Outcome Owl

See Clearly. Act Decisively.

Executive Briefing Book

Business Observability for your Organization

Your IT team monitors every server. Who monitors the business processes your customers actually experience?

Reading time: 10 minutes

Introduction

For Operations Leaders: This briefing explains why the gap between IT monitoring and business-process visibility is widening, how Outcome Owl closes it, and what getting started looks like.

The Observability Gap in Business Operations

Ask your team a simple question: what percentage of your end-to-end business processes can you see right now — from the customer's first interaction to the final outcome? Most organizations cannot answer this.

Your IT team knows the moment a server's CPU spikes. Your network operations center reports API response times to the millisecond. But ask where a specific customer's order, claim, or application is right now — across every system it touches — and silence follows.

This is the observability gap. The tools that transformed IT operations over the past decade (real-time telemetry, event-driven alerting, distributed tracing) have never been applied to the business processes your customers actually experience. Each department tracks its own slice. Each system reports its own status. The end-to-end process — the thing your customer is waiting on — belongs to no one.

The consequences show up every day:

- **An order stalls between picking and shipping.** Every system reports green. The customer waits two days before calling. The cost: expedited shipping, a service recovery credit, and a customer who now checks your competitor's prices.
- **A homeowner's insurance claim sits in adjudication for three weeks.** The dashboard shows "average cycle time looks fine" because the metric is an average. The cost: regulatory exposure and a policyholder filing a complaint with the state insurance commissioner.
- **A mortgage application stops dead between credit check and underwriting** because the borrower uploaded a document to the wrong portal. No one notices until the rate lock is about to expire. The cost: a rate-lock extension fee, re-disclosure, and a borrower who tells their realtor to stop referring your bank.

- **A telecom ships 10,000 modems in a quarter** and cannot answer how many are actually installed, because the answer lives in four systems with four owners. The cost: phantom inventory on the balance sheet, field technicians dispatched to customers who already self-installed, and revenue recognition delays.

These are not edge cases. They are the everyday reality of multi-system business operations. Each one carries a direct cost — in expedited recovery, manual investigation, delayed revenue, compliance exposure, and customers who leave without telling you why. Most organizations cannot quantify the total cost because the visibility required to measure it is the same visibility they lack.

Outcome Owl closes this gap.

The Forces Reshaping Operations

This gap is not new. What is new is the set of forces making it more urgent and more costly to ignore.

Operational complexity is accelerating, not stabilizing. Distributed workforces, hybrid supply chains, and multi-vendor ecosystems have made business processes span more systems, more geographies, and more handoff points than at any point in the past decade. Most organizations assumed this complexity was temporary. It is now the permanent operating model, and it continues to compound.

AI and automation are creating processes nobody is watching. Every new AI workflow, every new system integration, every new automation connecting tools that never talked to each other creates another end-to-end process that crosses departmental boundaries. The number of processes is growing faster than the visibility into them.

End customers now expect real-time accountability. B2B buyers increasingly expect the same transparency they experience as consumers: real-time status, proactive notification, and fast resolution. “We are looking into it” is no longer acceptable when the customer can track a package across continents but cannot get a status update on their insurance claim, their loan application, or their equipment order.

Regulatory and compliance requirements are tightening. Financial services, healthcare, and insurance regulators increasingly require demonstrable process controls, audit trails, and SLA compliance reporting. Organizations that rely on manual tracking or periodic spreadsheet reviews face growing audit exposure. The question is shifting from “can you describe your process?” to “can you prove your process is working as designed, right now?”

These pressures do not arrive one at a time. They compound. An organization adopting new AI integrations while facing tighter compliance scrutiny and rising customer expectations has a visibility problem that grows on three fronts simultaneously. The cost of the gap increases with every new system connected, every new regulation enacted, and every customer interaction where “we will get back to you” is the best available answer.

What Business Observability Means

Business Observability is a new operational capability, not a repackaging of existing tools.

It is not Application Performance Monitoring. Those tools tell your IT team whether servers are healthy, not whether your customer's order is stuck.

It is not Process Mining. Those tools reconstruct flows from structured event logs, but require clean data and a modeling phase before value.

It is not Business Intelligence. Those tools report on aggregated metrics, not individual process instances moving through your operations right now.

It is not Data Observability. Those tools monitor whether your data pipelines are healthy, not what the data says about your business processes.

Business Observability is something different:

> The ability to capture, monitor, and analyze end-to-end business processes in near-real-time, using the same event-driven observability principles that transformed IT operations, applied to the business processes your customers actually experience.

The core idea is straightforward. Every meaningful event in a business process (an order placed, a claim submitted, a loan application advancing) is captured as an **Observation**. Observations are linked into **Process Chains** that represent the complete end-to-end journey. Business rules detect when something goes wrong. Analytics reveal patterns that no one was looking for.

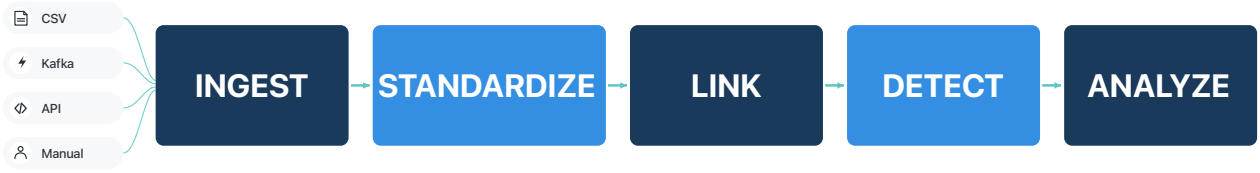
An Observation is not just a timestamp and a status. It carries the business context your operations team cares about: properties like Order ID, Customer ID, Customer Type, Channel, Team Name, Transaction Amount, Source System, or any other data your source systems can provide. The structure is flexible: you decide which properties matter for each process step, and the platform stores them without requiring schema changes or custom development.

The simplest linkage mechanism is a **shared identifier**, a golden thread like a Purchase Order ID or Claim Number that every system includes in its events. But processes do not always work that way. A claim spawns an adjudication case, which spawns a payment, each with its own natural identifier. Outcome Owl handles this through parent-child linkage: each step references the step that preceded it, and the platform traces the chain back to the originating event automatically. The end-to-end journey is assembled from the chain of references, not from a single identifier that every system must carry.

This means the data Outcome Owl collects becomes a valuable asset beyond observability. Teams across Care, Fraud, Finance, and Compliance can explore this consolidated, cross-system data through the platform's built-in analytics (Flow Explorer, Attribute Insights, and Attribute Drift) without writing queries or building dashboards. For teams that want to go deeper with their own tools (Tableau, Power BI, Python scripts, or AI models), an optional read-only analytics replica database makes the same data available

through defined views — one dataset assembled for operational monitoring, reused for many analytical purposes.

Outcome Owl supports two observation modes. **Process observations** track sequential workflows where steps follow a defined order: an order flows from placement to fulfillment, a claim moves from intake to payment. **Audit Trail observations** track standalone lifecycle events where the sequence is fluid and the timeline spans months or years. An asset is received, installed, serviced, and eventually retired, with events arriving from different systems over the asset's operational life. Both modes feed the same rules engine, the same analytics, and the same dashboards.



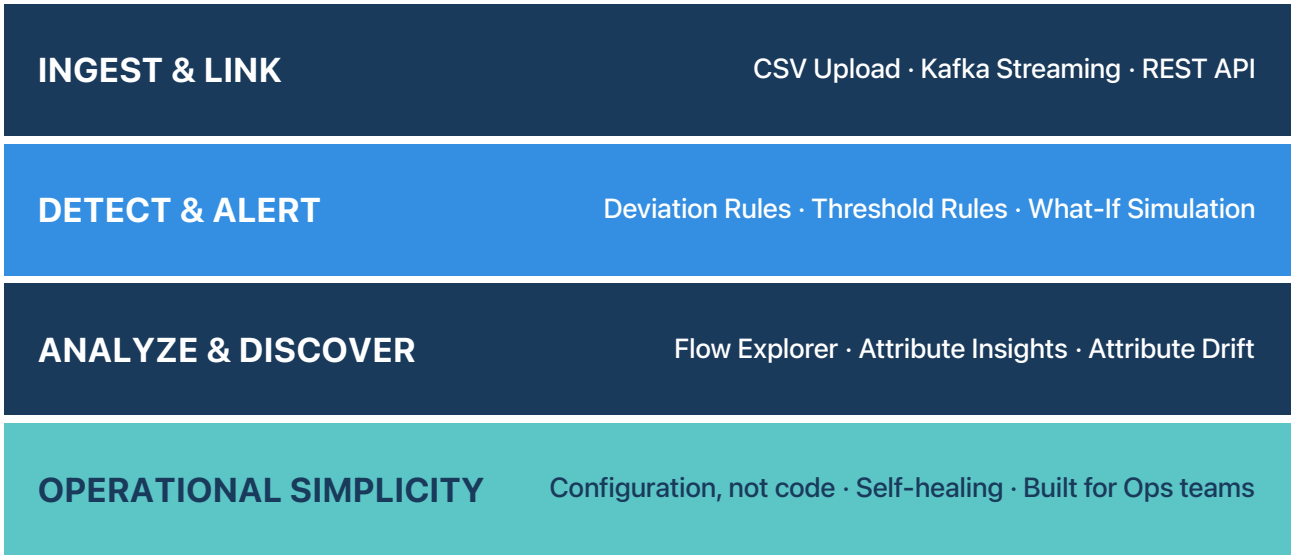
Three design principles make this work in the real world:

Source-agnostic ingestion. Business processes do not live in a single system. An order touches the e-commerce platform, the warehouse management system, the shipping carrier, and the delivery tracker. Outcome Owl accepts data from any source (CSV files, streaming event queues, REST APIs) and normalizes it into a common model. The principle is simple: **if your system can produce a data file or send a message, Outcome Owl can monitor it.** Any system that can emit an event, whether it is a modern cloud platform publishing to a REST API or a legacy mainframe producing a nightly flat file, can feed Outcome Owl. The platform does not require deep integration, proprietary agents, or vendor-specific protocols.

Out-of-order event resolution. In distributed operations, events do not arrive in sequence. A shipping confirmation may reach the system before the picking confirmation. A child event may arrive before its parent. Outcome Owl handles this natively, linking events into the correct process chain regardless of arrival order.

Graceful data quality handling. Bad data is a fact of operational life. Outcome Owl never silently rejects data. Every record is accepted into a visible staging layer, errors are surfaced with specific descriptions, and records are automatically re-evaluated on each processing cycle — so when the upstream system corrects the issue, the data flows through without manual intervention.

The Platform



Ingest & Link

Every event enters through a two-layer quality model. The staging layer accepts everything (valid data, incomplete data, even malformed data) using flexible text columns that never reject a record. A validation process evaluates each record against your configured business rules, promoting qualifying records to the production layer while keeping errors visible and correctable.

The platform automatically discovers and links events into end-to-end process chains, even when they arrive out of order or days apart. Historical data spanning months or years can be loaded in bulk and linked retroactively. Duplicate and updated observations are recognized by their unique identifier, merged gracefully, and everything downstream (process chain linkages, business rules, metrics, and analytics) is reprocessed automatically.

Your source systems do not need to be perfect. They just need to send what they have.

Four ingestion paths ensure you can connect any data source:

- **CSV Upload** — bulk data loads, historical backfills, and scheduled extracts. Files are uploaded through the UI or directly to cloud storage and processed automatically.
- **Kafka Streaming** — near-real-time event ingestion for organizations with existing event-driven architectures.
- **REST API** — application-to-application integration. A dedicated ingestion API lets your systems submit, correct, and delete observations in real time, with the same staging and validation as any other path.
- **UI Manual Entry** — corrections, edge cases, and workflows where human judgment is part of the process.

Detect & Alert

Outcome Owl's rules engine speaks the language of business operations, not technical thresholds.

Deviation Rules define time-based expectations between process steps. "If an order is placed, a quality check should follow within two business days." "If a claim enters adjudication, a decision should be reached within five days." When reality diverges from expectation, the system detects it, whether the expected step is late, missing, or out of sequence. Rules can specify the expected relationship (follows or precedes), whether the related step must exist, maximum allowed duration, severity level, and active date ranges that evolve as your business evolves.

Threshold Rules define numeric boundaries on any data property. "Flag if the dollar amount exceeds $10,000." "Flag if more than five credits post to the same account in 30 days." Thresholds support four comparison types (direct comparison, count, difference, and total) at three scopes: across all observations, within a single observation, or between two related activities. Each rule carries a severity level and an active date range.

Together, these rule types cover the vast majority of operational exception patterns: things that take too long, things that should have happened but did not, things that exceed acceptable boundaries, and things that happen too frequently.

Automatic performance regression detection. Beyond the rules you configure, the platform continuously monitors every process step for performance changes. If approval times double over a week, or a step that has been running smoothly starts consistently taking longer, the system surfaces it automatically — before the slowdown compounds into missed SLAs or customer complaints. You do not need to know which step to watch. The platform watches all of them.

Stalled process detection. Every originating process step has a configurable stalled threshold — the number of hours of inactivity before the platform flags a process as stalled. You set the threshold per process type to match the natural cadence of each workflow: 48 hours for returns, three months for long-lived asset tracking. The platform evaluates every active process against its specific threshold and surfaces stalled processes on the Process Explorer page, the Business Impact breakdown, and the Dashboard.

What If? simulation. Before changing a deviation rule, you can preview the impact. Select an existing rule, propose a different time window ("What if we changed 'quality check within 2 days' to 3 days?"), and the platform simulates how many violations would appear or disappear across your historical data. The simulation runs against real observations without modifying anything. Your operations team can tune rules with confidence, knowing the downstream effect before committing the change.

Analyze & Discover

Beyond rule-based detection, Outcome Owl provides discovery capabilities that reveal what you were not looking for, explorers for drilling into any individual process instance or entity lifecycle, and a Business Impact view that connects every finding to the concern areas your organization cares about.

"Where is my order?" Process Explorer lets you search by any identifier, locate the full end-to-end chain, and see every step on a visual timeline with duration and status. When two similar processes take different paths, a side-by-side comparison shows exactly where they diverged.

"What paths are my processes actually following?" Flow Explorer maps the actual paths your processes take in production — not what your process diagrams say should happen. It discovers every observed sequence, calculates volume and timing at each transition, and surfaces new variants automatically.

"What happened to this asset?" Audit Trail Explorer displays the complete lifecycle of long-lived entities (assets, contracts, certifications) as a chronological timeline with every attribute change surfaced inline. No query, no spreadsheet, no manual comparison.

"Why are some processes failing more than others?" Attribute Insights identifies which data attributes correlate with bad outcomes. If orders from a particular warehouse are disproportionately likely to produce deviations, it surfaces that correlation automatically, expressed as a lift ratio with statistical guards to filter out noise.

"What changed in my data that nobody told me about?" Attribute Drift detects when the shape of your data shifts: a new value appears, an expected value disappears, a frequency distribution moves beyond normal variation. These are early warning signals of upstream system changes or process modifications that no one communicated downstream.

"Which business areas need attention right now?" Business Impact connects findings to seven concern areas (Compliance, Customer Experience, Finance, Operations, Reputation, Revenue, Security) and displays a heat map graded Clear through Critical. A single low-priority item does not trigger alarm, but a pattern of high-severity issues across revenue-critical steps stands out immediately. One click drills to the specific observations driving the number.

Daily metrics across six categories (observation volume, processing status, deviations, thresholds, duration, and error aging) provide operational KPIs out of the box, without requiring anyone to build a dashboard.

Operational Simplicity

Outcome Owl is designed to be run by an operations team, not a dedicated engineering team.

Configuration, not code. Defining process steps, creating business rules, adjusting thresholds, managing users, and tuning system parameters: all of this happens through the platform's administration interface. No development projects. No sprint backlogs. No code deployments. When your business changes, your operations team updates the configuration and the platform adapts immediately.

Resilient by design. Processing jobs are built to recover from interruptions without manual intervention. Every job is idempotent: running it twice produces the same result as running it once. There is no corrupt state to diagnose, no manual recovery procedure to follow, and no data to reconstruct.

Global by design. Each user sets their own date format, time format, and time zone in their profile. The platform stores all timestamps in UTC for forensic consistency and translates them to each viewer's preferences when they appear in the UI.

Linkage guardrails. Configurable relationship constraints prevent observations from being linked into incorrect process chains. If a data error causes an observation to reference the wrong type of parent, the system flags it rather than silently building a false chain. Your operations team defines which process steps are valid parents for each step, and the platform enforces those relationships automatically.

Built for small teams. The platform does not require a database administrator, a dedicated DevOps engineer, or a team of developers to operate. Scheduled jobs run automatically. Alerts surface problems. The operational overhead is proportional to the complexity of your business processes, not the complexity of the platform.

Integration Points

OpenTelemetry compatibility. If your engineering organization already produces OpenTelemetry traces (and more of them do every year), Outcome Owl can ingest them as business observations. OTel Spans map directly to Observations. OTel Traces map to Process Chains. This means your existing instrumentation investment can serve double duty: engineering teams monitor technical performance through their APM tools, while operations teams monitor business process health through Outcome Owl, from the same underlying data.

Customer analytics access. For organizations that want to run their own queries (whether through Tableau, Power BI, custom Python scripts, or AI/ML models), Outcome Owl offers an optional read-only analytics replica database. Your analytics tools connect to a defined set of read-only views on the analytics replica database, never touching the primary transactional database.

How We Are Different

Business Observability, as a purpose-built capability for monitoring end-to-end business processes across heterogeneous systems, is a new category. There is no established product that does exactly what Outcome Owl does. The market is approaching from adjacent directions:

- **APM vendors extending upward.** Splunk, Dynatrace, and New Relic have added business process monitoring, but they still require instrumented application environments. They cannot accept a CSV file, a Kafka stream, or a manual entry in the same process chain.
- **Process Mining vendors broadening their reach.** Celonis and similar tools have expanded into real-time streaming and AI-assisted discovery, but they still require structured event data and a modeling phase before value is delivered. Enterprise deployments start at six figures per year.
- **Data Observability vendors monitoring pipeline health.** Monte Carlo, Sifflet, and Acceldata ensure your data infrastructure is reliable, but they do not monitor the business processes the data represents.

Each of these categories solves a real problem. None of them solves the problem Outcome Owl is designed for: giving operations leaders visibility into business processes that span multiple systems, accept data in any format, and need to be monitored end to end.

Competitive Positioning

Dimension	Process Mining	APM	Data Observability	Enterprise Workflow Platforms	BI Platforms	Outcome Owl
Primary entity	Case (ERP event)	Span / trace	Dataset / pipeline	Ticket / incident / case (varies by module)	Row / measure	Observation (any business event)
What it monitors	Process flows via automated discovery and conformance checking	Application and infrastructure health	Data pipeline freshness, schema, volume	IT, HR, customer service, security, and field service workflows; includes in-platform process mining	Whatever you build a dashboard for	End-to-end business processes across heterogeneous systems
Process model	Automated discovery + optional conformance checking	N/A	N/A	Predefined workflows + process mining with automated discovery	N/A	**Dynamically discovered from data — no data modeling prerequisite**
Data source	Event logs from ERP, CRM, databases, Kafka, APIs, data lakes	Application instrumentation	Data warehouses, lakes, pipelines	IntegrationHub (220+ connectors), imported audit/ event logs, REST/ SOAP APIs	Any (you build the pipeline)	Any (CSV, Kafka, API, manual) — source-agnostic
Time scale	Retrospective + near-real-time streaming	Real-time (milliseconds)	Near-real-time (pipeline runs)	Reactive to near-real-time (varies by module)	Retrospective	Near-real-time to historical
Primary users	Process analysts, operations, and finance teams	Site reliability engineers	Data engineers	IT, HR, customer service, security, and operations teams	Data analysts	Operations leaders
Rules engine	Conformance checking	Alert thresholds (technical)	Anomaly detection on data health	SLA management, workflow automation, predictive intelligence, AI agents	None (visualization only)	Business deviation + threshold rules
Data quality handling	Requires structured, well-formed event data	N/A	Monitors data quality (not business data)	Requires structured data conforming to platform schema	Assumes clean data	Two-layer staging; errors visible and correctable
Deployment model	Multi-tenant SaaS	Multi-tenant SaaS	Multi-tenant SaaS	Multi-instance SaaS (dedicated database per customer)	Multi-tenant SaaS	**Single-tenant dedicated infrastructure**

Industry Use Cases

The opening section described the observability gap through four industry scenarios. Here is how Outcome Owl addresses each one, and the kinds of insights it surfaces.

Order Fulfillment and Supply Chain

Each order event flows into Outcome Owl as an Observation through CSV extract or API integration. The system links them into an end-to-end process chain automatically, even when carrier confirmation arrives before the warehouse system reports the pack-complete event. Deviation rules flag any order where shipping confirmation does not follow picking within four hours. Threshold rules flag high-value orders above $5,000 that have not received manager approval. Daily metrics show fulfillment volume and cycle time by warehouse, by carrier, by product category.

> **In Practice:** Flow Explorer reveals that orders from Warehouse TX follow a different packing path that routes through a secondary quality check not present in Warehouse CA's process, and that TX orders take 2.3x longer to reach carrier handoff. Separately, Attribute Insights shows 'carrier = Regional Express' has a 4.2x lift for deviation outcomes, suggesting that carrier choice is an independent contributing factor worth investigating.

Claims Processing and Insurance

Claim events flow in via nightly CSV extract from the claims management system or API calls at each status change. Deviation rules detect claims where adjudication does not begin within 48 hours of intake completion. Threshold rules flag claims where the reserve amount exceeds jurisdictional thresholds. Attribute Drift detects when a new claim category starts appearing in the data, an early warning of market shifts or upstream system changes. Daily metrics track claims volume, aging, and cycle time, broken down by any attribute in the data.

> **In Practice:** Attribute Insights shows 'category = Water Damage' has a 4.1x lift for adjudication SLA deviations, far above any other claim category. Independently, 'region = Southeast' and 'adjuster_office = Tampa' also show elevated lift. The combination of these independent signals points the operations team toward a specific geography and office for investigation. Adjusting the date range filter reveals the pattern emerged three weeks ago, coinciding with a staffing change that was not communicated to operations leadership.

Loan Origination and Financial Services

Each loan event flows in as an Observation. The Process Chain model links every step of the origination workflow into a single end-to-end journey. Threshold rules flag applications where the credit-check-to-underwriting handoff exceeds three business days. Deviation rules flag applications where the appraisal has not been ordered within two days of underwriting approval. When underwriting policy changes, the Restatement pipeline re-evaluates all historical applications against the updated rules, providing immediate visibility into how the change would have affected past performance.

In Practice: Attribute Drift has detected that 'loan_type = FHA' applications have increased 40% over the past two weeks, flagged as a Frequency Change — an early signal that the applicant mix is shifting. Separately, Flow Explorer shows elevated duration between document verification and appraisal scheduling across all applications, suggesting a systemic bottleneck at that transition.

Network Equipment Asset Lifecycle

Each asset event is recorded as an Audit Trail observation, a standalone event tied to the asset's lifecycle rather than a sequential workflow step. Outcome Owl links all events for a given asset into a single lifecycle chain, regardless of which system generated the event or when it arrived. Audit Trail Explorer displays the complete asset timeline with attribute change tracking — an operations manager clicks any asset and sees every event from receipt to installation to service, with changes highlighted inline (location changed, firmware updated, technician assigned). Threshold rules flag assets serviced more than three times in twelve months, or assets in "shipped" status for more than 30 days without installation. Attribute Insights reveals which manufacturers, model numbers, or deployment regions correlate with premature failure or excessive service calls.

In Practice: Attribute Insights shows 'manufacturer = Vendor B' has a 3.8x lift for field service deviations within 90 days of installation. Independently, 'model = RX-4200' also shows elevated lift. Together, these signals point the procurement team toward a specific manufacturer and model for investigation.

Security and Compliance

Outcome Owl is built for organizations where data security, access control, and audit readiness are non-negotiable. This section details the technical security posture, the material your CISO, security team, or compliance auditors will want to review.

Single-tenant architecture. Every customer deployment runs on dedicated AWS infrastructure. Database, storage, and compute resources are not shared with any other customer. Isolation is physical, not logical.

Zero footprint in your environment. Nothing is installed in your data centers, on your servers, or in your cloud accounts. Outcome Owl runs entirely on isolated AWS infrastructure managed by Seattle Software Works. For data ingestion, the flow is one-way inbound. Your systems push data out and Outcome Owl receives it. No agents, connectors, or software packages are deployed into your infrastructure. When SSO is configured, the authentication flow involves standard browser-based redirects between Outcome Owl and your identity provider, and for OpenID Connect, a server-to-server token exchange where Outcome Owl calls your IdP's public token endpoint. This is standard OAuth 2.0 behavior and does not require network access into your environment, firewall exceptions, or VPN tunnels.

Network access controls. The UI and API endpoints sit behind an AWS load balancer that can be restricted to specific IP address ranges (corporate offices, VPN, or a defined set of network addresses). This is a standard deployment option, not a custom configuration.

AWS-native security foundation. Infrastructure credentials are managed through IAM Roles attached to EC2 instance profiles. No static access keys, no embedded credentials. Each customer's IAM Role is scoped to that customer's resources only.

Least-privilege database access. Thirteen dedicated database service accounts enforce separation of duties at the database level. Each service account has only the specific table-level grants required for its function and nothing more. The batch job that processes CSV files cannot access user session data. The metrics calculation job cannot modify observation records.

Authentication and identity. SAML 2.0 and OpenID Connect SSO integration with any compliant identity provider (e.g. Okta, Microsoft Entra ID, Ping Identity). Basic authentication can be disabled entirely once SSO is established. Bcrypt password hashing with per-hash salting. Seven configurable password policy parameters (minimum length, complexity, expiration). Automatic account lockout after configurable failed attempts within a rolling time window.

Access control. Fifteen fine-grained security roles separate concerns across API access, intake management, observation management, configuration, AI features, restatement, and administration. Role conflict rules prevent incompatible combinations. Service accounts are restricted to API roles and human users are restricted to UI roles.

Complete audit trail. Every authentication event, configuration change, user account modification, and administrative action is recorded with the identity of the actor, the timestamp, the originating IP address, and the details of what changed. The audit trail is retained according to a configurable retention policy and is accessible through the administration interface and API.

Configurable data retention. Retention periods are independently configurable for audit logs, session records, restatement history, AI prompt history, and intake staging records, from one day to ten years (3,650 days). Automated cleanup runs on a scheduled basis.

Primary database and storage isolation. The primary transactional database is accessed only by the application's own backend services and batch jobs. Customer-initiated queries are served exclusively through an optional read-only analytics replica database with a defined set of views. The primary database is never exposed to direct customer access. File storage (AWS S3) is equally isolated. Each customer deployment uses a dedicated S3 bucket with IAM policies scoped to that customer's resources only. No bucket, prefix, or object is shared across customers.

No secrets in code. Database credentials, JWT signing keys, SSO configuration secrets, and API keys are stored in AWS Secrets Manager and retrieved at runtime.

Business Continuity and Data Portability

Enterprise buyers evaluating any vendor (especially a focused company rather than a large platform vendor) should ask three questions: What happens to my data if I leave? Am I dependent on a shared platform I don't control? Am I locked into a proprietary format? These are fair questions, and the answers should be specific, not reassuring generalities.

Your data is yours, in a standard format. Every customer deployment runs on a dedicated PostgreSQL database — an open, industry-standard relational database engine. Your observations, configurations, rules, and analytics are stored in standard SQL tables, not a proprietary data format. If you leave Outcome Owl, your data can be exported as standard SQL or CSV at any time. There is no proprietary encoding, no format conversion required, and no data held hostage.

Your deployment is fully isolated. Because Outcome Owl uses a single-tenant architecture with dedicated AWS infrastructure per customer, your environment is physically separate from every other customer. There is no shared database, no shared compute, no shared storage. This isolation means your data and operations are unaffected by any other customer's activity, and your deployment can be managed, migrated, or wound down independently.

No vendor lock-in by design. The platform uses standard protocols and formats at every integration point: REST APIs, CSV files, Kafka, SAML 2.0 / OpenID Connect for authentication. Your source systems send data through standard interfaces. Your BI tools connect to a standard PostgreSQL replica through standard database drivers. Nothing in the architecture requires proprietary connectors, agents, or middleware that would bind you to Outcome Owl as the only option.

Many established SaaS vendors operate multi-tenant architectures where your data is interleaved with other customers' data in shared databases and shared compute. In those models, extracting your data requires the vendor's cooperation and proprietary export tools. In Outcome Owl's single-tenant model, extraction is a standard database operation that any DBA can perform.

Getting Started

Outcome Owl is built for rapid time-to-value. You do not need to connect every process across your entire organization to see results. You need one process and the willingness to look at it clearly.

Bring us one process. We will have it running in days or weeks, not months.

Standing up Outcome Owl on dedicated AWS infrastructure is fast, measured in days. The real variable is your data: how quickly your team can begin sending data from your source systems. A CSV export can flow into the platform on day one. API and streaming integrations follow as your engineering team connects them.

The onboarding path is straightforward:

1. **Define your process steps.** Identify the meaningful events in the business process you want to observe. This is a configuration exercise, not a development project. Your operations leaders know these steps already. You do not need to map out expected process flows; the platform discovers the actual flows from your data automatically. Our Process Discovery Worksheet walks your team through the conversation — from "why this process?" and which business areas it impacts, through the individual steps, linkage, and initial rules — so that configuration is a structured exercise, not a blank-page problem.

2. **Connect your data.** If your system can produce an event, Outcome Owl can monitor it. If you can produce a CSV file, you can start immediately. If your systems support streaming event publishing or API calls, we connect those as well. Outcome Owl meets your data infrastructure where it is today.

3. **Define your business rules.** What should happen after each step, and how quickly? What numeric boundaries matter? Outcome Owl codifies these into deviation and threshold rules that the platform evaluates continuously.

From there, the platform works. Process chains form automatically. Rules fire when reality diverges from expectation. Analytical trends and patterns emerge over days and weeks as the platform's statistical baselines build confidence from observed data.

You do not have to start from today. Outcome Owl can ingest and analyze historical data (months or years of it) through the same ingestion paths used for live data. Load your historical records and get the benefit of trend analysis and pattern detection from day one, without waiting weeks for enough live data to accumulate.

Early access, direct partnership. Outcome Owl is entering its initial pilot program with a limited number of organizations. Pilot participants work directly with the product team — the same people who designed and built the platform. Your processes, your data patterns, and your operational priorities shape how the platform evolves. This is not a sales demo followed by a support ticket queue. It is a working partnership where your team's experience directly informs the product.

Enterprise pricing has two components: an annual software license at the organization level (not per seat, not per event, not per process) and a pass-through infrastructure cost based on your actual AWS resource usage. Your entire operations team can access the platform without license management overhead. Infrastructure costs are reviewed on a regular cadence with a clear breakdown, so you always know what you are consuming and why. As your usage grows (adding read-only replicas, AI capabilities, or higher data volumes), the infrastructure component adjusts proportionally while the software license remains predictable.

Ongoing partnership is how we operate. Seattle Software Works is not a platform vendor that disappears after implementation. We work with your operations leadership to expand coverage process by process, refine rules as your business evolves, and ensure the platform delivers sustained operational value.

The observability gap in business operations is real, it is costly, and it is solvable. The tools exist. The architecture is proven. The only question is whether you are ready to see what has been invisible.

See Clearly. Act Decisively.

Outcome Owl is a product of Seattle Software Works, Inc.

Outcome Owl
See Clearly. Act Decisively.