Product Brief

Selector Platform Architecture

Inside Selector's Data-Centric Architecture for AI-Driven Observability

Executive Summary

Modern network operations teams face a structural challenge that no amount of additional tooling can fully solve. The data needed to understand an incident — metrics, logs, topology, configuration state, CMDB context, and change history — is collected in pieces, stored in different formats, processed through separate pipelines, and analyzed in isolation. When something goes wrong, this fragmentation doesn't just slow down the response. It forces the wrong kind of response: large, inefficient war rooms where engineers from every domain gather not to solve the problem together, but simply to stand by and validate that their own domain isn't to blame.

The consequence is a pattern that repeats across organizations of every size. A network outage triggers a call. Dozens — sometimes hundreds — of people join. Each team pulls up its own dashboard, runs its own queries, and waits for someone else to establish context before the real investigation can begin. Meanwhile, customers experience degradation, SLAs are missed, and the most experienced engineers spend their time in coordination overhead rather than problem-solving.

The root cause of this pattern isn't a lack of data. Most organizations have more telemetry than they can act on. The problem is structural: traditional observability architectures were designed around data types, not operational outcomes. They optimize for depth within a single domain — excellent metrics pipelines, excellent log aggregation, excellent APM — but leave the work of connecting those domains entirely to human operators who are already under pressure.

The Core Problem Selector Solves

In today's complex network environments, outages often trigger large, inefficient war rooms where numerous engineers attempt to correlate data across siloed tools. Each team relies on its own specialized tools and must be present to validate its domain's "innocence," resulting in wasted time and fragmented troubleshooting. Selector eliminates this by unifying data, context, and intelligence in a single platform.

The Selector Answer

Selector was built to address this challenge at the architectural level. Rather than adding another tool on top of existing fragmented pipelines, Selector introduces a fundamentally different structural approach: a data-centric platform that ingests raw telemetry and operational context together from the start, normalizes and enriches them in a unified programmable layer, applies a multi-stage AI and ML stack to surface genuine causal insights, and delivers those insights through a human-centric collaboration interface that meets teams where they already work.

The platform is organized around four core services — Collection, Data Hypervisor, Knowledge Service, and Collaboration — each purpose-built to address a specific structural gap in traditional observability. Together they form a data pipeline that transforms raw, fragmented telemetry into explainable, actionable intelligence at scale.

The result is a platform where every alert, anomaly, and incident arrives with the full context required to understand it, not just that something changed, but what it affected, why it happened, and what to do next. Where traditional approaches require operators to manually reconstruct that context from disconnected tools under pressure, Selector makes it available by default.

300+

Pre-built integrations across network, cloud, infrastructure, and ITSM tools — analysis begins from existing data on day one

99.9%

Noise reduction achieved through ML-based log clustering and self-supervised signal detection — surfacing only what matters

Millions/min

Data points ingested per minute with workload-independent horizontal scaling across all telemetry types

The Data-Centric AI Approach

Why Data-Centric Architecture Matters

There are two primary approaches to building AI systems for network operations. The conventional model-centric approach focuses on continuously training larger and more complex models — an approach that struggles to adapt to new data types without significant retraining and that produces outputs that are technically sophisticated but operationally unreliable. Because these models are optimized for the data they were trained on, they tend to degrade as environments evolve, require specialist expertise to maintain, and produce recommendations that engineers are reluctant to act on without extensive manual verification.

Selector takes a fundamentally different path: data-centric AI. Rather than building models that must absorb increasing complexity, Selector focuses on standardizing, enriching, and contextualizing the data itself before any model sees it. By transforming raw telemetry from any source into a unified metric and event framework, Selector enables its models to reason consistently and reliably across inputs regardless of their origin, format, or structure. The intelligence becomes durable because it is grounded in data quality, not model complexity.

Model-Centric vs. Data-Centric AI

Model-Centric AI	Selector: Data-Centric AI
Trains increasingly larger, more complex models	Standardizes and enriches data at the point of ingestion
Struggles to adapt when new data types are introduced	Models reason consistently across all inputs, regardless of source
Technically sophisticated but operationally brittle	Produces reliable, explainable, and trusted outputs
Requires specialist expertise to maintain and retrain	Adapts to new sources and protocols without re-instrumentation
Outputs are often second-guessed rather than acted on	AI recommendations engineers act on, not second-guess

How Selector Applies This Philosophy

Selector's data-centric approach is not a single technique — it is a design principle that runs through every layer of the platform. At the collection layer, it means ingesting telemetry and context together rather than in separate pipelines. At the Data Hypervisor layer, it means enriching every record with topology, ownership, maintenance state, and business criticality before intelligence is applied — so that models never have to reason from incomplete information. At the Knowledge Service layer, it means applying self-supervised and unsupervised learning directly to the data stream, building lightweight models for every time series that continuously learn what normal looks like and detect deviations in context.

This approach produces a qualitatively different kind of output. Because every insight Selector generates is grounded in normalized, enriched, and contextually complete data, the recommendations it surfaces are explainable: engineers can trace every alert, correlation, and root-cause finding back to the specific signals, relationships, and reasoning that produced it. That explainability is what makes Selector's output trustworthy enough to act on — and ultimately, trustworthy enough to automate.

Enabling Agentic AI and Automation

Reliable automation requires high-confidence insights. This is the fundamental constraint that limits most AIOps implementations: when AI outputs are uncertain, incomplete, or difficult to trace, operators are right to override them. The result is a system that generates recommendations but doesn't reduce workload, because every recommendation still requires manual verification before any action is taken.

Selector's architecture is designed to break through this constraint by filtering massive volumes of raw telemetry into precise, deterministic outputs that operations teams can genuinely trust. When Selector confidently identifies a root cause — a fiber cut, a device reboot, a BGP flap cascading from a configuration change — the evidence chain is clear, the affected scope is known, and the confidence is high enough to act on. That creates the conditions for genuine automation: an agentic AI workflow can drain affected links, reroute traffic, update the ITSM ticket, and notify the on-call team without requiring a human to verify each step first.

This moves organizations from reactive troubleshooting — where the goal is to understand what happened after the fact — to controlled, intelligent automation where issues are detected, understood, and resolved faster than human operators could manage alone. It is the practical outcome that Selector's data-centric architecture is ultimately designed to enable.

A Fundamental Skills Bridge

A persistent challenge in network operations is the gap between domain expertise and AI capability. Network engineers have deep knowledge of how infrastructure behaves but limited experience building or evaluating AI systems. Data scientists understand AI but lack the networking context needed to interpret telemetry meaningfully. Selector's data-centric architecture bridges this gap — by encoding operational context directly into the data layer, it ensures that AI models work from the same vocabulary and understanding that experienced network engineers use, without requiring engineers to become AI specialists or data scientists to become network experts.

Platform Architecture Overview

Selector's platform is organized around four core services that work in sequence, each enriching the data and intelligence passed to the next layer. This layered architecture is what allows Selector to deliver reliable, explainable, and actionable insights at scale.

Collection

Horizontal ingestion architecture that unifies raw telemetry and operational context from 300+ sources across network, cloud, infrastructure, application, and edge domains.

Data Hypervisor

Programmable normalization and enrichment layer that standardizes structure, extracts entities, adds operational context, suppresses noise, and routes data to the appropriate storage backend.

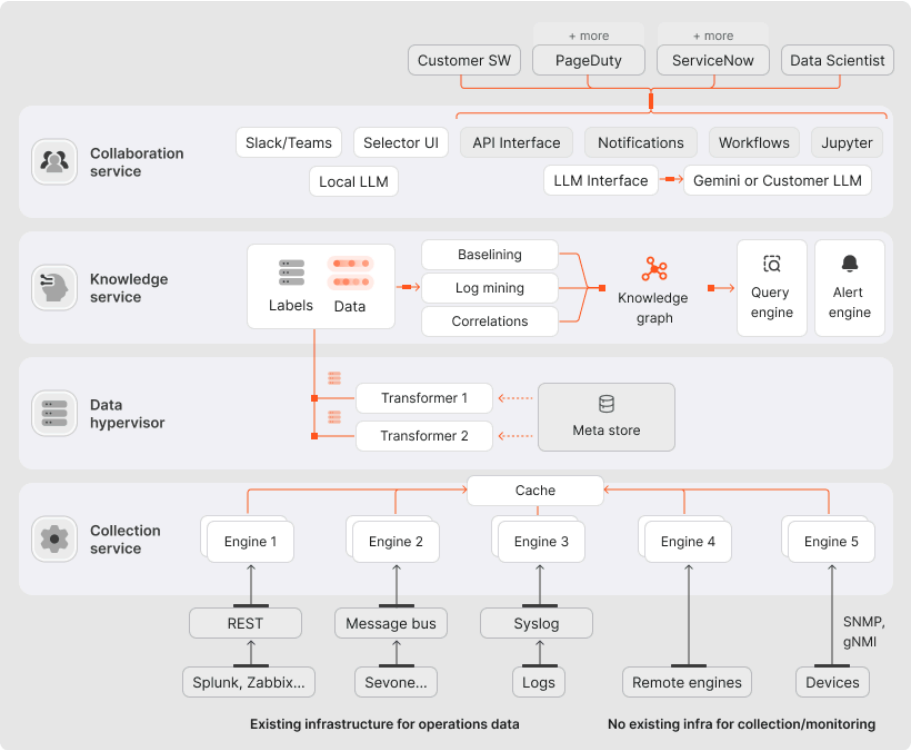
Knowledge

AI and ML intelligence layer that applies adaptive anomaly detection, temporal and contextual correlation, root-cause analysis, forecasting, and pattern analysis to normalized operational data.

Collaboration

Human-centric presentation layer combining natural language interaction, dynamic dashboards, ChatOps integration (Slack, Teams), alerting, and workflow automation so teams can act on intelligence where work already happens.

Each layer is purpose-built and feeds the next, ensuring that by the time insights reach operators, they are normalized, enriched, correlated, and presented in a directly actionable form. This architecture reflects Selector's data-centric AI philosophy: rather than building ever-larger models, Selector standardizes and enriches the data itself so that AI models can reason consistently regardless of source, format, or domain.



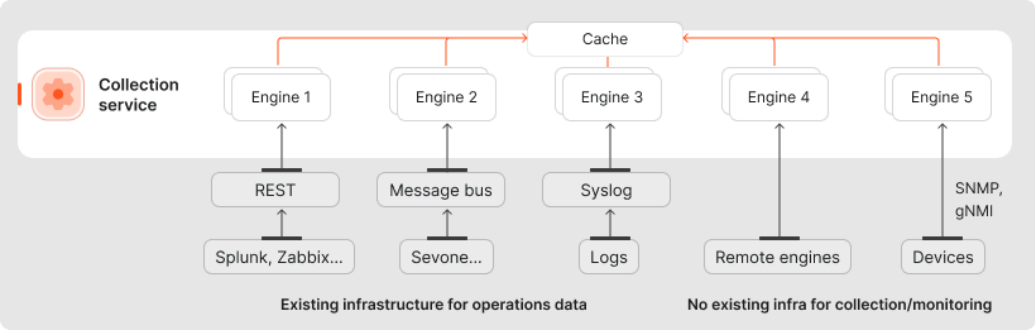
Selector Platform Architecture — all four services working together

Layer 1: Collection Service

A horizontal ingestion architecture for full-stack operational context

Selector's Collection Service eliminates a structural limitation at the core of traditional observability: the separation of telemetry from operational context. In most platforms, telemetry and the context needed to interpret it are collected in different systems, processed through separate pipelines, and correlated only after an incident has already begun.

Selector takes a different approach. Its collection layer sits horizontally at the bottom of the platform, allowing organizations to ingest raw telemetry and contextual data from across network, cloud, infrastructure, application, and operational systems into a shared model for cross-domain correlation, root-cause analysis, and action.



Collection Service — horizontal ingestion engine architecture

The Problem with Traditional Correlation

Traditional monitoring platforms were architected around a vertical data model: logs into one pipeline, metrics into another, traces into a third, while topology, CMDB, configuration state, and operational events remain in separate tools. Each pipeline operates with its own schema, storage format, and query engine — optimized for depth within a single data type, not breadth across types.

- **Type-specific pipelines preserve data silos:** Correlation happens later through predefined rules or manual investigation, rather than during ingestion.
- **Operational context is structurally absent:** Logs, metrics, configurations, topology, and change signals live in different tools, slowing root-cause analysis.
- **Scaling one workload forces overbuilding the entire stack:** SNMP polling, high-frequency streaming telemetry, and log volume often require infrastructure scaling beyond the specific collection tier under load.

How the Collection Service Works

Horizontal Placement

Collection engines are deployed close to the data source — in data centers, branch locations, cloud regions, or network segments — reducing latency between signal generation and ingestion and maintaining collection continuity in distributed and hybrid environments.

Push and Pull Ingestion

Selector supports direct device collection via SNMP, gNMI, syslog, and CLI alongside upstream-tool ingestion from platforms such as Splunk, Prometheus, and Kafka. Both push-based streaming and pull-based polling are supported, allowing teams to consolidate existing tool investments into a single pipeline without forklift replacement.

Context-Rich Ingestion

Collection is not limited to time-series data and logs. Selector simultaneously ingests topology and dependency graphs, CMDB and inventory records, configuration state, routing policy, interface aliases, and change signals. Every telemetry record enters the shared intelligence layer with the operational context required to correlate symptoms to causation.

Scale by Workload

Because the collection layer is horizontal, each ingestion workload scales independently. If SNMP polling volume increases, capacity is added to that tier without provisioning additional infrastructure across the rest of the platform. Each workload — gNMI streaming, NetFlow export, syslog ingestion, API-based pull — scales on its own terms.

What Selector Can Ingest

Capability	Description
Metrics / Time-Series	CPU, memory, interface counters, and custom device and service metrics collected directly via SNMP, gNMI, and streaming telemetry, or pulled from Prometheus-compatible sources and REST APIs.
Logs / Syslog	Syslog streams from network devices, infrastructure, and application layers. ML-based clustering groups similar log patterns to surface anomalies and reduces noise before signals reach the correlation layer.
Events / Alerts / Anomalies	Operational events, SNMP traps, syslog-derived events, generated incidents, and anomaly signals from monitoring, ITSM, and observability tools — normalized and enriched on ingestion.
Flow and Network Telemetry	NetFlow, sFlow, and IPFIX records for traffic analysis and anomaly detection, alongside high-frequency gNMI streaming for near-real-time KPI monitoring
Configuration State	Running and intended device configuration ingested via Netconf/CLI, used to track configuration state, detect drift, and correlate change events directly with performance degradation or incidents.
Topology / Dependency Context	Device relationships, service dependency graphs, and digital twin context built from SNMP, configuration data, and telemetry. Stored in a Knowledge Graph and used to enrich every ingested signal.
Metadata and CMDB Context	ServiceNow CMDB records, inventory data, ownership attributes, site and region labels, interface aliases, and routing policy context — enriching every telemetry record with operational relationships.
Third-Party Tool Telemetry	APM, log, metric, and trace data from Splunk, NetBox, InfluxDB, Kafka, ThousandEyes, Prometheus, SolarWinds, LogicMonitor, and ServiceNow. Selector connects to 300+ pre-built integrations.

Operational Benefits

One architecture across domains: A single horizontal collection layer replaces the need for separate vertical stacks per domain. Network, infrastructure, cloud, application, and edge signals all feed the same shared intelligence layer.

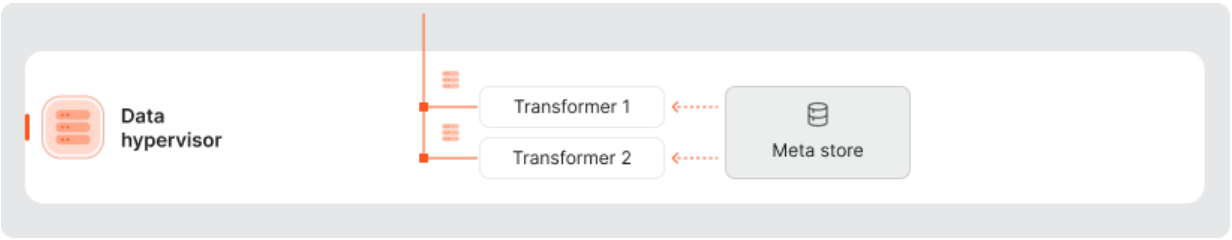
- **Faster time to value:** Selector ingests directly from tools already in production — Splunk, Prometheus, SolarWinds, ServiceNow, and 300+ others — without requiring schema changes, re-instrumentation, or replacement of existing infrastructure. Cross-domain analysis begins from day one.
- **Better root-cause analysis:** Because telemetry and operational context are ingested together, correlation happens during ingestion rather than after. The platform connects symptoms to causation across domains, not just within a single telemetry type.
- **Operational flexibility:** The collection layer is source-agnostic and schema-agnostic. Teams can expand telemetry coverage, swap upstream tools, or migrate to new protocols without redesigning the ingestion architecture.

Layer 2: Data Hypervisor

A programmable data layer that normalizes, enriches, and unifies operational data for cross-domain intelligence

Selector's Data Hypervisor addresses a structural gap in traditional observability and AIOps stacks: data arrives in different formats, with different schemas, different levels of structure, and almost none of the business and operational context required to interpret it correctly.

The Data Hypervisor sits between the collection and intelligence layers, where it normalizes, enriches, de-duplicates, suppresses noise, and connects metrics, logs, events, flow records, configuration state, topology, CMDB metadata, and operational signals into a common data model. The result is a programmable, source-agnostic data plane that preserves raw detail, adds context before intelligence is applied, and prepares every signal for correlation, root-cause analysis, and action.



Data Hypervisor — normalization, transformation, and meta store layer

Why Traditional Data Normalization Fails

- **Context is lost before analysis begins:**
  Raw telemetry rarely includes the business and operational context needed to make it useful across domains. Critical metadata — site, owner, device role, service dependency, maintenance state — must be added before signals can be analyzed accurately.
- **Early transformation strips away useful signals:**
  Many traditional ETL pipelines transform, collapse, or abstract data before it is fully stored. That removes the original timestamps, source details, and event boundaries needed to reconstruct what actually happened across infrastructure layers.
- **Static schemas do not scale:**
  Predefined schemas, source-specific mappings, and hand-built regex rules are difficult to maintain as vendors, software versions, and message formats evolve. As environments change, these brittle normalization models cause data to be misclassified, dropped, or stripped of meaning.

How the Data Hypervisor Works

ML-Driven Entity Extraction and Normalization

Rather than relying on static regex libraries or predefined schemas, the Data Hypervisor uses machine learning to infer structure from unstructured and semi-structured content. Log messages, SNMP traps, syslog events, and CLI outputs are parsed without requiring hand-crafted templates, and similar events are clustered into meaningful types across vendor and format variations.

Programmable Context Layer

Customer-specific business logic is digitized directly into the hypervisor through YAML-based configuration and DevOps workflows. Maintenance windows, provider notification rules, criticality tags, suppression rules, device ownership labels, and circuit identifiers are applied programmatically and evolve alongside the customer environment without requiring platform changes.

ELT-Style Raw Data Preservation

The Data Hypervisor follows an ELT (Extract → Load → Transform) model: raw data lands first with full source detail and original timestamps intact. Transformation and enrichment are applied afterward. This approach preserves the exact signal fidelity needed to reconstruct cross-domain timelines — critical for identifying causal chains that span network, cloud, infrastructure, and application domains.

Source-Agnostic Data Routing

Different data types require different storage and processing models. Metrics, logs, topology, configuration state, and flow records are normalized into a shared semantic layer, then routed to the backend most appropriate for each type. Applications, dashboards, and intelligence services interact with a unified data model rather than isolated storage systems.

Data Hypervisor Capabilities

Capability	Description
Automatic Schema Inference	Uses ML to infer structure from unstructured and semi-structured data, including free-form logs, syslog, SNMP traps, and CLI output, reducing dependence on manually authored parsing rules.
Multi-Source Normalization	Normalizes data from network devices, cloud platforms, applications, monitoring tools, and ITSM systems into a consistent internal model regardless of vendor, protocol, or encoding format.
Contextual Enrichment	Enriches records with operational metadata such as topology relationships, service dependencies, ownership, location, maintenance state, and business criticality.
Noise Suppression and Deduplication	Identifies duplicate signals, suppresses operationally irrelevant noise, and reduces flapping or repetitive events before they reach the intelligence layer.
Programmable Business Logic	Applies customer-specific logic such as maintenance windows, provider notices, suppression rules, escalation tags, and criticality settings through YAML and DevOps-managed workflows.
Cross-Domain Signal Linkage	Uses shared metadata and enrichment to connect records across metrics, logs, events, configuration changes, topology, and flow data on a unified timeline.
Intelligent Data Routing	Directs normalized records to the storage backend appropriate for their type while preserving a consistent query model across all data domains.
Live Query and Analysis	Supports real-time dashboards, ad hoc investigation, and AI-driven analysis across all ingested signal types through a unified data layer.

Operational Benefits

- **More reliable AI and correlation:** Selector's intelligence layer operates on normalized, enriched, and de-duplicated data rather than fragmented raw inputs. This improves correlation accuracy, reduces false positives, and strengthens root-cause analysis across domains.
- **Lower operational overhead:** Automatic schema inference and ML-based extraction reduce the burden of maintaining large libraries of regex rules, field mappings, and source-specific transformations as environments evolve.
- **Preserved signal fidelity for investigation:** Because raw data is retained before transformation, teams can revisit historical incidents with full source detail intact and analyze them using current topology, metadata, and operational
- **Operational logic that evolves with the environment:** Business rules are managed as code through YAML and standard DevOps workflows, making them auditable, version-controlled, and easier to update as infrastructure and policy change.
- **One data model across the full stack:** Network, infrastructure, cloud, and application signals are normalized into a shared model, enabling unified search, consistent dashboards, and cross-domain root-cause analysis without separate tools or query paths.

Layer 3: Knowledge Service

An intelligence layer for anomaly detection, correlation, root-cause analysis, and predictive operational insight

Selector's Knowledge Service is the analytical engine that helps operations teams interpret what their data is actually telling them. In complex environments, metrics, logs, alerts, and events may indicate that something has changed — but not how the signals relate, which conditions matter most, or where the investigation should begin. The Knowledge Service addresses this by applying machine learning, correlation analysis, root-cause analysis, and predictive techniques to operational data so that teams can evaluate issues with more context and less manual effort.

Positioned between the Data Hypervisor and the Collaboration Service, the Knowledge Service operates on normalized and enriched operational data and applies ML- and heuristics-based analysis to metrics, logs, alerts, events, and related context to identify deviations from expected behavior, determine how signals are related, and surface the most likely causal explanation.

Knowledge Service — labels, data, baselining, log mining, correlations, knowledge graph

Why Traditional Operational Intelligence Breaks Down

- **Static thresholds do not reflect live operating conditions:** Fixed rules are difficult to maintain in environments where baselines shift by workload, time of day, topology, and service behavior. This creates noise in some conditions and blind spots in others.
- **Signals are evaluated in isolation:** Metrics, logs, alerts, and changes are often processed through separate analytical paths and correlated only after an incident has already begun. The result is slower investigation and higher operator effort during the situations where speed matters most.
- **Operational context is incomplete:** Telemetry without dependency, topology, and configuration context makes it harder to distinguish symptoms from likely causes. In distributed environments, this limits the reliability of correlation and root-cause analysis.

How the Knowledge Service Works

Adaptive Anomaly Detection

Baselines are established for time-series data and deviations are detected using models that account for cyclic and seasonal behavior. This allows evaluation of changing operating conditions in context rather than against fixed thresholds — reducing both alert noise and missed detections as environments evolve.

Temporal and Contextual Correlation

Selector evaluates relationships between events using both time and operational context. By connecting related signals across infrastructure domains — rather than surfacing each alert independently — the Knowledge Service reduces noise created by independent alert streams and surfaces coherent incident narratives.

Root-Cause and Pattern Analysis

Using recommender and association models, Selector ranks related events and evaluates likely causation. Sequential mining identifies patterns where the order of events is operationally significant — so teams can understand not just what happened, but in what sequence and why.

Query and Workflow Integration

Analysis is available through a unified query layer and supports downstream collaboration workflows, including natural language interaction via the Selector UI, Slack, or Microsoft Teams. Results feed directly into the Collaboration Service for coordinated incident response.

Knowledge Service Capabilities

Capability	Description
ML-Driven Anomaly Detection	Detects deviations in time-series behavior using adaptive baselines that account for cyclicity, seasonality, and changing operating conditions.
Temporal and Contextual Correlation	Connects related signals across metrics, logs, alerts, and events using time, topology, and operational context.
Root-Cause Analysis	Uses recommender and association models to rank related signals and identify likely causal relationships.
Log Analytics & Entity Extraction	Applies clustering and named entity recognition (NER) to extract operationally useful structure from machine-generated log data.
Forecasting and Capacity Analysis	Supports forecasting for KPIs such as memory and interface utilization to help teams evaluate likely future behavior.
Event Prediction and Sequential Mining	Identifies patterns in historical event behavior, including cases where event order is important.
Topology-Aware Analysis	Improves correlation by incorporating physical and logical relationships across infrastructure and services.
Unified Query & Investigation Support	Provides a common interface for querying operational data and preparing results for dashboards, workflows, and collaboration channels.

Customization with S2ML

Every network environment is unique. Selector provides flexibility through Selector Software Modeling Language (S2ML), enabling teams to tailor the platform by customizing enrichment and labeling, defining noise reduction rules, adjusting event prioritization, and refining LLM outputs. This ensures the system aligns with each organization's operational workflows and terminology.

Operational Benefits

- **Faster detection and investigation:** Supports lower MTTD, MTTI, and MTTR by combining anomaly detection, event correlation, and root-cause analysis in a single service layer.
- **Less manual rule maintenance:** Reduces dependence on fixed thresholds, regex-heavy parsing, and hand-built logic that is difficult to sustain in changing environments.
- **Better context during incidents:** Improves visibility into how issues relate across infrastructure, topology, and change activity rather than presenting signals in isolation.
- **More consistent operational analysis:** Provides a shared analytical layer across metrics, logs, alerts, and events instead of separate investigative paths for each data type.

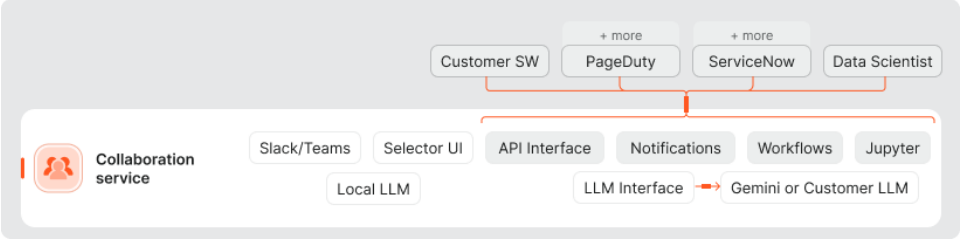
Layer 4: Collaboration Service

A human-centric but increasingly agentic operational command center for investigation, coordination, and response

Selector's collaboration service is the operational layer where intelligence becomes coordinated action. In complex environments, identifying an issue is only part of the problem. Teams must also align on what is happening, determine the most likely cause, engage the right stakeholders, decide the safest next step, and maintain a clear record of the response as conditions evolve.

Positioned at the top of Selector's architecture, the Collaboration Service moves beyond traditional ChatOps patterns in which users ask questions, review summaries, and manually drive the next step. Instead, it establishes an operational command center in which Selector's AI interface can participate directly in the incident lifecycle: understanding the investigation goal, assembling relevant evidence across domains, coordinating people and systems, recommending next actions, and keeping the response loop active until the issue is resolved or formally escalated.

Rather than functioning as a passive interface to platform outputs, the Collaboration Service makes Selector's intelligence operational. It brings together natural language interaction, dynamic visualizations, alerting, workflow investigations, and agentic coordination in a common service layer so teams can investigate, decide, and act with far less manual orchestration.



Collaboration Service — interfaces, integrations, and LLM connectivity

Why Traditional Operational Workflows Break Down

- **Tool fragmentation increases operator effort:** Operations teams often work across multiple systems for monitoring, ticketing, collaboration, and reporting. This creates unnecessary overhead during incidents and makes it harder to maintain a shared view of the problem.
- **Context switching slows response:** When alerts, dashboards, and collaboration happen in separate tools, engineers spend time moving between systems to gather basic context before they can begin remediation.
- **Access to operational intelligence is inconsistent:** Many platforms still depend on query languages, static dashboards, or specialist interfaces that limit who can investigate issues and how quickly teams can respond.

How the Collaboration Service Works

Natural Language Interaction and Goal-Oriented Investigation

Users can engage the platform in natural language, but the interaction is no longer limited to question-and-answer exchanges. Selector's AI can interpret the operational goal behind the conversation — for example, isolating the cause of rising latency, validating whether a change triggered and incident, or identifying the safest remediation path — and then guide the investigation accordingly. This makes the experience more accessible to a broader set of operators while also making incident handling more directed and outcome-oriented.

Actionable Recommendations, Not Just Summaries

The Collaboration Service is designed to make each interaction operationally useful. Instead of only reporting what happened, Selector's AI can present the current state of evidence, identify the safest next actions, and frame decisions in a way operators can immediately act on. This helps teams move from information consumption to guided response, with clear options for investigation, remediation, escalation, or approval.

Persistent Operational Context

As incidents unfold, the Collaboration Service maintains continuity across the conversation and the workflow. Selector can retain the working context of the event — what has been investigated, which hypotheses have been ruled out, which teams are engaged, and what approvals or actions are pending — so the organization does not have to repeatedly reconstruct state across tickets, chat threads, and dashboards. The result is a more consistent and auditable response process.

Workflow Integration and Human-in-the-Loop Execution

Selector connects collaboration directly to ticketing, notifications, reporting, and downstream workflows so that insight can be translated into action across operational systems. Where organizations are ready, Selector can help initiate or coordinate automated actions. Where governance requires oversight, the same workflows can remain human-in-the-loop, with recommended actions presented for approval before execution. This allows users to progress from assisted operations toward controlled automation without sacrificing trust or accountability.

Cross-System Operational Response

By integrating with platforms such as Slack, Microsoft Teams, ServiceNow, Jira, PagerDuty, and ServiceDesk, the Collaboration Service allows Selector to operate where teams already work while maintaining alignment with broader operational processes. The goal is not simply to embed observability into chat, but to turn collaboration surfaces into an extension of Selector's operational intelligence and response model.

Collaboration Service Capabilities

Capability	Description
Natural Language Querying	Allows users to interrogate the environment in plain language and receive grounded findings, recommended next steps, and relevant visual outputs.
Copilot & Conversational Investigation	Supports goal-driven follow-up during active incidents so Selector's AI can continue investigating, track context, and guide the next step.
Slack & Microsoft Teams Integration	Delivers operational intelligence into common collaboration channels so teams can review findings, coordinate response, and act faster.
Dynamic Dashboards & Visualizations	Supports interactive dashboards, device-health views, drill-down workflows, and other visual outputs tied to live operational investigation.
In-Channel Coordination & Escalation	Connects with systems such as ServiceNow, Jira, PagerDuty, and ServiceDesk to support escalation, approvals, and workflow management.
Persistent Incident Context	Maintains incident history, key findings, and pending actions so teams can resume work without reconstructing the investigation.
Workflow-Ready Outputs	Makes platform outputs available through APIs so external systems can retrieve findings, recommendations, and related results programmatically.

Operational Benefits

- **Faster incident coordination:** Helps teams move from alerts to evidence, ownership, and next steps in a more direct workflow.
- **Less tool switching during response:** Reduces the need to move across separate dashboards, messaging tools, and ticketing systems to keep the investigation aligned.
- **Broader access to operational insight:** Makes operational intelligence easier to use through natural language interaction and shared collaboration interfaces.
- **More consistent execution across teams:** Supports common workflows for investigation, escalation, approval, and response across day-to-day operations and active incidents.

Business Values and Customer Outcomes

Selector's data-centric architecture is not just a technical design choice — it directly translates into measurable operational improvements for the organizations that deploy it. By addressing the structural causes of operational fragmentation rather than adding more tooling on top of existing silos, Selector enables outcomes that point tools and reactive approaches cannot deliver.

Reduced Mean Time to Resolution

By combining anomaly detection, correlation, and root-cause analysis in a single intelligence layer, Selector reduces the investigation effort required to understand and resolve incidents — supporting lower MTTD, MTTI, and MTTR across all operational domains.

Tool Consolidation Without Disruption

Selector's 300+ pre-built integrations and schema-agnostic collection layer allow organizations to unify signals from existing tools without replacement or re-instrumentation. Cross-domain analysis begins from existing data on day one.

Democratized Operational Intelligence

Natural language querying, shared dashboards, and ChatOps integration make operational intelligence accessible to engineers and operators regardless of their familiarity with specialized query languages or domain-specific interfaces.

Smaller, More Effective War Rooms

Where traditional outages required hundreds of engineers to validate their respective domains, Selector's cross-domain intelligence allows smaller teams to isolate root causes with confidence — without requiring domain specialists from every team to be present.

Trusted Foundation for AI Automation

Reliable automation requires high-confidence insights. Selector's architecture filters massive volumes of data into precise, deterministic outputs, creating a trusted foundation for agentic AI workflows.

Operational Resilience at Scale

Selector's horizontal, workload-independent scaling model means that growth in any single data type doesn't require overbuilding the entire stack. The platform grows with operational needs without architectural redesign.

A Different Kind of Platform for a Different Kind of Problem

The operational challenges that Selector addresses are not new. Network outages have always required cross-domain investigation. Telemetry has always been distributed across multiple systems. The gap between the data teams collect, and the insight they can act on has always created pressure on operations engineers during incidents.

What has changed is the scale and complexity of modern infrastructure, and the corresponding gap between what traditional architectures can support and what operations teams actually need. As environments have grown more distributed — spanning on-premises infrastructure, multiple cloud providers, edge deployments, and a growing diversity of application and network layers — the fragmentation that was always present has become structurally unmanageable. Teams are not failing because they lack talent or effort. They are failing because the tools they depend on were designed for a world that no longer exists.

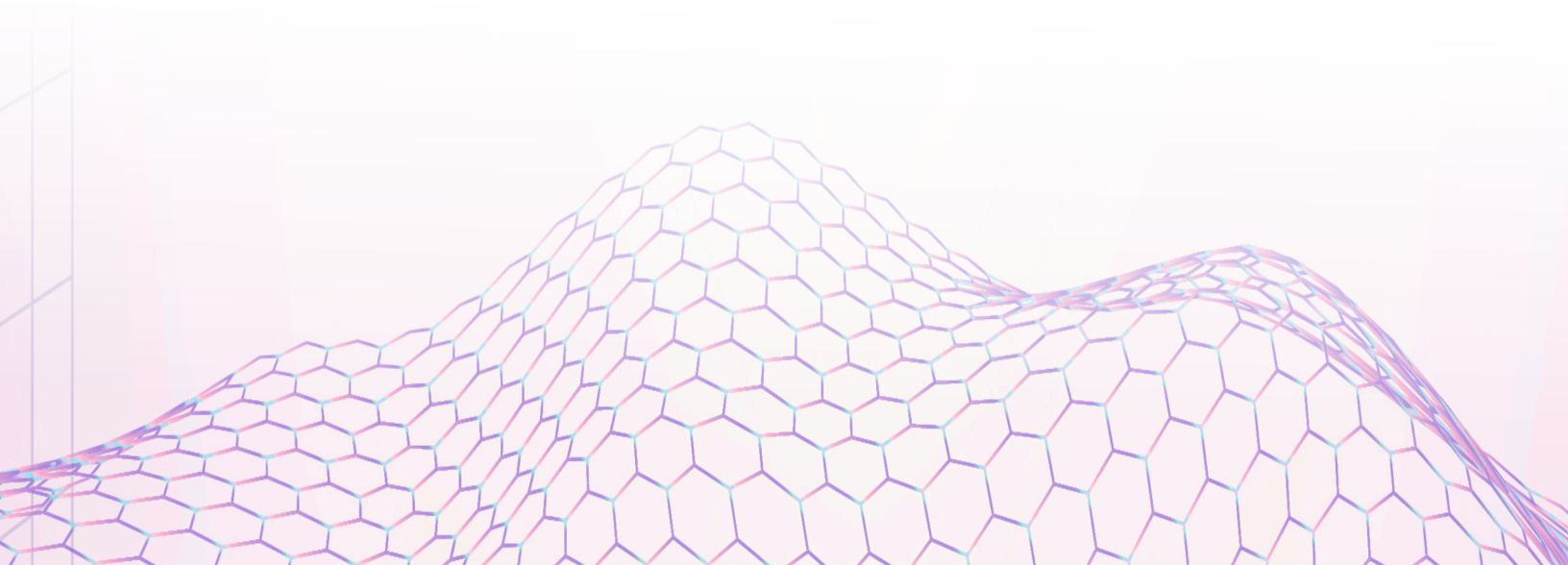
Selector was built for the world that does exist. Its data-centric architecture treats context, enrichment, and correlation as foundational requirements — not optional features. Its horizontal collection model means that growth in telemetry volume or data type diversity doesn't require architectural redesign. Its programmable normalization layer means that business logic and operational context evolve alongside the environment without manual intervention. Its AI pipeline produces deterministic, explainable insights that operations teams can genuinely trust. And its collaboration interface means that the intelligence generated across the platform reaches engineers in the tools where they already work, in a form they can immediately use.

Stop Reacting. Start Anticipating

See how Selector helps network operations teams cut through noise, pinpoint root cause faster, and build a trusted foundation for AI-driven automation.

Request a Demo

www.selector.ai | sales@selector.ai

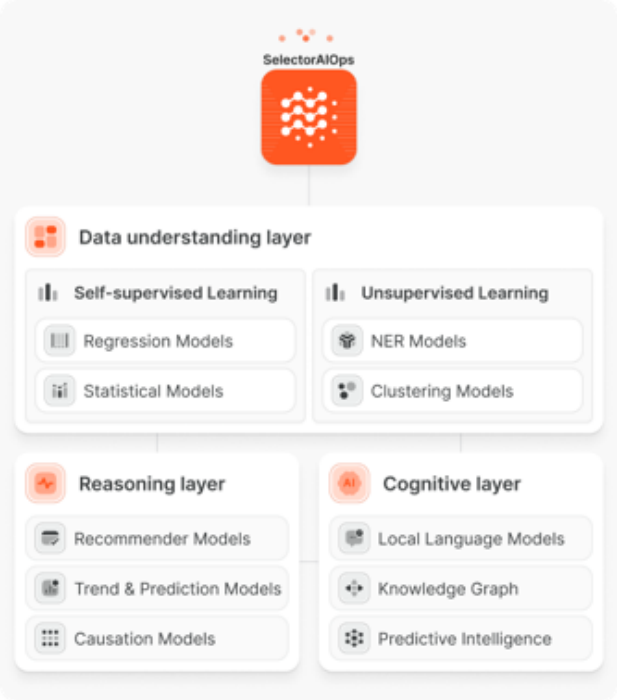


Appendix A: Inside the AI/ML Pipeline

How Selector's multi-layered model architecture transforms raw telemetry into operational insight

Selector's AI and ML architecture is what distinguishes the platform from conventional monitoring and AIOps tools. Rather than applying a single general-purpose model to raw telemetry, Selector uses a layered approach that mirrors how experienced network engineers actually diagnose problems: starting with data, moving through signal detection, building contextual relationships, establishing causation, and finally presenting findings in a form that operators can immediately understand and act on.

Each stage is purpose-built for its role in the pipeline. Together they form a system that can ingest millions of signals per minute, reduce them to a small set of high-confidence causal findings, and explain each one in plain language with full traceability to the underlying data.



Model Types Across the Pipeline

The AI pipeline draws on a purpose-built set of model types, each selected for its specific role in transforming data into operational insight:

Layer	Model Types	Role in Pipeline
Data Understanding	Self-supervised Learning, Unsupervised Learning, Regression Models, Statistical Models, NER Models, Clustering Models	Establishes baselines, detects anomalies, clusters log patterns, and extracts structured entities from unstructured data
Reasoning	Recommender Models, Trend & Prediction Models, Causation Models	Evaluates relationships between signals, ranks likely causes, and models how events propagate across infrastructure
Cognitive	Local Language Models, Knowledge Graph, Predictive Intelligence	Maintains a continuously updated model of the environment, predicts future behavior, and translates findings into operator-ready narratives

The Five-Stage AI Architecture

Data Ingestion and Normalization

Enterprise networks generate millions of metrics and logs per minute from sources that span vendors, protocols, and generations of infrastructure. Selector's foundational layer is a high-throughput data pipeline capable of ingesting from virtually any source. During deployment, Selector works closely with customers to understand their environment and normalize incoming data into a standardized representation aligned with their operational vocabulary — ensuring that AI models reason from consistent, meaningful inputs from day one.

Signal Detection and Self-Supervised Learning

The next challenge is separating meaningful signals from overwhelming noise. For metrics, Selector applies self-supervised learning to create a lightweight model for every time series, continuously learning normal behavior — retraining every 30 minutes on recent data — and enabling accurate anomaly detection even as network conditions evolve. For logs, Small Language Models (SLMs) and NLP interpret log semantics rather than relying on static pattern matching. This allows the system to understand context — distinguishing a BGP flap from a temperature alert, for example — and eliminate up to 99.9% of irrelevant noise before signals reach the correlation layer.

Reasoning and Correlation

After reducing millions of inputs to a manageable set of anomalies, Selector applies a reasoning model that builds contextual embeddings and evaluates relationships across two dimensions: spatial similarity (shared devices, interfaces, IPs, or topology relationships) and temporal similarity (events occurring within the same time window). These relationships are used to group anomalies into coherent clusters — or episodes — forming a structured graph of network activity that captures how events relate across the full stack.

Causation Modeling

Correlation alone is insufficient for reliable operations. Selector's causation model incorporates network topology and domain logic to determine true cause-and-effect relationships within each episode. For example, a link failure causes a BGP session flap; a device reboot causes the link failure. This hierarchical reasoning ensures operators are directed to the root cause rather than chasing downstream symptoms — and that any automated response addresses the actual problem, not a secondary effect.

Presentation and Large Language Models

Raw graph outputs are not actionable for operators. Selector uses Large Language Models to translate deterministic insights into structured, human-readable narratives. Critically, Selector is LLM-agnostic: core intelligence resides entirely in the underlying AI models, and LLMs are used strictly for presentation and summarization — not for reasoning. Each incident includes full explainability and traceability, allowing engineers to drill down into the underlying data and reasoning at any level of detail.