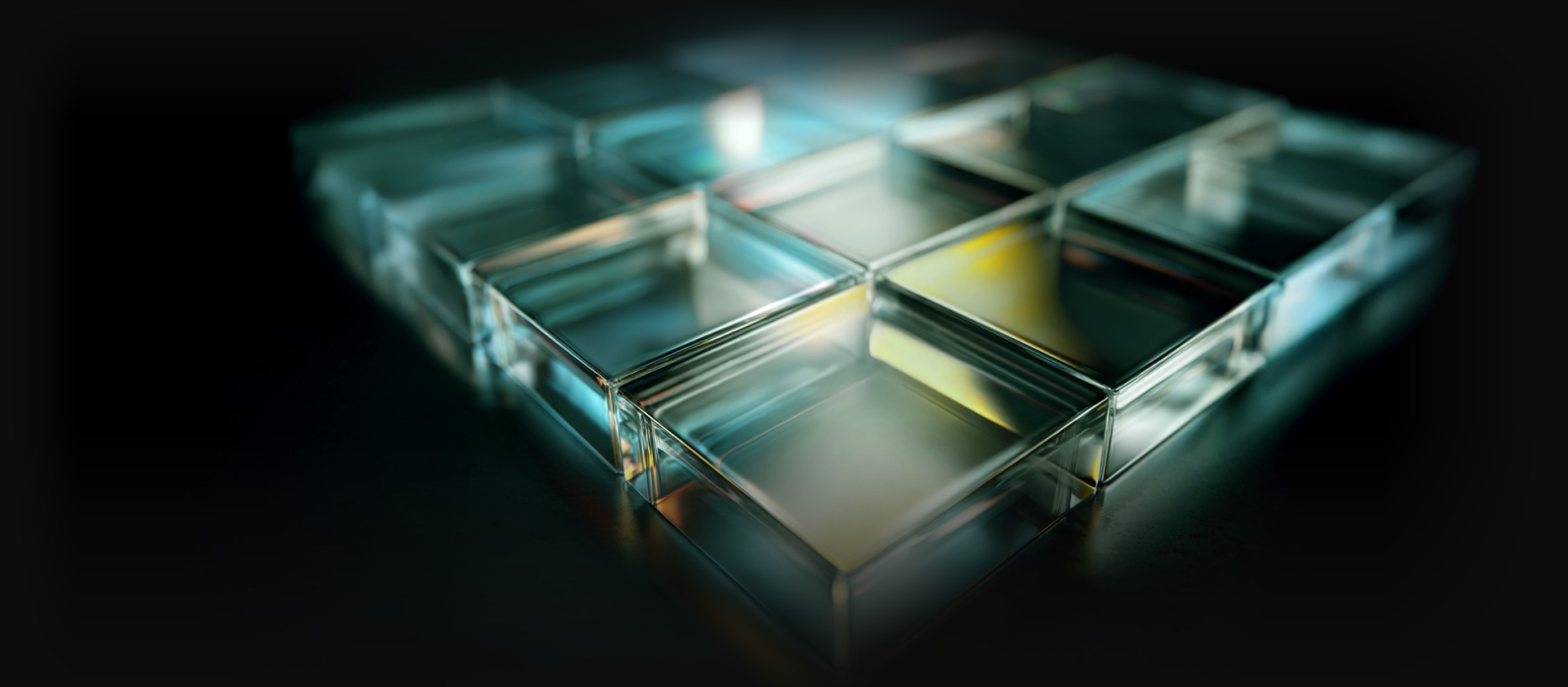
Prefactor

WHITEPAPER / 2026



FOR HEADS OF AI, PLATFORM LEADS, AND GRC REVIEWERS

The Agentic Contract.

How enterprises move AI agents from pilot to production with speed, control, and evidence.

> Enterprises are no longer blocked by whether agents can work in pilots. They are blocked by whether agents can be **owned, reviewed, monitored, and amended** in production. The agentic contract gives Heads of AI, platform teams, and GRC reviewers a shared operating model for getting agents to production quickly without losing control over risk, quality, or cost.

SUBJECT	READ TIME	COMPANION	VERSION
Framework + Lifecycle	~15 min	docs.prefactor.ai	1.0

SECTION 01

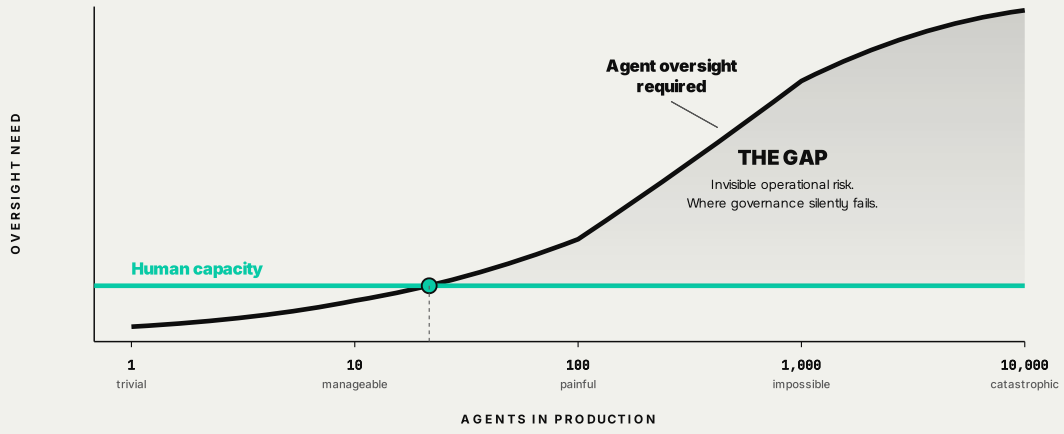
Most agent projects do not stall because the technology is hard.

They stall because the organisation around them is not set up to get them into production. This document is for the people closest to that problem.

Who this is for

Heads of AI and platform leads moving agents from pilot to production. The engineers and SREs carrying the operational weight. The GRC and security reviewers evaluating something their playbooks were not written for. The business owners whose work the agent is doing. **The framework has to work for that reality, or it does not work.**

Why the math is breaking



Three-week use case → nine-month delivery. Seven approvals: business case, InfoSec, legal, data governance, enterprise architecture, CISO sign-off, deployment readiness. **The cost is the delta — roughly eight months of opportunity per agent, multiplied across the queue.**

SURVIVAL THRESHOLD	The current model breaks past **a few dozen agents**. Every enterprise serious about agents crosses that line in the next 12 months.

The structural failures

AMBIGUOUS OWNERSHIP	Production breach at 2am. The SRE pages the engineering manager, who pages the sponsor, who is on holiday. **Nobody knows who should have been called first.**
SLOW APPROVAL CYCLES	**The slowness is rational.** GRC reviewers sign off carefully because the alternative is approving something they do not understand. The framework has to give them a faster path that is still rigorous, not a faster path that pretends the rigour is someone else's problem.
GRC AND ENGINEERING IN DIFFERENT ORBITS	Engineering ships to a spec engineering wrote. GRC reviews a spec GRC wrote. **Neither spec is the contract the business agreed to.** The agent passes both reviews and fails its first real-world incident.

SECTION 02

The tools the org already owns do not work for this.

Five categories of governance every enterprise already has. None of them were built for software that decides on its own. Each fails differently when the count goes up.

Most enterprises will reach for what they already own. The instinct is correct; the tools are wrong. **Agents break each category at a different point**, and the failure modes compound. Naming them is how the inevitability becomes legible.

> **This is not workflow automation with extra steps.** Traditional software scaled linearly with engineering teams. **Agents generate actions, decisions, and operational surface area autonomously.** One agent in production produces ten thousand decisions a week without a human ever writing them. Ten agents produce a hundred thousand. The governance models the org already owns assume each unit of work has a human author at the start of it. **Remove that assumption and every category below fails the same way: the volume crosses a threshold the tool was not designed for.**

01	**Ticketing & approval workflows** BUILT FOR HUMAN REQUESTERS	Each ticket assumes a person waiting. **Agents can generate hundreds of approval-required actions per hour.** The queue length itself becomes a denial of service.
02	**GRC processes** BUILT AROUND POINT-IN-TIME REVIEW	An agent reviewed at deployment is not the agent running in production a week later. The contract changes, the model changes, the data changes. **GRC has nothing to re-bind to.**
03	**IAM tooling** BUILT FOR STABLE IDENTITIES	**Ephemeral agents run for a single task and disappear.** Most never get an identity at all — but they read customer records, hit financial APIs, and OCR sensitive attachments. The sensitive data IAM was built to gate, accessed by software with no meaningful identity behind it.
04	**SaaS application governance** BUILT AROUND VENDOR SURFACES	Agents do not have a vendor. The behaviour is defined by the team that deployed them, the model they call, and the prompt they were given. **None of that lives in the SaaS catalogue.**
05	**Observability stacks** BUILT TO SURFACE ANOMALIES	An agent silently doing the wrong thing at 99.9% availability is not an anomaly. **It is normal traffic with a wrong answer.** A support agent closing tickets that should have been escalated. A claims agent paying out claims that should have been routed for review. Static evals don't catch this either — they run against test data, and production data is different data. The observability stack reports green while the business outcome moves the wrong direction.

None of these tools fail loudly

Each one keeps reporting normal. The ticket queue is processing. GRC sign-off is complete. The dashboards are green. **The signal has to come from somewhere these tools were not built to look. That somewhere is the contract.**

SECTION 03

The agentic contract is the operating model.

Not a document. Not a runtime policy file. The artefact that captures what an agent has agreed to do, who owns it, and what it is being measured against.

DEFINITION

Agentic contract

The live operating agreement between business, engineering, and governance teams that defines **what an agent may do, who owns it, how it is measured, and what happens when it breaches its limits.**

WHAT THIS IS NOT

Not policy-as-code. Policy engines enforce rules at the request layer; the agentic contract defines what the agent has agreed to do across its whole lifecycle and who owns the consequences. **Not an eval framework.** Evals measure model output; the contract measures the agent against thresholds the business signed. **Not a governance doc.** A Confluence page describes intent; the contract is the live artefact every breach, amendment, and audit binds back to.

THE POSITION, IN ONE PARAGRAPH

Most enterprises will get one or two agents to production through sheer effort. **The ones that scale will do it because they have an operating model and a runtime that runs it.** The agentic contract is the operating model. **Human work scales with risk, not with agent count.** Prefactor does the rigour invisibly on the agents that do not need attention. The team shows up where the contract says they have to.

What changes when you adopt the contract

	WITHOUT AN AGENTIC CONTRACT		WITH AN AGENTIC CONTRACT
01	Every review starts from scratch	→	**Reviews happen against agreed clauses**
02	Incidents require reconstruction	→	**Incidents produce an audit trail**
03	Owners are inferred during failure	→	**Owners are named before deployment**
04	Agent count increases human workload	→	**Human work scales with risk**

Where software becomes necessary. Past dozens of agents the framework needs a runtime. **That is where Prefactor fits.**

SECTION 04

The contract: what the agent has agreed to.

Authored where it matters. Inherited everywhere else. The customer chooses where authoring kicks in.

The contract is what the agent has agreed to: what it can do, on whose behalf, within what limits, owned by whom. It survives the developer leaving, the sponsor changing teams, and the GRC reviewer rotating off. Every agent has one. **What changes by risk profile is whether a human writes it.**

Contract anatomy	v1.0 / template
Identity & class	Agent name, business unit, build pattern. **What kind of agent this is.**
Scope	The bounded set of tasks the agent is permitted to attempt. **Stated as inclusions, not exclusions.** Anything not listed is out of scope by default. PREFACTOR BEST PRACTICE
Tools & data access	Allowed tools, MCP servers, data domains, and read/write scopes. **The most common source of risk.**
Decision authority	What the agent can do unilaterally. What requires human approval. What requires escalation. The boundary between these three is where production incidents live.
Ownership & accountability	**Owner.** Business stakeholders. Sign-off chain. Operator on call. Escalation path. The agent operates under an ownership structure on equal footing with its risk envelope.
Risk envelope	The risk profile assigned to the agent: data categories, action multipliers, thresholds. Drives per-run classification. *See risk profile, page 5.*
Quality bar	What "doing the job correctly" looks like, measurable. Technical health, evals, real-world feedback signals. Each criterion has an evaluation method.
Cost ceiling	Token spend, infrastructure cost, cost per accepted output. The agent has a budget and an action when the budget is breached.
Failure behaviour	What the agent does when it cannot comply. Refuse, escalate, defer to a human, fall back to a deterministic path. Defined in advance, exercised in test.
Versioning	Contract version, code version it binds to, last reviewed date. The contract is a living document.

IN THE FIELD / **FINANCIAL SERVICES**

A product team needed an agent to surface **product-specific settlement information** without crossing into financial advice. Standard guardrails couldn't tell the difference. The fix was per-agent contract clauses that **separated "contextual product information" from "regulated advice,"** with thresholds tied to scope. The contract did the work the guardrails could not.

PATTERN OBSERVED ACROSS THE DISCOVERY SET

SECTION 05

The lifecycle: when each owner has to show up.

Who approves what. Who gets paged. Who owns rollback. Six stages, five gates, three places a change can re-enter when something breaks.

Agent development is anomalous because the business owns the work the agent is doing. The lifecycle exists to make that joint development tractable. Each gate names who has to show up and what they have to bring with them. Without the gates, the org structure does not arrive on time, and agents stall in stages between meetings.

G1	Is this worth scoping?	OWNER Head of AI + Business sponsor
G2	Is the contract scoped enough to build against?	OWNER Engineering lead + Business sponsor
G3	Is the agent honouring its own contract in test?	OWNER Engineering lead + QA
G4	Is the contract operable under real-world risk?	OWNER GRC
G5	Is the live signal trustworthy?	OWNER Operator + SRE
G6	Is the agent still doing what we agreed?	OWNER Operator + Business sponsor

The contract is what decides

A properly authored contract makes most fallbacks unnecessary. **The pattern of where you fall back reveals whether the contract was right.** A fallback to Build means the implementation missed it; to Scoping means the contract was wrong from the start. **G6 is where backward movement is normal:** a parameter tweak stays in G6, an implementation fix re-enters Build, a re-scope re-enters Scoping. Mechanics at **docs.prefactor.ai**.

> IN THE FIELD / **INSURANCE**
>
> A claims-automation team's POC came back at **53% confidence against an 80% threshold**. The implementation was sound; **the contract was wrong. "On day one, your tech, your SME, and your business should sit together."**
>
> PATTERN OBSERVED ACROSS THE DISCOVERY SET

SECTION 06

Performance: how the contract is managed.

A composite of risk, quality, and cost. Three pillars, one score. The dashboard is the contract management surface.

The contract is what the agent has agreed to. Performance is how the agreement holds up in production, run after run. **Three pillars, each with its own mechanics, rolling up to one score the team and the business can both read.** Most agents do not need a human looking at them every day. They need Prefactor watching, scoring, and surfacing the runs that breach. **The team's attention is the scarce resource.**

PILLAR 01 / RISK

Risk.

A property of the run, not the agent. Calculated per agent-instance from the data the agent touches and the actions it performs.

- Data category weights (PII, financial, health)
- Action type multipliers (read, write, transact)
- Thresholds: Low, Medium, High, Critical
- Profile design as the lever — single-user vs multi-user, customer-facing weighting

PILLAR 02 / QUALITY

Quality.

Whether the agent is doing the job correctly, measurable. Starts technical. Extends into the signals that actually matter to the business.

- Technical health: latency, errors, availability
- Evals: model output graded against contract clauses
- Real-world feedback: ServiceNow, support sentiment, downstream metrics
- SME review where the stakes warrant it

PILLAR 03 / COST

Cost.

What the agent costs to operate, against what it is producing. The pillar that quietly determines whether the agent is worth running at all.

- Token spend per run and per agent
- Infrastructure cost (compute, storage, network)
- Cost per accepted output
- Budget ceilings with defined response on breach

THE ROLL-UP

Three pillars, one score, surfaced in the dashboard that doubles as the contract management surface. **Risk says how much attention the run deserves. Quality says whether the agent is doing the job. Cost says whether the agent is worth running.** The team intervenes when any pillar breaches its agreed threshold. Prefactor handles the rest.

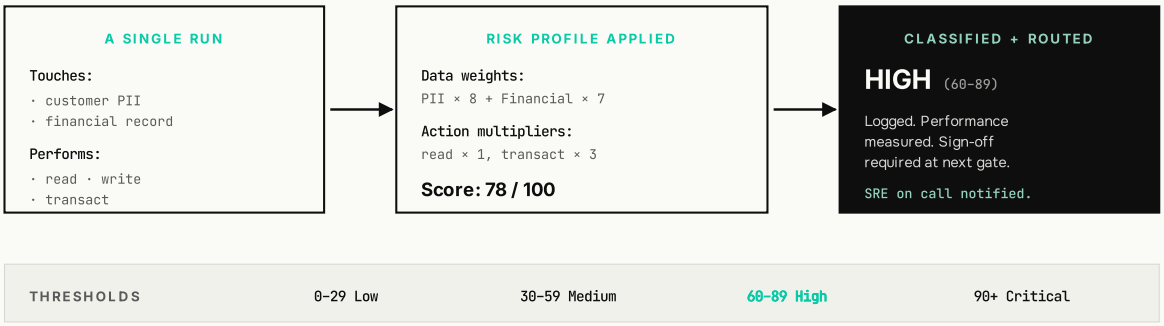
Risk profiles, cost tracking, and full performance mechanics at **docs.prefactor.ai**.

SECTION 07

Risk in operation: how a run gets classified.

Risk is a property of the run, not the agent. Each instance is scored from the data it touches and the actions it takes. The score determines the response.

How a run gets classified



What changes when a threshold breaches

WITHOUT PREFACTOR

Production threshold breach at 2am.

1. Datadog alert fires. Goes to a generic on-call channel.
2. SRE wakes up. Doesn't know which team owns the agent. **Pages engineering manager.**
3. Engineering manager doesn't know the contract. **Pages business sponsor.**
4. Business sponsor is on holiday. Agent runs degraded for nine hours.
5. Postmortem next week. **Nobody knows whether the contract changed or the agent did.**

WITH PREFACTOR

Production threshold breach at 2am.

1. Run classified Critical. Auto-rollback fires per contract failure clause.
2. Operator on call paged by name. Contract version on screen.
3. Diff against last known-good version surfaces the change.
4. Operator approves rollback or escalates per contract escalation path.
5. **Audit trail is the artefact, not a reconstruction.** Postmortem is a meeting, not an investigation.

SECTION 08

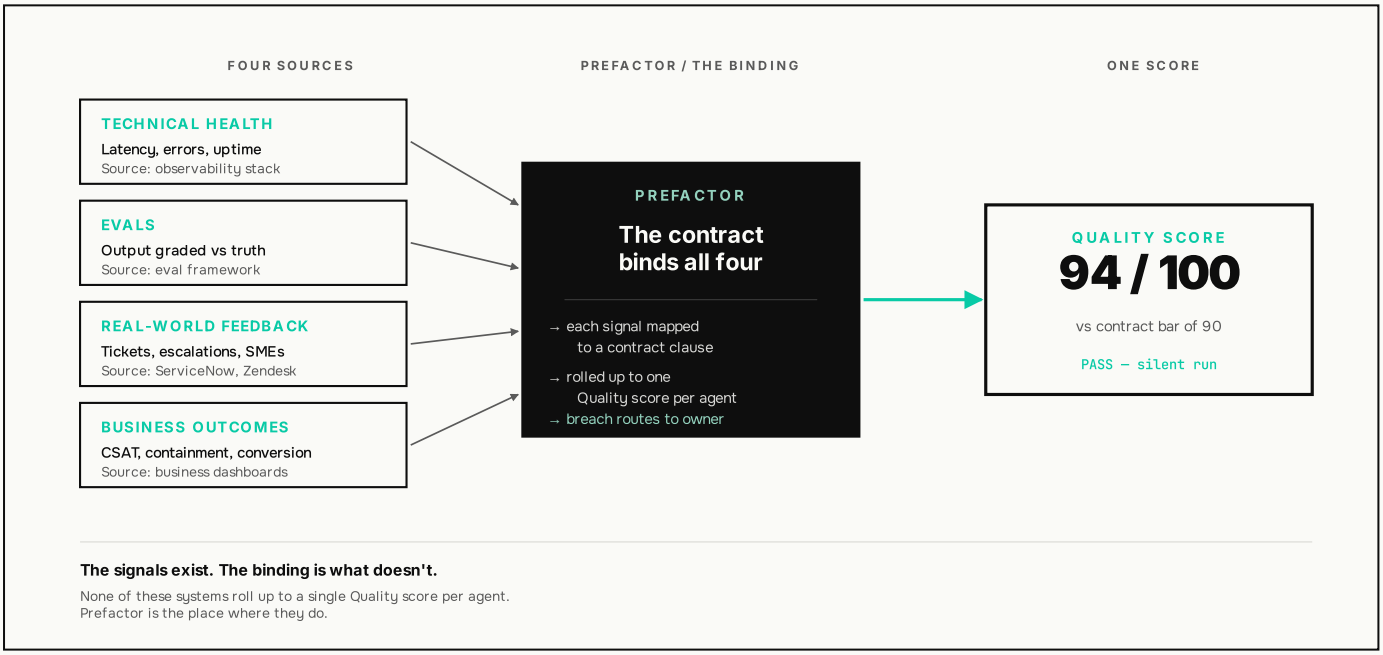
Quality is what "doing the job correctly" means.

Starts technical. Grows into the signals the business actually cares about. The only pillar that requires the agent's owner to define what good looks like.

Risk and cost are calculable from data the agent generates. **Quality is calculable only against a definition the business has supplied.** Without that definition, every quality measurement is an opinion. With it, every measurement becomes a contract clause the agent can be held to.

Defining quality is easy. Measuring it is the hard part.

"Accuracy above 95%." "Containment above 80%." "First-contact resolution above 60%." Any business owner can write a quality definition. **The hard part is that those definitions cannot be measured from the agent alone.** Quality lives across four signals owned by four different teams, in four different systems, that don't know each other exist.



This is the parallel to risk. **Risk on Prefactor is a scoring path: data weights × action multipliers, classified by threshold. Quality on Prefactor is a binding path: four signals × contract clauses, rolled up into one score per agent.** Both produce a single number. Both route the breach to the named owner. The difference: risk reads from one source (the agent's own spans). Quality has to pull from four.

IN THE FIELD / **MINING AND RESOURCES**

A data engineering team tied production-readiness to **SME-rated accuracy at 90%**. **"We use above 90% as of now. So if the score is above 90%, then only we think it's a good agent, otherwise not."** One signal, bound to one clause, with one number. **The other three signals — technical health, real-world feedback, business outcomes — were unbound.** The team had the definition of "good" but no way to roll up all four into a single Quality score. That binding is the Prefactor advantage.

PATTERN OBSERVED ACROSS THE DISCOVERY SET

SECTION 09

A claims processor, end to end.

One agent. One contract. One threshold breach. The framework, applied.

SECTION 10

Where Prefactor fits.

Prefactor sits inside the runtime path of agents and MCP-connected systems. It captures spans, classifies actions, enforces policy thresholds, routes approvals, and maintains audit evidence automatically.

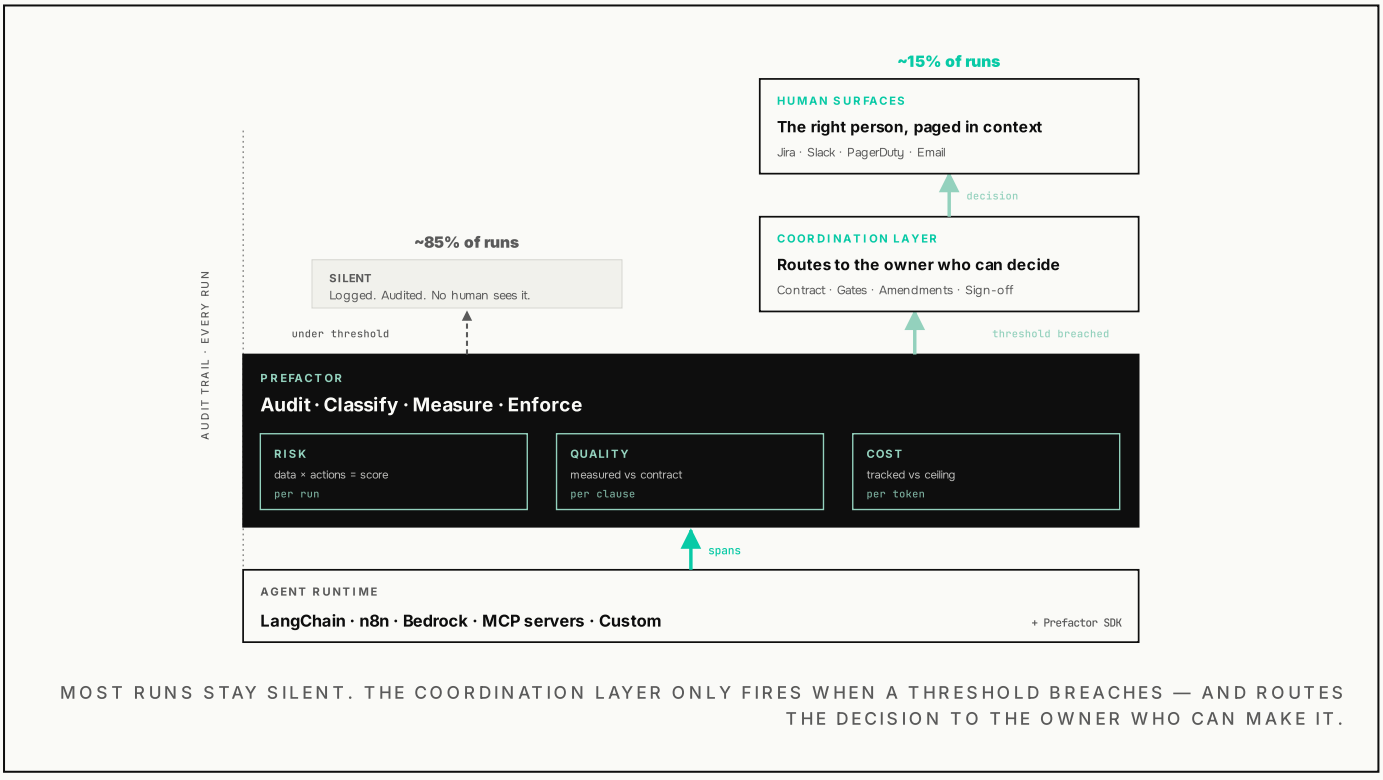
By the time agent count or risk concentration crosses what a single human can hold, the framework needs a runtime to enforce it. **That runtime is Prefactor.** The SDK instruments the agent at the call site. Every span — model invocation, tool call, data access — flows into the Prefactor core, which classifies, measures, tracks, and enforces the failure clause automatically. **When a threshold breaches, Prefactor pauses the run, pages the named owner, and surfaces the diff against last known good.** When nothing breaches, Prefactor logs and stays out of the way.

The target operating envelope

What a contracted production agent should look like, by design. These are operational targets, not deployed metrics.

~85% of runs stay silent. Logged, audited, no human sees them.	**< 4 min** from threshold breach to the named owner paged in context.
~90 sec to roll back to last known good. Auto-fired by the contract.	**0** incident reconstructions. The audit trail is the artefact.

The architecture



MOST RUNS STAY SILENT. THE COORDINATION LAYER ONLY FIRES WHEN A THRESHOLD BREACHES — AND ROUTES THE DECISION TO THE OWNER WHO CAN MAKE IT.

REFERENCE

Glossary.

The terms used in this document, briefly defined. Full definitions and platform mechanics live at docs.prefactor.ai.

Agent

A unit of work that uses an AI model to perform a task on behalf of a person, team, or system. Has a contract, a lifecycle, and a performance profile. docs.prefactor.ai/platform/concepts/agent

Agentic contract

The living artefact that captures what an agent has agreed to do, owned by, against which risk envelope, quality bar, and cost ceiling. The source of truth across every layer of governance.

Span

A single observable event within an agent run: a tool call, a model invocation, a data access. Spans are how risk and quality are measured. docs.prefactor.ai/platform/concepts/span

Amendment routing

Where a change re-enters the lifecycle when an agent breaches in production. Three options sized by the change: a parameter tweak stays in G6, an implementation fix re-enters Build, a re-scope re-enters Scoping. The size of the change determines the route.

Performance score

A unified composite of risk, quality, and cost, surfaced per agent in the dashboard that doubles as the contract management surface.

Agent instance (run)

A single execution of an agent. The unit at which risk is classified and performance is measured.
docs.prefactor.ai/platform/concepts/instance

Risk profile

A configuration assigned to an agent that classifies the data risk of each run. Built from data category weights, action multipliers, and thresholds. docs.prefactor.ai/platform/concepts/risk-profile

Lifecycle

The six stages and five gates an agent passes through from discovery to production and into continuous monitoring. Defines when each owner has to show up.

Gate fallback

Backward movement at any lifecycle gate. The pattern of where a fallback goes reveals whether the contract was right: a fallback to Build means the implementation missed the contract; a fallback to Scoping means the contract was wrong from the start.

Workspace defaults

The inherited contract and risk profile that low-risk agents operate under without explicit per-agent authoring. Set once at the workspace level, applied automatically.

CLOSING

Speed with rigour.

The agentic contract is the operating model. **Prefactor is the runtime control plane that runs it.**

BOOK YOUR READINESS REVIEW

Scan to book a demo.

A working session on contract authoring, risk profile design, threshold definition, and breach response.

prefactor.ai

Prefactor

The agentic operational layer giving enterprises **speed with rigour** on AI agents from POC to production.

WEB	DOCS	CONTACT
prefactor.ai	docs.prefactor.ai	hello@prefactor.ai