Seven Engineering Lessons from Running AI in Production

07

FIELD NOTES

Engineering controls for reliable AI systems

Nirmal Kumar Jingar

Raptors Conference 2026 / 21 October / Online

Proceedings of the Raptors Conference, Vol. 1 (2026)

doi.org/10.68495/rc.2026.C054

The model is only one part of the production system

A plausible answer can still produce an unsafe or unreliable outcome.

Production reliability depends on the controls around the model: what enters, what leaves, and what happens next.

Model + **Controls** + **Outcome**

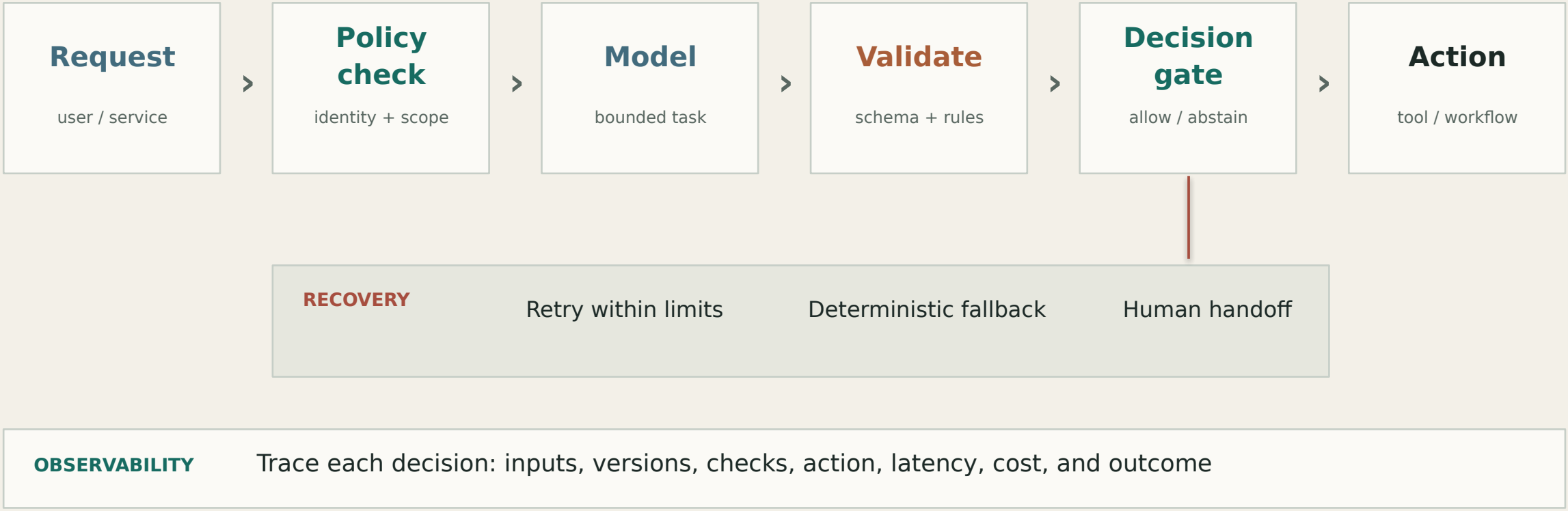
probabilistic | deterministic | operational

Dependability is an end-to-end property.

Demo success	Production success
one good response	bounded, recoverable behavior

A dependable AI feature has a control path

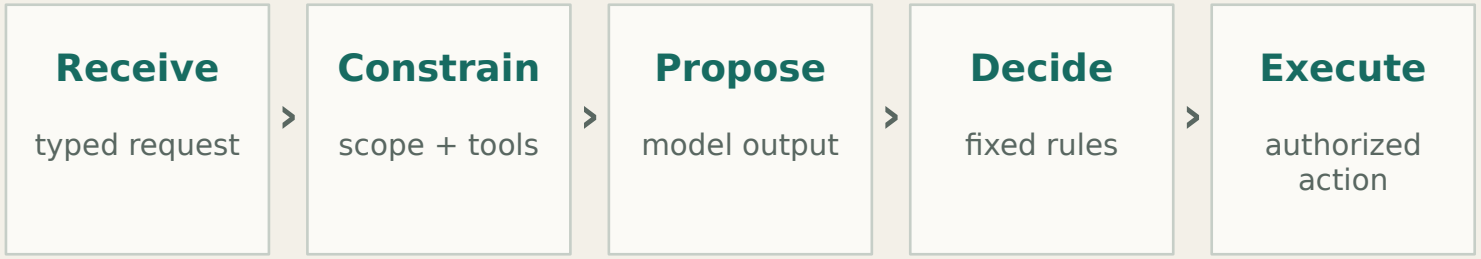
The model proposes. The system checks, decides, and records.



Put a deterministic control layer around AI

AI should operate inside a workflow with explicit limits.

CONTROL PATH



ENGINEERING RULE

Keep policy, state changes, and execution rules in code you can inspect and test.

WHAT TO MEASURE

Track which layer made each decision and why.

Validate at every boundary

Validate before model input and before any output affects a real decision.

ENGINEERING RULE

Treat generated text as untrusted input until it passes the checks required by its use.

VALIDATION LAYERS

Input	required fields, size, permissions
Shape	schema, types, allowed values
Meaning	domain and policy rules
Action	authorization and state checks

Reject, repair, or escalate invalid output. Never silently coerce a risky value.

Design the fallback before the failure

Every model call needs a defined timeout, retry budget, and safe next step.

ENGINEERING RULE

Retries handle transient faults. Fallbacks preserve the workflow when the model cannot help.

FAILURE RESPONSE

Retry	Fallback	Handoff
bounded and jittered	known safe behavior	person or queue

OPERATING SIGNAL

Measure fallback rate, timeout rate, recovery time, and user-visible impact.

Make decisions observable and replayable

A final answer alone cannot explain a production incident.

ENGINEERING RULE

Record enough structured context to reconstruct behavior without retaining unnecessary sensitive data.

DECISION RECORD

Request
correlation ID, safe input reference

Versions
model, prompt, tools, policy

Checks
validation results, rule outcomes

Decision
action, fallback, reviewer

Runtime
latency, cost, status

Result
downstream outcome

Replay with the recorded versions and controlled fixtures. Redact or tokenize sensitive fields.

Make governance part of runtime behavior

Permissions, budgets, and audit rules must apply to every invocation.

ENGINEERING RULE

A policy document cannot stop an over-scoped request at runtime.

VALIDATION LAYERS

Identity	who or what is calling
Scope	which data and tools
Limits	budgets, rate, action caps
Audit	policy decision + actor

Test denied access, exhausted budgets, and missing identity as normal production paths.

Deploy with a controlled blast radius

Measure behavior before increasing exposure.

CONTROL PATH



ENGINEERING RULE

Use staged traffic, explicit stop conditions, and a rollback path that operators can trigger.

WHAT TO MEASURE

Define thresholds for quality, safety, latency, cost, and fallback before rollout.

Separate reasoning from execution

Let the model propose intent. Let deterministic code decide whether to act.

ENGINEERING RULE

Keep side effects behind authorization, validation, idempotency, and explicit state transitions.

FAILURE RESPONSE

Model proposes	Executor checks	System acts
structured intent	policy + current state	idempotent operation

OPERATING SIGNAL

An uncertain proposal can be reviewed. An unchecked side effect is difficult to undo.

A practical path from prototype to production

Start with the decision the system can influence, then add controls around that boundary.

1 — **Map the decision** — What can the AI change? What is the cost of a wrong action?

2 — **Set boundaries** — Define inputs, allowed outputs, permissions, and abstention cases.

3 — **Build recovery** — Choose timeout, retry, fallback, and handoff behavior.

4 — **Instrument it** — Trace decisions and define alerts before launch.

5 — **Release in stages** — Shadow, canary, measure, and roll back against agreed thresholds.

Dependable AI is a system property

The central engineering move is to make probabilistic behavior governable.

Before action

What must be true before this output can change state?

On failure

What is the safe behavior when the model is unavailable or wrong?

After a decision

Can an operator explain and replay what happened?

Seven Engineering Lessons from Running AI in Production

Nirmal Kumar Jingar / Raptors Conference 2026 / doi.org/10.68495/rc.2026.C054

© 2026 Nirmal Kumar Jingar. Licensed under CC BY 4.0.