MSA

강병호

이런 분이면 더 좋아요!

- Golang을 활용한 개발 경험이 있으신 분
- 마이크로서비스 아키텍쳐 기반의 서비스 경험이 있으신 분
- 비동기 프로그래밍, 대용량 데이터 처리, 확장성을 고려한 설계, 혹은 이에 준하는 경험이 있으신 분
- 서비스하는 프로덕트의 Back-office admin 을 직접 구현하고 관리해본 경험이 있으신 분
- Geographic Information System 관련 지식이 있거나 관련 프로젝트 경험이 있으신 분

- 전체적인 관점에서 좋은 아키텍처에 대한 고민을 하고 개선해 나가요. 배포하기 쉬운 구조, 운영하기 쉬운 구조, 안정성을 위한 구조, 성능 개선을 위한 구조에 대해서 고민하고 실행에 옮겨요.
- Java/Kotlin, Spring Framework를 기본으로 Redis, Kafka, Kubernetes, ELK를 적극적으로 사용하며, 대용량 트래픽이나 동시성 처리에 대한 고민을 많이 하시게 될 거예요.

Monolithic Architecture

모든 구성 요소가 단일 애플리케이션으로 구성된 아키텍처

Monolithic Architecture

Monolithic Architecture

👍

- 간단한 유지보수
- 개발 속도의 향상
- 통합 및 테스트의 간단함
- 기술 스택의 동일성

👎

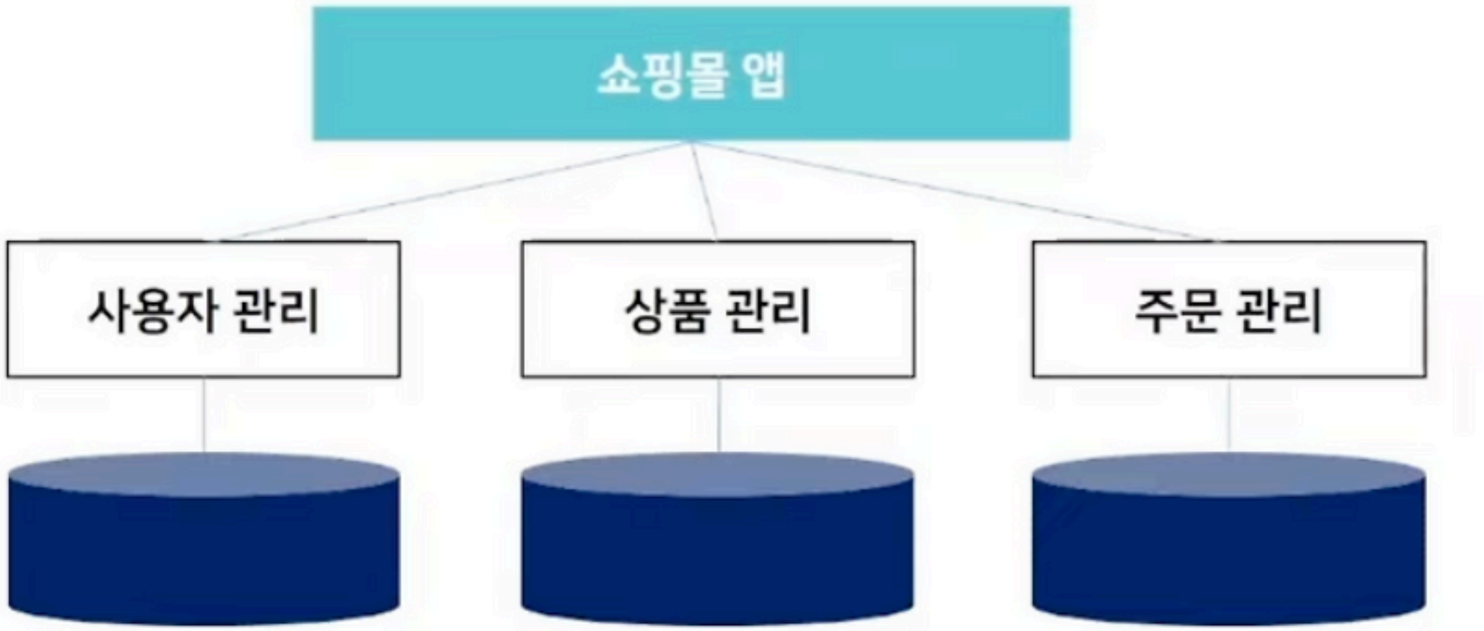
- 확장과 배포의 어려움
- 기술 스택의 제한성
- 여러 서비스들의 강한 결합성
- 협업의 어려움

대규모 시스템에서의 부적절함

Micro Service Architecture

각 서비스 기준으로 애플리케이션이 분리되어 관리되는 아키텍처

Micro Service Architecture



Micro Service Architecture

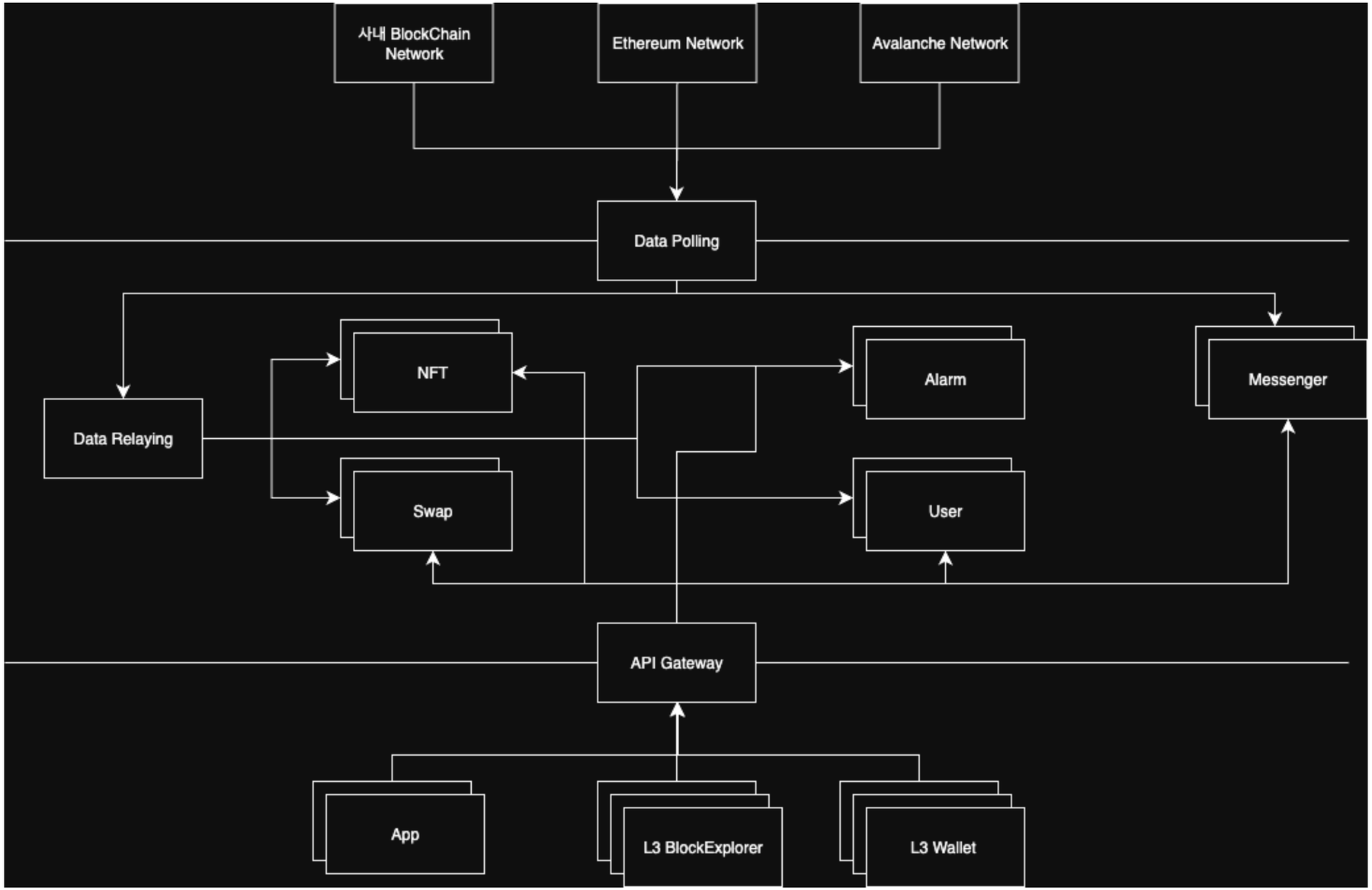
👍

- 서비스 단위 개발 배포
- 독립적인 확장성
- 장애 격리
- 기술 스택의 유연성

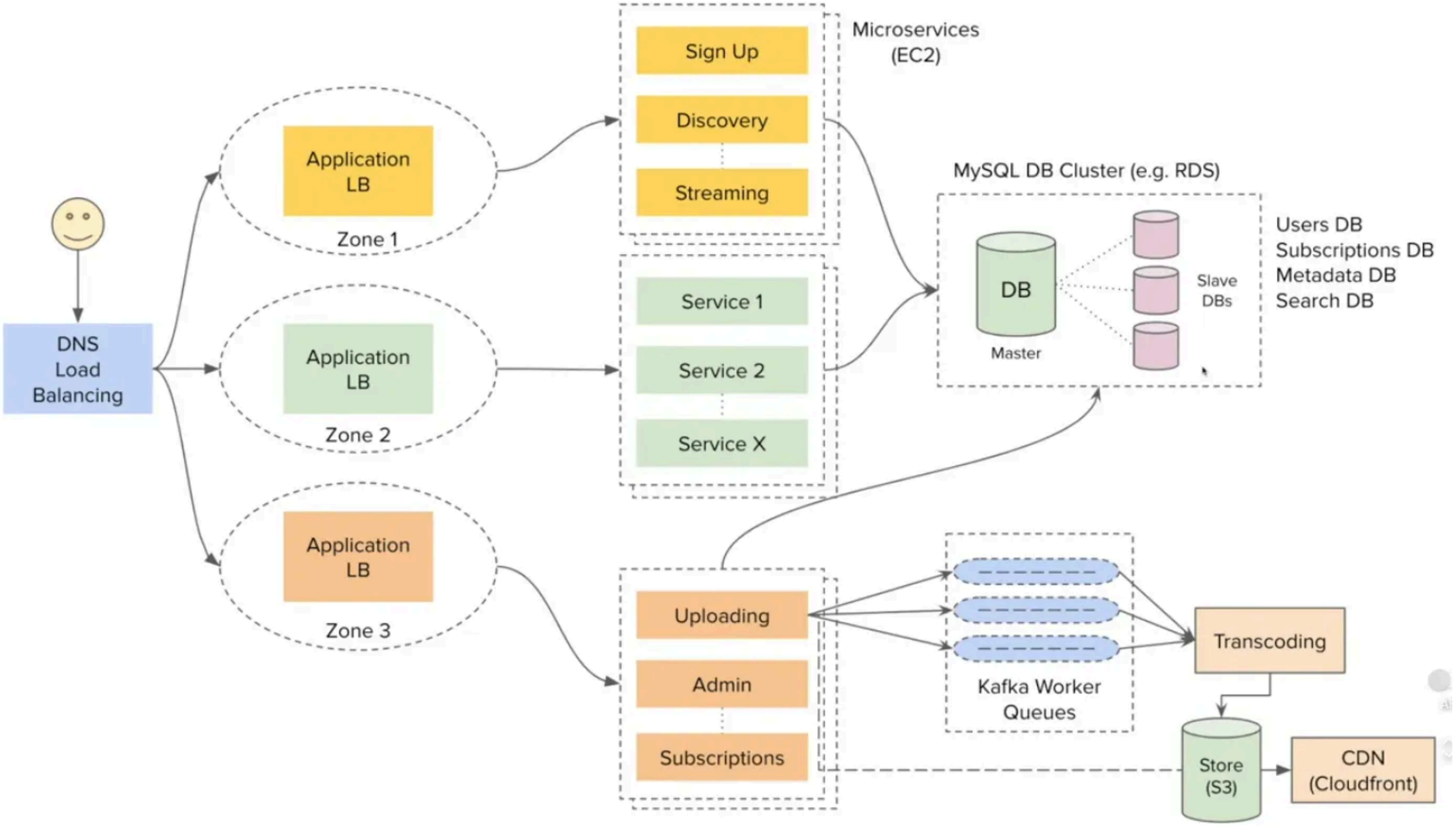
👎

- 운영 복잡성 증가
- 분산 시스템 이슈
- 개발 초기의 러닝 커브

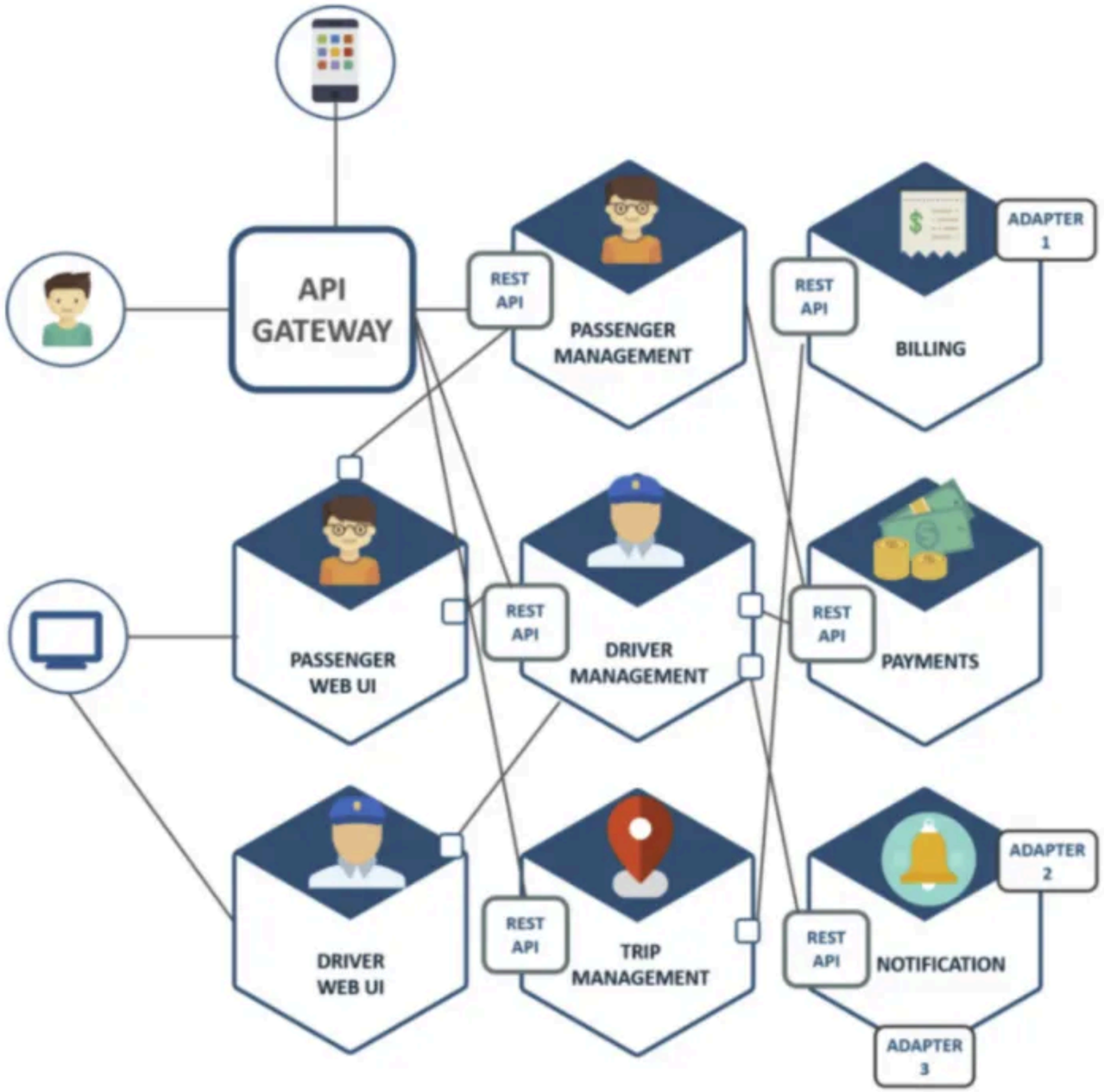
구분	모놀리식 아키텍처 (Monolithic)	마이크로서비스 아키텍처 (Microservices)
구조	하나의 통합된 애플리케이션	여러 개의 독립된 작은 서비스들
배포	전체 애플리케이션을 하나로 빌드·배포	서비스 단위로 개별 배포 가능
데이터베이스	하나의 통합 DB 사용	각 서비스가 자체 DB를 가질 수 있음
통신 방식	내부 메서드 호출	HTTP/REST, gRPC, 메시지 브로커 등 사용


사내 BlockChain Network
Ethereum Network
Avalanche Network
Data Polling
NFT
Alarm
Messenger
Data Relaying
Swap
User
API Gateway
App
L3 BlockExplorer
L3 Wallet

Netflix

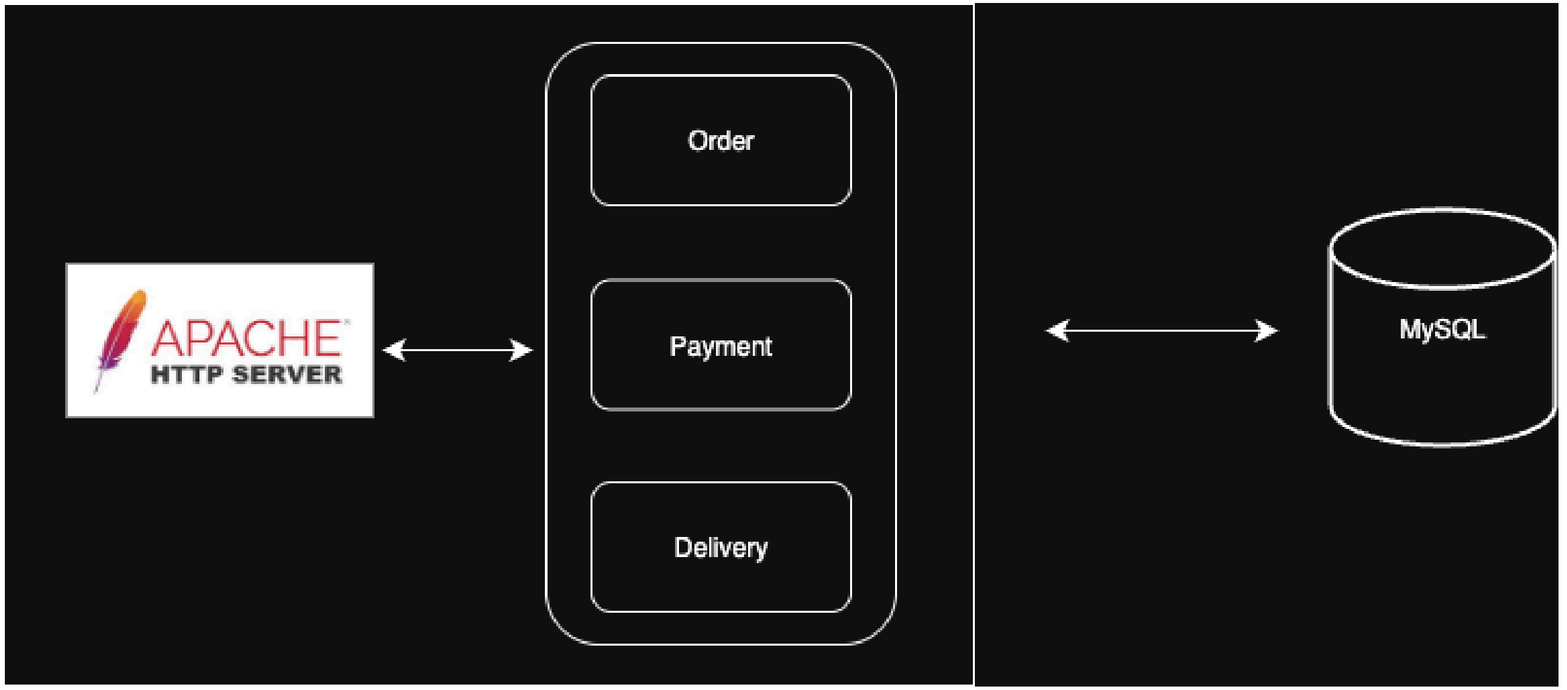

DNS
Load
Balancing
Application
LB
Zone 1
Application
LB
Zone 2
Application
LB
Zone 3
Microservices
(EC2)
Sign Up
Discovery
Streaming
Service 1
Service 2
Service X
Uploading
Admin
Subscriptions
MySQL DB Cluster (e.g. RDS)
DB
Master
Slave
DBs
Users DB
Subscriptions DB
Metadata DB
Search DB
Kafka Worker
Queues
Transcoding
Store
(S3)
CDN
(Cloudfront)

Uber


API
GATEWAY
REST
API
PASSENGER
MANAGEMENT
REST
API
BILLING
ADAPTER
1
PASSENGER
WEB UI
REST
API
DRIVER
MANAGEMENT
REST
API
PAYMENTS
DRIVER
WEB UI
REST
API
TRIP
MANAGEMENT
REST
API
NOTIFICATION
ADAPTER
2
ADAPTER
3

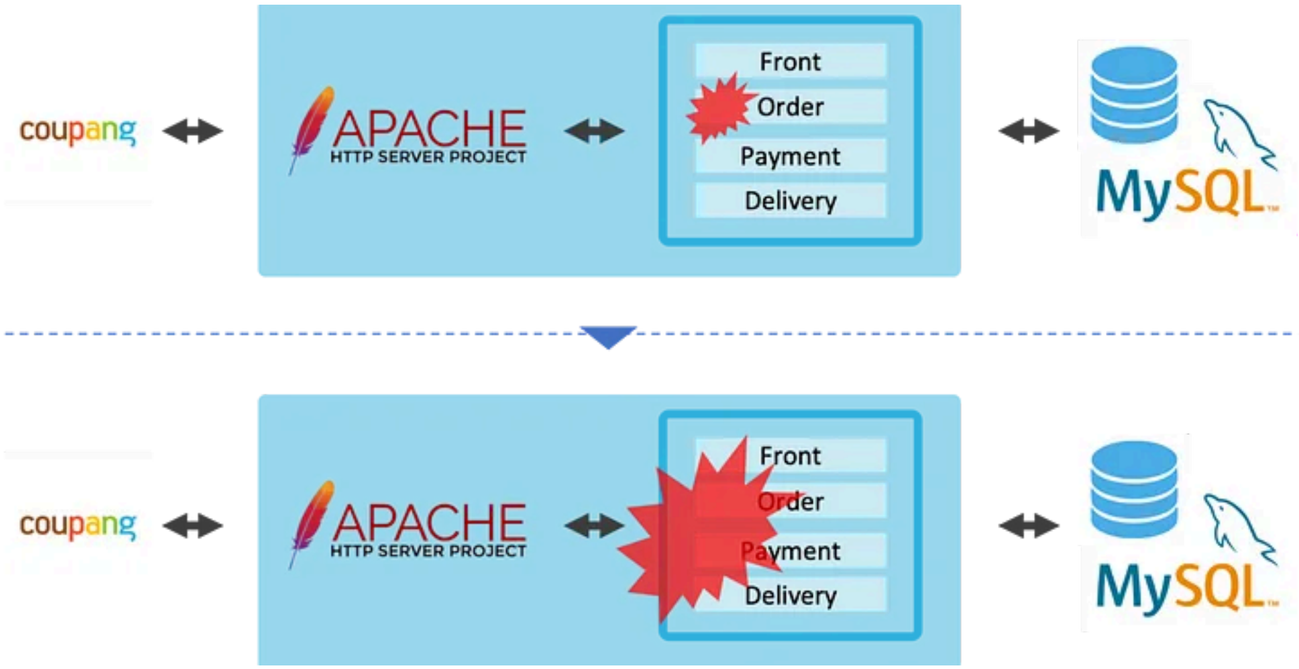
Coupang

2011년 쿠팡의 서버 아키텍처



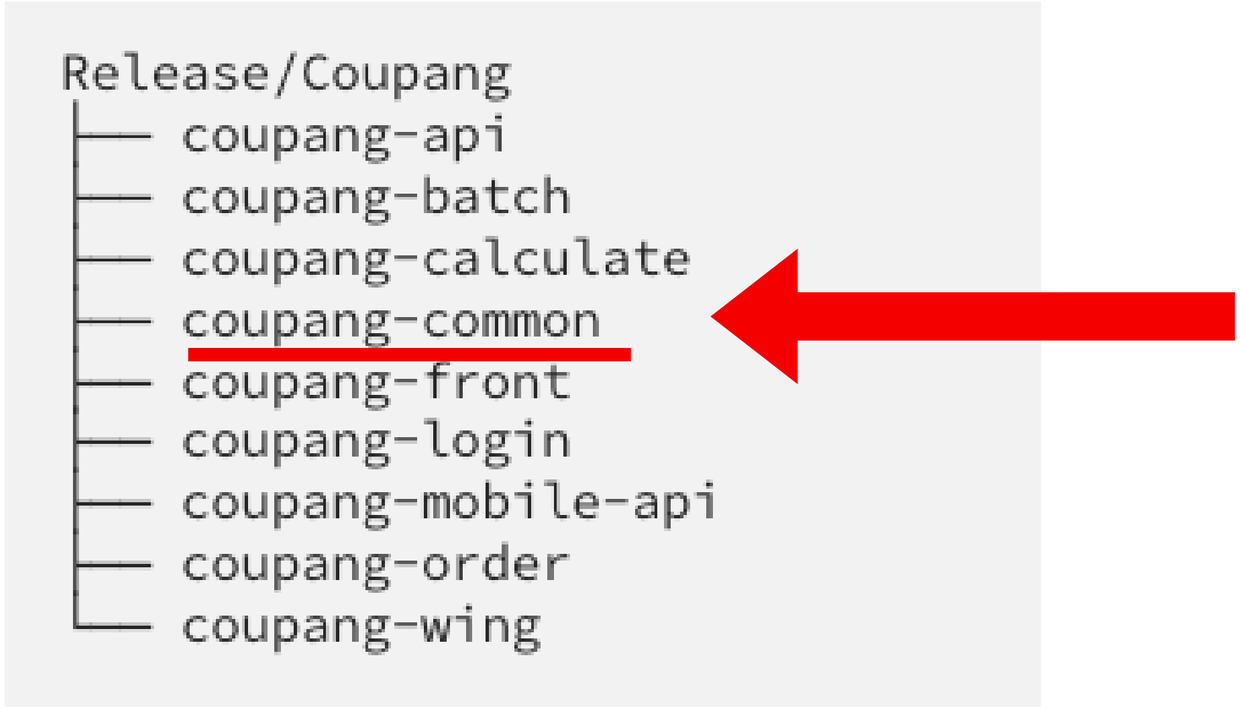
Monolothic에서의 Pain Point

부분 장애의 전체 장애로의 확대



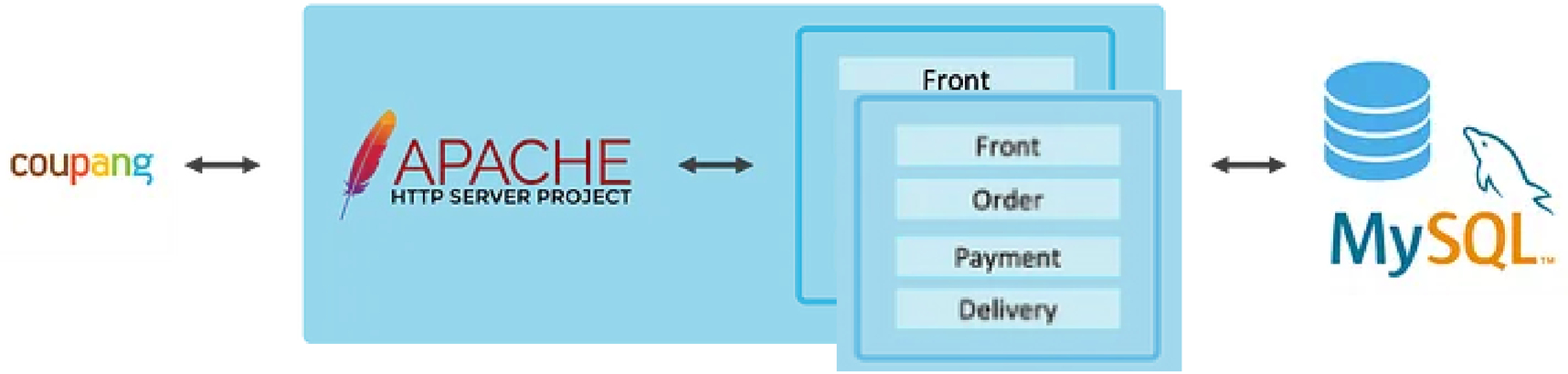
Monolothic에서의 Pain Point

레거시화의 가속



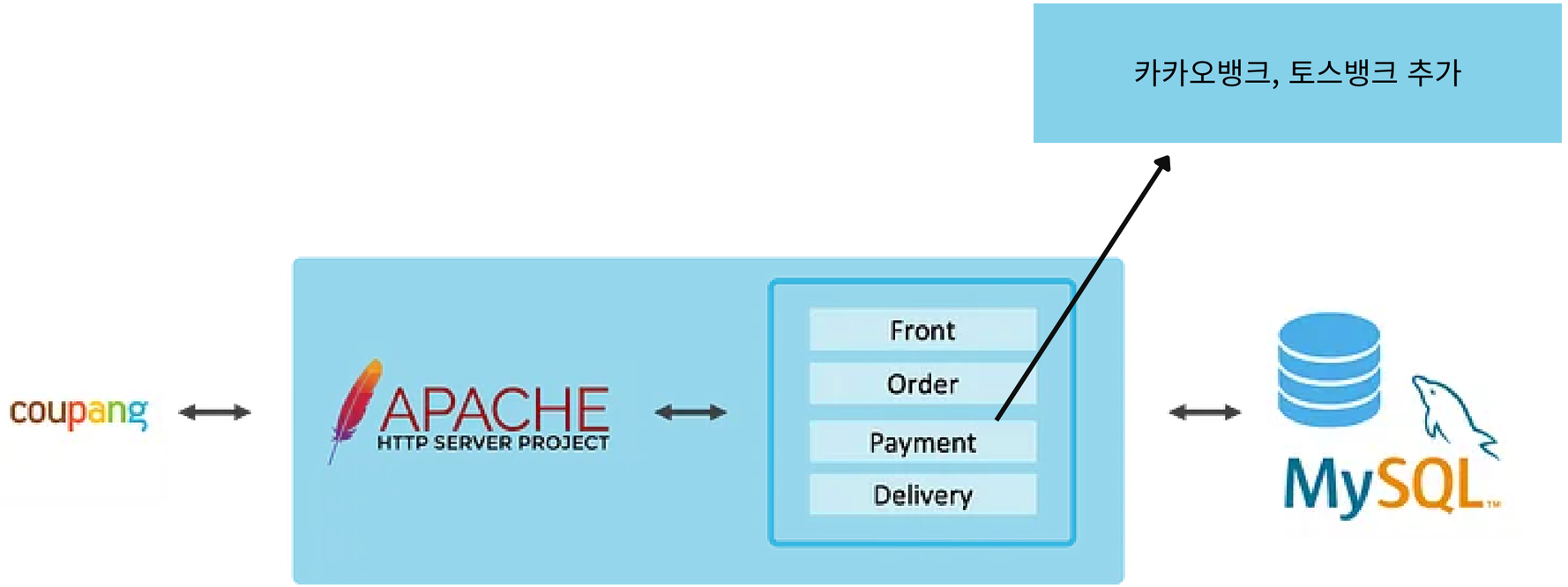
Monolothic에서의 Pain Point

확장성의 부족함



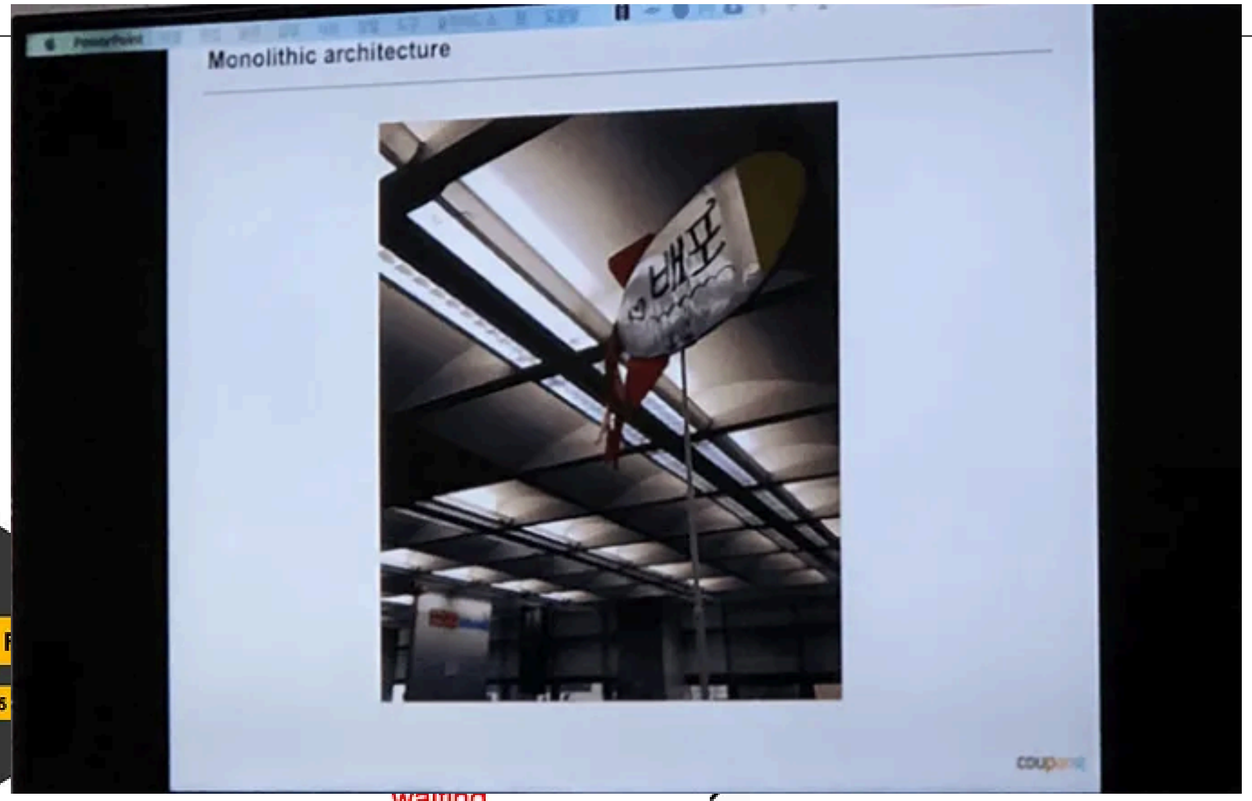
Monolothic에서의 Pain Point

테스트 비용의 증가



Monolothic에서의 Pain Point

배포 시간에 대한 기하급수적인 증가



Feature11
Feature 9
Feature13
Feature 6
Feature 7
Feature11
Feature 5
Feature10
Feature 8
Feature12

waiting

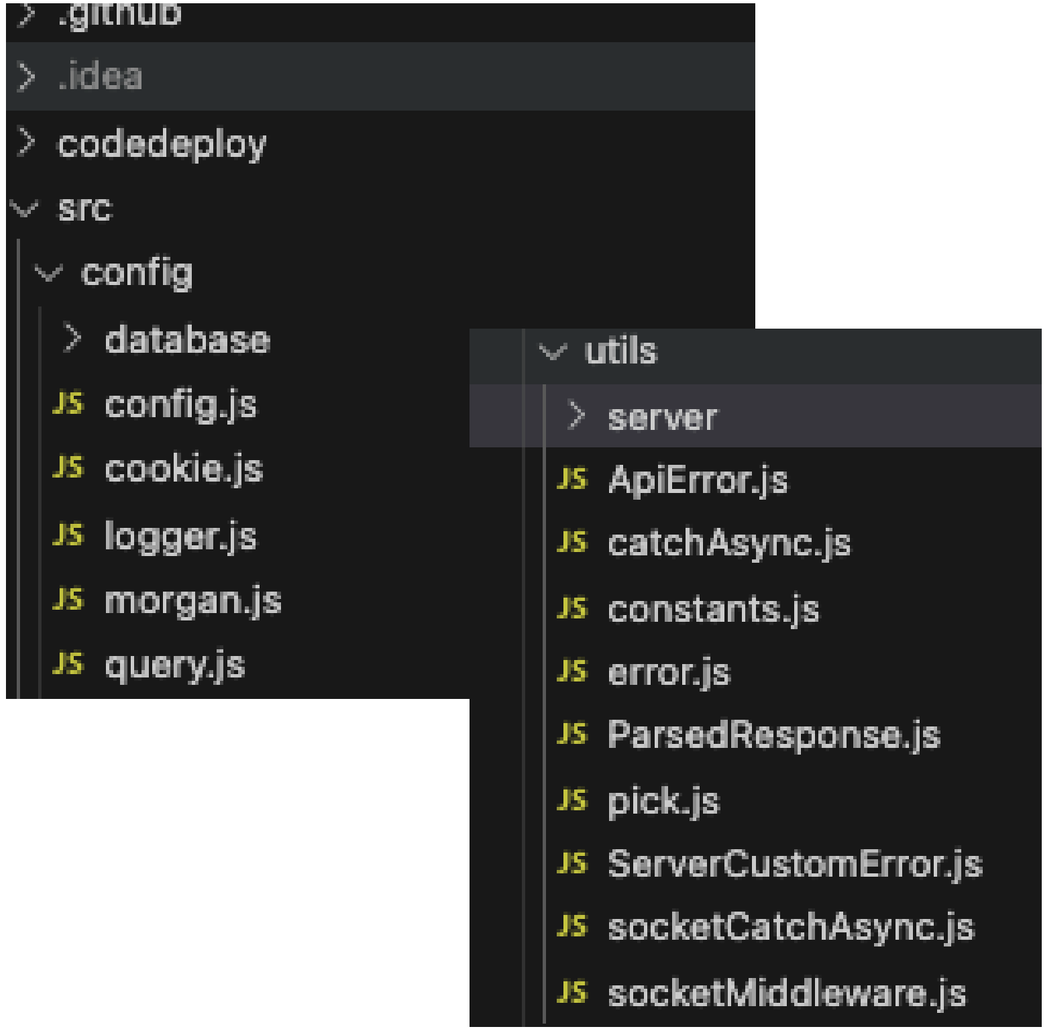
waiting

전환 프로젝트 Vitamin Project

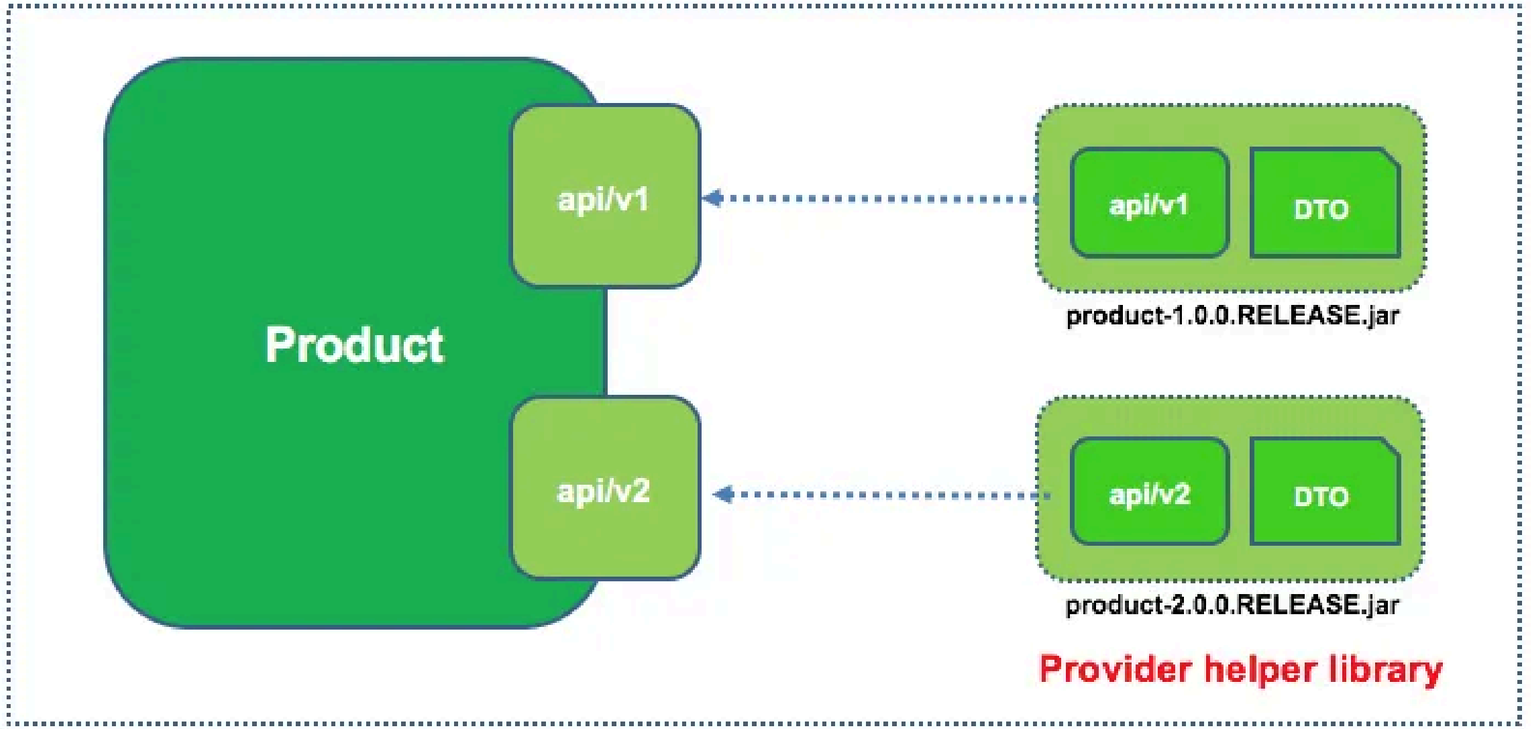
전략 1. Vitamin Framework의 제공

도메인 단위의 개발을 돕는 기본 코드 템플릿

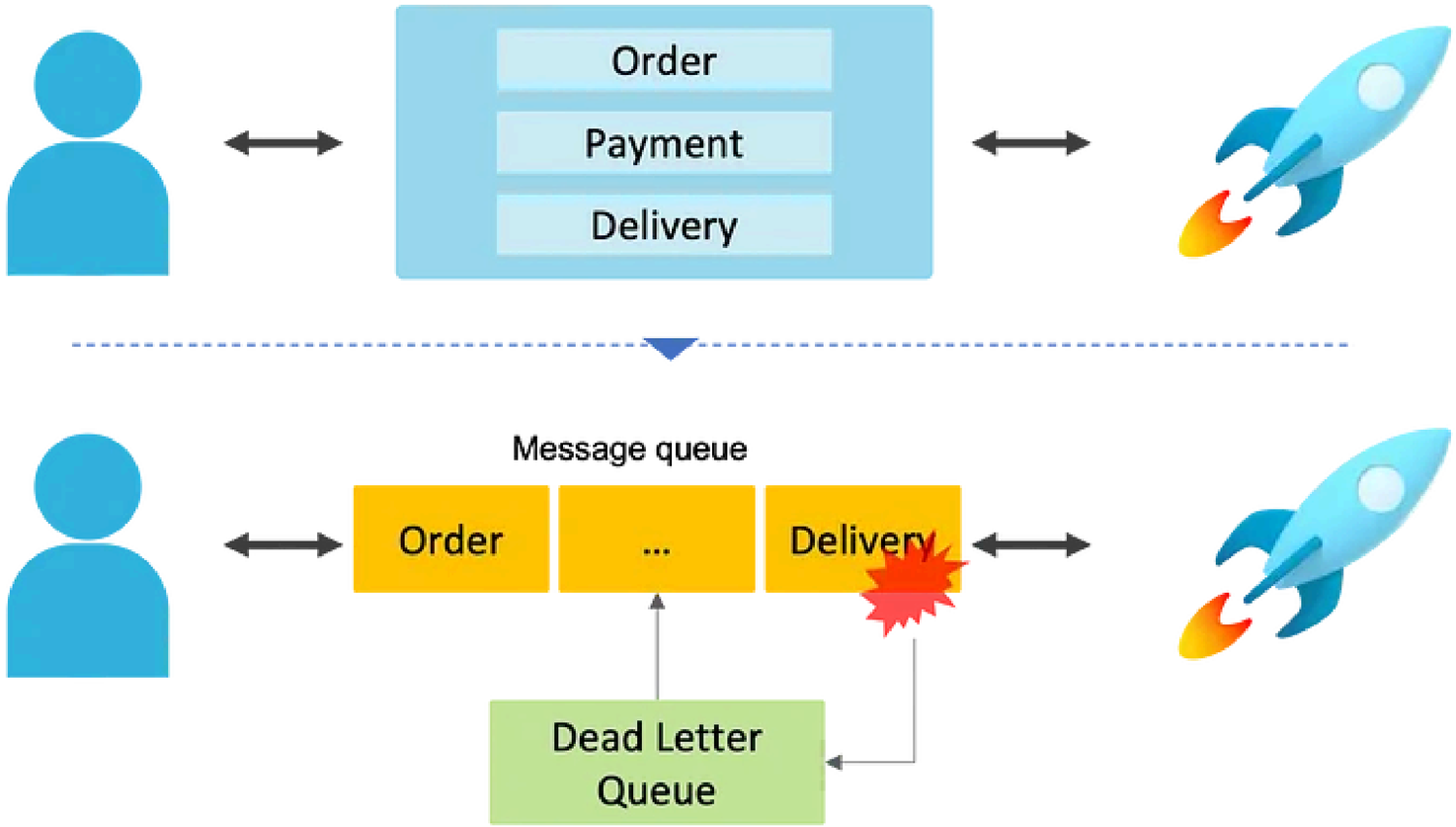
테스트, 배포, 내부 플랫폼 서비스와의 간단한 연동



전략 2. API Helper Library 제공

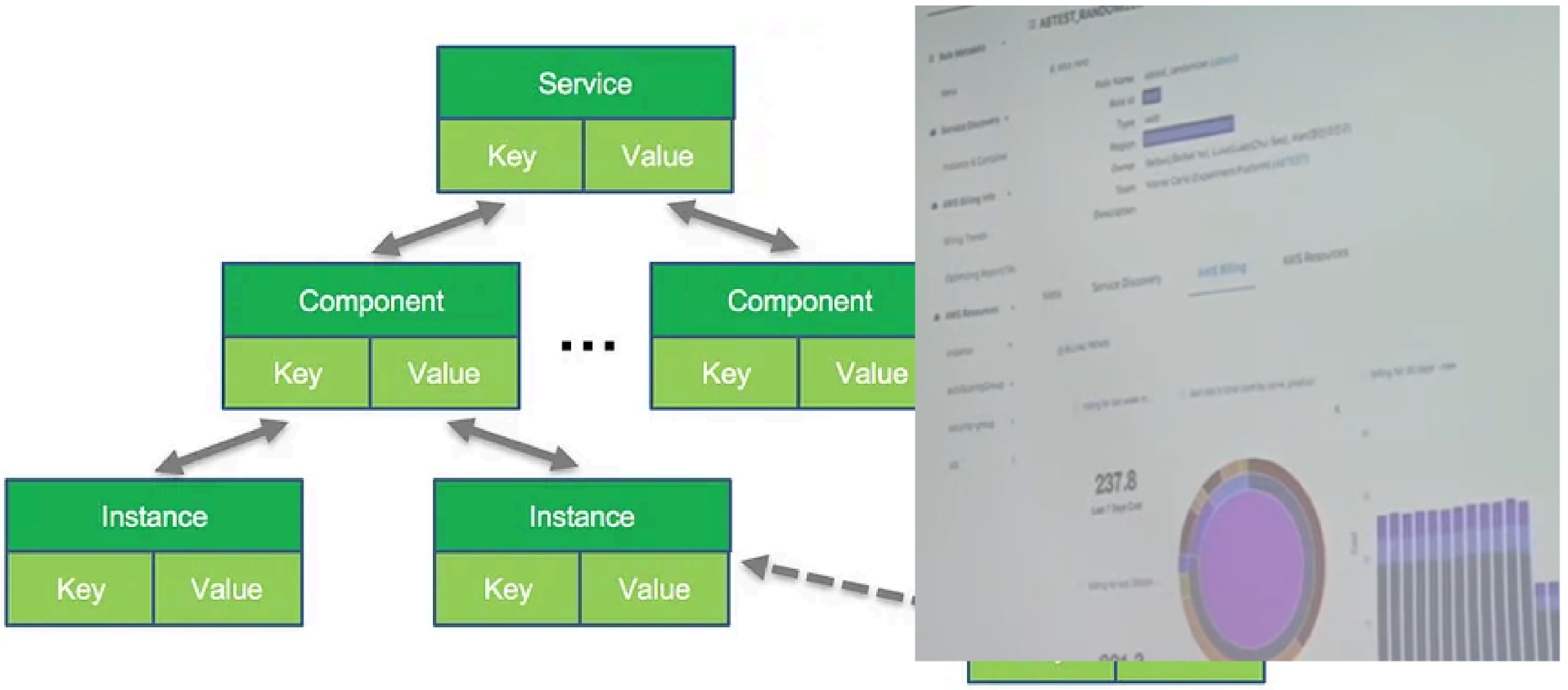


전략 3. 메시지 큐의 활용


Order
Payment
Delivery
Message queue
Order
...
Delivery
Dead Letter Queue

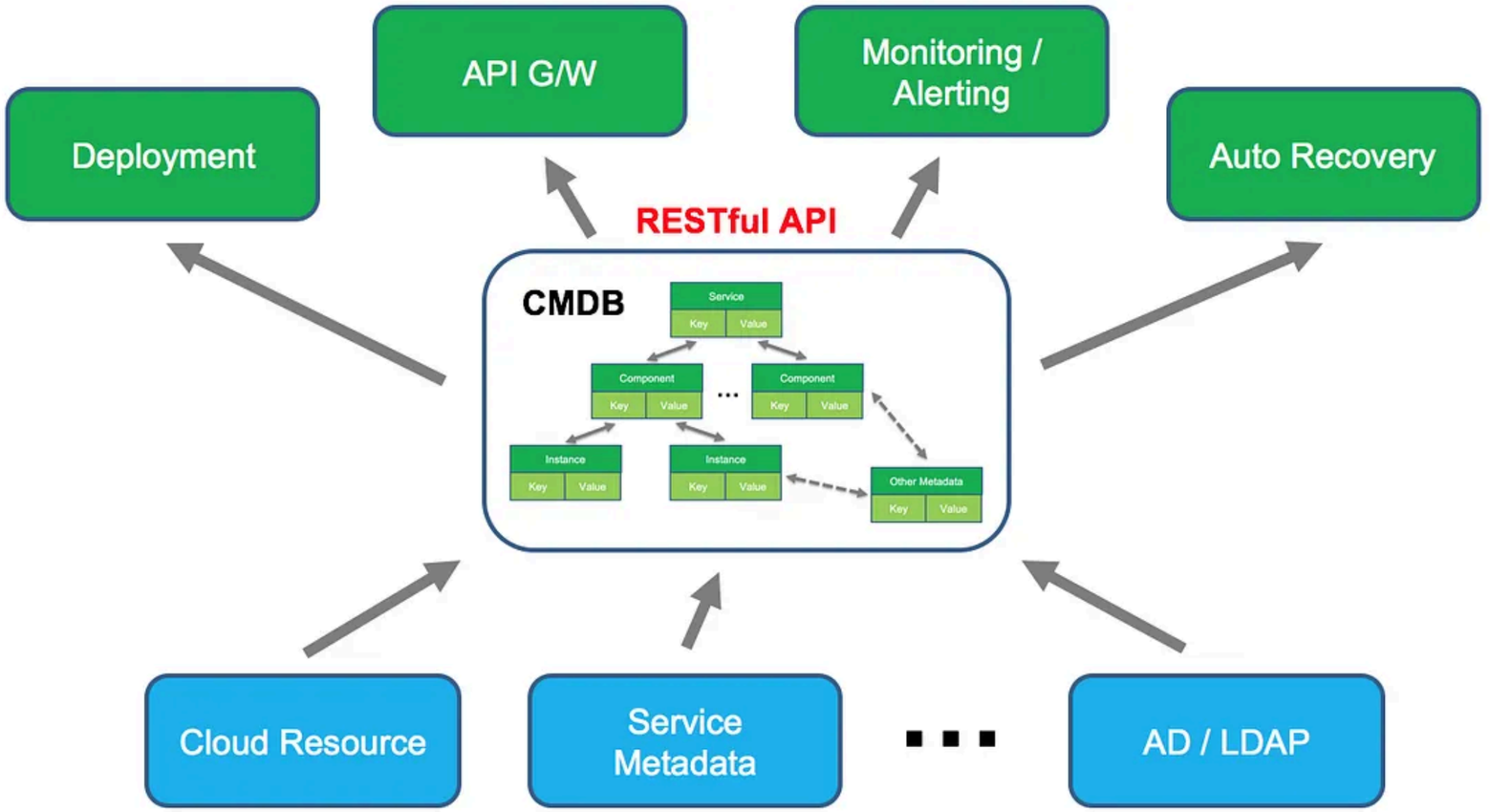
전환 후 운영 및 장애 극복을 위한 시스템

인프라 및 자원 관리 : CMDB(Configuration Management Database)



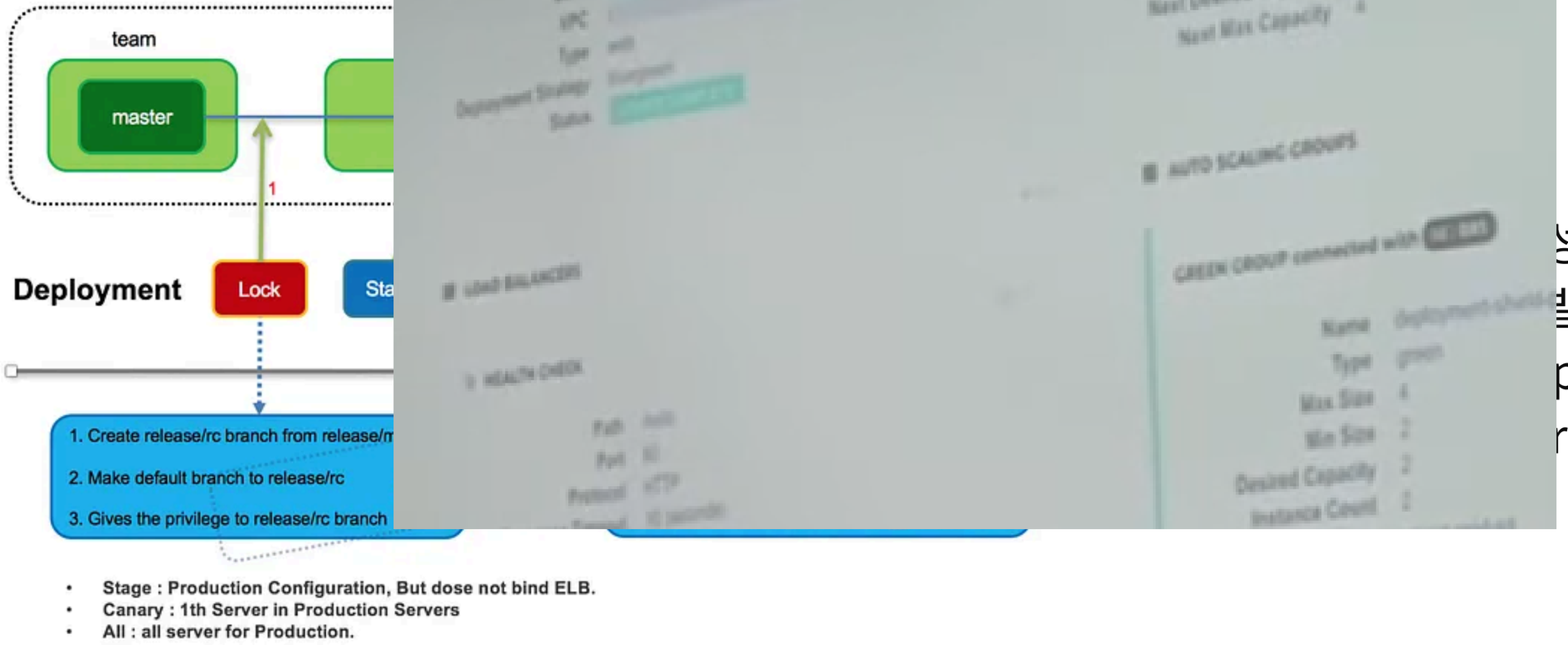
전환 후 운영 및 장애 극복을 위한 시스템

인프라 및 자원 관리 : CMDB(Configuration Management Database)



전환 후 운영 및 장애 극복을 위한 시스템

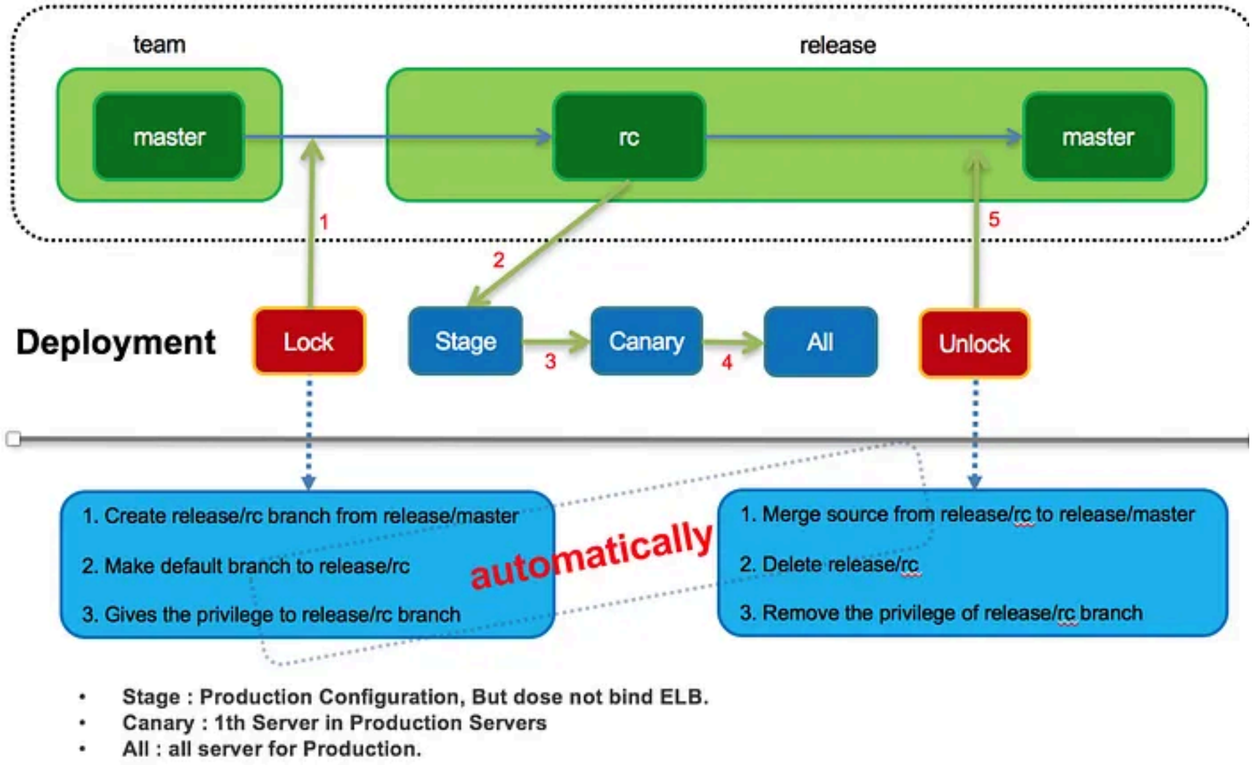
배포 관리 : Bolt2 배포 시스템



넝
발 비용 최소화
p 통한 리소스 효율화
rvice warm-up

전환 후 운영 및 장애 극복을 위한 시스템

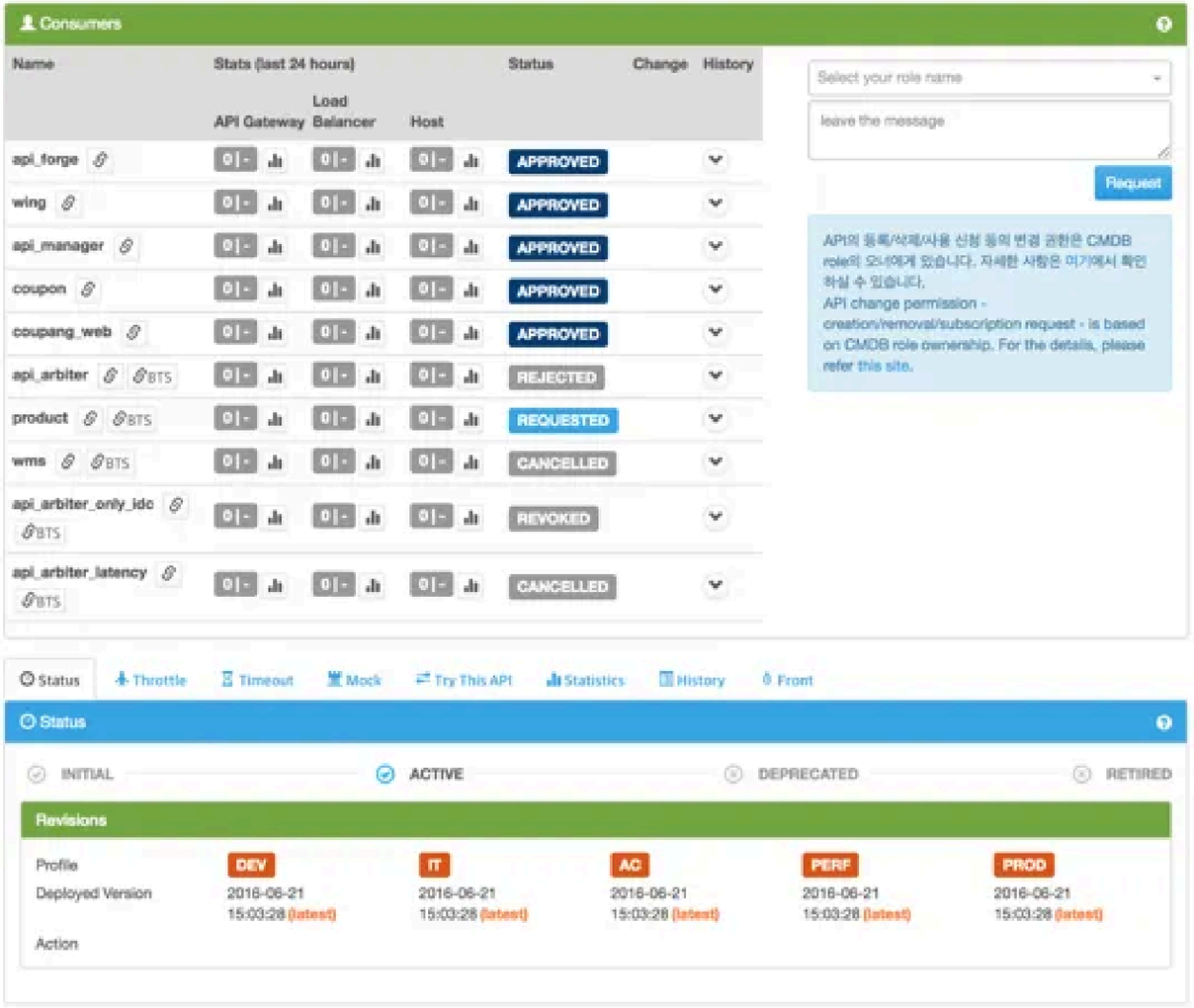
배포 관리 : Bolt2 배포 시스템



1. Lock
2. Stage
3. Canary
4. All
5. Unlock

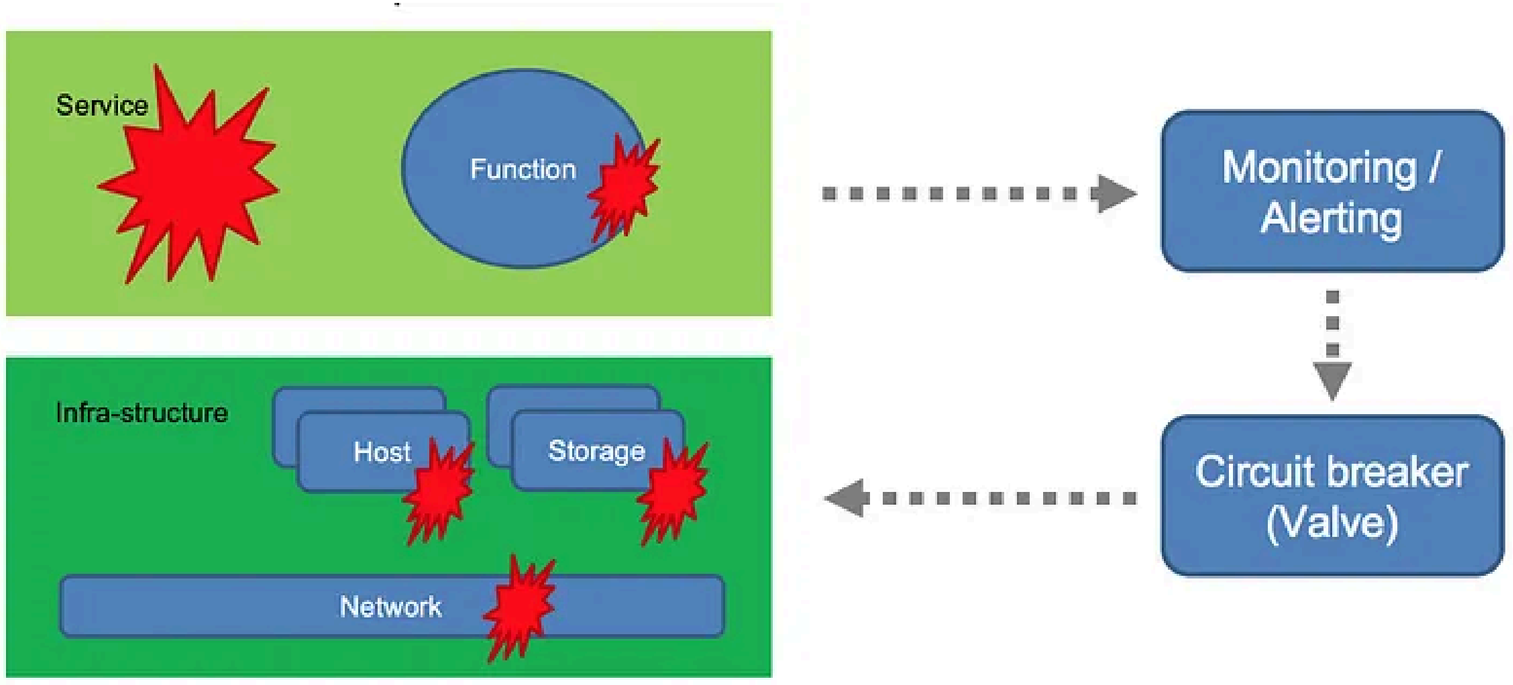
MSA 전환 후 운영 및 장애 극복을 위한 시스템

API Gateway



MSA 전환 후 운영 및 장애 극복을 위한 시스템

장애 탐지 및 복구 : Circuit breaker system



[Go] 당근마켓에서의 Golang 도입, 그리고 4년 간의 기록

1. Starvation mode of Mutex
2. CPU Throttling
3. GC Frequency가 높은 현상

GC 튜닝

CPU 사용량이 40% → 25%

GC Frequency 400회 → 8회

CPU Time 30% → 1%

Pod count 245개 → 120개

autopprof

Automatically profile the Go applications when CPU or memory utilization crosses specific threshold levels against the Linux container CPU quota and memory limit.

그래서 우리는?

왜에 대한 이해

CS 지식의 깊이

간단한 경험

Spring Cloud Netflix

3.0.0-SNAPSHOT

This project provides Netflix OSS integrations for Spring Boot apps through autoconfiguration and binding to the Spring Environment and other Spring programming model idioms. With a few simple annotations you can quickly enable and configure the common patterns inside your application and build large distributed systems with battle-tested Netflix components. The patterns provided include Service Discovery (Eureka), Circuit Breaker (Hystrix), Intelligent Routing (Zuul) and Client Side Load Balancing (Ribbon).

Thank You