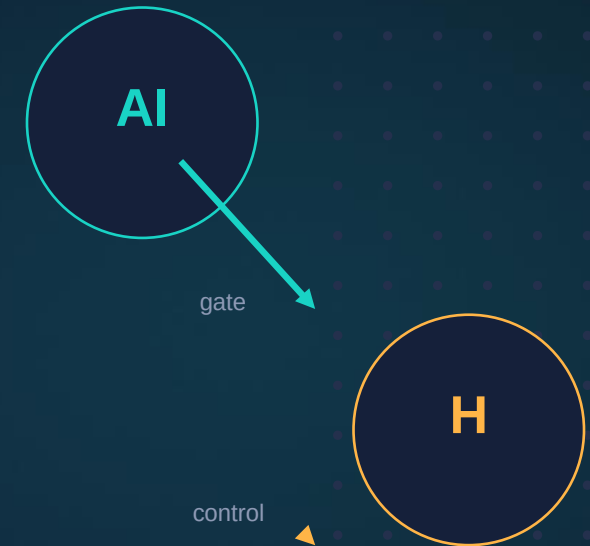


Feature-Flagged Autonomy

Confidence-Gated Human–AI Logistics Systems

How to progressively turn on agentic decision-making without turning off operational control.

Pavan Boppidi



“The views and opinions expressed in this presentation are my own and do not represent the views of my current and previous employers.”

Turning on an agent is easy. Governing it is hard.

Feature flags solve deployment exposure. They do not, by themselves, solve decision authority.

RUNTIME CONTROL

Software feature

You can enable a behavior for 1% of traffic, watch telemetry, and roll back without redeploying.

NEW BLAST RADIUS

Agentic behavior

The “feature” can now choose tools, sequence actions, spend money, reroute freight, and affect downstream systems.

AUTHORITY

Governance gap

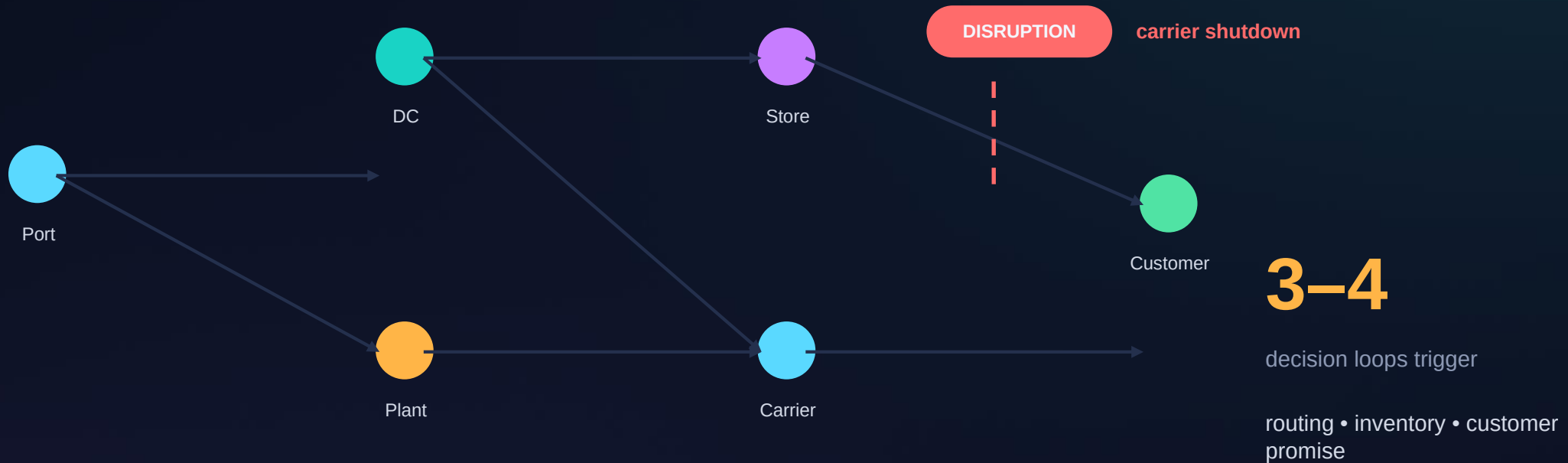
A boolean flag says “on/off.” It does not answer: who may act, on which decision, with what confidence, and under what conditions?

Core thesis

Progressive delivery becomes far more valuable when the rollout control plane and the decision-governance plane are connected.

Logistics is where autonomy meets real-world consequences.

Supply chains are dynamic networks; disruptions propagate faster than humans can re-plan by hand.



Recent industry research also shows logistics is moving from AI-assisted analysis toward AI that plans, decides and acts — while humans remain central to operations.

What a bad Tuesday looks like without a gate

One disruption, one confident agent, and no control plane. An illustration, not a real incident.

T+0 MIN

Carrier suspends a lane

A disruption hits. The routing agent proposes a reroute at 0.99 confidence, based on thin history.

T+2 MIN

No gate, so it executes

Confidence is treated as permission. The booking tool runs with no stakes or novelty check.

T+10 MIN

A second agent joins in

An unregistered agent calls the same carrier API. Two plans now compete for the same capacity.

T+30 MIN

Nobody can explain it

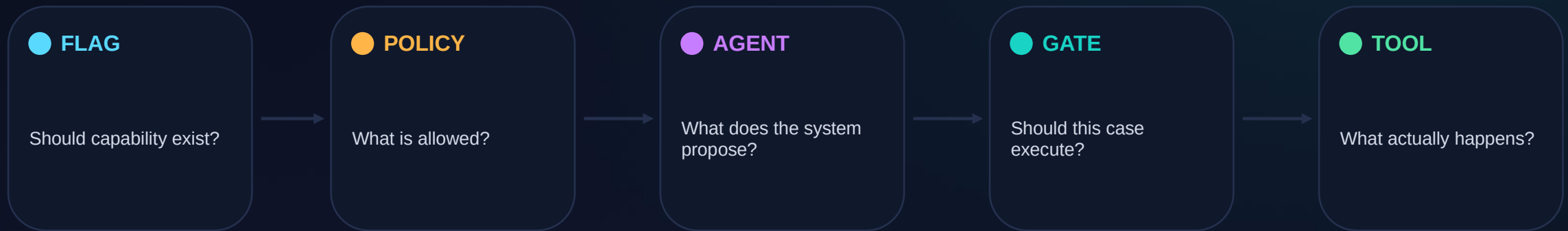
The flag is off, but in-flight runs continue. The logs show what happened, not why it was allowed.

Takeaway

Every missing control on this slide has an answer later in the talk.

From feature flag → autonomy policy → decision gate

Think of autonomy as a capability you progressively expose, not a binary architecture you switch on.



Two planes, one control loop

DEPLOYMENT

Control plane

Feature flags determine exposure, rollout and rollback.

EXECUTION

Decision plane

Confidence, stakes, novelty and policy determine authority per decision.

The architecture in one picture (Intelligent Logistics Orchestration Architecture)

1 INFRASTRUCTURE

Kubernetes / Knative • Kafka • Always-warm WebAssembly edge

2 AGENT INTELLIGENCE

MARL + CTDE • LLM reasoning • orchestrator • tool executor • memory

3 GOVERNANCE

Confidence gate • feature flags • MCP policy • agent identity • audit trail

4 LOGISTICS APPLICATIONS

Warehouse • routing • demand • cold chain • customs

CONTROL PLANE

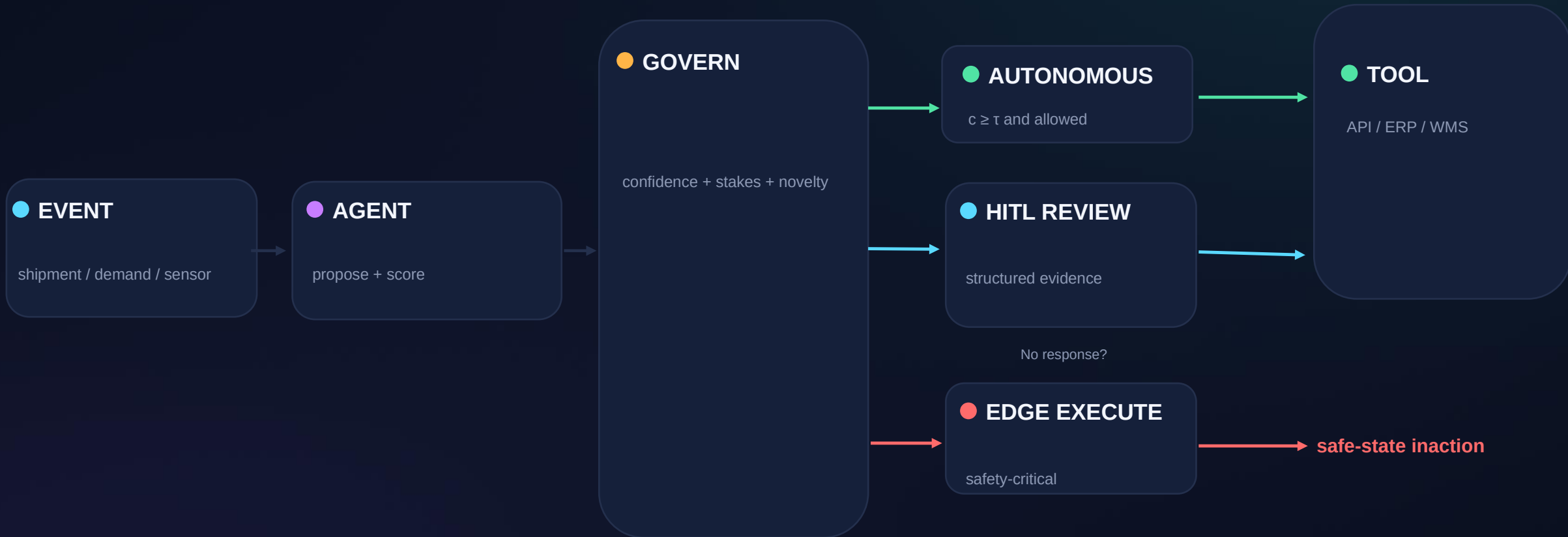
Feature flags

- + confidence
- + stakes
- + novelty
- + identity
- + rollback

**Every action
passes here.**

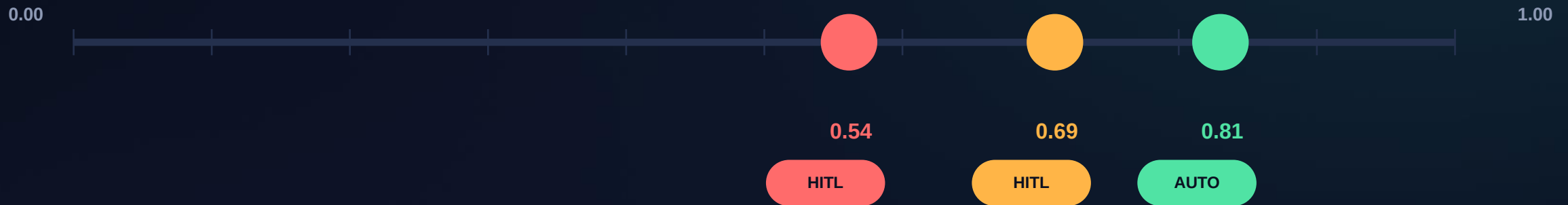
Every autonomous action passes through a gate

The architecture turns “AI confidence” into an explicit routing decision.



Confidence is a routing signal — not permission

An agent's score tells the governance layer how much autonomy to expose. It should never be the only condition.



SIGNAL

Confidence

Calibrated probability / decision-quality signal; monitor for drift.

CONSTRAINT

Stakes

High-consequence actions require stricter policy regardless of confidence.

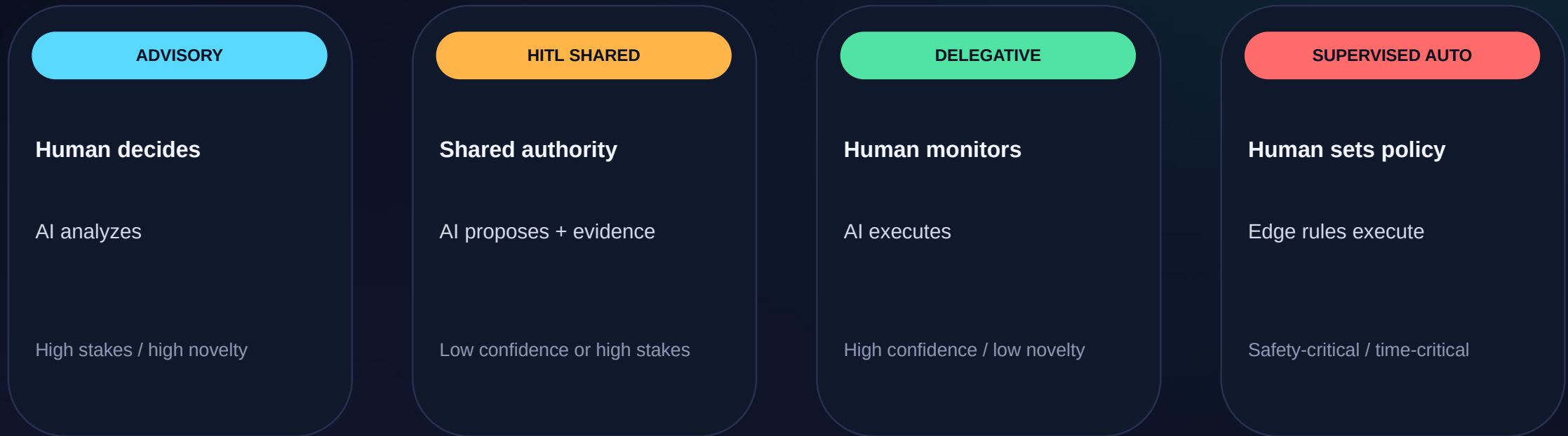
CONTEXT

Novelty

Sparse historical analogues should increase escalation pressure.

Four authority modes make governance operational

People do not have to “approve AI.” They need a clear contract for when they are the authority, co-pilot, supervisor or exception handler.



The design objective is not “maximum autonomy.” It is “the right authority for each decision.”

A useful autonomy flag is richer than true / false

Treat the flag as a versioned policy object, not a boolean.

autonomy_policy

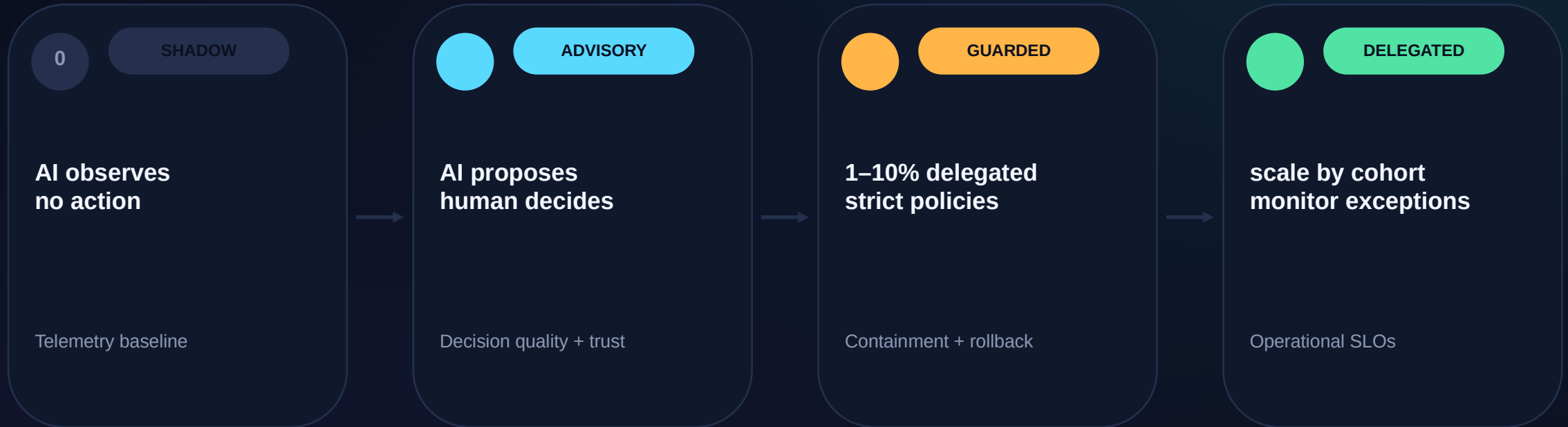
enabled	true
mode	delegative
rollout	10%
confidence_floor	0.78
allowed_tools	[route, booking]
max_cost	500 USD
novelty_limit	0.25
kill_switch	incident OR drift
expires_at	2026-11-15

What can be controlled

Capability	reroute / replenish / customs
Environment	prod / canary / shadow
Cohort	region / site / lane / partner
Authority	advisory / HITL / delegated
Safety	tool allow-list / spending cap
Lifecycle	expiry / owner / rollback

Four steps from “observe” to “delegate”

Autonomy should be rolled out like a production capability — with evidence at each stage.



Promotion criteria quality • containment SLO • drift • human escalation load • incident rate

A flag value is intent. Containment is something you prove.

A disabled flag can block new work while in-flight work continues — the autonomous equivalent of a partial shutdown.

ADMISSION CONTROL

flag = OFF

New runs blocked

Existing runs may still hold
queued tool calls, retries or leases.



CONTAINMENT

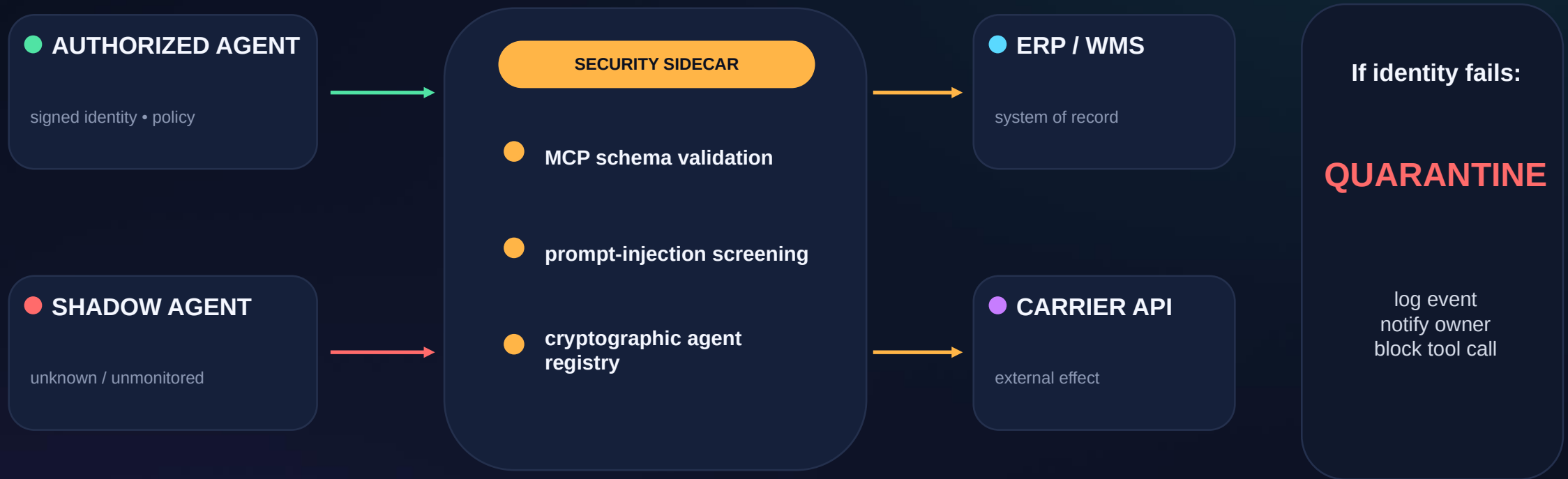
STOP

- cancel / drain active runs
- revoke tool leases
- invalidate cached policy
- block fallback paths
- verify tool-call rate $\rightarrow 0$

Containment SLO = time from disable $\rightarrow$ no externally consequential tool calls

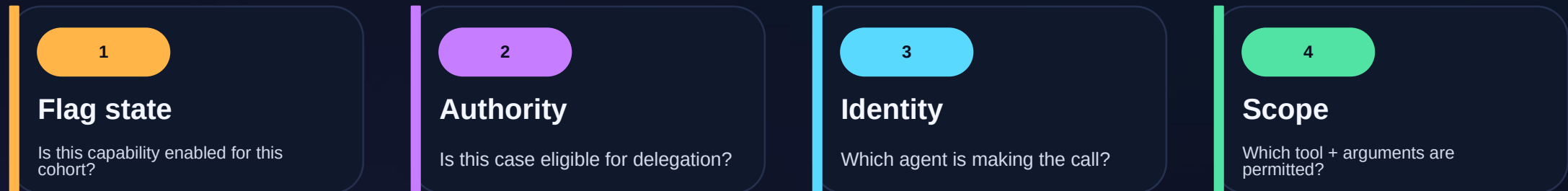
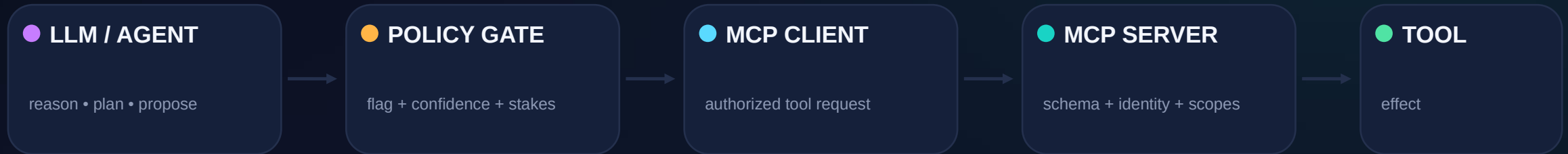
The second control plane you did not authorize

Governed autonomy fails if an unauthorized agent can still reach the same tools.



Tools are the real trust boundary

The model can reason; the tool boundary decides what the system is allowed to do.



This is where autonomy becomes enforceable.

Five assumptions that do not survive production

Each one sounds reasonable. Each one has an answer in this talk.

MYTH

High confidence means it is safe to act.

Put a human in every loop and you are safe.

A feature flag is a kill switch.

Our network perimeter covers every agent.

Guardrails belong in the prompt.

REALITY

→ **Confidence is a routing signal. Stakes and novelty decide the rest.**

→ **Escalation is scarce. Match the authority mode to the decision.**

→ **A flag value is intent. Containment is something you prove.**

→ **An unregistered agent can reach the same tools. Enforce identity at the tool.**

→ **The model is not the trust boundary. The tool is.**

If the system acts, the system should explain its authority trail

The audit event should capture not only what happened, but why the action was allowed.

event_id	9f2c...	novelty	0.12
agent_id	agent.route.v3	decision	reroute: carrier B
flag_version	routing-auto@12	authority	delegative
confidence	0.81	tool_scope	carrier.book
stakes	medium	outcome	success

AUDITABLE AUTHORITY

Feature-flag state + decision gate + tool identity + outcome = reconstructable autonomy.

Escalation is a scarce resource

A governance policy that overloads operators can fail even when the AI is “correct” on average.



when queues overload

Design goal: preserve decision quality across the shift, not just at 9:00 AM.

Self-check: where is your autonomy today?

Answer yes or no. An illustrative self-check, not a standard.

1

Can you list every agent that can call your operational tools?

2

Does each agent have its own signed identity?

3

Is autonomy a versioned policy with an owner and an expiry?

4

Have you tested the kill switch under load?

5

Do logs capture flag version, confidence and authority mode?

6

Do you track escalation queue depth and confidence drift?

COUNT YOUR YES ANSWERS

0–2

Start with the foundation
(days 0–30)

3–4

Run advisory mode and build
evidence (days 31–60)

5–6

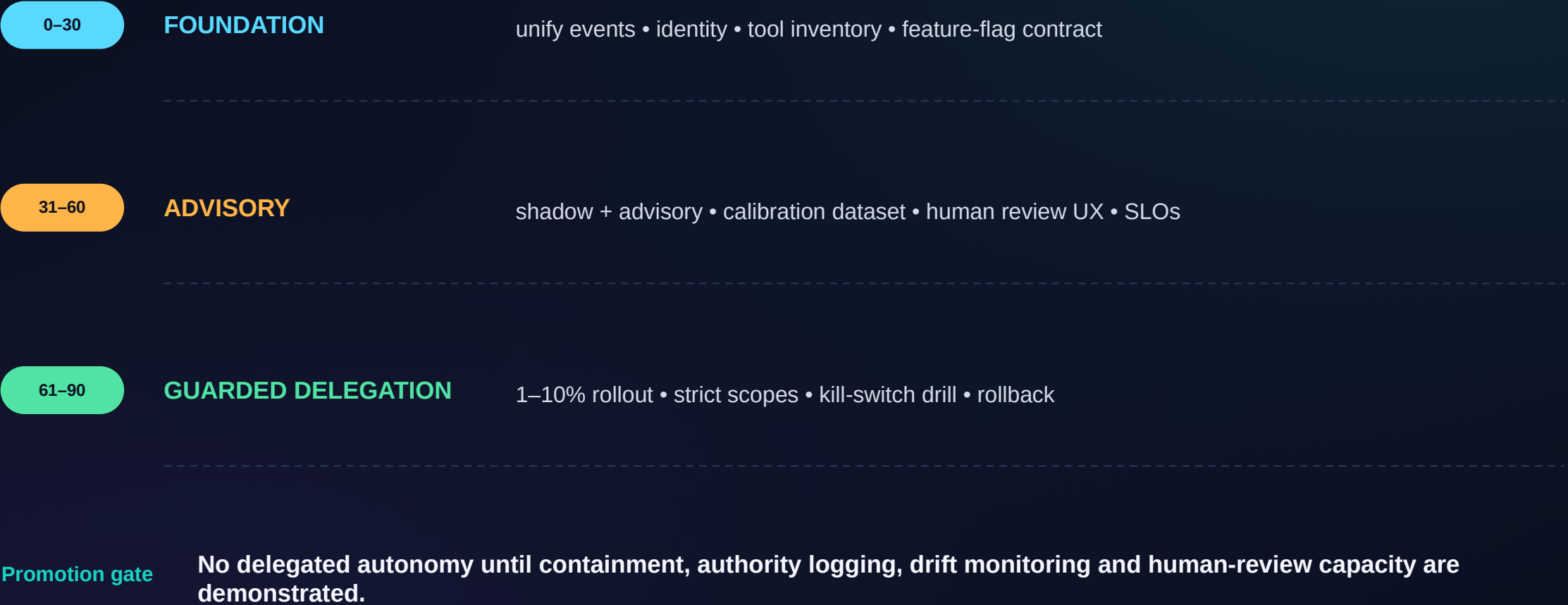
Plan guarded delegation
(days 61–90)

Use it

Your score is a starting point, not a grade.

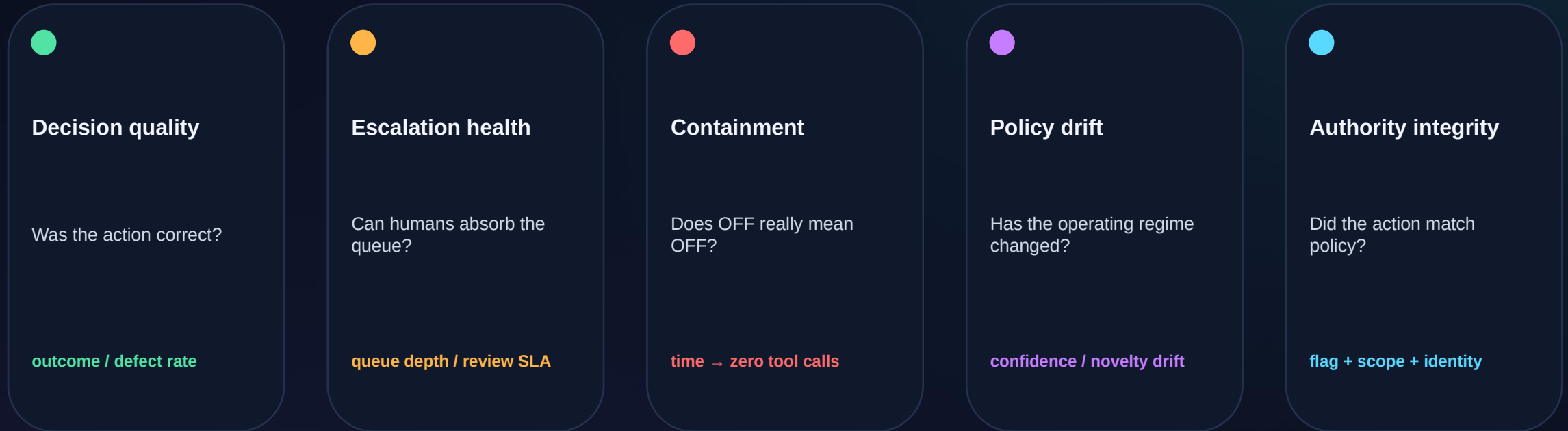
A practical 90-day path to governed autonomy

Start with evidence and containment. Earn authority incrementally.



Five SLOs for agentic autonomy

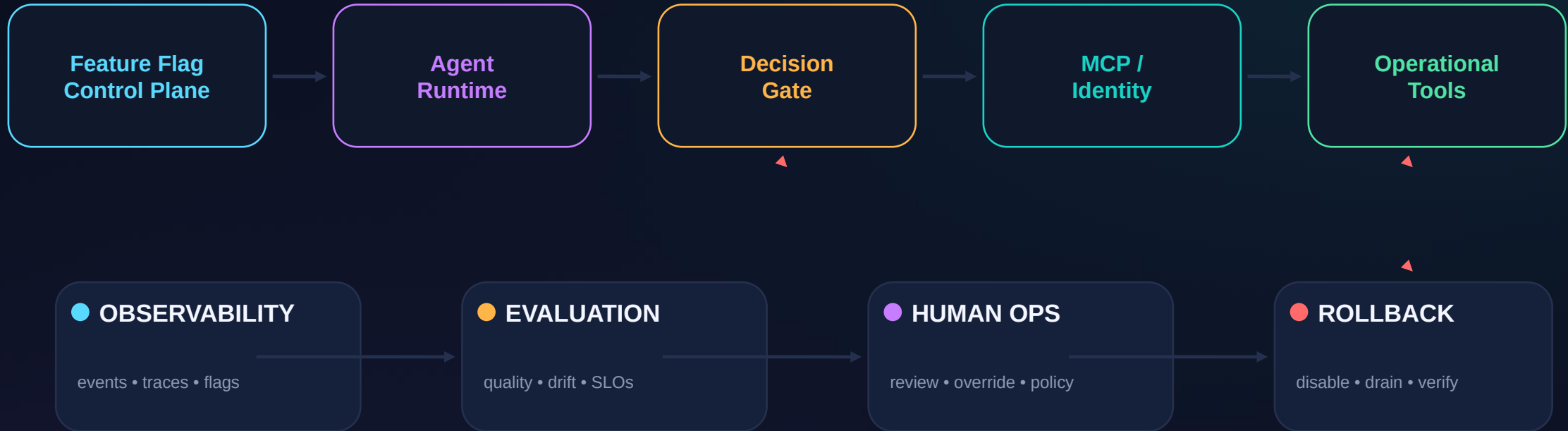
If you cannot measure the boundary between “allowed to act” and “actually acted,” you are not ready to scale it.



These are governance SLOs — not just model metrics.

A production-ready autonomy control plane

Connect feature delivery, decision governance, tool authorization and operational telemetry in one loop.



The loop is the product.

Autonomy should be a feature you can gate, observe and reverse.

The goal is not to keep humans in every loop. The goal is to keep authority explicit, evidence-driven and reversible.

- | | | | | | |
|----------|---|----------|--|----------|--|
| 1 | Gate the capability | 2 | Gate the decision | 3 | Prove the boundary |
| | Feature flags let you control exposure and rollout. | | Confidence + stakes + novelty decide authority per case. | | Identity, tools, logs and containment make autonomy enforceable. |

“The future of logistics is not human OR autonomous. It is governed collaboration.”

Thank you

Pavan Boppidi • Conf42 Feature Flags 2026