SIGNAL FABRIC

Turning live data into agent-ready inputs.

Language models are increasingly asked to act on real-time operational data. Raw firehoses are the wrong shape for that work. A governed real-time signal layer resolves the data deterministically — normalize, correlate, predict — and hands each agent a small, evidence-backed task packet. The result is lower token cost, a narrower surface for hallucination, and a feedback loop that improves at the speed of the data.

01

Lower token cost

Resolve context once, not on every model call.

02

Less hallucination

Selection and evidence are decided, not guessed.

03

AI-speed feedback

Every outcome tunes routing and prediction.

The model should reason, not re-derive.

Autonomous agents are good at reasoning over a clean, bounded input and producing an outcome. They are expensive and unreliable when forced to inspect raw fields, infer what the data means, and decide what is true — yet that is exactly what happens when an agent is pointed straight at a live firehose.

A **real-time signal layer** sits between the data and the agent. It performs the deterministic work — ingestion, normalization, enrichment, correlation, scoring, and prediction — once, and packages the result as an **Agent Task Packet**: a normalized signal object plus an explicit work instruction. The probabilistic component is bounded to interaction and narrative. The decisions that must be correct — which streams matter, which thresholds apply, whether the evidence is complete — are resolved outside the model and returned as explicit objects.

This separation produces three compounding effects. **Token cost falls** because the agent no longer re-parses the data universe on every call. **Hallucination falls** because stream selection, thresholds, and evidence requirements are resolved deterministically rather than invented. And the system **improves at AI speed**, because every agent outcome feeds back into routing, scoring, and prediction without waiting on a human retraining cycle.

The argument that follows is grounded in a working architecture and a live firehose registry. It treats the signal layer not as a feature bolted onto a model, but as the operating layer between high-velocity data and governed agent execution — **decision-grade signals** an agent can act on because it can show the evidence, delivered with infrastructure-grade reliability.

1×

Context is resolved a single time, ahead of delivery — not repeated inside every agent call.

0

Hidden firehoses. Anything a packet relies on must appear in its selected set.

2

Evidence roles required before a cross-domain packet is eligible for action.

Raw data is the wrong shape for an agent.

Enterprises, infrastructure, and research environments emit enormous volumes of telemetry: network flows, weather and disaster alerts, market ticks, scientific records, water and space-weather signals. Each carries meaning, but it arrives continuously, in incompatible formats, at different rates.

Most agents today operate on documents, prompts, static extracts, or narrow API calls. They rarely receive the full temporal context of real-time telemetry, so large categories of operational intelligence go unused. The instinctive fix — point the model at the firehoses and let it sort things out — fails in predictable ways. When the model is treated as the source of truth for selection, thresholds, and eligibility, deterministic decisions are being made probabilistically.

FAILURE MODES OF MODEL-AS-ROUTER

01 **Hidden stream usage**
A recipe quietly depends on a firehose that was never surfaced, so behavior can't be audited or explained.

02 **Silent threshold drift**
User-stated limits are paraphrased away on the next call; the watch no longer means what was asked.

03 **Context drowning evidence**
Hundreds of contextual streams crowd out the primary evidence the decision actually depends on.

04 **Partial-evidence packets**
An action is generated from one role's evidence when the thesis required convergence across several.

05 **Repeated field inspection**
Every question re-reads and re-ranks the same schemas, paying full token cost to re-derive known context.

06 **Endpoint invention**
A code-generation agent fabricates API paths or sources that do not exist, then builds on them.

> **The cost of inference-time reasoning**
>
> Pushing selection and judgment into the model imposes three taxes at once — a **token tax** from re-reading the data universe, a **hallucination tax** from guessing at sources and thresholds, and a **latency tax** from doing both on the critical path. The taxes scale with the number of streams, so the system gets worse precisely as it grows.

Move the reasoning out of the model.

The core move is to separate natural-language intent from execution authority. A model is well suited to proposing interface logic and narrative structure. It is poorly suited to being the source of truth for which firehoses to select, which endpoints to call, which thresholds to preserve, and whether a packet is eligible for action.

In a signal layer, those decisions resolve inside a deterministic **semantic resolver**. A user expresses a thesis in plain English; the resolver interprets it and returns explicit objects — a selected set of streams, the evidence roles the thesis requires, a data plan, a watch plan, and a packet policy. The model consumes that resolved plan. It does not re-derive it.

This is a different contract than retrieval or a chatbot wrapper. Retrieval hands the model more text to reason over. The signal layer hands the model **fewer, higher-authority objects** and removes the decisions it should not be making. The deterministic engine is the authority on structure; the model is left to do the part it is genuinely good at.

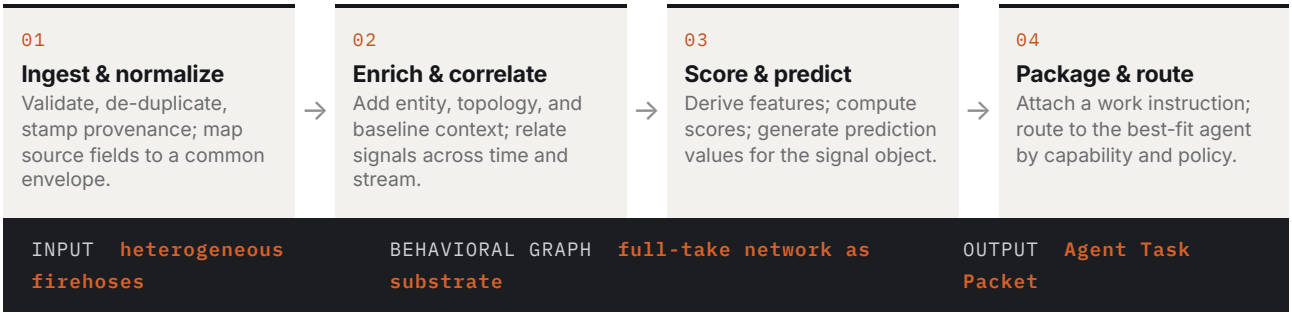
What stays deterministic

`selection`	Which firehoses and semantic groups a thesis maps to — bounded by role, never enumerated wholesale.
`thresholds`	User-stated limits are preserved verbatim through resolution; they cannot be paraphrased by a model.
`eligibility`	Whether evidence is complete enough for a packet to be created, or only surfaced for review.
`endpoints`	The exact API paths a builder may use — returned by the resolver, so there is nothing to invent.

The principle is simple to state and consequential in practice: the probabilistic component never holds operational authority. It interacts, narrates, and assembles — on top of a plan whose correctness was established before the model was invoked.

From firehose to packet, in one governed pass.

The signal layer is a pipeline of deterministic transformations. A raw record enters from a tap or adapter and leaves as an **Agent Task Packet**: a context-bearing, prediction-bearing, policy-bearing object an agent can act on without parsing the original source format. The same pass builds a behavioral graph from full-take network telemetry, which serves as connective tissue between otherwise separate streams.



THE PACKET AS A CONTRACT

A telemetry source uses a source-specific schema; an agent interacts with one normalized representation. The packet is the standardized contract between arbitrary sources and arbitrary agents. Add a new firehose and existing agents keep working — the agent never learns the source, only the common signal object.

SCALE BY GROUPING, NOT ENUMERATION

A flat catalog breaks at hundreds of streams. The resolver maps a thesis to semantic groups, selects bounded firehoses within them, and summarizes the rest. Selected streams stay role-relevant even as the universe grows to thousands — payloads and token use stay small.

> **Why this shape matters downstream**
>
> Because parsing, enrichment, correlation, and prediction happen once — before delivery — the agent receives precomputed context instead of raw records. That single design choice is what makes the three pillars on the following pages possible.

Pay for context once.

The dominant cost of an agent operating on live data is not the final reasoning step. It is the repeated work of re-reading and re-ranking the data universe on every call. The signal layer pays that cost a single time and amortizes it across every downstream consumer.

Instead of sending hundreds or thousands of firehose schemas to a model, the builder receives a small selected-firehose set, a small selected-group set, compact summaries for the remaining context, data endpoints the resolver already chose, refresh guidance, and the packet and watch policy. Parsing, normalization, enrichment, feature extraction, correlation, governance transforms, and prediction are performed ahead of delivery — so the agent spends its tokens on the task, not on rediscovering the world each time.

	AGENT ON RAW FIREHOSES	AGENT ON THE SIGNAL LAYER
What the model receives	The full schema universe, re-sent per query	**A bounded, resolved plan** plus group summaries
Where reasoning happens	Inside the model, every call	In the deterministic resolver, once
Cost driver	Size of the data universe	Size of the resolved plan
Effect of adding streams	Per-query cost inflates	**Per-query cost is flat**
Re-parsing of raw records	Repeated by each agent	Eliminated — context is precomputed

THE SCALING PROPERTY

Token spend tracks the plan, not the data.

When per-query cost is decoupled from the number of streams, expansion becomes safe. New firehoses are onboarded into a group and an evidence-role profile; they enrich what the system can resolve without enlarging what any single agent must read. Growth improves coverage instead of inflating the bill.

Decide it; don't guess it.

In operational settings, hallucination is not an abstract risk. It looks like inventing an endpoint, using a stream that was never surfaced, quietly relaxing a threshold, or declaring an action eligible on incomplete evidence. The signal layer closes each of these by making the decision deterministic and auditable.

THE RESOLVER CONTRACT

01 No hidden firehoses
If a recipe, watch, or data plan uses a stream, it must appear in the selected set. Behavior stays explainable.

02 Thresholds preserved
User-stated limits survive resolution unchanged — a model cannot soften them on a later turn.

03 Evidence-gated packets
A cross-domain packet requires evidence from every required role; partial evidence is a review candidate, never a completed action.

04 No post-hoc inference
Builders may not add streams after the resolver succeeds; the resolved plan is the boundary of what is in play.

05 Source-data boundary
Consumers use the layer's APIs — they do not scrape the console, poll upstream sources, or fabricate paths.

Evidence-role gating, concretely

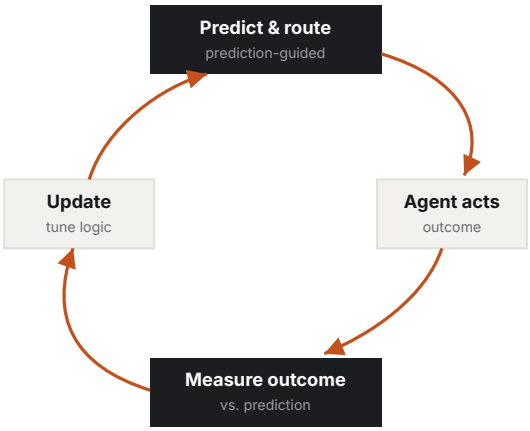
```
required_evidence_roles:
  - network_movement_evidence
  - real_world_disruption_context
→ packet is eligible only when both roles are present.
```

Movement-only or context-only evidence cannot be promoted into an operational packet when the thesis demands convergence. The gate is structural, not a matter of model judgment.

A human-review boundary sits on top of all of this. The layer may draft packets; execution, dispatch, ticketing, or blocking requires deliberate action. Drafting is automated — authority is not. This is how the system stays preemptive without overclaiming what automation should be trusted to do on its own.

A loop that closes at AI speed.

A static pipeline degrades. The signal layer is built as a closed loop: agent outcomes, human actions, approvals, and suppressions are recorded and fed back into the decisions that produced them — at the cadence of the data and the agents, not an episodic retraining schedule.



WHAT THE OUTCOME TUNES

Each outcome record carries the prediction value, the actual result, latency, and approval state. The system compares predicted to observed — a forecast anomaly against a confirmed one, expected agent success against measured — and uses the delta to update scoring, routing, enrichment, prediction, suppression, and the agent capability registry. Provenance is preserved throughout; improvement happens inside the layer, not by editing the source.

REPLAY AS A SAFETY VALVE

Historical signal objects can be re-emitted through the **same router** that serves live traffic. A new agent is qualified against real past conditions before it ever touches a live stream; routing rules are validated against known outcomes. This is how the loop moves fast without moving recklessly.

PREDICTION-GUIDED ROUTING

Packets are matched to the best-fit agent by capability, latency class, governance, and prior performance. As outcomes accumulate, routing sharpens — low-latency work reaches low-latency agents, sensitive work stays in controlled environments, and weak performers are routed around.

Anatomy of an Agent Task Packet.

The packet pairs a **processed signal object** — what is true — with a **work instruction** — what to do about it. Together they let an agent execute without seeing raw telemetry. Each field maps to one of the three pillars.

AGENT TASK PACKET

```
# processed signal object
{
  "event_time": "…T12:00:00Z",
  "stream": { network / flow },
  "entity": { asset, relations },
  "features": { rate, periodicity },
  "correlation": { related_ids },
  "prediction": { value, confidence },
  "governance": { tenant, policy }
}
# work instruction
{
  "task_type": "investigate",
  "evidence_required": [ … ],
  "latency_ms": 250,
  "allowed_tools": [ … ],
  "approval": "policy_engine",
  "feedback_required": true
}
```

HOW THE FIELDS EARN THEIR PLACE

`prediction` **Feedback.** The forecast that later gets compared to the real outcome — the input to the closed loop.

`correlation` **Token cost.** Cross-stream relationships resolved once, so the agent doesn't reconstruct them.

`evidence_required` **Hallucination.** The roles that must be present; the gate the packet must pass to be actionable.

`governance` **Hallucination.** Tenant and policy tags travel with the data, so restricted fields never reach the wrong agent.

`allowed_tools` **Governance.** The bounded toolset for this task — and, by omission, what is prohibited.

`approval` **Review boundary.** What must sign off before action, keeping drafting automated and authority human.

`latency_ms` **Routing.** The class the router uses to match the packet to a suitable agent.

One object, two readers

The signal object answers a question; the work instruction issues a command. Keeping them in one packet means an agent never has to assemble context and intent from separate places — and the layer can enforce that both halves agree before anything is dispatched.

Cyber-physical convergence.

Consider a single plain-English thesis and follow it through the layer. The point is not the dashboard; it is how the three pillars show up in one governed flow.

> OPERATOR THESIS
>
> Detect when real-world operational stress is converging with cyber and network risk before it becomes an incident. Create an Agent Task Packet when a critical external counterparty or a high-volume network route is simultaneously exposed to abnormal data movement and a relevant real-world disruption signal — severe weather, a FEMA disaster, space-weather, or earthquake activity — within the same geographic or operational window.

HOW THE RESOLVER ANSWERS

DOMAIN · INTENT	Geo-operational context · network-and-environment intersection watch
EVIDENCE ROLES	`network_movement_evidence` + `real_world_disruption_context`
SELECTED GROUPS	AOT collectors · Earth operations · Water & climate · Space operations
BOUNDED STREAMS	`aot:collector:*` · `sf.earth.noaa.nws.alert` · `sf.earth.usgs.earthquake.event` · `sf.emergency.fema.disaster_declaration` · `sf.space.noaa.goes.xray_flux`
PACKET POLICY	Require all evidence roles; partial evidence surfaces as a review candidate only

THE THREE PILLARS, IN ONE FLOW

TOKEN COST

Selection is bounded to four groups out of the whole registry. The agent reads a handful of role-relevant streams, not the universe.

HALLUCINATION

No packet fires on network movement alone. Both evidence roles must converge in the same window, or it is only a candidate.

FEEDBACK

Whether the packet proved out — incident or false alarm — feeds back to tune the convergence thresholds and routing.

Infrastructure, not a feature.

The signal layer is the operating layer between high-velocity data and governed agent execution. It is what makes agents economically viable at firehose scale, trustworthy enough to act, and capable of improving at the speed of the data they run on.

Each pillar reinforces the others. Bounding what an agent reads **lowers token cost** and, at the same time, removes most of the surface where a model could invent a source or drift a threshold. Resolving evidence deterministically **reduces hallucination** and produces clean prediction-versus-outcome records. Those records **close the loop**, and a sharper loop tightens selection and routing — which lowers cost again.

The position is deliberate. This is a **real-time signal layer** producing **decision-grade signals** that **complement existing systems** rather than replace them — built for **preemptive security** and operational foresight, with **infrastructure-grade reliability**. The model is freed to do what it does well, on top of a foundation engineered to be correct.

THE TAKEAWAY

Resolve once. Gate on evidence. Learn from every outcome.

Three disciplines, one substrate. The signal layer turns raw, heterogeneous, real-time data into bounded, explainable, agent-ready inputs — and keeps getting better as it runs.

SD Streaming Defense

Architecture and mechanics in this document reflect the Signal Fabric Predictive Operations system and the provisional patent *Systems and Methods for Real-Time Telemetry Normalization, Prediction-Guided Agent Routing, and Automated Outcome Generation.* Streaming Defense Confidential.

SIGNAL.
STRUCTURE.
STRENGTH.