## Project Development Progress Report

### 1. Error Handling & Resume from Breakpoint

* **Status:** Completed ✅
* **Implemented Solution:**
  + Kafka inherently provides automatic breakpoint resumption through its offset mechanism, ensuring reliable recovery after any exceptions or restarts.
  + Optimized Kafka consumer configuration (max.poll.interval.ms increased from the default 300,000 ms to 600,000 ms), significantly reducing risks related to message duplication and enhancing system stability under abnormal scenarios.

### 2. Performance Optimization

* **Status:** Completed ✅
* **Implemented Solution:**
  + Introduced Kafka offset auto-commit interval (auto.commit.interval.ms), optimized to commit offsets asynchronously every second, substantially reducing the risk of message duplication.
  + Switched from single-threaded processing to multi-threaded batch processing (leveraging CustomThreadPool), significantly improving message handling efficiency.
  + The current system performance benchmark demonstrates capability of handling millions of messages within a 5-minute window.

### 3. Remove Dependence on HBase

* **Status:** Completed ✅
* **Implemented Solution:**
  + Successfully removed the dependency on HBase. All data storage and caching functionalities have been migrated entirely to Redis and MongoDB.

### 4. Unit Test & Code Coverage (SonarQube)

* **Status:** In Progress (current coverage 30%+) ⚠️
* **Next Steps:**
  + Explicit goal set to increase unit test coverage to 80% or above.
  + Continuous improvements to code quality as measured by SonarQube metrics.

### 5. Monitoring System (Grafana & Prometheus)

* **Status:** Not yet started ❌
* **Next Steps:**
  + Planned for next iteration: Implement metrics collection using Prometheus and visualization using Grafana to achieve comprehensive system monitoring.

### 6. Combine Cusip Data for Related Cusips

* **Status:** Completed ✅
* **Implemented Solution:**
  + Related Cusip information is cached in Redis, and the associated refData is stored in MongoDB. Efficient data merging is implemented by integrating Redis with MongoDB, enabling rapid data consolidation and distribution.

### 7. Live-live Subscription Mode & Extensibility

* **Status:** Partially completed, further enhancement needed ⚠️
* **Implemented Solution and Future Plans:**
  + Currently, the system does **not yet support real-time dynamic listening** to MongoDB configuration changes for rules. Explicit future plans include implementing dynamic rule loading and enabling real-time configuration changes.
  + Additional APIs will be developed to dynamically start and stop specified Kafka listeners, further enhancing system flexibility.
  + Horizontal scalability is already supported through the expansion of Kubernetes Pods, providing strong extensibility for future growth.

### 8. System Recovery Time

* **Status:** Completed ✅
* **Implemented Solution:**
  + Optimized Kafka client reconnection strategy (reconnect.backoff.max.ms reduced to 10 seconds), dramatically enhancing rapid recovery capability after interruptions.

## Project Review Summary

| **Review Items** | **Status** | **Comments & Next Steps** |
| --- | --- | --- |
| 1. Error Handling & Resume from Breakpoint | ✅ Completed | Kafka parameter tuning significantly reduced risks |
| 2. Performance Optimization | ✅ Completed | Multi-threaded batch processing achieved millions of messages processed within 5 mins |
| 3. Remove Dependence on HBase | ✅ Completed | Fully migrated to Redis and MongoDB |
| 4. Unit Test & Code Coverage (SonarQube) | ⚠️ In Progress (30%+) | Target 80%+, actively ongoing |
| 5. Monitoring (Grafana & Prometheus) | ❌ Not started | Scheduled explicitly for the next iteration |
| 6. Combine Cusip Data for Related Cusips | ✅ Completed | High-efficiency data merging via Redis and MongoDB integration |
| 7. Live-live Subscription Mode & Extensibility | ⚠️ Partially completed | Dynamic rule configuration and API-driven listener management planned; Horizontal scaling through Pods supported |
| 8. System Recovery Time | ✅ Completed | Optimized Kafka reconnection strategy; rapid recovery assured |