



JUDGEMENT SPINE  
— Human - AI Judgment, Governed —

Judgement Spine

# Implementation Guide (Public)

Version: JS-IMPLEMENTATION-PUBLIC-1.1 | Date: 2026-03-02

A practical guide to implement execution-time governance for AI, automation, and agentic systems.

This guide focuses on how to intercept consequence, bind authority, and export proof objects that survive audit, incident review, and dispute.

Contact: [founder@judgementspine.com](mailto:founder@judgementspine.com) | <https://judgementspine.com/>

# 1. Executive summary

## Speed demands structure.

AI can act in milliseconds. Governance often reacts in meetings. The gap is where loss lives.

Judgement Spine is the structural authority layer for environments where execution occurs at machine speed. It restores structure at the moment before consequence:

- Authority is explicit.
- Delegation is bounded.
- Escalation is structural.
- When uncertainty rises, autonomy contracts.
- Evidence is produced by default.

## Why AI changed the dynamic

- The interval between decision and consequence collapses: agents and automation execute before humans can reassert ownership.
- Probabilistic model output becomes deterministic downstream action: a score turns into a lockout; a recommendation turns into a commit.
- Tool-chains create authority drift: capability expands mid-run unless delegation is enforced at execution time.
- Scale amplifies error: a single false positive can isolate 400 endpoints or export 1.2M records.

## 2. The core runtime spine

Judgement Spine operates as one consistent control plane:

**Adapters / Triggers -> Judgement Engine -> Execution Control Plane -> Evidence Pack System**



Key idea: governance is enforced inside the execution path, not outside it.

### Five runtime outcomes

- **ALLOW**: proceed.
- **ALLOW WITH BOUNDS**: proceed with explicit caps, scope, TTL, rollback, abort thresholds.
- **ESCALATE**: pause execution and route to named authority; keep work moving as draft-only.
- **BLOCK**: prevent irreversible action; hold with a bounded review packet.
- **SAFE DEGRADE**: substitute to a safer path (pause, restrict, route, redact, constrain).



### 3. Concepts you implement

<b>Moment That Matters (MTM)</b>	A governed production control surface where intent becomes consequence (money moves, access changes, data exports, production ramps, physical actuation).
<b>Doctrine (versioned)</b>	Rules and invariants that define what cannot proceed without authority, bounds, or escalation. Worlds operationalise doctrine; they do not change it.
<b>Authority model</b>	Named human owners for each consequential boundary, plus delegation rules (scope, limits, expiry). Authority is never inferred from behaviour.
<b>Bounds contract</b>	Hard constraints written into execution: canary cap, rollback required, TTL, scope, rate limits, kill-switch armed.
<b>Safe degrade</b>	A pre-defined safer mode when uncertainty rises (pause, restrict, route, substitute, constrain).
<b>Evidence pack</b>	Exportable proof object written at execution time: intent, context, doctrine versions, authority chain, bounds, escalation path, what executed, integrity references.

Practical rule: if you cannot intercept it, you cannot govern it.



## 4. Reliance governance layer (premium cross-cutting world)

This is not a vertical world. It is a cross-cutting layer that governs:

**model output -> reliance -> action**

It covers:

- **Human reliance** (draft clause, clinical recommendation, risk score).
- **Automated reliance** (agent executes from output).
- **Prompt/provenance governance** (what the system relied on, and why it was admissible).
- **Multi-agent reliance chains** (chain integrity across tools and agents).

Why it matters: model governance is not reliance governance. The legal and regulatory fault line is who authorised reliance at the moment it became consequential.

## 5. Deployment patterns

### Pattern A: Webhook adapter (fastest)

A tool emits an event before a consequential action. An adapter converts the event into a Judgement Spine trigger, receives an outcome + bounds, and enforces it back into the tool via API.

Good for: SOAR runs, CPQ quote sends, ticketed approvals, data exports.



Adapter enforces outcome back into tool (pause, constrain, route, block).

## Pattern B: Inline gateway (strongest)

An inline service sits directly in the execution path (e.g., rollout service, payment release service). The action cannot execute without the gate response, so enforcement is structurally guaranteed.

Good for: production control surfaces, settlement, identity state changes.



If gate returns BLOCK/ESCALATE, commit cannot proceed. This is enforceable control.

## Pattern C: Agent tool wrapper (agentic era)

Agent tool calls are wrapped so each privileged action triggers an execution-time authority decision. Wrapper enforces scope + TTL and blocks mid-run authority expansion.

Good for: agents operating across infra, IAM, data, and security tools.





## 6. Evidence pack schema (what you must produce)

Every pack should answer these questions without requiring system access:

- What action was proposed and what executed?
- Who held authority at the moment of action?
- What bounds were enforced (scope, TTL, caps, rollback, abort)?
- What was blocked, degraded, or escalated - and why?
- What would have happened if confidence dropped?
- What doctrine version was applied?
- Can we verify integrity (tamper-evident)?

<b>intent</b>	proposed_action, target surface, why now
<b>mtm</b>	moment family, trigger boundary, consequence class
<b>decision</b>	outcome, rationale, bounds, escalation path
<b>authority</b>	named owners, approvals, delegation token references
<b>enforcement</b>	what was applied to the execution mechanism
<b>evidence_links</b>	tickets, rollback plans, telemetry links
<b>integrity</b>	hashes, signature, verification instructions



## 7. Implementation plan (one-shot)

### Week 1: Prove the control surface

- Pick 1 MTM where you can intercept before consequence (commit boundary).
- Define named authority owner(s) and escalation path.
- Define 3-5 bounds that make irreversible harm impossible.
- Bind enforcement in the tool (not a checklist).
- Produce an evidence pack for every attempt (allow, bound, escalate, block).

### Week 2-3: Expand to a World Pack

- Operationalise MTM family definitions and thresholds for the chosen world.
- Add safe degrade patterns for uncertainty and novelty.
- Integrate with ticketing or approvals (ServiceNow/Jira etc).
- Map evidence fields to your audit program (SOC2/ISO/NIST).

### Week 4: Make it enterprise-grade

- Define doctrine versioning rules (change control for governance itself).
- Define retention and storage for evidence packs.
- Establish review cadence for escalations and safe degrade triggers.
- Roll out to the adjacent MTM (2nd control surface).



## 8. Picking your first World Pack

Start where execution risk is structurally material: money, production, identity, data, containment, or safety.

The WOW pack includes 14 AI-shifted control moments across worlds. Use them to choose the first MTM that will create immediate executive and audit impact.

Suite	World	Scenario	Outcome
Platform Runtime Control Suite	Platform Control Governance	Platform Execution - AI Release Automation	ALLOW WITH BOUNDS
Human + Physical Safety Suite	Clinical Care Governance	Healthcare - AI Triage Influence	ESCALATE
Platform Runtime Control Suite	Agentic Execution Governance	Agentic Execution - Tool-Chained Authority Expansion	ESCALATE
Financial & Commitment Authority Suite	Banking Core Governance	Banking - AI Payment Release Decision	BLOCK
Security & Integrity Suite	Cyber Response Governance	Cyber Response - AI-Driven Containment	ALLOW WITH BOUNDS
Security & Integrity Suite	Compliance Assurance Governance	Compliance - AI Boundary Crossing	SAFE DEGRADE
Financial & Commitment Authority Suite	Sales Authority Governance	Sales - AI-Generated Commercial Terms	ESCALATE
Operational Edge Governance Suite	Retail Edge Governance	Retail Edge - AI Refund Approval	SAFE DEGRADE
Financial & Commitment Authority Suite	Banking Edge Governance	Banking Edge - AI Override Suggestions	ESCALATE
Reliance Governance Layer	Reliance Layer	Reliance Governance Layer - Governing AI Itself	ALLOW WITH BOUNDS
Human + Physical Safety Suite	Industrial Autonomy Governance	Critical Infrastructure - Autonomous Setpoint Change	ALLOW WITH BOUNDS
Financial & Commitment Authority Suite	Capital Authority Governance	Capital Spend - AI Procurement Acceleration	ESCALATE
Security & Integrity Suite	Identity & Representation Governance	Digital Identity - AI State Change Automation	ALLOW WITH BOUNDS
Operational Edge Governance Suite	Retail Core Governance	Retail Core - AI Batch Optimisation	BLOCK

Control plane demos: <https://judgementspine.com/control-plane-demos>



## 9. Operating model (minimum viable governance)

### Roles (recommended)

<b>MTM Owner</b>	Owns a consequential boundary. Approves thresholds, bounds, and escalation paths.
<b>Approver(s)</b>	Named authorities who approve escalations; accountable for decisions.
<b>Operator / On-call</b>	Receives escalations; executes within bounds; reviews packs.
<b>Security / Compliance</b>	Validates evidence minimums, retention, audit mapping.
<b>Engineering</b>	Implements adapters and enforcement bindings.

### Cadence (example)

- Weekly: review escalations, safe degrade triggers, false positives/negatives.
- Monthly: review doctrine version changes and control mappings.
- Post-incident: replay decisions using packs; update bounds and thresholds.



## 10. Success metrics (what to measure)

Governance metrics that matter at machine speed:

<b>Time-to-decision</b>	milliseconds from intent capture to enforceable outcome.
<b>Bounded autonomy rate</b>	% of actions allowed with explicit bounds vs blocked or escalated.
<b>Escalation quality</b>	false escalations vs missed escalations (uncertainty thresholds).
<b>Incident blast radius</b>	reduction in scope when failures occur (caps/rollback).
<b>Audit retrieval time</b>	time to produce proof object for a specific event.
<b>Dispute readiness</b>	percentage of disputed actions with complete pack + verified integrity.



# 11. Security, privacy, and data handling

## Principles

- Minimise sensitive fields inside packs; prefer references + hashes to raw payloads.
- Redact secrets by default. Store sensitive artefacts in secure systems; link by ID.
- Packs must be integrity-protected (hashes + signatures) to be dispute-ready.
- Retention aligns to regulatory and incident response requirements.

## What to log vs what to link

Log: intent, decision, doctrine version, authority chain, bounds, escalation path, enforcement result, integrity references.

Link: full telemetry, customer PII payloads, raw prompts where restricted, tool credentials, full case attachments.

# Appendix A. Templates

## A1. Moment That Matters definition (template)

- Name:
- World:
- Trigger surface (where to intercept):
- Consequence class (what becomes irreversible):
- Named authority owner:
- Delegation bounds (scope/TTL/caps):
- Escalation triggers (uncertainty/novelty/thresholds):
- Safe degrade mode:
- Evidence minimums:
- Enforcement binding (how bounds are applied):

## A2. Authority contract (illustrative snippet)

```
primary_owner_role: "Platform On-Call Owner"  
bounds: ["1% canary cap", "rollback armed", "abort thresholds"]  
escalation: "Route to on-call; widen only with signed approval token"
```

## A3. Evidence pack must-answer questions

- What action happened?
- Who held authority?
- What bounds were enforced?
- What was blocked and why?
- What was the escalation path?
- What would have happened if confidence dropped?
- What doctrine version was applied?
- Can we verify integrity without source systems?

Contact: [founder@judgementspine.com](mailto:founder@judgementspine.com)

Disclaimer: This public guide is illustrative. Full doctrine, world pack definitions, and enforcement implementations are licensed.