Service Architectural Patterns

PTS3 / FMFI UK / 7.4.2025

Michal Kostič

Outline

- Major architectural patterns
  - Monolith
  - Service Oriented Architecture
  - Microservices
  - Event sourcing
- Bonus Patterns
  - Monolith to Microservices migration
  - Circuit Breaker
  - CQRS - Command and Query Responsibility Separation

Example recap

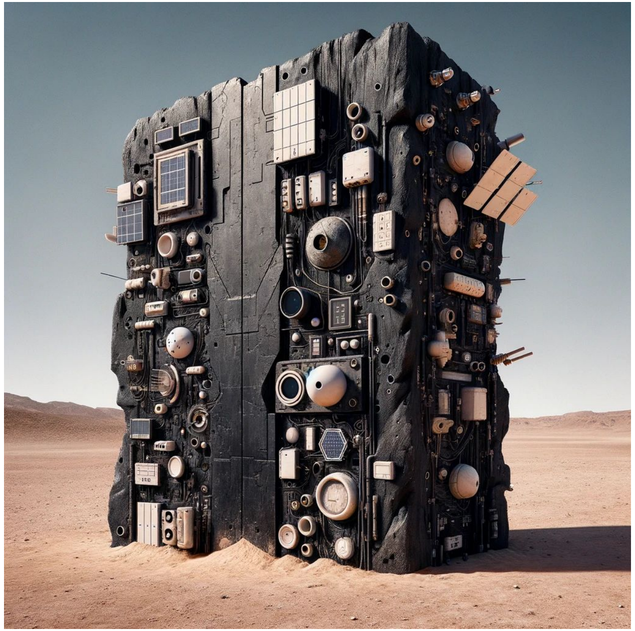
- E-commerce site
  - Customer buys goods
  - Company employees manage the inventory and orders

Major Architectural Patterns and Styles

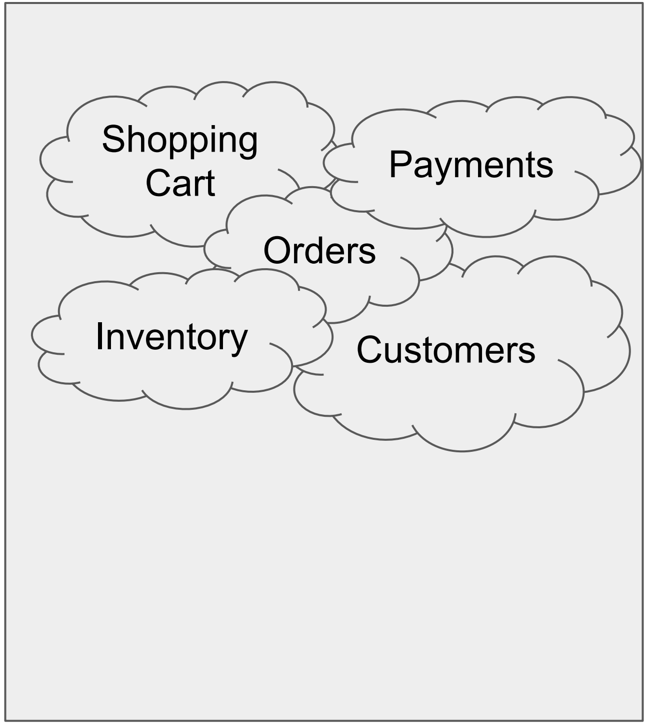
Monolith

Expectation

Reality



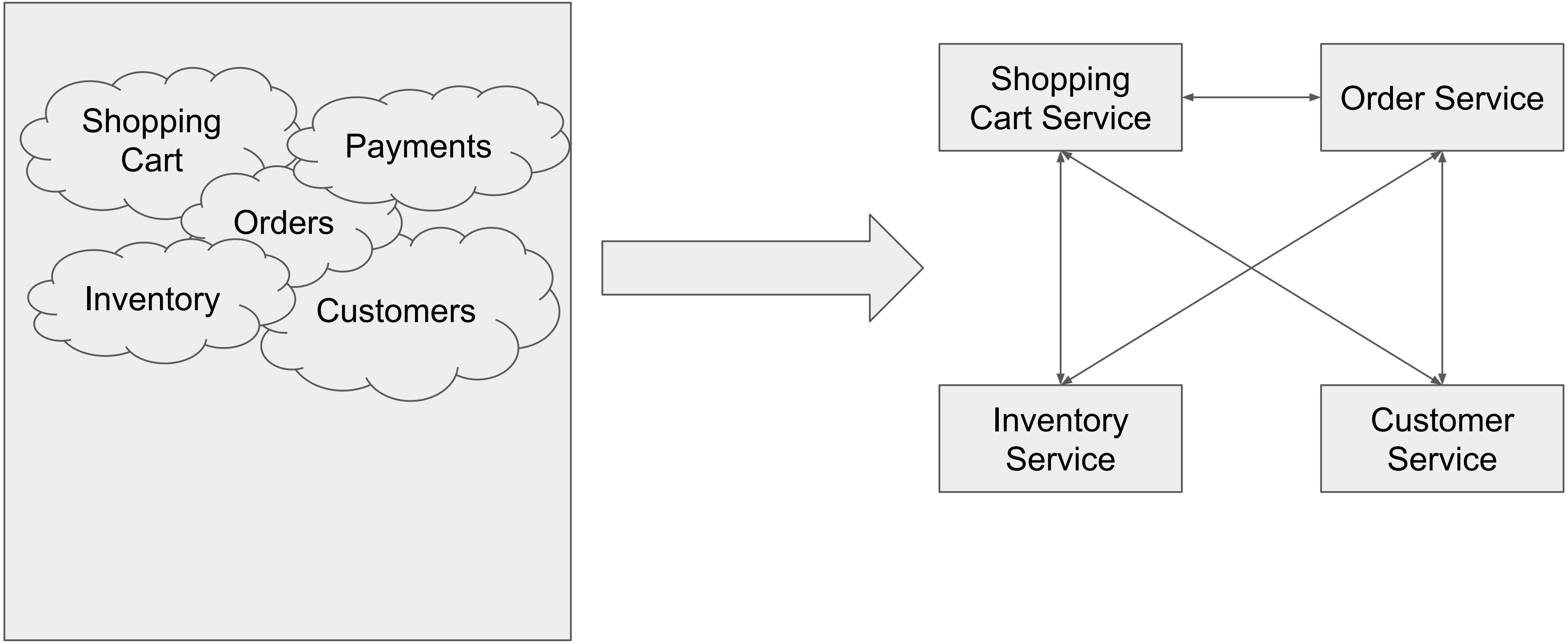
Monolith



Monolith

- Single unified software application, that is self-contained
- Pros
  - Simplicity - everything in single codebase
  - Efficiency - fast communication between sub-modules
  - Ease of development - running locally, debugging, …
- Cons
  - Maintainability
  - Scalability
  - Agility - adding new features can be complex
  - Single point of failure

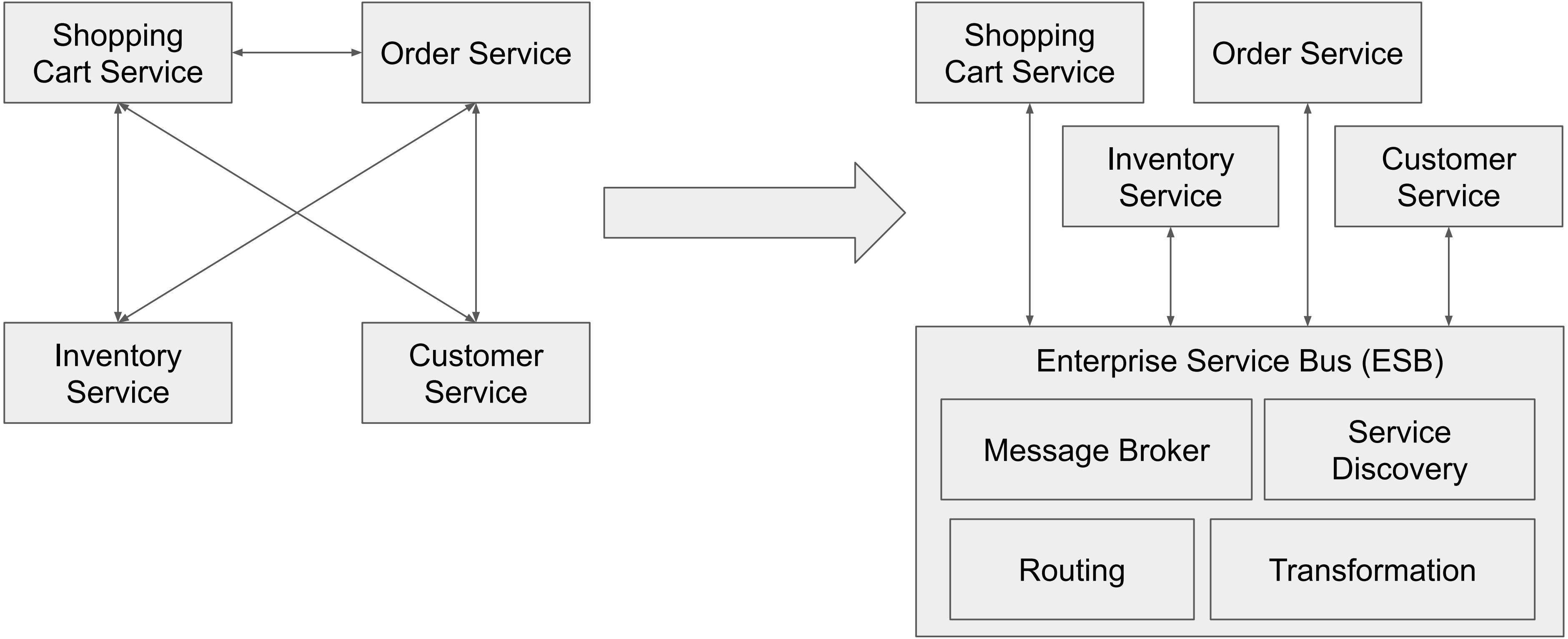
Service Oriented Architecture



Service Oriented Architecture

- Service-oriented architecture (SOA) is a method of software development that uses software components called services to create business applications
- Pros
  - Reusability
  - Scalability
  - Agility
  - Loose Coupling
  - Platform Independence
- Cons
  - Complexity
  - Network overhead
  - Ease of development - distributed debugging, difficult to run locally

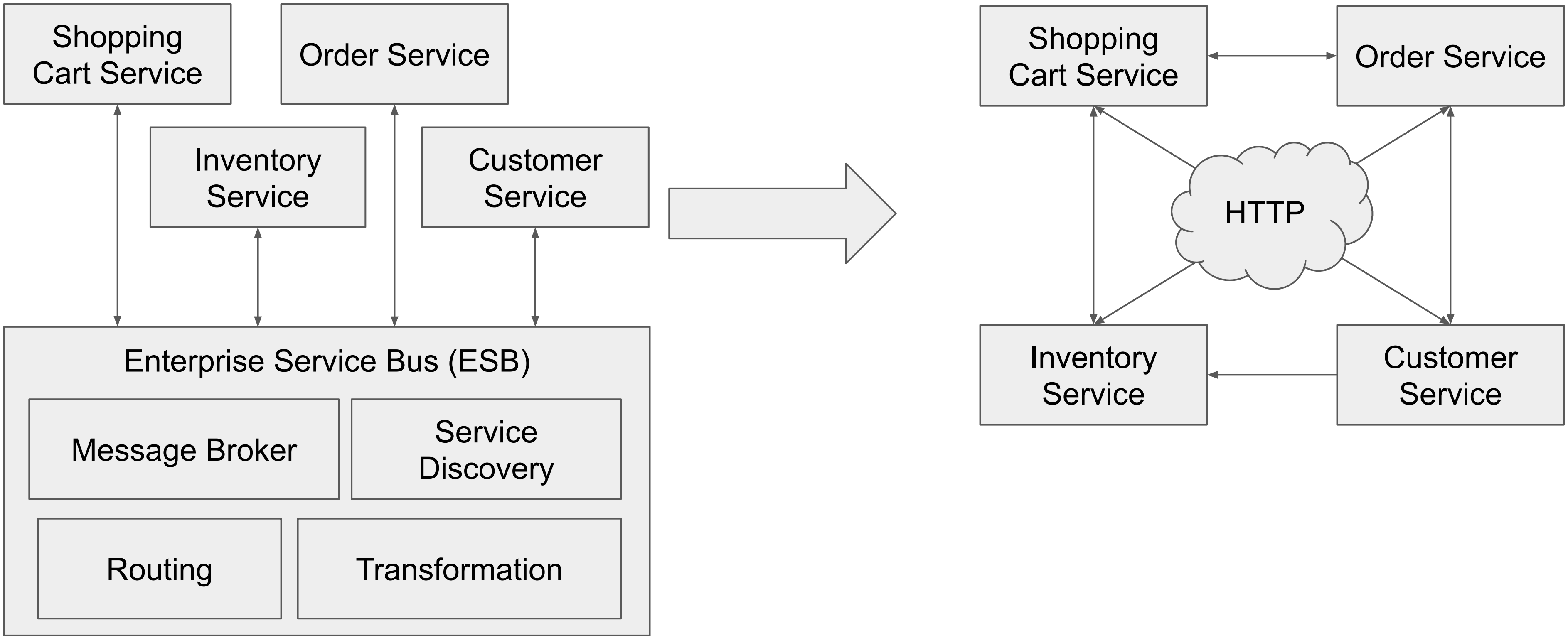
Enterprise Service Oriented Architecture



Enterprise Service Oriented Architecture

- Simplify communication and service discovery in settings with high number of services
- Enterprise Service Bus
  - Message routing
  - Data and protocol transformation
  - Security enforcement
  - Monitoring and management
- Pros
  - Central integration point
  - Loose coupling
  - Standardization
- Cons
  - Increased complexity
  - Performance overhead
  - Increased cost and vendor lock-in

Microservices



Shopping Cart Service

Order Service

Inventory Service

Customer Service

Enterprise Service Bus (ESB)

Message Broker

Service Discovery

Routing

Transformation

Shopping Cart Service

Order Service

HTTP

Inventory Service

Customer Service

Microservices

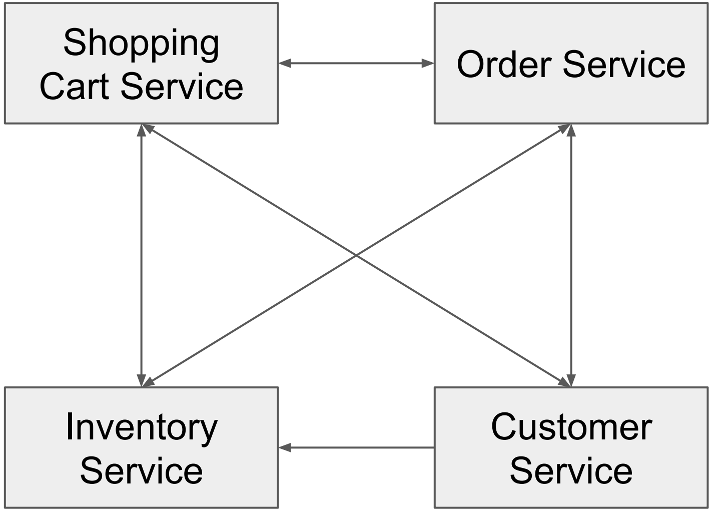
- Microservices are an architectural approach to software development where software is composed of small independent services that communicate over well-defined APIs
- Pros
    - Scalability
    - Agility
    - Fault tolerance
    - Improved maintainability
    - Technology independence
- Cons
    - Increased complexity
    - Distributed debugging
    - Communication overhead
    - Deployment complexity

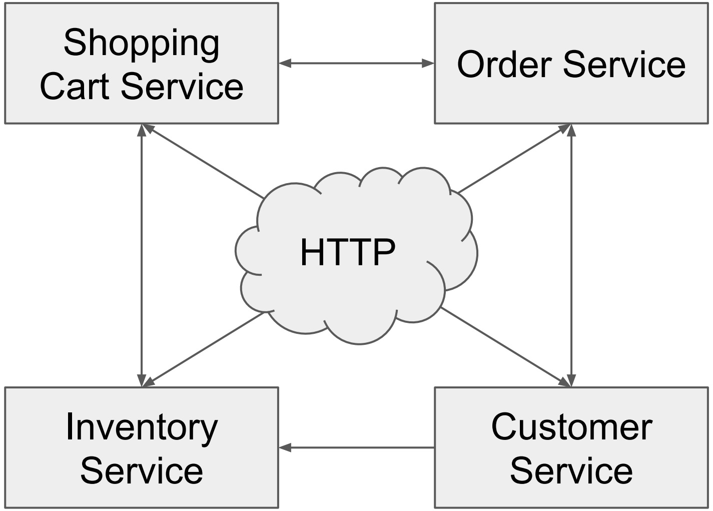
Microservices - related patterns

- How to split?
  - Business capability - models organization
  - Subdomain
  - Self contained service
  - Team
- Database per service vs Shared database
- API Composer
  - Builds out a facade and unifies multiple services
- Critiques
  - Right granularity - https://thenewstack.io/year-in-review-was-2023-a-turning-point-for-microservices/
  - Focus on composability - https://thenewstack.io/composable-architectures-vs-microservices-which-is-best/

Microservices vs Service Oriented Architecture

Spot the difference



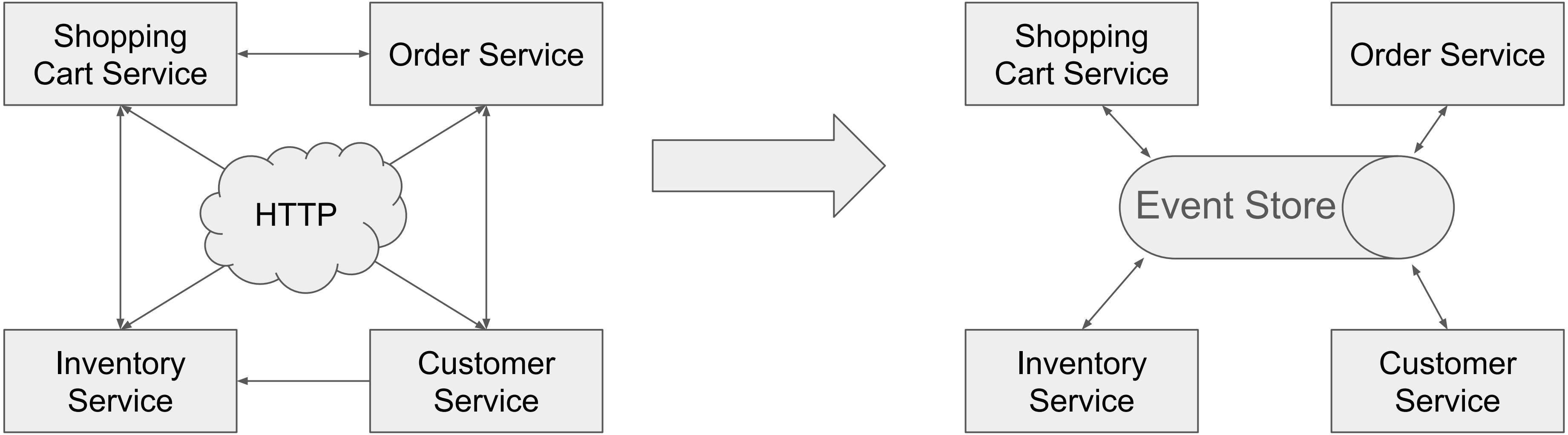


Bottom line: Well designed SOA is very similar to microservice architecture

Microservices Ecosystem

- Composable Architecture
  - API first design
  - Focus on reusability
  - API gateway for publishing APIs
- Service Mesh
  - Communication via proxy
  - Goals
    - Observability
    - Discovery
    - Load balancing
    - Scalability

Event Sourcing



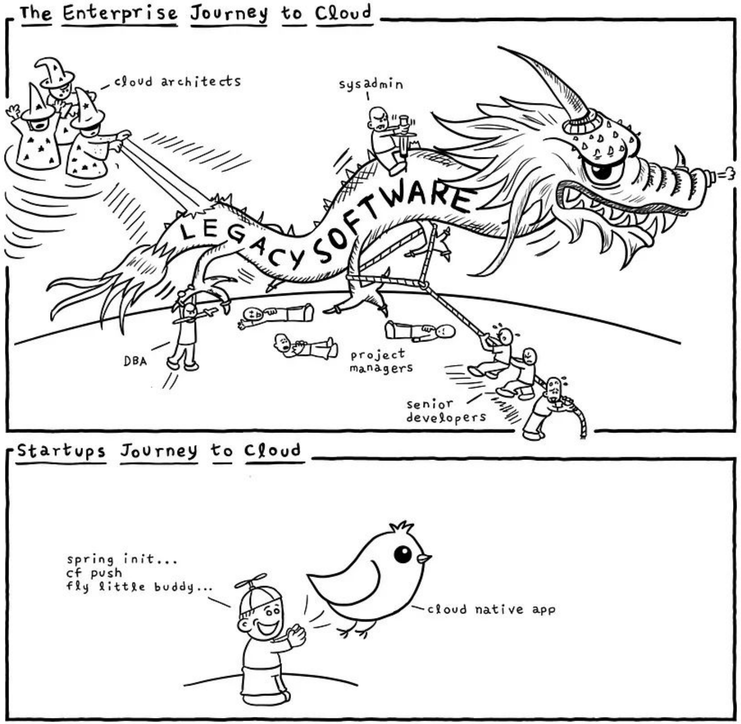
Summary

- Metrics to evaluate your architecture
  - Simplicity/Complexity
  - Agility
  - Scalability
  - Maintainability
  - Developer experience (local development, debugging)
  - Fault tolerance
  - Technology independence
  - Communication overhead
- Service Oriented Architecture, Microservices, Composable architecture are all refinement of the similar concepts - do not get too hung on the details!
- Beware of the marketing lingo - focus on the underlying principles!
- Complexity kills - apply just enough architecture!

Bonus Patterns

Monolith to Microservices Migration

- Big bang
- New features are created as microservices
- Extract functionality (strangler pattern)
- Anticorruption layer

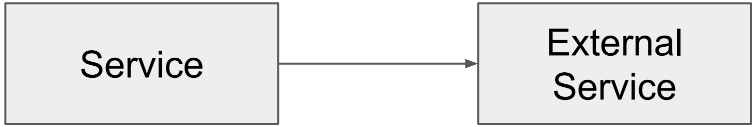


Daniel Stori {turnoff.us}
Thanks to Michael Tharrington

https://turnoff.us/geek/enterprise-vs-startup-journey-to-cloud/

Circuit Breaker

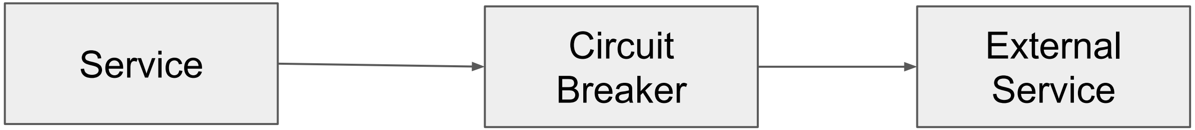
Timeout blocks our service as well



Service

External Service

Circuit breaker detects failure and returns immediately

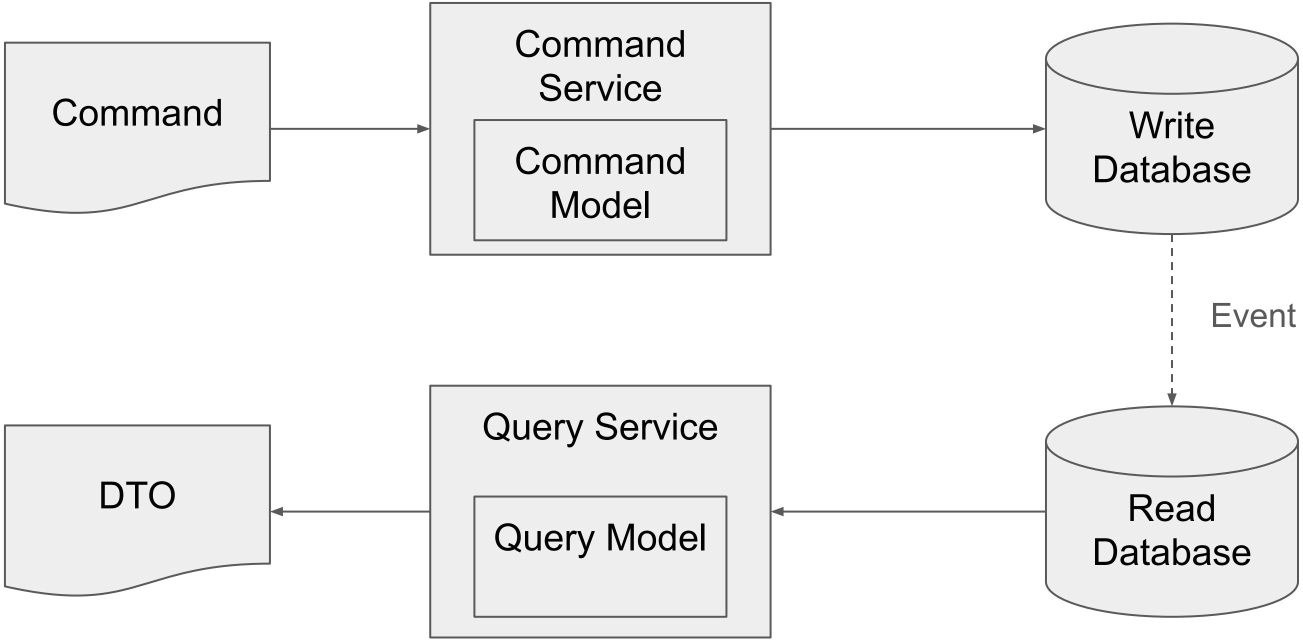


Service

Circuit Breaker

External Service

Command and Query Responsibility Segregation (CQRS)



Command

Command Service

Command Model

Write Database

Event

Read Database

Query Service

Query Model

DTO