Eliminating The 90% Noise: How Teamvoy Re-Engineered Trade Surveillance Software For A Global Exchange

From Research to Reality: Engineering a Production-Grade AI Co-pilot for Real-time Trade Monitoring

Book a Consultation

SERVICES

Ai/Ml Engineering, Java Backend, R&D-To-Production, Synthetic Data

INDUSTRY

Fintech, Trade Surveillance

CLIENT

Global Stock Exchange & Financial Services Provider

TRUE-POSITIVE RETENTION

100%

CALIBRATION TIME

<30s

mization Engine for High-Frequency Trading Environments. Scaling Intelligence: Building an AI-Native Optimization Engine for High-Frequency Tra

Summary

Teamvoy cut false positive alerts by 60-90% for a leading global stock exchange – without losing a single true positive – by translating experimental research into a production-grade trade surveillance system built in Java.

The client's existing trade surveillance software was generating over 90% false positive alerts, overwhelming investigators and masking real market abuse signals. Their data science team had developed promising optimization algorithms in Python, but the research existed only as notebooks – not deployable, not compliant, not fast enough. No internal team had turned it into a working system.

Teamvoy took ownership of that gap. We re-engineered the core optimization logic into a high-concurrency Java backend, built the surrounding microservices architecture, and delivered a trade surveillance monitoring solution that processes full historical datasets in under 30 seconds. Parameter calibration that previously took compliance teams several weeks now runs automatically overnight.

60–90% False positive reduction	**100%** True positive retention	**<30s** Parameter calibration	**10+** Variables optimized

Key Outcomes And Takeaways

- False positive alerts reduced by 60-90%, with 100% True Positive retention rate maintained throughout.
- Parameter calibration time dropped from several weeks of manual work to under 30 seconds of automated processing.
- Experimental Python notebooks fully re-engineered into a thread-safe, high-concurrency Java system.
- A proprietary benchmarking and synthetic data generation platform built as a reusable asset for future trade surveillance software engagements.
- Complex AI research integrated into live financial infrastructure without compromising regulatory compliance or determinism.

01. Our Client

Global Stock Exchange Operating A High-Volume Trade Surveillance System

Our client is a leading global stock exchange and financial services provider. They operate a high-availability trade surveillance system designed to monitor market activity and detect potential abuse. To maintain market integrity, the system generates real-time alerts based on specific trading signals.

The exchange operates under MiFID II and equivalent cross-border compliance frameworks. Any trade surveillance system handling their alert volume must meet strict standards for determinism, auditability, and reproducibility – requirements that ruled out any off-the-shelf solution and made a bespoke engineering engagement the only viable path.

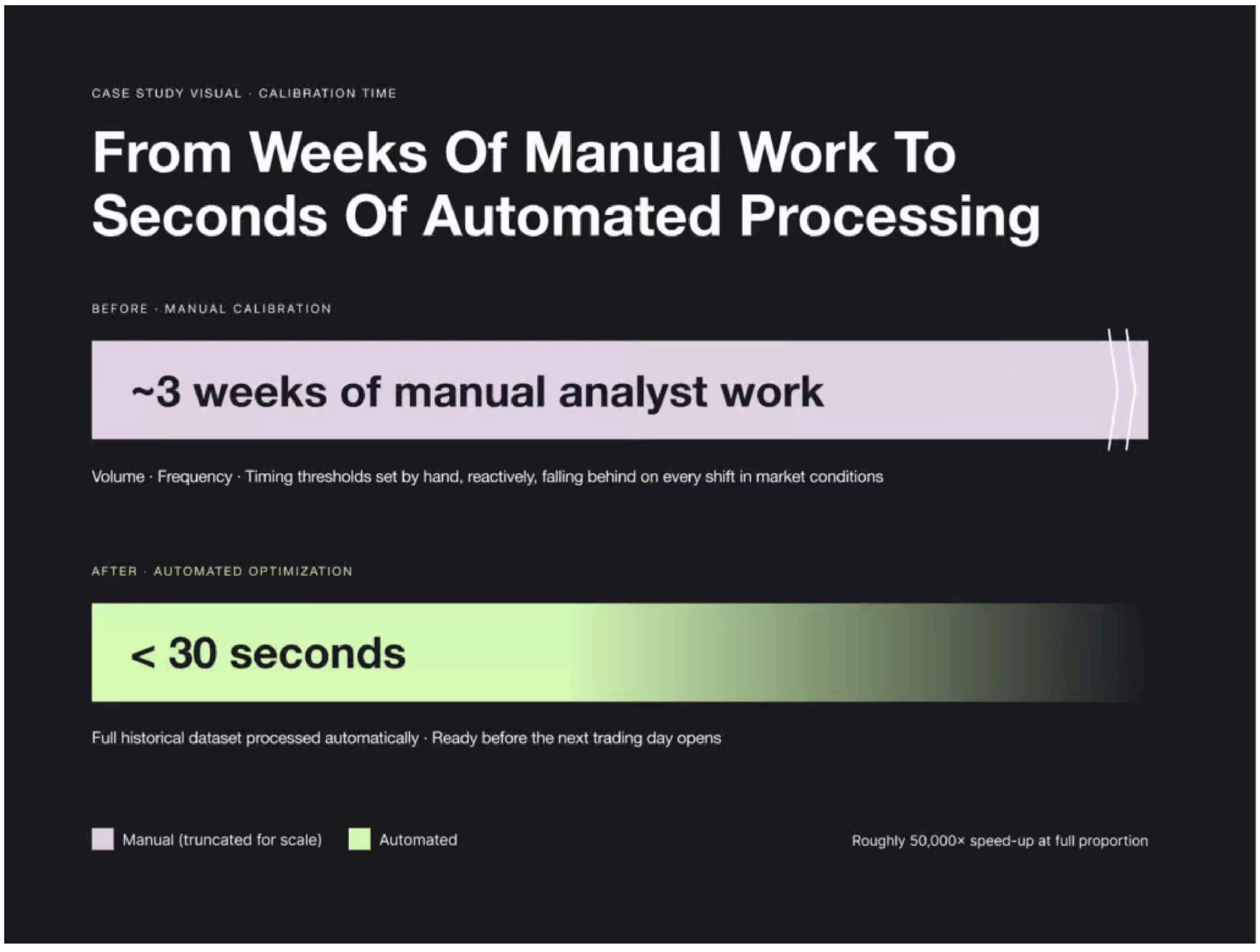
The project focused on building a sophisticated AI Co-pilot – an analytical tool designed to refine alert signals and identify truly suspicious trading behavior with documented, auditable accuracy. As one of the leading trade surveillance software providers in the financial sector, the client required a system built to the highest standards of precision and regulatory reliability.

02. Challenge

90% False Positive Rate Blocking Effective Trade Surveillance Monitoring Solution

When investigators spend most of their day triaging alerts that turn out to be nothing, the system has failed – not technically, but operationally. The exchange's trade surveillance software was generating alerts at scale, but over 90% were false positives. Compliance teams were buried. Real market abuse signals were competing for attention with noise.

The underlying cause was manual parameter calibration. Analysts set thresholds for volume, frequency, and timing by hand – a slow, reactive process that couldn't keep pace with changing market conditions. When trading volumes shifted, calibration fell behind. The only traditional fix was hiring more investigators – more headcount chasing the same noise.



Three Constraints Defined The Problem

- Accuracy: Reduce false positives by 60-90% without dropping a single true positive. Zero tolerance for missed market abuse signals.
- Speed: Parameter calibration had to shift from weeks to near real-time. Compliance teams needed updated thresholds before the next trading day opened.
- Compliance: Any optimization system operating within a regulated exchange must produce deterministic, auditable results – a black box, however accurate, was unacceptable to regulators or to the client's own risk function.

Main Goals

Goal	Metric	Target Objective
Accuracy	False Positives	Reduce by 60-90% with 100% True Positive retention
Automation	Calibration speed	Replace manual calibration with automated AI-driven workflow
Reliability	System confidence	Systematic, data-backed parameter validation

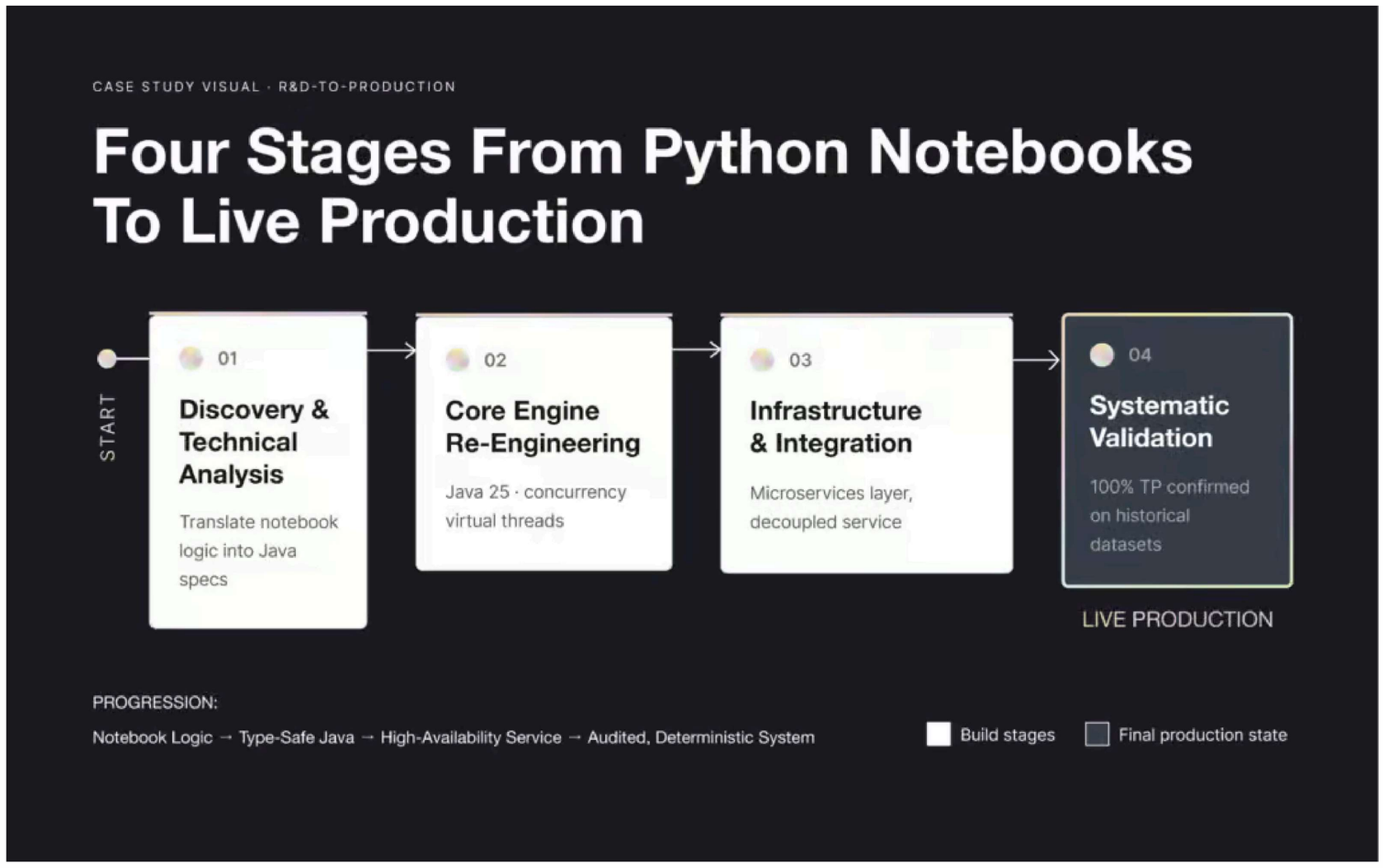
03. Our Approach

R&D-To-Production Re-Engineering For Trade Surveillance Financial Services

The client's ML team had done the hard algorithmic work. They had identified an optimization approach that could reduce false positives significantly – proven in controlled experiments, documented in Python notebooks, and validated against sample datasets. What they didn't have was a path to production.

Python notebooks don't run in high-availability financial infrastructure. They can't handle the concurrency requirements of a live exchange. They don't produce the audit trails that trade surveillance regulations demand. And they can't process the data volumes required for same-day parameter calibration.

Teamvoy's role was to take that research – logic intact – and rebuild it as a system that could actually run. That meant re-engineering the optimization algorithms in Java, designing the microservices layer, and building a benchmarking environment capable of validating every parameter change before it touched production data.



Our Strategic Role

- Algorithmic Re-Engineering: We translated experimental Python research into type-safe Java code. This gave the system the throughput required by a global stock exchange while preserving the algorithmic integrity of the original work.
- Architecture Ownership: We built the integration layer that allowed AI-driven insights to function within the client's existing microservices infrastructure.
- Performance Engineering: By implementing optimization logic directly in the production backend, we eliminated cross-language latency and enabled the engine to handle high-scale trading volumes in near real time.

The Delivery Team

- Production Delivery Lead (Teamvoy): Managed roadmap, business logic prioritization, and the transition from R&D to a market-ready system.
- Backend Engineer (Teamvoy): Solely responsible for re-engineering experimental Python models into a high-concurrency Java environment – microservices design, data pipelines, and core optimization logic.
- ML Advisory Group (Client): Provided the algorithmic blueprints and domain-specific insights for implementation.
- Frontend Team (Client): Handled UI integration for parameter visualization and user workflows.

Stages

Four stages took the project from fragmented Python research to a running Java production system:

Discovery & Technical Analysis. Deep analysis of experimental Python notebooks to translate theoretical algorithmic models into production-ready technical specifications for the Java backend.

Core Engine Re-Engineering. End-to-end re-implementation of optimization algorithms in Java, prioritizing thread-safety, low-latency execution, and large-scale data processing.

Infrastructure & Integration. Decoupled service integration strategy to embed the AI Co-pilot into the client's high-availability microservices infrastructure without disrupting existing alert workflows.

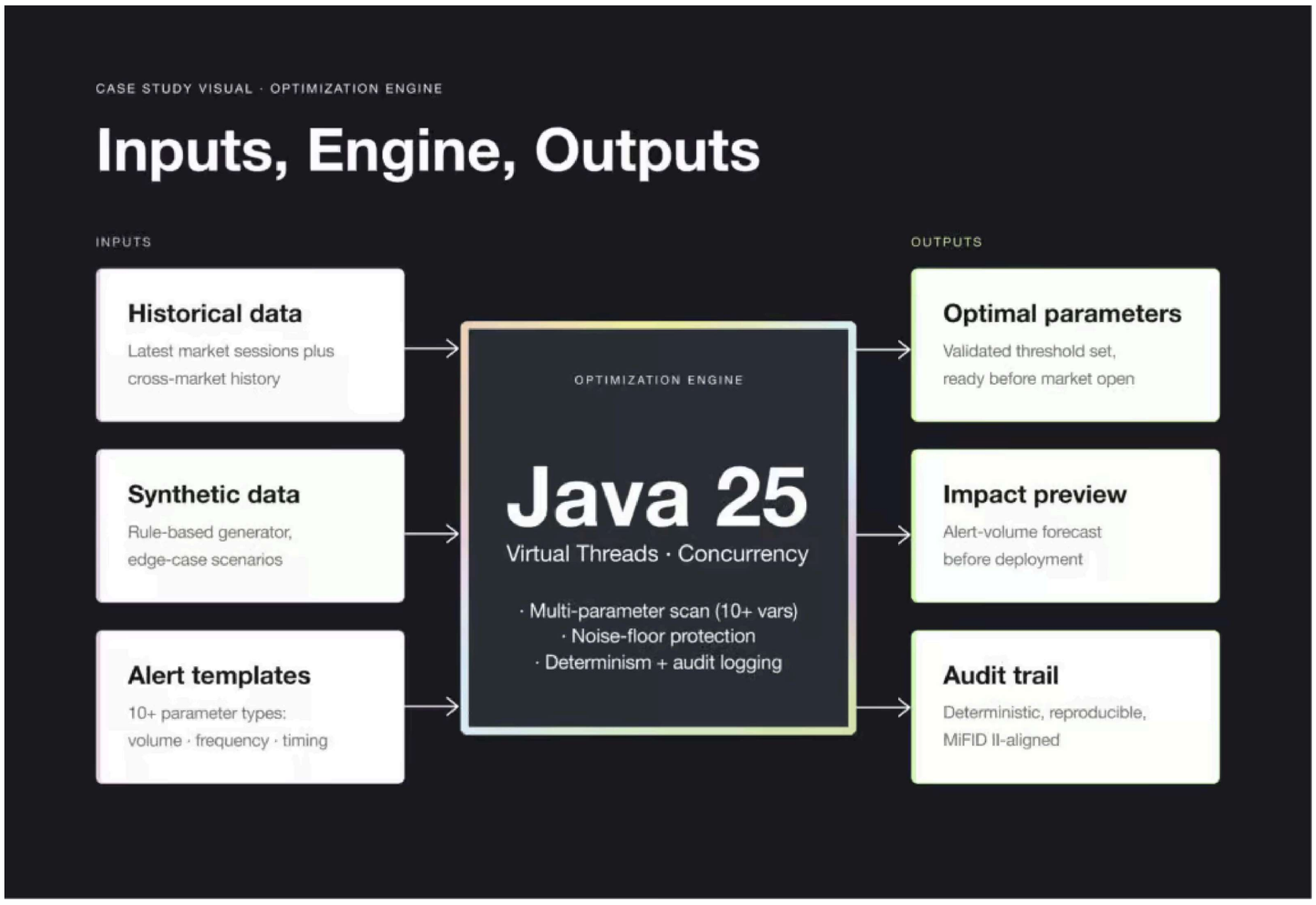
Systematic Validation. Rigorous testing against extensive historical datasets to confirm the 100% True Positive benchmark and verify architectural stability under peak market loads.

04. Solution

AI-Powered Trade Surveillance Software That Cuts Noise Without Missing Threats

The AI Co-pilot is an analytical engine designed to optimize trade surveillance parameters based on historical data. By analyzing specific timeframes, the system identifies the optimal parameter configuration – the exact combination of 1 to 10+ variables – that minimizes false positives without compromising true positive detection.

Built on the Java 25 ecosystem, this trade surveillance monitoring tool processes massive datasets at speed, allowing stock exchanges to refine their monitoring thresholds for upcoming trading days with mathematical certainty.



Key Features

- Predictive Impact Transparency: A clear, data-backed preview of exactly how many alerts will be eliminated by specific parameter sets – so users understand operational impact before deployment.
- Universal Data Flexibility: Runs optimizations across diverse data samples – from the latest market sessions to historical cross-market data – supporting any alert type or trading anomaly pattern.
- Noise Floor Protection: A built-in boundary layer prevents aggressive noise reduction from overlapping with true positive signatures, maintaining 100% integrity of market abuse detection.
- Multi-Parameter Scalability: Simultaneously optimizes 10+ complex variables – a capability made possible by the high-concurrency architecture of the Java backend.

Key Engineering Decisions

1. High-Concurrency & Real-Time Perfomance

To achieve execution times under 30 seconds across massive datasets, we implemented multi-level parallelization using Java 25's advanced concurrency models and virtual threads. This native approach eliminated cross-language overhead and enabled the engine to handle high-scale trading volumes in near real time.

2. Custom Algorithmic Refinement

The initial research provided only a basic algorithmic blueprint. We engineered a more sophisticated optimization engine from scratch, addressing complex edge cases overlooked in the experimental phase – extending standard optimization logic with custom heuristics to keep the engine stable under extreme market volatility.

3. AI-Accelerated Prototyping

We explored multiple algorithmic approaches, including Gradient-based and Evolutionary methods, using AI agents to prototype these alternatives and run high-speed comparative testing. This allowed us to select the most dependable approach without extending the development cycle.

4. Synthetic Data Engineering

Faced with restricted access to live production data, we developed a Synthetic Data Generator using rule-based logic to create diverse experimental environments – simulating different trading volumes, market complexities, and rare edge-case scenarios. This provided a measurable baseline for performance tracking and accelerated the validation cycle.

5. Regulatory-Grade Determinism

In a stock exchange environment, trade surveillance technology must meet mandatory compliance standards. We built a dedicated test suite focused on Robustness, Determinism, and Input Sensitivity, integrated directly into the CI/CD pipeline – so every update maintains strict reproducibility and aligns with MiFID II and global financial regulatory standards.

6. Automated Benchmarking Platform

We built a proprietary benchmarking environment running thousands of automated scenarios, with real-time monitoring of CPU, RAM, and execution time. The platform validates changes instantly and has become a reusable asset – a sandbox for future algorithmic work across trade surveillance software engagements.

60–90% Noise Reduction Across A Production-Grade Trade Surveillance Solution

The result isn't just a faster system – it's a different way of working. Investigators who previously spent most of their shift reviewing false positives now spend it on alerts that matter. Parameter calibration that blocked compliance teams for weeks runs in under 30 seconds. The exchange can update its trade surveillance system overnight and open the next trading day with validated thresholds.The result is a system that sets the benchmark for speed, precision, and regulatory confidence across trade surveillance services.

Business Results

- Operational Noise Reduction: Systematic validation on historical datasets confirmed a 60–90% reduction in false positive alerts, freeing investigators to focus on high-probability threats.
- Rapid Parameter Calibration: Reduced threshold optimization from several weeks of manual effort to under 30 seconds of automated processing.
- Zero-Risk Filtering: A validation layer maintains 100% True Positive retention – aggressive noise reduction never compromises regulatory security.
- Scalable Compliance: A deterministic, benchmarked environment meets the transparency requirements of global financial regulators across compliance trade surveillance operations.
- Infrastructure as Asset: A reusable benchmarking and synthetic data generation platform serves as a foundation for future AI-driven trade surveillance software initiatives.

Overall Impact

For Teamvoy, this project demonstrates a specific capability: taking research that works in theory and making it work in production, inside regulated financial infrastructure, without sacrificing the algorithmic integrity of the original work. The Python-to-Java re-engineering path we established here – with its synthetic data environment, deterministic testing suite, and automated benchmarking platform – is a reusable foundation for future trade-surveillance financial-services engagements.

The client's compliance team has a tool that scales with trading volume, not with headcount. That was the goal from day one.

> *Teamvoy took something we couldn't productionize ourselves and turned it into the system our compliance team now relies on every day. The reduction in noise was immediate – and we haven't missed a single true positive since deployment.*
>
> **– Head of Trade Surveillance, Global Stock Exchange**