



Judgement Spine

Execution Governance Doctrine (Public)

Version: JS-DOCTRINE-PUBLIC-1.2 | Date: 2026-03-02

A governance doctrine designed for environments where execution occurs at machine speed.

This document is a public summary. Full canonical doctrine and world pack definitions are licensed.

Website: <https://judgementspine.com/>



1. The problem: when speed outruns authority

In AI-native systems, the interval between decision and consequence collapses.

A model output or agent recommendation can become a production change, a payment release, a data export, or a safety action in seconds.

Traditional governance assumes human pacing. It relies on meetings, checklists, and after-the-fact audits. That model fails when execution is automated.

Judgement Spine exists to make authority explicit at the moment before consequence.



2. Core doctrine (invariants)

These invariants define what a governed execution path must preserve:

- Judgement precedes execution.
- No consequential action proceeds without a named human authority.
- Automation acts within bounds.
- Agents may execute but never originate authority or own consequence.
- Ownership is explicit; if accountability is unclear, execution does not proceed.
- Escalation is structural: when risk expands or confidence declines, the system pauses, constrains, or transfers authority by design.
- Drift is assumed; governance must withstand speed and pressure at runtime.
- Autonomy contracts under uncertainty: systems move to safer modes, not faster ones.
- Evidence is produced by default before consequence.

If an invariant cannot be enforced, the system must degrade to a safer mode or halt the consequential action.



3. How judgement is applied

Judgement Spine is applied at Moments That Matter: the boundaries where intent becomes consequence.

A standard execution-time flow:

Capture intent	The system captures the action intent in a structured form.
Resolve MTM	Identify which Moment That Matters family applies and which doctrine checks are required.
Verify authority	Resolve named human owners; validate delegation and approvals.
Evaluate uncertainty	Assess confidence/novelty/risk thresholds; uncertainty triggers escalation or safe degrade.
Compute outcome	Return one of five outcomes: ALLOW, ALLOW WITH BOUNDS, ESCALATE, BLOCK, SAFE DEGRADE.
Bind enforcement	Write bounds into the execution mechanism at the last safe moment.
Mint evidence	Write an evidence pack before consequence, with integrity references.



4. Outcomes (why five modes matter)

Enterprise control requires more than allow/deny. Five outcomes preserve speed while preventing irreversible harm.

ALLOW	Proceed.
ALLOW WITH BOUNDS	Proceed, but only inside explicit constraints (caps, scope, TTL, rollback/abort).
ESCALATE	Pause commit and route to named authority; keep work moving as draft-only.
BLOCK	Prevent the irreversible commit point; hold with a bounded review packet.
SAFE DEGRADE	Proceed only via a safer substitute path (restrict, redact, throttle, isolate one host, etc.).



5. Evidence as a first-class output

In modern enterprises, governance must be defensible. Evidence cannot be reconstructed after the fact from ambiguous logs.

A defensible execution evidence pack is minted at execution time and designed to be portable: it can be inspected by an auditor, regulator, incident reviewer, or counterpart in a dispute without requiring access to the originating systems.

Minimum must-answer questions

- What action happened (proposed vs executed)?
- Who held authority at the moment of action?
- What bounds were enforced?
- What was blocked, degraded, or escalated - and why?
- What would have happened if confidence dropped?
- What doctrine version was applied?
- Can integrity be verified (tamper-evident)?

6. Worlds and suites

One doctrine can be operationalised in multiple trust boundaries via World Packs (suites).

World Packs do not change the doctrine. They operationalise it: defining MTM families, thresholds, evidence minimums, and enforcement bindings for a given domain.

Platform Runtime Control Suite	Agentic Execution Governance, Platform Control Governance
Human + Physical Safety Suite	Clinical Care Governance, Industrial Autonomy Governance
Financial & Commitment Authority Suite	Banking Core Governance, Banking Edge Governance, Capital Authority Governance, Sales Authority Governance
Security & Integrity Suite	Compliance Assurance Governance, Cyber Response Governance, Identity & Representation Governance
Operational Edge Governance Suite	Retail Core Governance, Retail Edge Governance
Reliance Governance Layer	Reliance Layer



7. Reliance governance layer

The reliance layer is a cross-cutting premium layer that governs:

model output -> reliance -> action

It covers human reliance (draft clause, clinical recommendation, risk score), automated reliance (agents executing from outputs), prompt provenance governance, and multi-agent reliance chains.

This is the missing governance layer for AI-native systems: the fault line is not only the model. It is who authorised reliance on the model at the moment it became consequential.

8. What this is not

- It is not a compliance certification. It helps you produce and verify evidence for your controls.
- It is not policy paperwork. It is designed to be enforced at runtime.
- It is not a generic workflow tool. It is a control plane for consequential boundaries.

Contact: founder@judgementspine.com

Disclaimer: This public doctrine summary is illustrative and not legal advice.