workflow that learns. Unless the system records **what it decided**, **what actually executed** and **what happened next**, every consequential decision starts from scratch — and the organization's own verified outcomes never improve the next one.

WHAT THE RUNTIME DOES

A shared layer around your agents that closes the feedback loop.

Your agents, frameworks and systems of record stay exactly where they are. The Runtime provides the shared concerns every app rebuilds — plus the one frameworks omit: it turns each **verified outcome** into Experience that improves the **next decision**.



LOOP CLOSES `Learn → future decision` · verified Experience changes the next decision, with model weights unchanged

CROSS-CUTTING Governance · Evidence · Replay constrain the whole loop — learning may re-rank and re-prioritize, but never rewrite authority, approval or security policy

Governance is a boundary, not a reward function. Your framework authors the workflow; the Runtime operates the decision — durable state, governed execution, independent verification, and outcome-grounded learning applied selectively with fallback.

WHY IT'S MEASURABLE

Same workload. Same tools. Same model. Before and after.

We freeze the model, the toolset and the workload, then instrument one decision end to end. Every claim in this deck and in the pilot is a **delta you can reproduce** — not a vibe. The instrumentation splits into system health and decision quality.

SYSTEM & EXECUTION METRICS

BEFORE / AFTER

- **Task quality** — held-out, oracle-scored where an oracle exists
- **Tokens · latency · peak RAM** — the cost of a decision
- **Replay equivalence** — deterministic reconstruction of the run
- **Recovery** — completion after injected failure
- **Policy violations · duplicate side-effects** — safety of execution
- **Human minutes** — intervention actually required

DECISION-QUALITY METRICS

THE LEARNING-SYSTEM DELTA

- **Decision utility / regret** — vs the best available action
- **Learning benefit** — lift as verified Experience accumulates
- **Harmful-learning rate** — the subset made *worse*, tracked explicitly
- **Coverage / fallback** — when the system defers to frozen policy or a human
- **Evidence value** — did acquiring an observation change the action

Positive average lift is not the bar. A pilot can raise the mean while harming a material subset of decisions — so we pre-register the harmful-learning and coverage/fallback thresholds alongside the success criteria, and keep nulls in the record.

COMPONENT EVIDENCE

Before your pilot begins, three pieces are already measured.

These are published, held-out, reproducible results on independent data — not the pilot's hypotheses. Each isolates one Runtime concern, so a pilot builds on measured mechanism rather than a promise.

CONTEXT RUNTIME — MEASURED

Sufficient, not maximum, context

On ContractNLI, an **800-token** selected-and-closed evidence set beats stuffing the **4,003-token** full document — **~5×** fewer tokens, higher accuracy.

`0.511 → 0.700 → 0.733 NLI acc`

GOVERNED AUTONOMY — MEASURED

Answer when supported, abstain when not

Selective resolution reaches **0.817** autonomous accuracy at **67%** coverage — and defers the rest. The adversarial null is kept in the record.

`0.817 acc @ 67% coverage`

DECISION QUALITY — MEASURED

Same frozen model, better decisions

On **726** held-out chess positions with an independent oracle, the complete Runtime cut blunders from **0.408** to **0.197** — the model unchanged.

`-52% blunders · weights frozen`

Evidence discipline. Chess, Context and Governance are **measured**. Business-decision generalization (Agentic Apps) is **controlled** with the real-data gate open; sequential asset allocation (Quantify.club) is **in research**. We never label a hypothesis as a result.

COMPONENT PROOF · CONTEXT RUNTIME

Sufficient context beats maximum context. ~5× fewer tokens.

More context is not better context. Selecting a sufficient, **closed** evidence set — and verifying it is enough to resolve the task — beats filling the window. Measured on **ContractNLI**, held-out.

800 / 4,003

selected budget vs full-document tokens — **~5×** fewer moved into the window

81%

of cases where closure recall of the **union** ≥ either method alone

END-TO-END NLI ACCURACY — THE SAME TASK, THREE CONTEXT STRATEGIES



Contradiction detection rises further: **0.800 → 0.933** — the hardest label benefits most from a closed evidence set.

WHY IT MATTERS FOR A PILOT

MEASURED

Selection · closure · verification are distinct questions

Selection — which evidence. **Closure** — is that set sufficient to resolve the task. **Verification** — is the answer actually supported. This is not "better RAG"; it is three separable Runtime jobs, each measured.

For the Context Efficiency wedge this converts directly: **same task quality at a fraction of the tokens**, and smaller-model viability for the same workload.

COMPONENT PROOF · GOVERNED AUTONOMY

Answer when supported. Abstain when not.

A governed system knows when **not** to act. Coverage and fallback are first-class metrics, not failures — and the results that did **not** improve stay in the record.

0.817

autonomous accuracy on the **covered** slice — selective resolution

67%

coverage — the remaining **33%** defers to fallback or a human

THE ABSTENTION MECHANISM

Applicability decides autonomy

When the evidence supports a confident, verifiable answer, the Runtime acts autonomously. When applicability is insufficient, it **abstains** — falling back to the frozen policy, requesting evidence, or escalating. That guard is exactly what keeps average lift from hiding a harmful subset.

THE NULL WE KEEP

REPORTED, NOT HIDDEN

HippoRAG adversarial · no lift

On an adversarial multi-hop retrieval set, the graph-augmented arm showed **no improvement** over baseline. We report it as a null rather than dropping it.

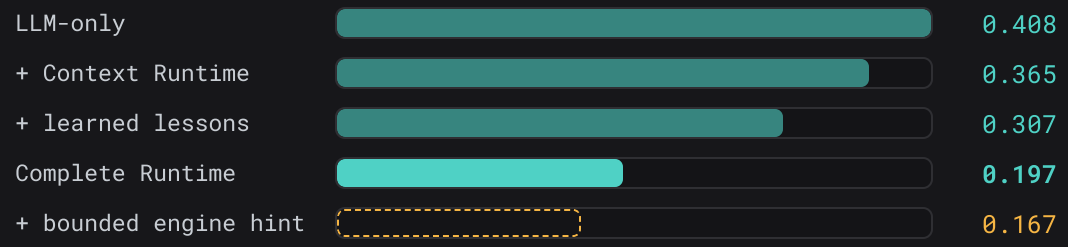
Why this belongs in a pilot deck: a research-grade Runtime earns trust by publishing where the mechanism does *not* help. Nulls and failures stay in the record — that is the proof philosophy the 30-day pilot inherits.

COMPONENT PROOF · DECISION QUALITY

Same frozen model. Different Runtime. ~52% fewer blunders.

The clean mechanism proof: a deterministic environment with an independent oracle. Frozen Qwen, **726** held-out positions, **Stockfish 17.1 depth-16** evaluator. **Two separate experiments — never merged.**

EXPERIMENT A · ARCHITECTURE ABLATION — MEASURED



Blunder rate; lower is better. The bounded engine hint arm is **external assistance** — shown dashed and labeled, never folded into the Runtime claim.

EXPERIMENT B · LEARNING CURVE · WEIGHTS FROZEN — MEASURED



Held-out blunder rate falls as verified Experience accumulates — **no fine-tuning, no weight change**. Separate from Experiment A.

Read it correctly. Holding the model constant, changing the Runtime changed decision quality. This is **not** "a 52% smarter LLM" — the model is unchanged; the system around it changed. Chess is the laboratory, not the product: it gives an independent oracle for measuring whether the Runtime itself improves decisions.

RUNS AROUND YOUR FRAMEWORK

Frameworks build workflows. Models reason. ReDevOps operates and improves decisions.

The same closure / verification / governance contracts run **inside** the framework you already use — not a fork of it. Eight measured, conformance-classified reference integrations.

EIGHT MEASURED INTEGRATIONS

MEASURED

NVIDIA NeMo agent	**PydanticAI** typed agents	**LlamaIndex** retriever	**CrewAI** multi-agent
Azure cloud	**Google** cloud	**AWS** cloud	**DigitalOcean** cloud

0.800→0.867

NeMo · SciFact · **0.412→0.913** GDPR closure

0.320→0.693

LlamaIndex · code closure · **6/6** unsupported answers abstained

THE BOTTOM LINE

Not a replacement — an operating layer

Keep your model, your agent framework and your systems of record. ReDevOps adds evidence selection, governed execution, verification and outcome-grounded learning **around** them.

Every integration is classified by conformance and reproducible with frozen environments and acceptance hashes at **redevops.io/integrations**.

ENTERPRISE

Hardening is additive. The decisions are identical.

The pilot runs on the open-core Runtime. Enterprise adds controls — it does not change answers. Two guarantees keep that honest.

PARITY

BEHAVIORALLY CONFORMANT

Open-core ≡ Enterprise, decision for decision

Every shared contract produces the same decision in both editions, and **Python ≡ Go** is conformance-tested. What you measure in the pilot on open-core is what you get in production on enterprise — no silent divergence, no "the paid version decides differently".

- same Context / Decision / Mission / Governance contracts
- same replay and verification semantics
- cross-runtime hash agreement on golden fixtures

HARDENED_DELTA

ADDITIVE CONTROLS

Controls layered on, decisions untouched

Enterprise adds the operational envelope a regulated deployment needs — around the same decision loop, never inside the learning objective.

- **Identity & SSO** — deny-by-default principals
- **ABAC + break-glass** — attribute-scoped authority
- **Tenant isolation** — per-tenant boundaries
- **Hash-chained access ledger** — tamper-evident audit

Governance stays outside learning. Learning may change ranking, expected utility, strategy, evidence preference, timing and abstention thresholds within allowed policy. It may never independently change identity, authority, permission, approval requirements, security policy or monetary limits.

PILOT WEDGES · RELIABILITY

Pick the wedge that matches your pain.

Four entry points, one Runtime. The first two convert directly from the measured component proofs; the next two (following slide) test the learning-system thesis on your data.

WEDGE 1 · CONTEXT EFFICIENCY

MEASURED MECHANISM

Same task quality, a fraction of the tokens

For high-volume workloads where context cost — tokens, latency, RAM — or smaller-model viability is the constraint. Built on the ContractNLI closure result.

MEASURE

- task quality
- tokens
- latency
- peak RAM
- smaller-model viability

WEDGE 2 · AGENT RELIABILITY

MEASURED MECHANISM

Missions that complete, don't double-fire, and recover

For workflows where invalid actions, duplicate side-effects or unrecoverable failures cost real money. Built on durable mission execution, replay and governed autonomy.

MEASURE

- mission completion
- invalid actions
- duplicate side-effects
- replay
- recovery
- human minutes

Where the workflow becomes a learning system.

These wedges test the thesis directly. The mechanism is validated in controlled worlds; the real-data gate opens with **your** historical decisions and outcomes.

WEDGE 3 · DECISION IMPROVEMENT

CONTROLLED

WAITING FOR REAL DATA

Can the frozen model decide better from verified outcomes?

Repeated consequential decisions: **collections, lead follow-up, fraud review, reconciliation, exception handling, renewal, prioritization** — including when HOLD is the right action.

MEASURE

- utility / regret
- HOLD / abstention quality
- human escalations
- outcome delta
- learning curve
- harmful-learning rate

Core question – **Can the same frozen model make better decisions after learning from verified operational outcomes?**

WEDGE 4 · DECISION-RELEVANT DISCOVERY

OPTIONAL · HYPOTHESIS

Value of information before acting

For workflows where action quality is limited by **missing information**. The system identifies which observation would change the decision — and whether it is worth acquiring.

MEASURE

- evidence requests that change the action
- value of information
- acquisition cost
- avoided intervention
- decision latency

Core question – **When it doesn't know enough, can it identify which observation is worth buying before acting?**

THE 30-DAY PILOT

Freeze → shadow → bounded activation → report.

Read-only and shadow first. We pre-register **success and failure criteria** before week one ends — and if the Runtime does not improve the workload, we say so and remove it.

WEEK 1

Freeze the decision problem

Define state, observable evidence, candidate actions incl. **HOLD**, governance constraints, execution semantics, verified outcome and a utility/regret proxy. Establish current and human baselines. **Pre-register success and failure criteria.**

WEEK 2

Shadow the Runtime

Integrate Context, Decision Intelligence, Mission, verification and Experience capture. **No learned behavior touches production.** Validate state-transition parity — a rejected or unexecuted action never becomes the assumed next state.

WEEK 3

Counterfactual / bounded activation

Compare, on the same workload: current workflow · frozen-model Runtime · Runtime + verified Experience · guarded learning with fallback where data is sufficient. Run adversarial and failure/recovery tests.

WEEK 4

Engineering + decision report

Baseline, architecture diff, decision-quality, context/evidence and execution/reliability results, learning curve **if statistically meaningful**, harmful-learning analysis, coverage/fallback — and a plain recommendation.

Shadow-first is the safety guarantee. Nothing in production changes until the counterfactual clears its pre-registered gates. The deliverable is a report you could publish — including the case where the answer is "remove it".

THE OFFER

Pick one repeated, consequential decision.

The one that is expensive when it is late, wrong, or escalated to a senior person. We begin **read-only**, replay your historical process, establish a baseline, and test whether Context, Decision Intelligence or verified Experience changes the result — before any production action is enabled.

WHAT WE NEED TO RECONSTRUCT IT — LEAKAGE-SAFE

OPENS THE REAL-DATA GATE

- **State as-of decision time** — the record before the action
- **Context** — the account / case / customer situation
- **Action taken** — the intervention or the decision to hold
- **Execution confirmation** — what actually went through
- **Subsequent outcome** — what happened next
- **Recurrence / history** — prior cases for the same entity
- **Timestamps** — sufficient for leakage-safe reconstruction
- **Permission / license** — for evaluation use

REPRODUCIBILITY

Every number is checkable

- redevops.io/integrations
- github.com/redevops-io
- dataset hashes
- reproduce commands

Same workload. Same model where possible. Frozen protocol. Before / after. **Nulls and failures stay in the record.**

Alex Rybachek · Miami, FL · redevops@redevops.io