### **1. Overview**

We’re facing intermittent WriteConflict errors in our ECS-based service that processes IP\_COST\_DETAILS and IP\_COST\_SUMMARY events and persists transactional data into multiple MongoDB collections.

### **2. Tech Stack & Design**

* **MongoDB**: Atlas Replica Set (M50)
* **Driver**: Spring Data MongoDB (via MongoTemplate)
* **Transactional Layer**: Spring-managed using @Transactional
* **Retry Strategy**: Implemented using Spring Retry (@Retryable)
* **Bulk Ops**: Performed using bulkWrite(replaceOne, upsert=true)
* **Audit Logging**: Inserted at every transaction boundary (insertOne, never updated)
* **No use of withTransaction**, only Spring @Transactional

### **3. Actual Flow**

#### **A. IP\_COST\_DETAILS → save:**

1. ECS receives cost detail event
2. **BulkWrite** into lvl4CostDetails$$Scenario
3. **refBU lookup** to enrich each cost
4. **Year-wise grouping**
5. **BulkWrite (replaceOne)** into lvl3CostDetails$$Scenario per year
6. **DeleteMany** only for years not present in this event
7. Audit log inserted with L3 + L4 UPSERT and L3 DELETE

#### **B. IP\_COST\_DETAILS → delete:**

1. ECS receives delete event
2. deleteMany on lvl3CostDetails$$Scenario
3. deleteOne on lvl4CostDetails$$Scenario
4. Audit log inserted with DELETE type

#### **C. IP\_COST\_SUMMARY → save:**

1. ECS receives summary event
2. Upsert to lvl1FinancialsSummary (set scenario block)
3. Lookup from ipTaxonomy
4. Transform scenario:{} into scenario:[] (flat array with year-level records)
5. Upsert (replaceOne with upsert:true) into lvl2FinancialsSummary (1 doc per proposal)
6. Audit log inserted

#### **D. IP\_COST\_SUMMARY → delete:**

1. ECS receives summary delete event
2. ECS sets that scenario block to {} in lvl1FinancialsSummary
3. Triggers reaggregation that produces empty array
4. BulkWrite replaces lvl2FinancialsSummary with an empty scenario array
5. Audit log inserted

### **4. Important: Version Check**

Before **any save or delete**, we **verify version ID** against existing MongoDB document.

If the event is **not the latest**, we **skip processing** and insert audit log with type SKIPPED.

This is critical for both details and summary flows.

### **5. Errors Observed**

We’ve seen the following errors even though retries are in place:

#### **Error A – WriteConflict:**

Write conflict during plan execution and yielding is disabled  
Error Code: 112

* Observed during bulkWrite with replaceOne into lvl3CostDetails$$Scenario
* Happens more often under concurrency when multiple events target same proposal/planId/scenario

#### **Error B – MongoSocketWriteException:**

InternalStreamConnection.transitivelyRetryable exception   
Caused by: com.mongodb.MongoSocketWriteException: Exception sending message

* Observed sporadically under high load
* We don’t have any manual socket-level handling configured

### **6. Request for Guidance**

1. **Spring Retry vs Native withTransaction**
   1. Is Spring @Transactional with @Retryable enough?
   2. Would moving to native withTransaction give better control over TransientTransactionError?
2. **WriteConflict Handling**
   1. Best practices for retrying bulkWrite with replaceOne when transactional deletes follow upserts?
   2. Should we yield or manually stagger writes?
3. **Connection Errors**
   1. Any best practices for avoiding MongoSocketWriteException in ECS Spring Boot services?