



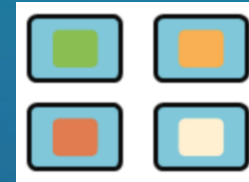
Microservice Architecture: Orchestration to Reactive Approach

Divya Goel, Chandni Jain

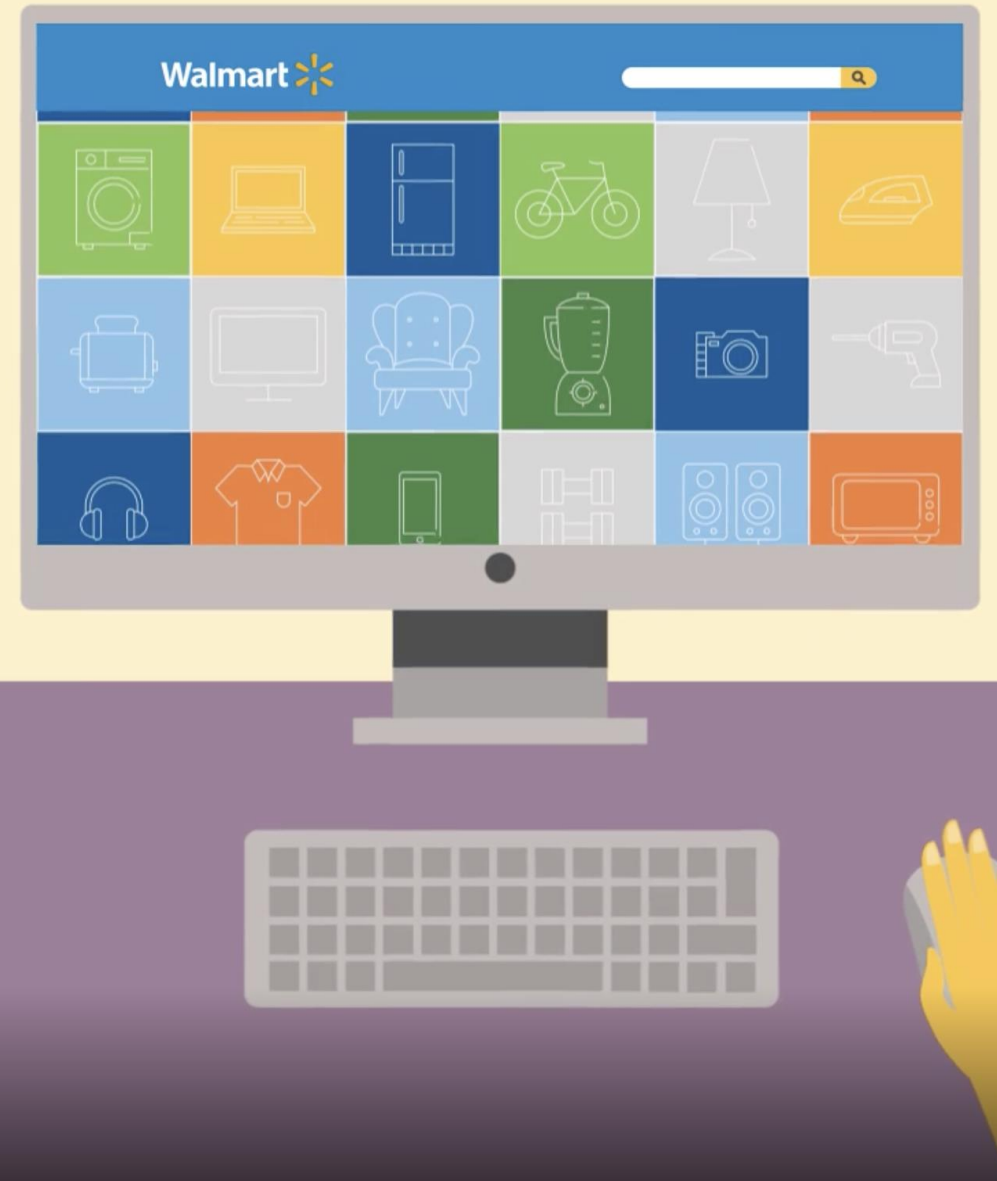
Walmart's journey to build a scalable system for hundreds of millions of items



Monolithic



Microservices



The Scale

CATALOG SIZE



70M

NUMBER OF STORES



4,700

ITEMS IN STORES



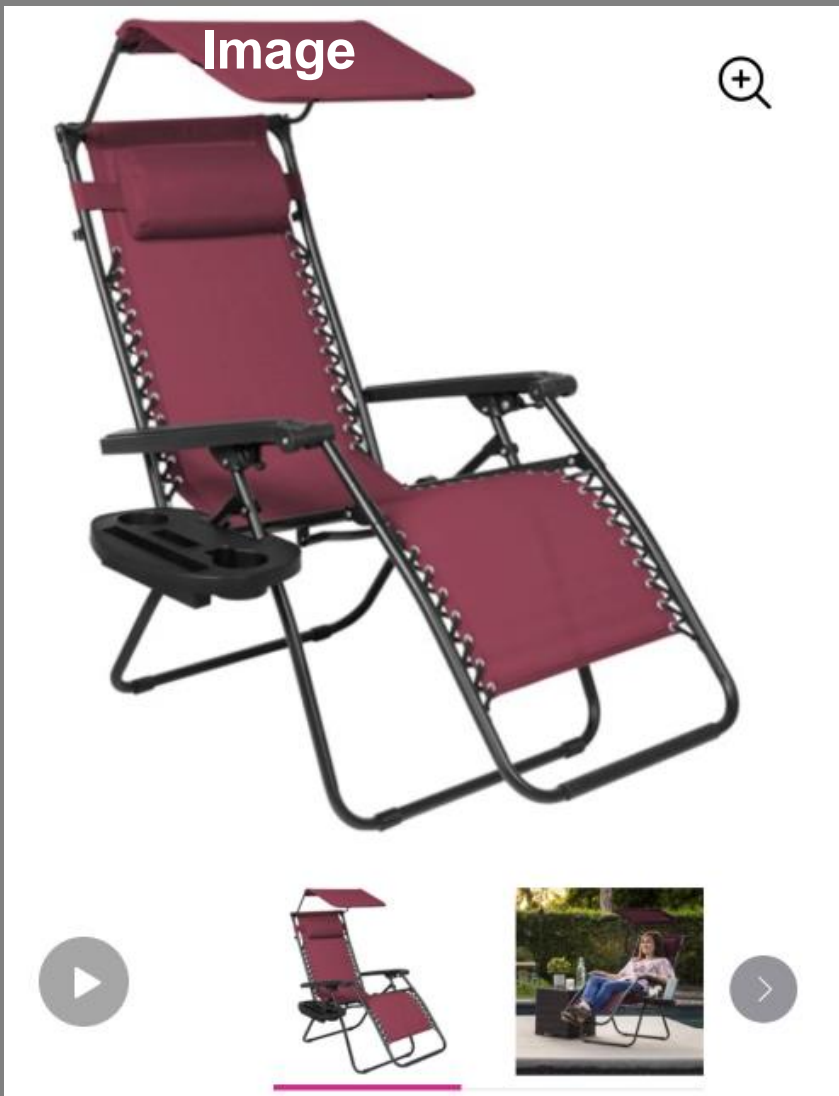
1M

Shelf

Patio & Garden / Patio Furniture / Patio Chairs & Seating / Zero Gravity Chairs / Shop by Number of Chairs / Zero Gravity Chair Single Pack

f p t

REDUCED PRICE

Best Choice Products Zero Gravity Chair w/ Canopy
Shade & Magazine Cup Holder

Name

★★★★★ [450 reviews](#) [Best Choice Products](#)

Reviews

\$51.99 List \$112.99

Actual Color: Burgundy



Grouping

Add-on services (0 Selected) [Select](#)



Available Options: Walmart Protection Plan.

Care Plan

Qty:

1 ▼

Add to Cart

 **Free 2-day delivery**
Arrives by Tue, Oct 1 Pickup not available[More delivery & pickup options](#)

#GHC19

Product Setup: Monolithic Architecture

Normalization

Matching

Merge

Easy To:

- Develop
- Deploy
- Test
- Monitor

Challenges With Growing Scale

Validation

Normalization

Matching

Merge

Ranking

Classification

Grouping

Images

- Difficult to onboard new developers
- Longer cycles between releases
- Scaling the application can be difficult
- Increases infrastructure costs
- Obstacle to scaling development
- Technology lock-in

Microservice Architecture

Normalization

Validation

Matching

Classification

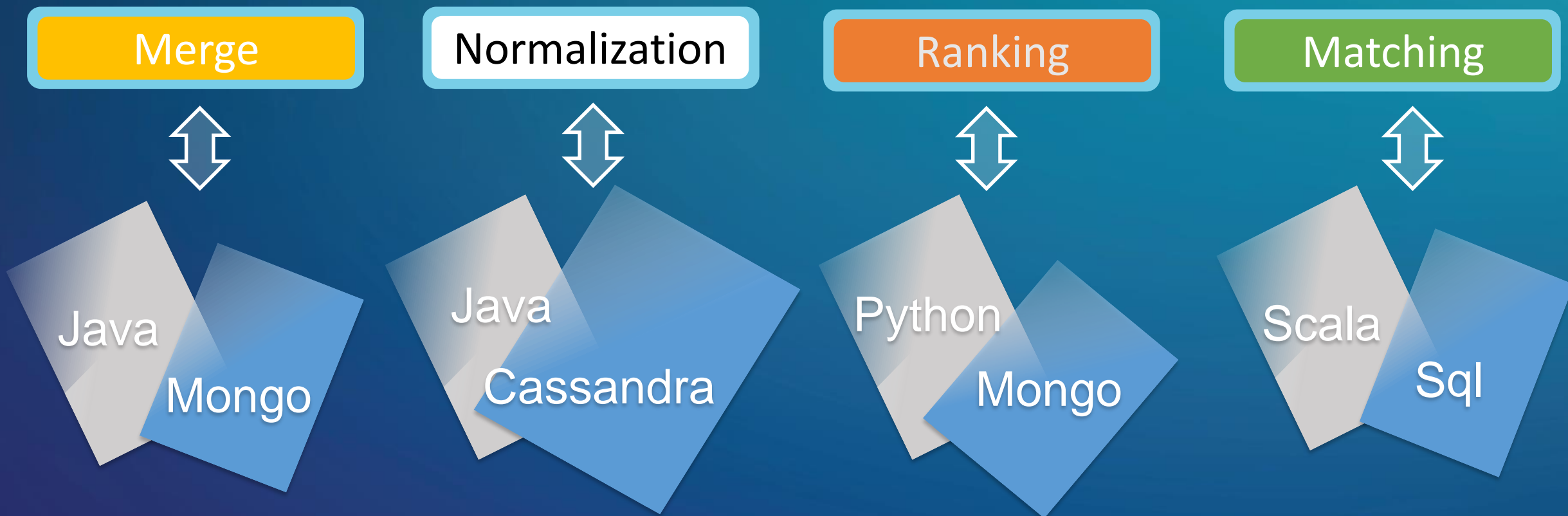
Ranking

Images

Merge

Grouping

Tech Stack Flexibility



Hardware Scaling Flexibility

Validation



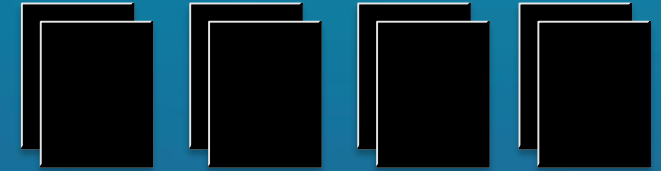
2 X (2 core CPU, 2 GB RAM)

Images



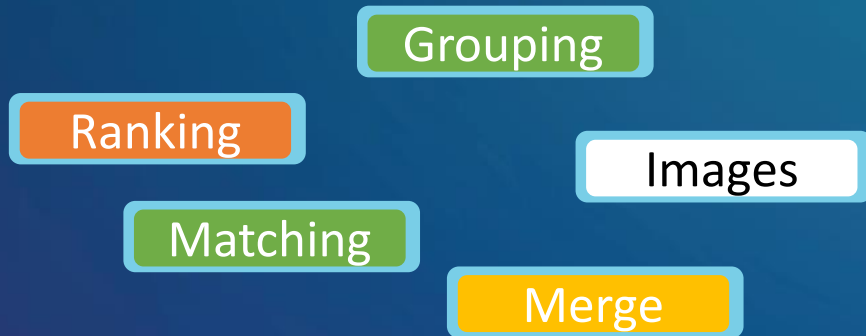
5 X (4 core CPU, 4 GB RAM)

Classification

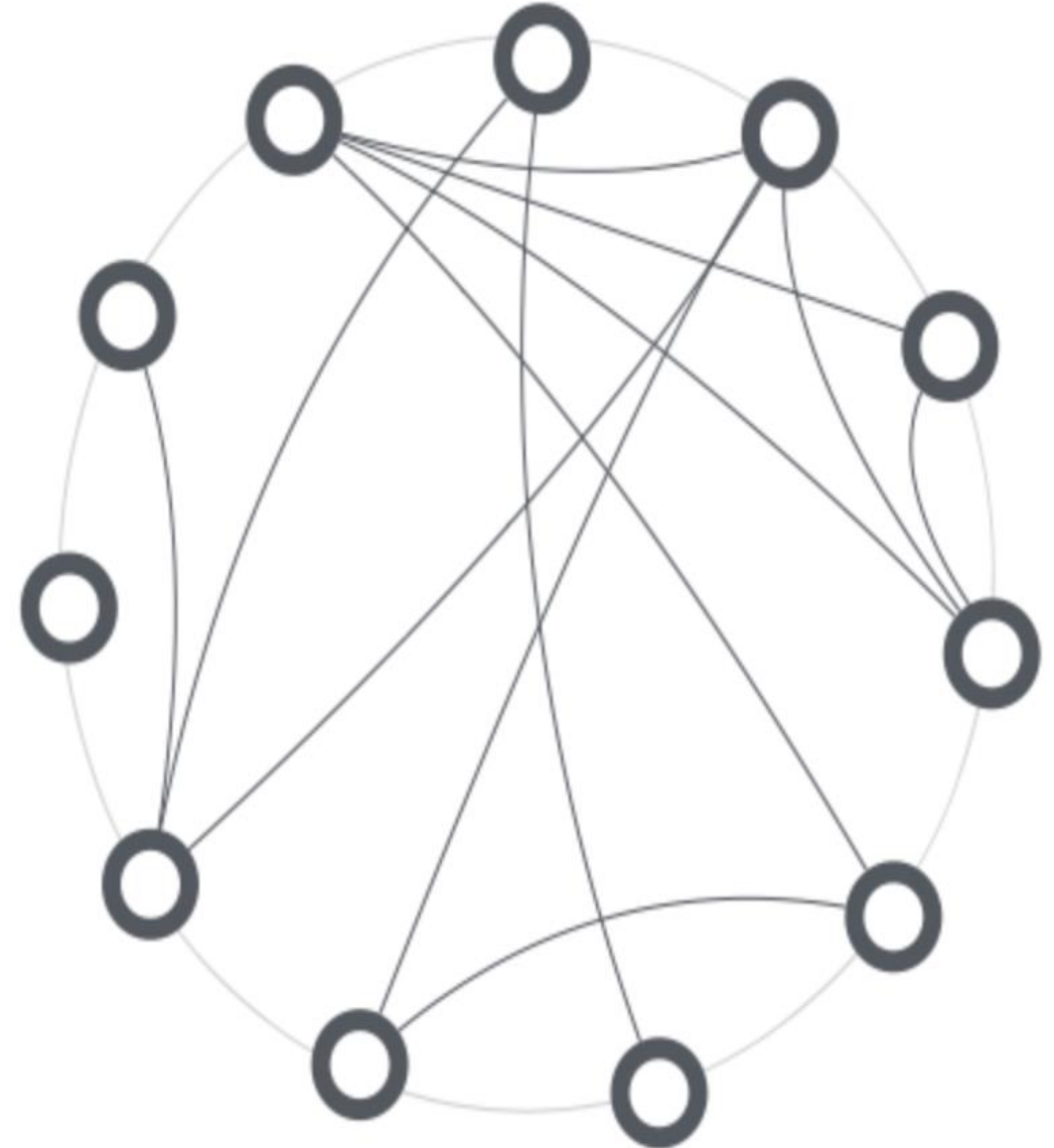


4 GPU

Microservice Architecture

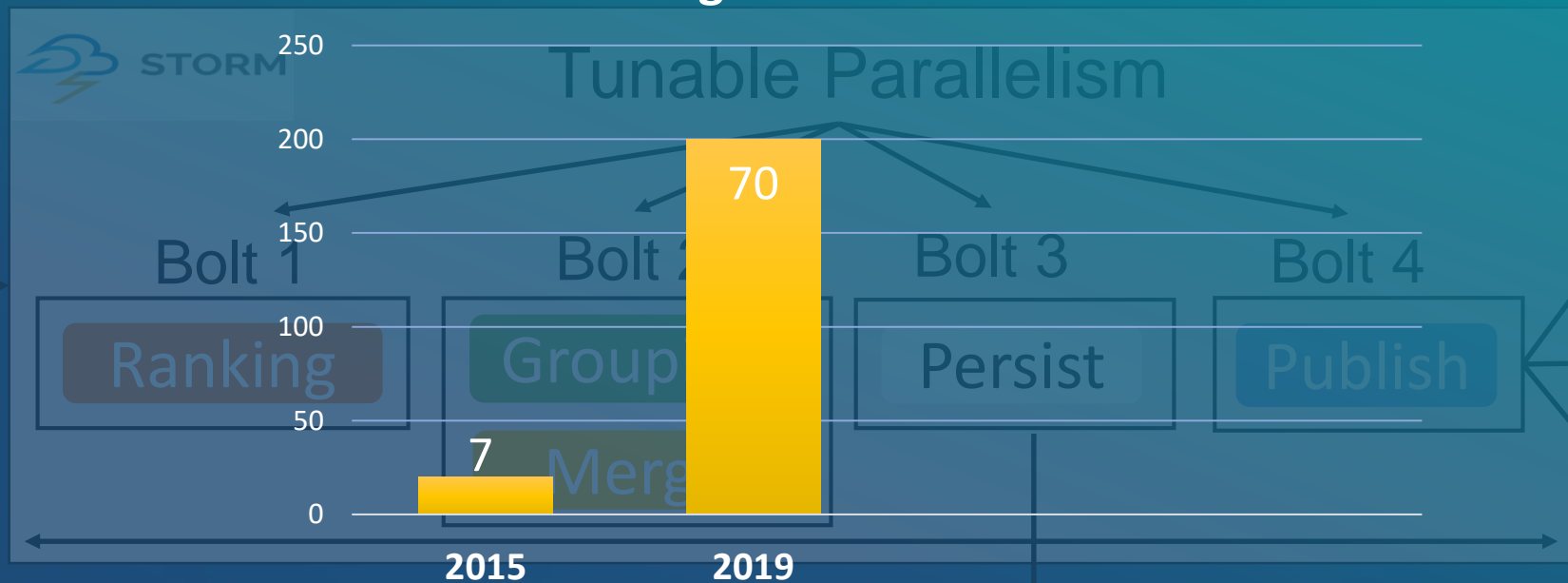


Services Orchestration



Orchestrator

Catalog Growth In Millions



In-Memory Processing

■ Catalog Growth

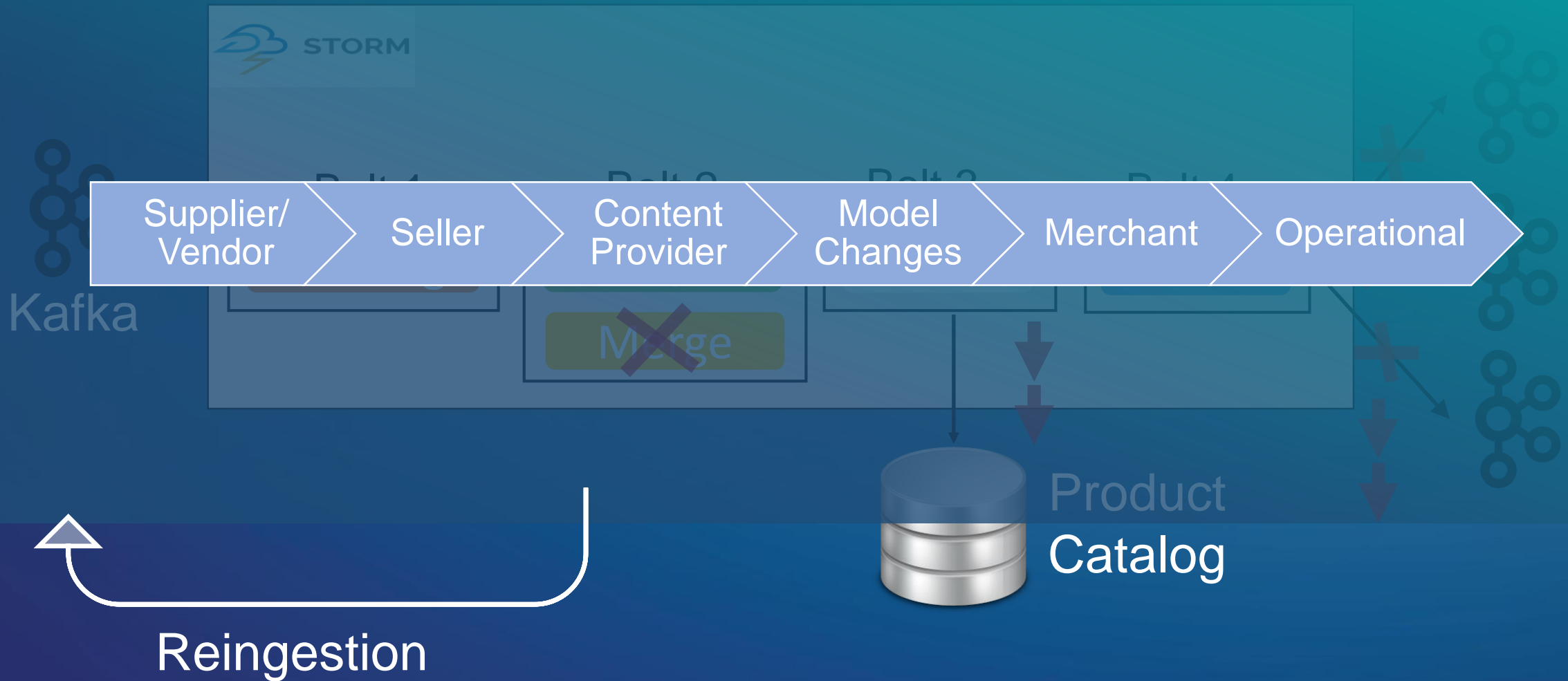


Product Catalog

Scaled for millions of item ingestions

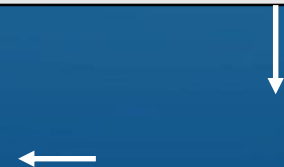
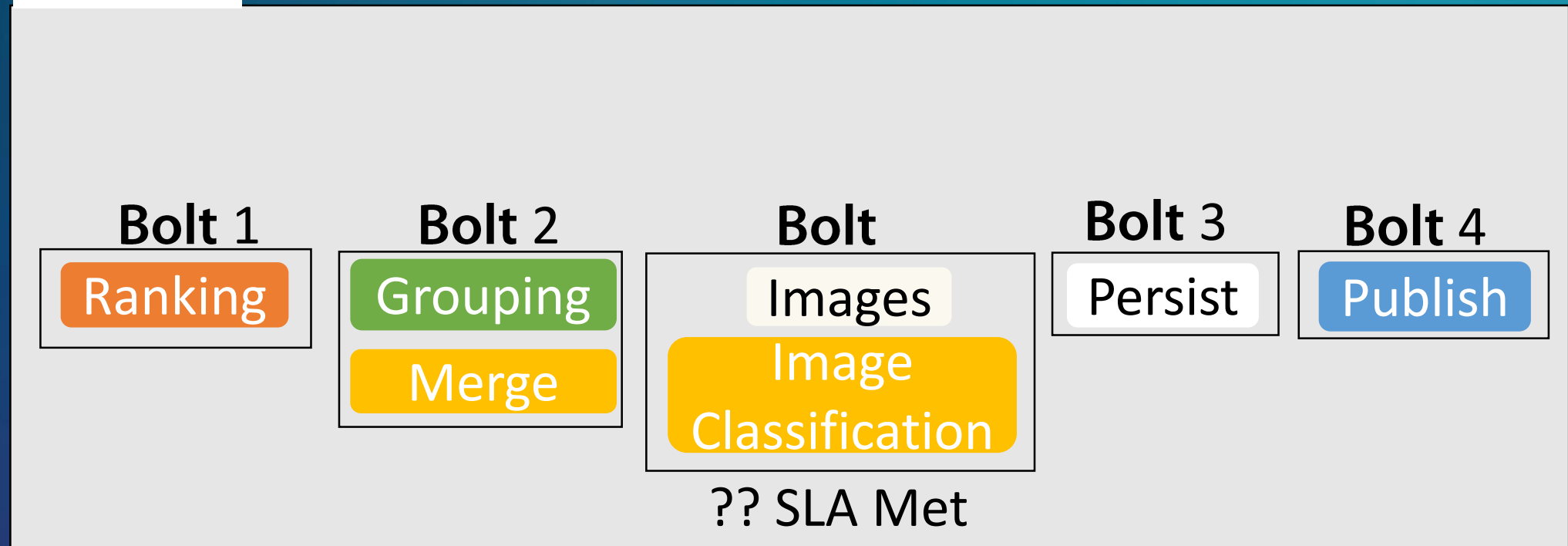
Challenges With Growing Scale

Orchestrator



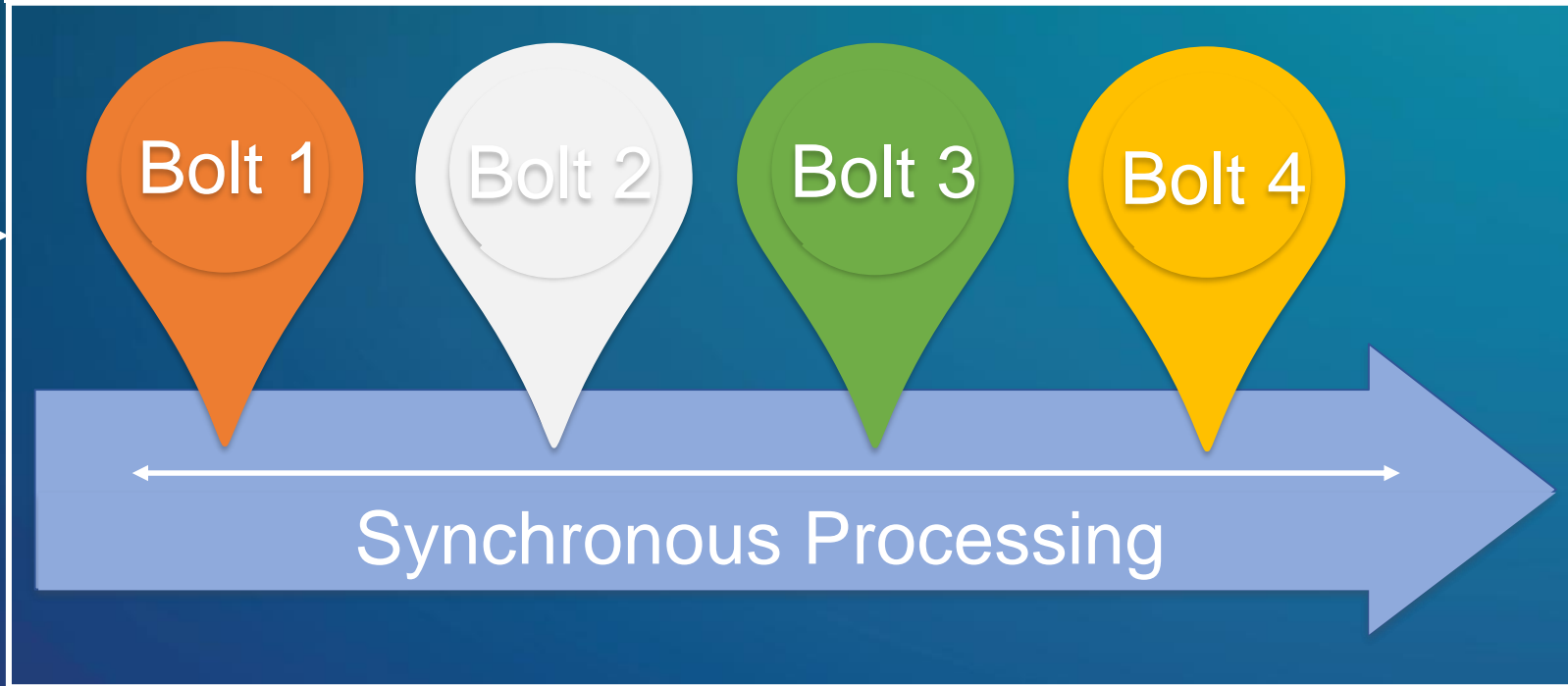


Orchestrator





Orchestrator

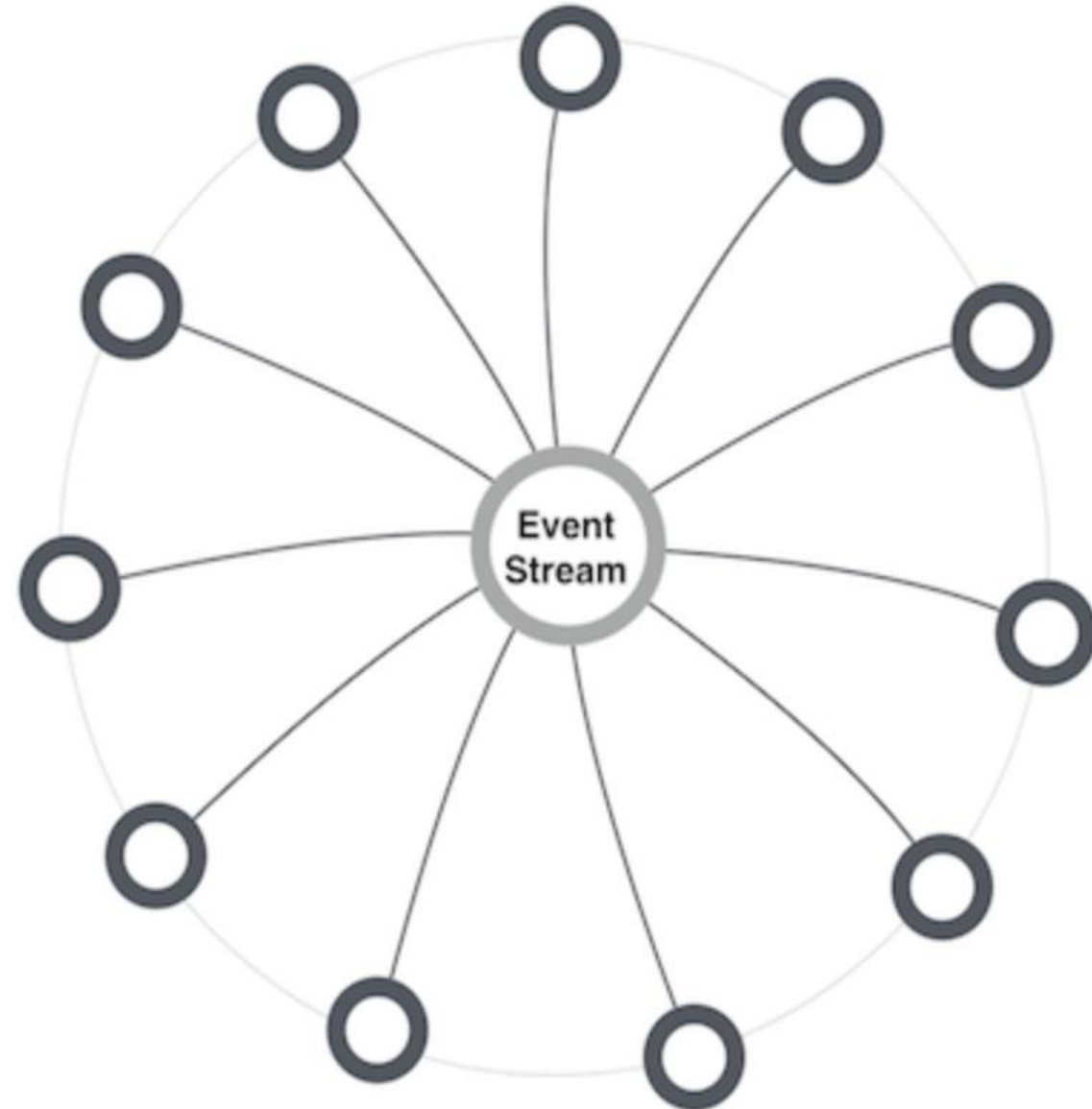


Monitoring

Message Ordering not reliable

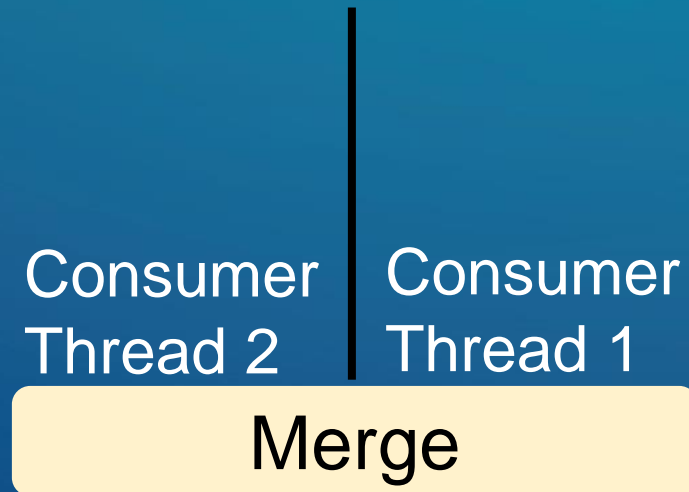
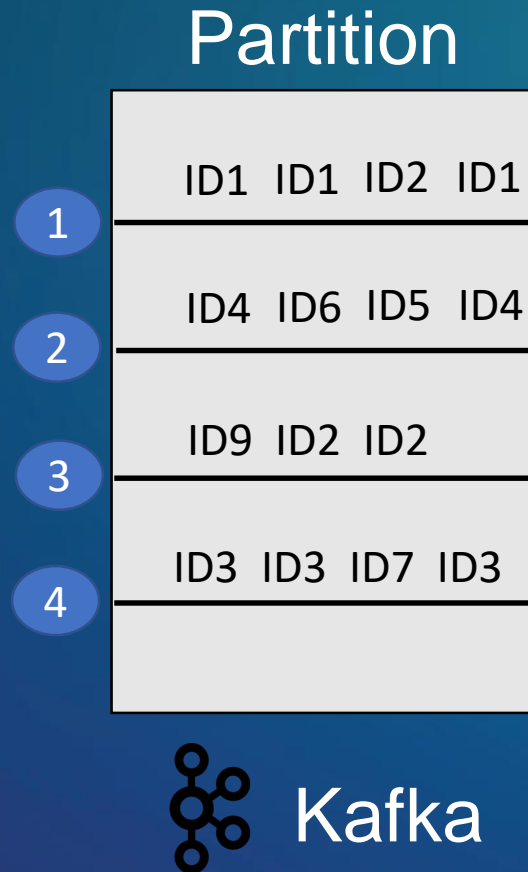
Microservice Architecture

Reactive Services

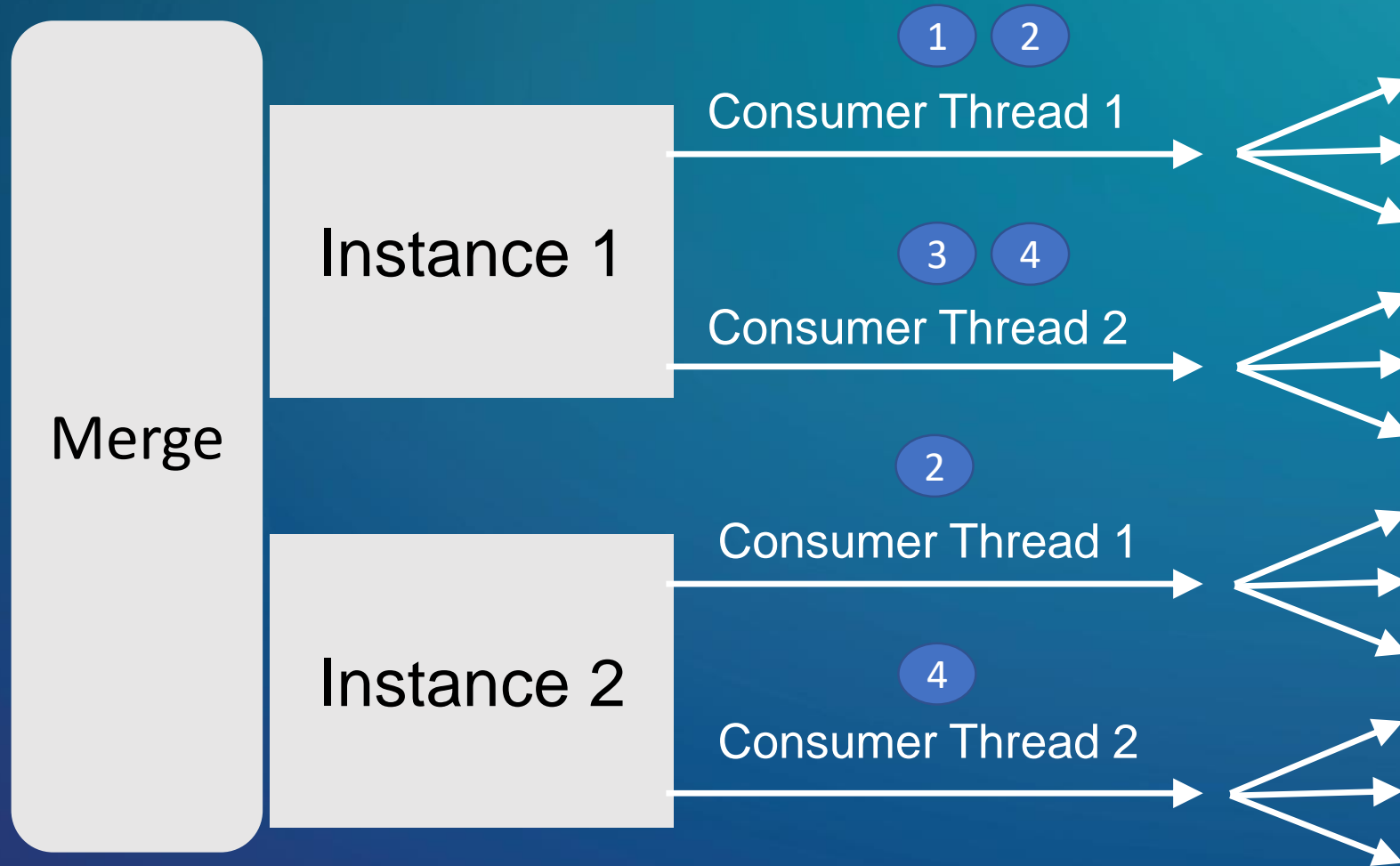




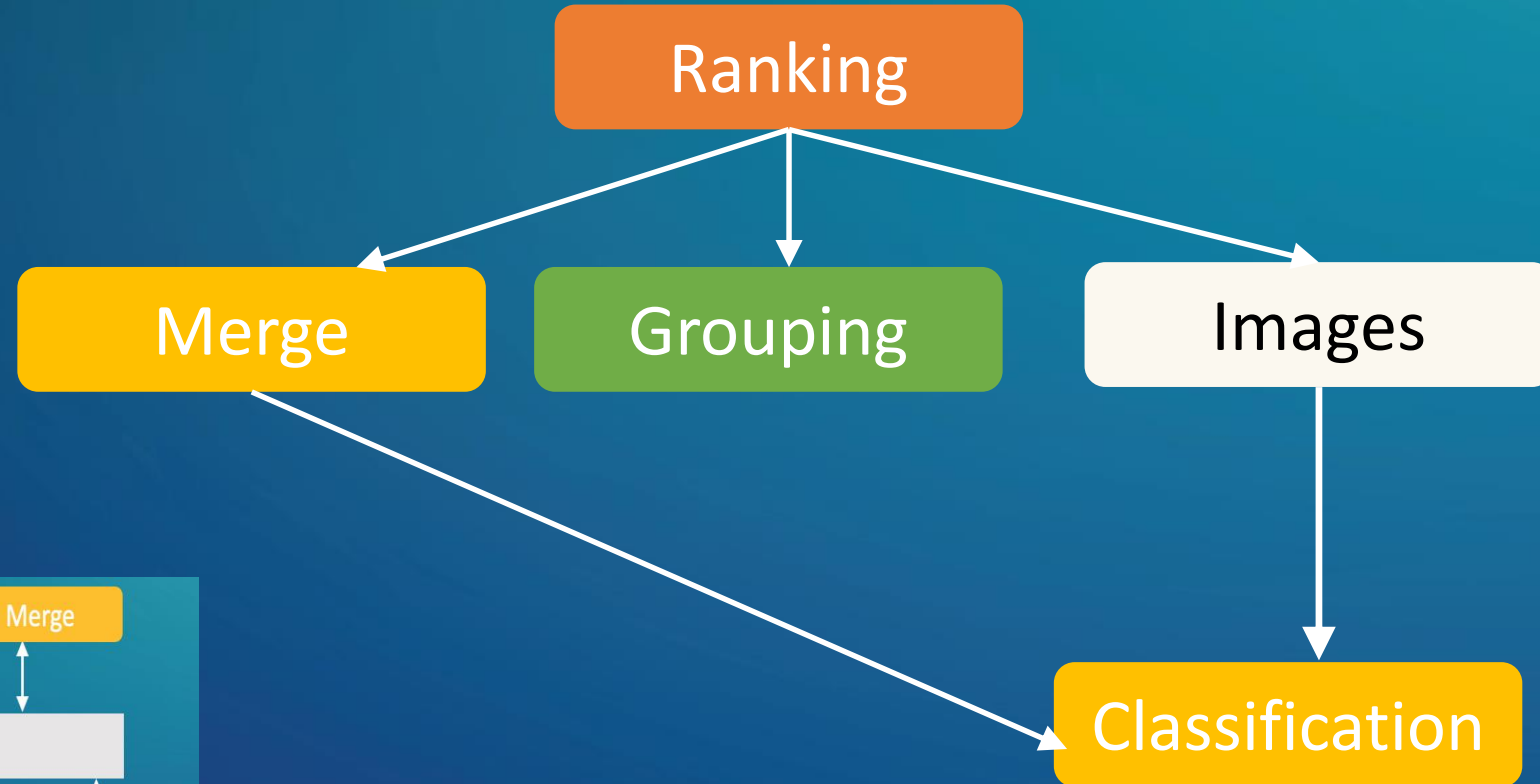
Message Ordering

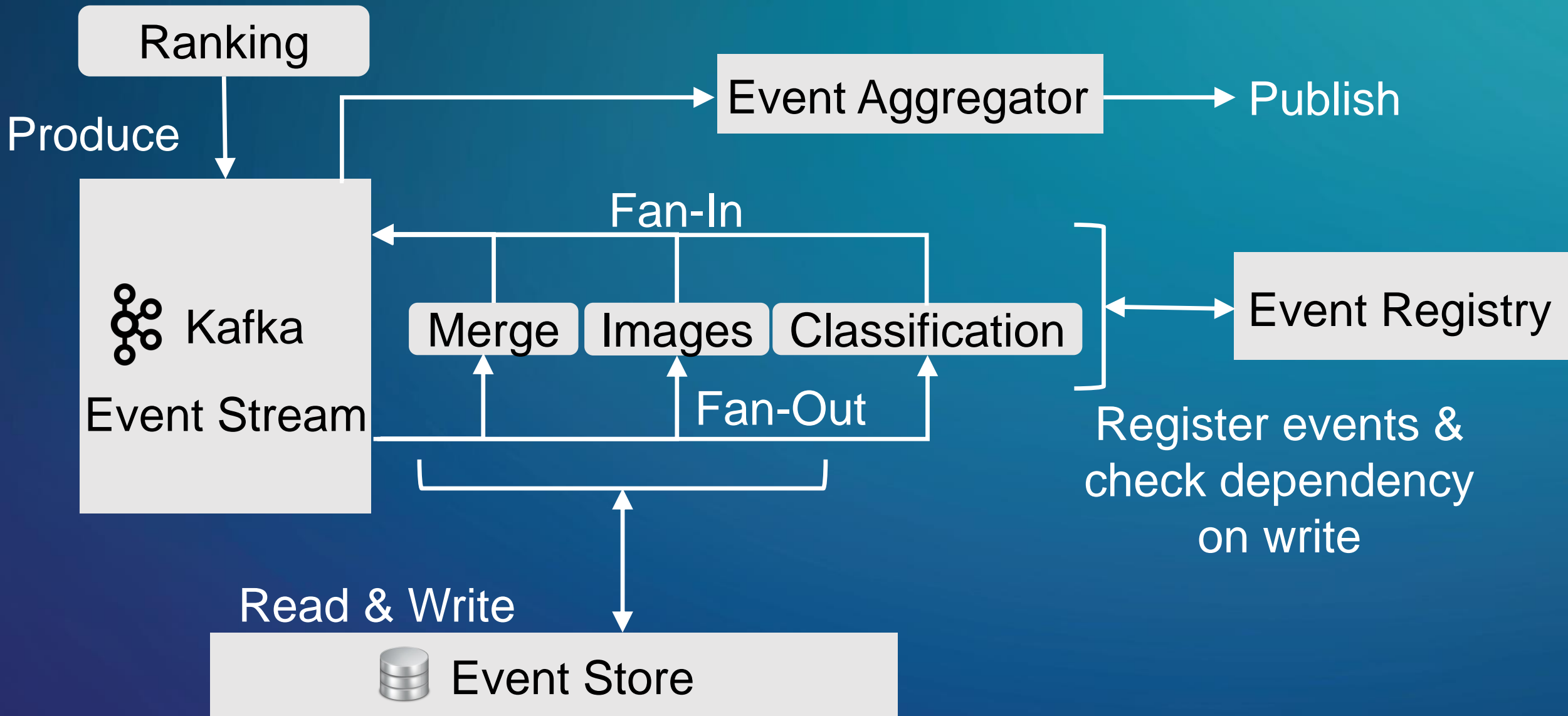


How To Scale



Service Dependency





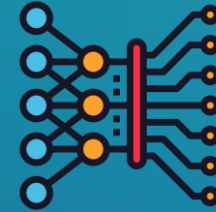
Final Thoughts



**Future Scale &
Requirements**



**Monitoring &
Analytics**



**Revisit & Evolve
Architecture
Periodically**

Appendix

Q&A

- **When monolithic should be broken-down to microservices?**
- **Serverless Architecture**
- **Cloud Native**
- **Kubernetes**

Why Reactive?

- Easy to plug in and plug out microservices
- No single point of failure
- Flexible Orchestration
- Scaling for individual service requirement
- Parallel processing for independent microservices
- Long running service can participate in event stream
- Process and publish intermediate state

**Applying a microservice architecture is not about building the perfect system. Instead, it is about building a framework in which a good system can emerge over time as the understanding grows”
- Martin Flower**

Links

- Anil Kumar. Apache Kafka for Item Setup. [Apache-Kafka-Item-Setup](#)
- Andrew Bonham. Microservices—When to React Vs. Orchestrate. [React-vs-Orchestrate](#)
- Martin Fowler, [Microservices Architecture](#)
- Microservices Architecture, <https://microservices.io/patterns/microservices.html>