Entreprise Integration Pattern

Vincent AGI
SIGLIS M1
2025/03/20

What’s EIP

Enterprise Integration Patterns (EIP) are standardized solutions
 for common challenges in integrating enterprise applications.
They cover aspects like data exchange, message routing,
and process automation, ensuring efficient and reliable system communication

Introduction

Markets are environments where prices fluctuate quickly

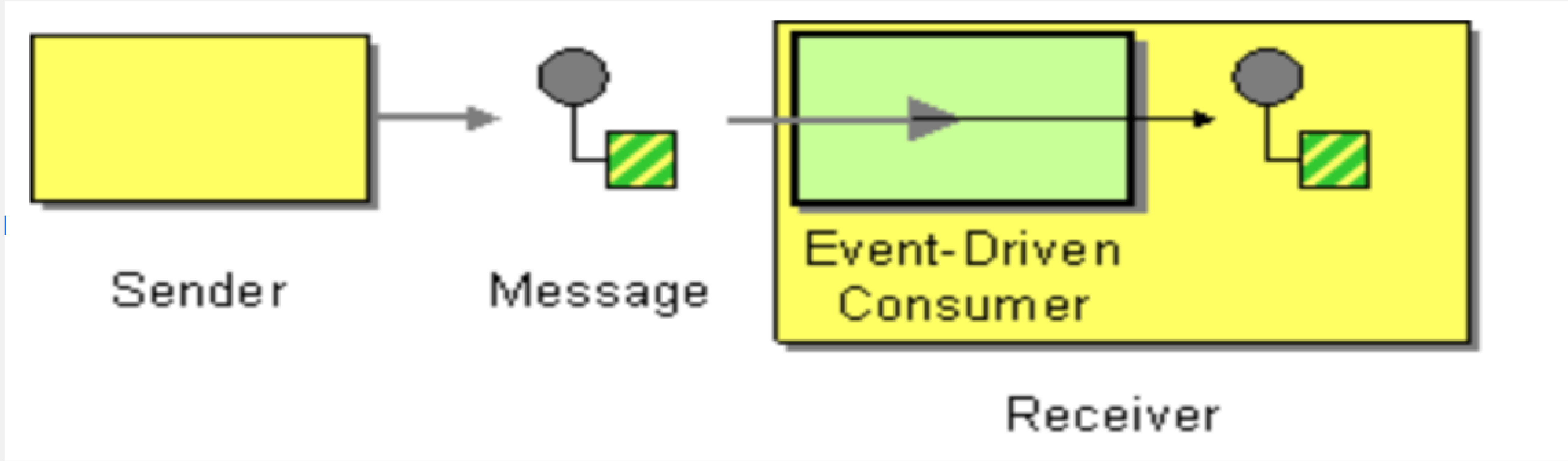
All stakeholders must be informed continiously

My Subject

Make a system which implements price variations for markets and notifications in real-time with 3 EIPs :

- Aggregator
- Dead Letter Channel
- Event Driven Consumer

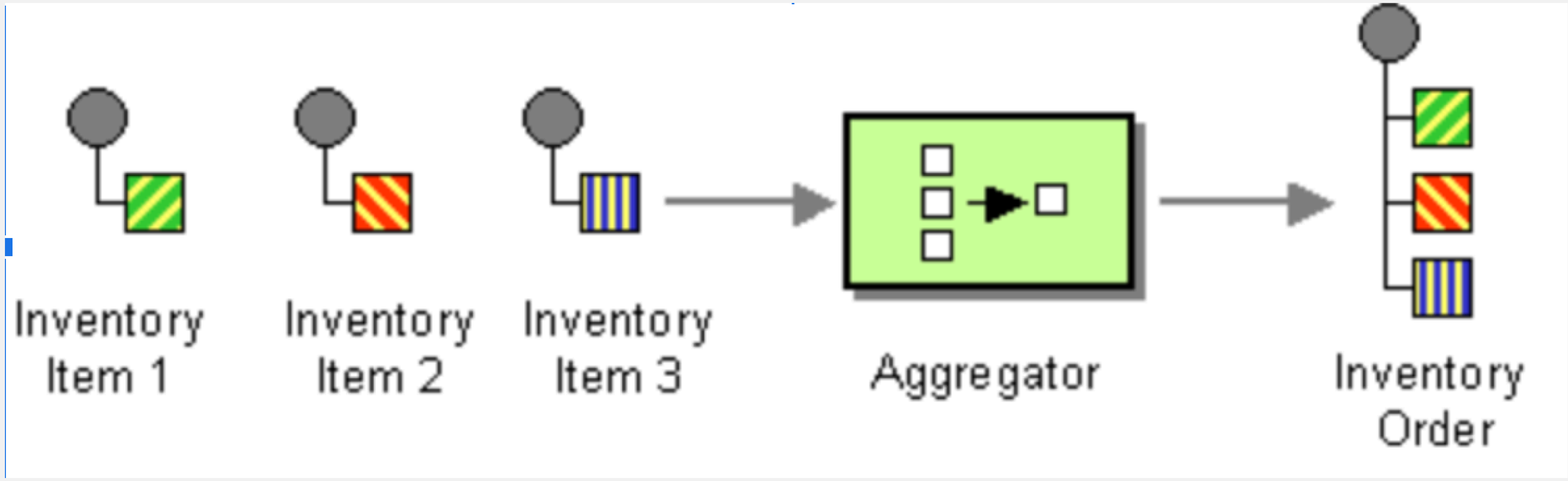
What’s Event Driven Consumer



A message was emitted and received by consumers.
The consumer emitted a new message to the client.

For exemple, the first message may be “price update” and the consumer emits this message to all consumers

What’s Aggregator

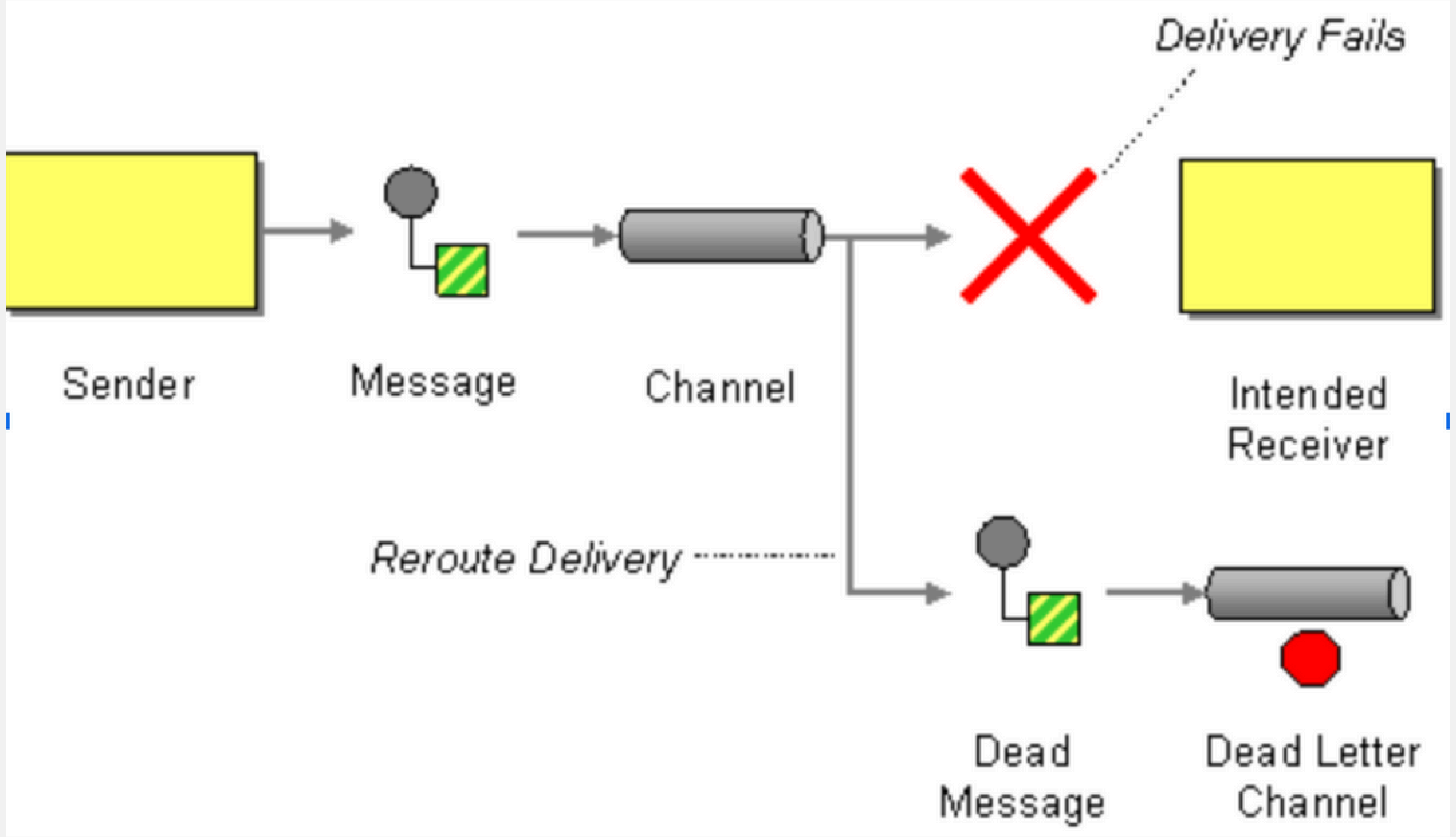


This pattern aggregator is used to aggregate some messages from differents origins.

It temporarily stocks each message to produce new messages which then merge with the first one.

For example if I have 3 messages (1, 2 and 3) from 3 origins, the aggregator will produce a new message whose value will be 6 (1 + 2 + 3).

What’s DLC



Imagine your application generates sometimes errors. How do you retry this scenario and solve it ?

That’s the solution provided by Dead Letter Channel. It stocks failed messages to a queue until the scenario replayed.

My challenges

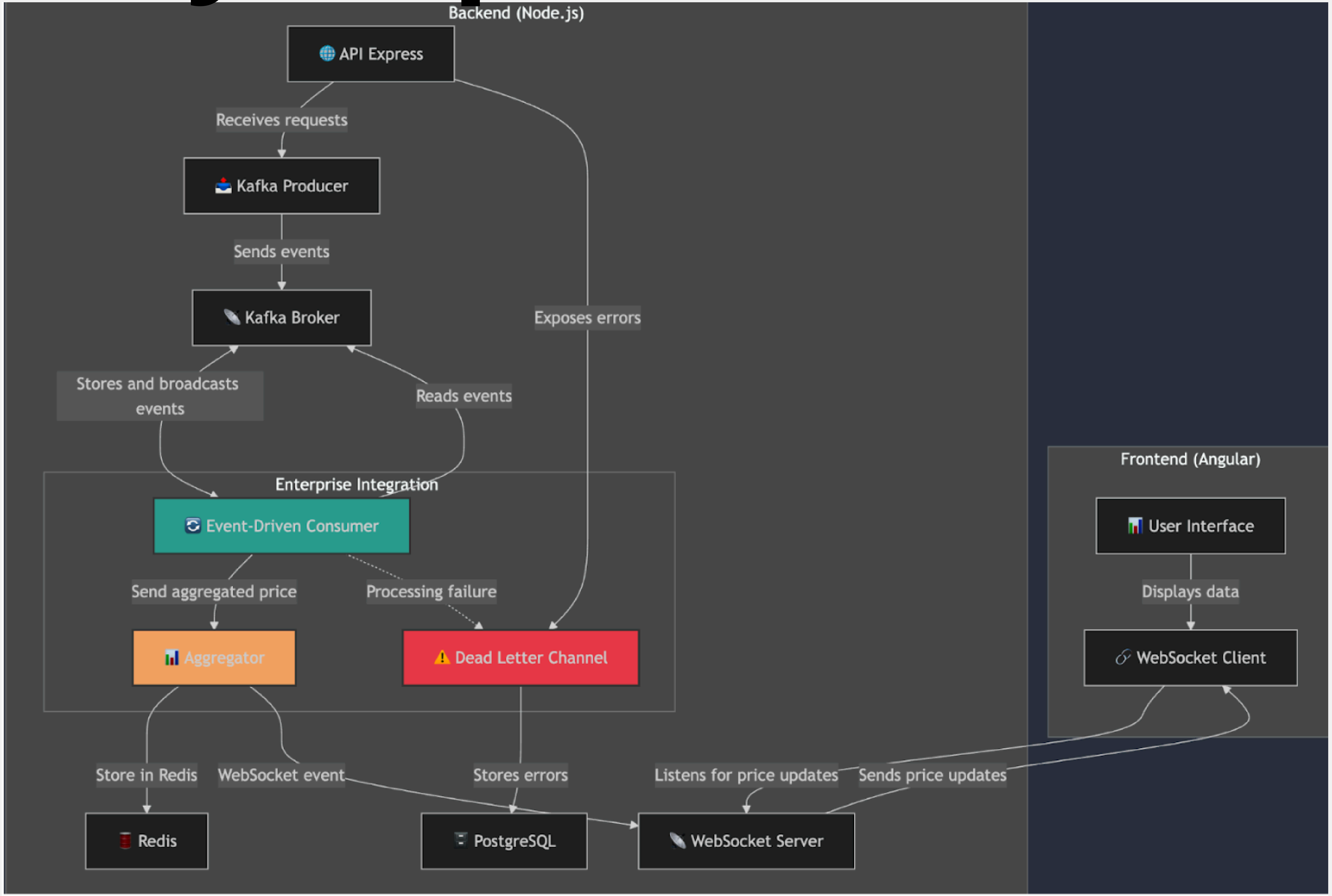
Manage large data streams

Ensure data consistancy

Ensure correct data aggregation

Project driven by events

My Implementation



Impacts of EIP : EDC

EDC is reactive, scalable, reduces coupling and use resources efficiently.

Can bring more complexity in events order

Impacts of EIP : Aggregator

Computes some information from a period and provides quality information for traders.

It normalizes data and reduces noise and data corruption

However the Aggregator introduces a delay to compute a lot of information

Impacts of EIP : DLC

DLC improves fault tolerence by preventing lost failed data

It allows easier debugging during replayed failed scenarios

However need for storage increases because there are a lot of failed messages

Technical environments

Kafka as messages queue

Redis as in-memory database for aggregator

Postgresql for datastorage and DLC

WebSockert for real-time and bidirectionnal communication

Conclusion

EIP take part of best practices

Great solutions to improve software robustness

I learned to be better developer

I was able to take a step back from my job