



An-Najah National University

Course: Advanced Software Engineering

Semester: First 2025/2026

Refactored the Event Processing System Refactoring

Issue Analysis Report

Instructor: Dr. Mustafa Assaf

Prepared By

Fathi Al Heelo

12323885

Date: DEC 12,2025

Issue 1:violation of SRP

The original code had a single class response for multiple concerns, such as event validation and data transformation and database logging and monitoring .

Why it is problematic?

This makes the class Hard to maintain and hard to test, and any small change break functionality .

For refactoring solution :

Decorator for data transformation and strategy for behavior and observer for logging and dashboard updates and proxy for validation and performance monitoring

Issue two : tight coupling between components

Problem the original implementation directly instantiated concrete classes such as database connections and loggers also event handlers

Why it is problematic ?

Violates DIP

And makes the system rigid and difficult to extend or test, and prevents replacing implementations easily .

Refactoring solution :

Makes interfaces for database event strategy ,
eventObserver , eventProcess

And used factories .

Components now depend on abstractions , not concrete implementations .

Issue 3: violation of OCP :

Problem event handling logic was implemented using large if/else or switch blocks based on event type .

Why it is problematic ?

Adding a new event type requires modifying existing code and high risk introducing bugs also code becomes difficult to scale .

Refactoring solution ?

Applied the strategy pattern and each event type has its own strategy class also new event types can be added without modifying existing logic .

Issue 4 : Code duplication

Problem similar logic for encryption, compression, and metadata handling was duplicated across multiple places.

Why it is problematic ?

Harder to maintain and inconsistent behavior and any change must be repeated in multiple locations .

Refactoring solution :

Introduced the decorator pattern and each transformation is implemented as a separate decorator and transformations can be combined dynamically .

Issue 5 :Poor Resource management

Problem the original code created database connections directly without reuse .

Why it is problematic ?

Performance degradation and resource exhaustion and not scalable under load .

Refactoring solution :

Implemented connection pool pattern and reused database connections efficiently and improved performance and scalability

Issue 6 : Lack of centralized object creation

Problem is events were created manually with repeated configuration logic.

Why it is problematic ?

inconsistent event initialization and code duplication also difficult to manage default configurations .

Refactoring solution :

Applied the prototype pattern and used EventTemplateRegistry to clone predefined templates also ensured consistent and flexible event creation .