Cloud-Native Platform Engineering: Modernizing Fixed-Income Index Systems

A comprehensive examination of how platform engineering and cloud-native technologies transformed a legacy fixed-income index system, delivering substantial performance improvements and operational benefits in a regulated financial environment.

By: **Tarun Chataraju**

Agenda

01

Legacy System Challenges

Identifying critical pain points in the existing fixed-income index infrastructure

02

Modernization Strategy

Platform engineering approach and cloud migration framework

03

Technical Architecture

Kubernetes implementation and microservices design

04

Implementation & Results

Performance improvements, cost reduction, and compliance adherence

05

Lessons Learned

Key takeaways and actionable recommendations for similar transformations

Legacy System Challenges

Significant Downtime

Monthly index rebalancing required 12+ hour maintenance windows, creating critical service disruptions

Data Volume Explosion

5x increase in fixed-income market data volumes over 3 years overwhelmed existing processing capabilities

High Latency

Index calculations requiring 30+ minutes in peak periods, impacting client SLAs and trading capabilities

Limited Scalability

Monolithic architecture constrained horizontal scaling during high-demand periods

The existing architecture couldn't adapt to market demands for real-time data, faster calculation cycles, and increasing security universe coverage.

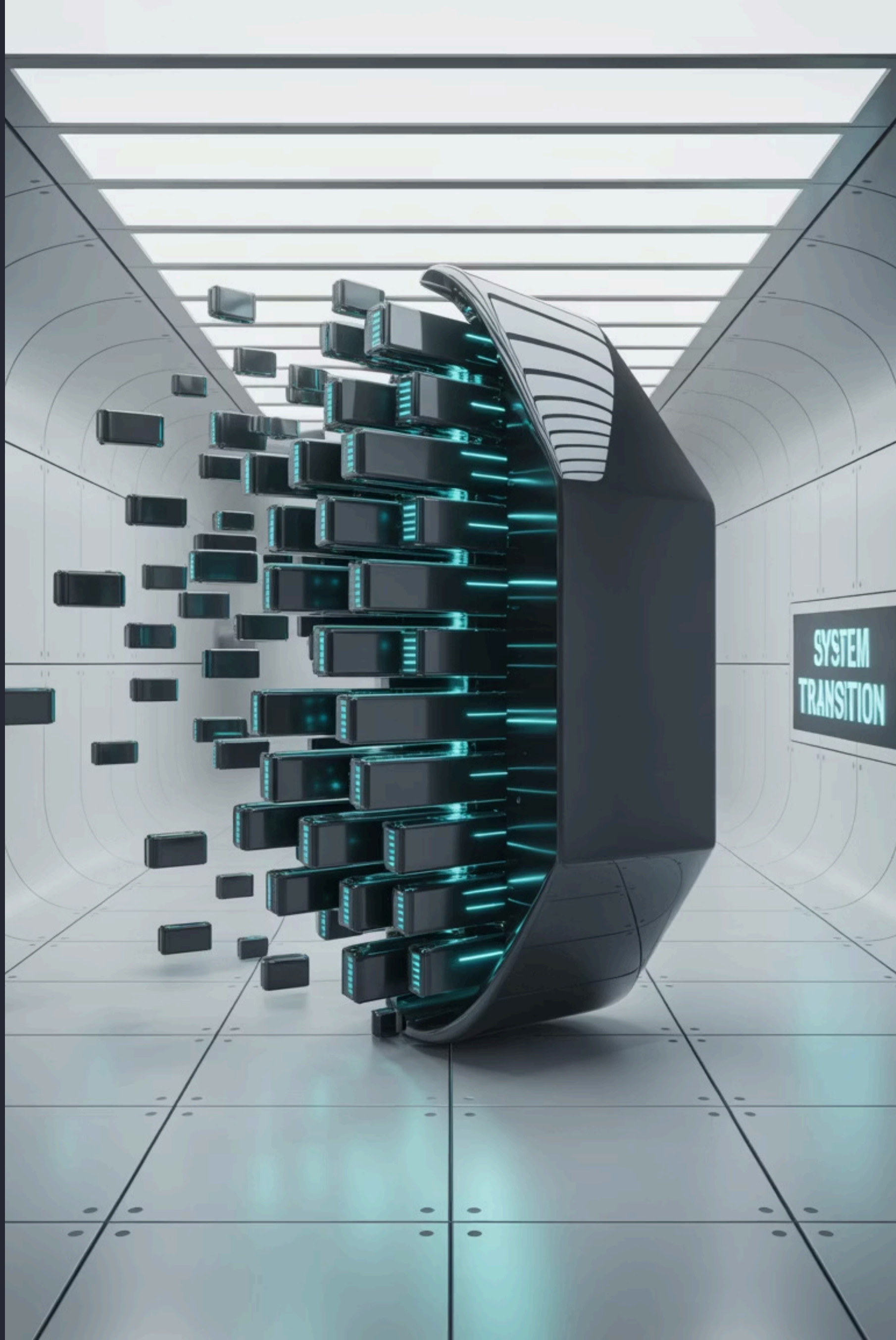
Modernization Strategy

Platform Engineering Approach

- Phased migration maintaining **zero service disruption**
- Infrastructure-as-Code (IaC) for consistent environments
- Microservices architecture with domain-driven design
- CI/CD pipeline implementation with automated testing

Key Strategic Decisions

- Containerization of all system components using Docker
- Implementation of asynchronous processing workflows
- Distributed caching for frequently accessed market data
- Security and governance controls built into deployment pipelines



Technical Architecture Evolution

Before: Monolithic Application

- Single codebase with tight coupling
- Vertical scaling only
- Manual deployment processes
- Limited redundancy
- Batch-oriented processing

After: Cloud-Native Microservices

- Domain-bounded services
- Horizontal auto-scaling
- GitOps deployment model
- Multi-region redundancy
- Event-driven architecture

Kubernetes & Airflow Implementation

Data Ingestion Services

Containerized adapters for multiple data providers with stateless processing

Calculation Engine

Scalable compute pods with auto-scaling based on workload

Data Persistence

Cloud-native time-series databases optimized for financial data

API Layer

RESTful and GraphQL interfaces with built-in compliance controls

Service mesh implementation enables advanced traffic management, observability, and security policies across all microservices. Airflow orchestrated the workflow end-to-end.

Critical System Components

Request Queuing System

Implemented Apache Kafka for message brokering, enabling:

- Asynchronous processing of calculation requests
- Peak load management without service degradation
- Message persistence for system recovery
- Event sourcing for accurate system state reconstruction

Distributed Caching

Redis cluster implementation providing:

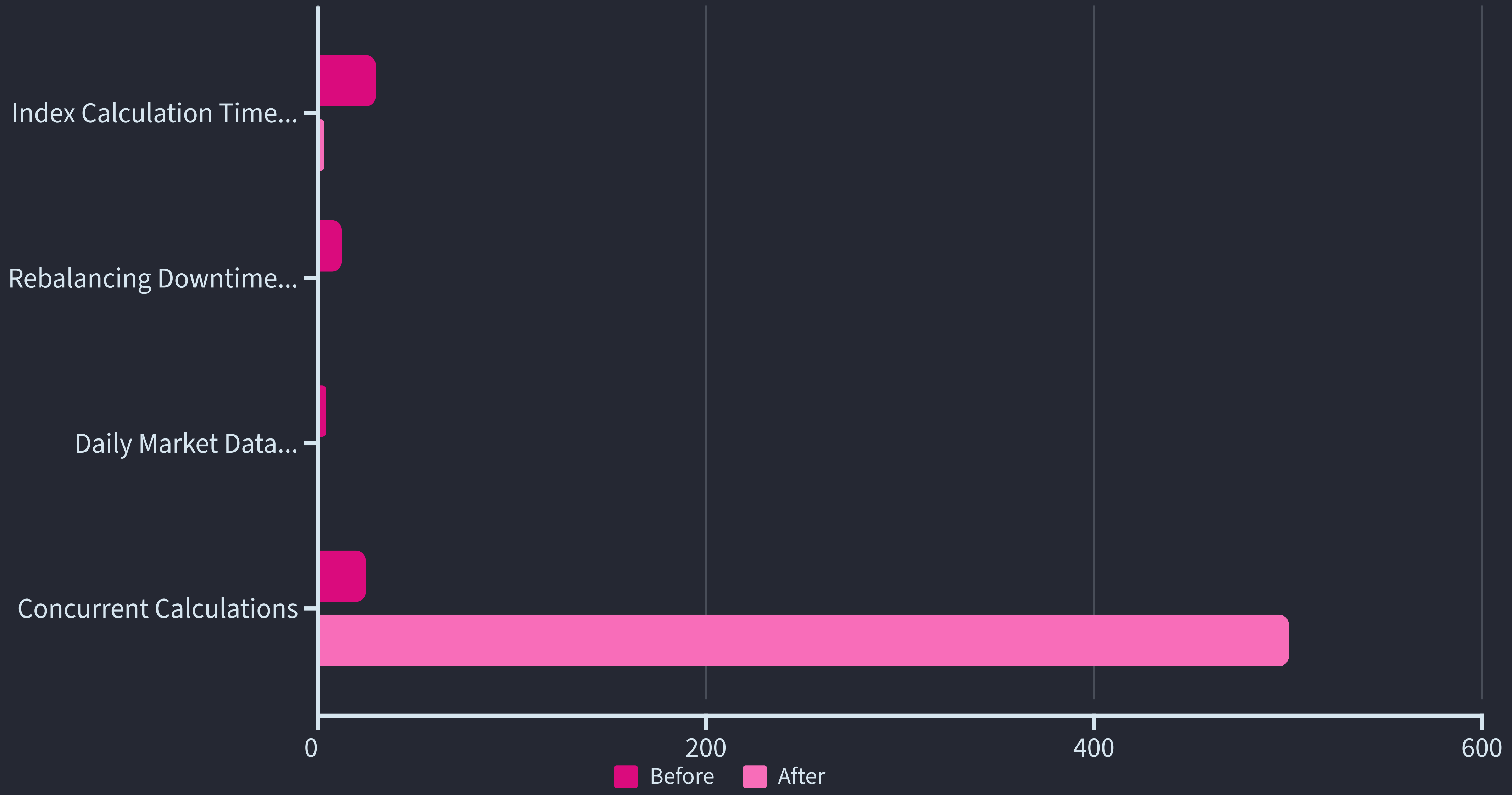
- 90% reduction in database read operations
- Sub-millisecond access to frequently used market data
- Cross-region data replication
- Failure resilience with automatic failover

Container Orchestration

Kubernetes features leveraged:

- Horizontal Pod Autoscaler for dynamic scaling
- Custom Resource Definitions for financial domain objects
- Network policies enforcing regulatory boundaries
- StatefulSets for ordered deployment and scaling

Performance Improvements



The modernized architecture delivered an 88% reduction in index calculation time and eliminated scheduled downtime, while supporting 20x more concurrent calculations.

Migration Approach

Phase 1: Infrastructure Foundation

Established cloud environment with regulatory controls, implemented IaC, and built CI/CD pipelines with security scanning. Deployed monitoring and logging infrastructure with financial services compliance controls.

Phase 2: Data Layer Migration

Migrated historical market data to cloud-native databases, implemented change data capture systems, and established dual-write patterns during transition. Created data validation frameworks to ensure accuracy post-migration.

Phase 3: Service Decomposition

Identified domain boundaries and extracted microservices following the strangler pattern. Containerized services with Kubernetes manifests and deployed to production with feature flags enabling gradual cutover.

Phase 4: Full Production Transition

Implemented blue-green deployment for final cutover, executed comprehensive service verification, and decommissioned legacy infrastructure after successful parallel operation period.

Organizational Transformation

Platform Team Structure

Established dedicated platform engineering team serving as an internal service provider to index development teams. Implemented SRE practices with error budgets and SLOs.

Developer Experience

Created self-service portal for infrastructure provisioning, standardized development environments in containers, and built comprehensive documentation with runbooks.

Knowledge Transfer

Conducted immersive training programs on cloud-native technologies, paired platform engineers with index developers, and established internal tech talks and knowledge sharing sessions.

Key Takeaways

Performance Transformation

Cloud-native modernization delivered 8x faster index calculations and eliminated rebalancing downtime, directly improving client experience and enabling new product capabilities.

Compliance Integration

Regulatory requirements successfully integrated into CI/CD pipelines and infrastructure automation, maintaining continuous compliance while accelerating deployment frequency.

Organizational Impact

Platform engineering approach reduced time-to-market for new indices by 70% while enabling development teams to focus on business logic rather than infrastructure management.

Next Steps for Platform Evolution

- Machine learning integration for anomaly detection in market data
- Multi-cloud deployment strategy for additional resilience
- Enhanced self-service capabilities for business users
- Extending Airflow DAGs to support streaming workflows like triggering intraday recalculations

Thank You