Deploy on K8S

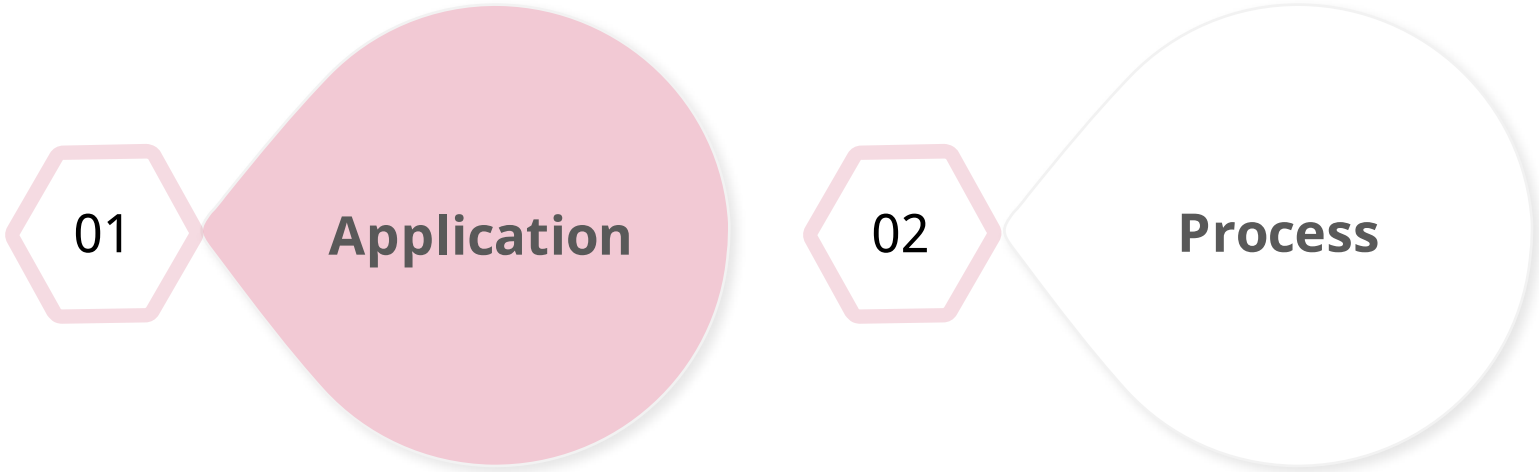
V.1.0

2018.08.29

SK 주식회사

Table of Contents

PART 01. Overview




어벤져스:인피니티 워
새로운 조합을 이룬 어벤져스, 역대 최강 빌런
타노스에 맞서 세계의 운명이 걸린 인피니티스
톤을 향한 못한 대결이 펼쳐진다! 마블의 클라이
맥스를 목격하라!
MARVEL STUDIOS
AVENGER
INFINITY WA
APRIL 25

액션 모험


오션스 트웰브


트로이


ROBIN HOOD


MARVEL STUDIOS
어벤져스

코미디


러브 액츄얼리


킷 스마트


섹스 앤 더 시티 2


JIM CARREY
트루먼쇼


어벤져스:인피니티 워
새로운 조합을 이룬 어벤져스, 역대 최강 빌런
타노스에 맞서 세계의 운명이 걸린 인피니티스
톤을 향한 못한 대결이 펼쳐진다! 마블의 클라이
맥스를 목격하라!
MARVEL STUDIOS
AVENGER
INFINITY WA
APRIL 25

액션 모험


오션스 트웰브


트로이


ROBIN HOOD


MARVEL STUDIOS
어벤져스

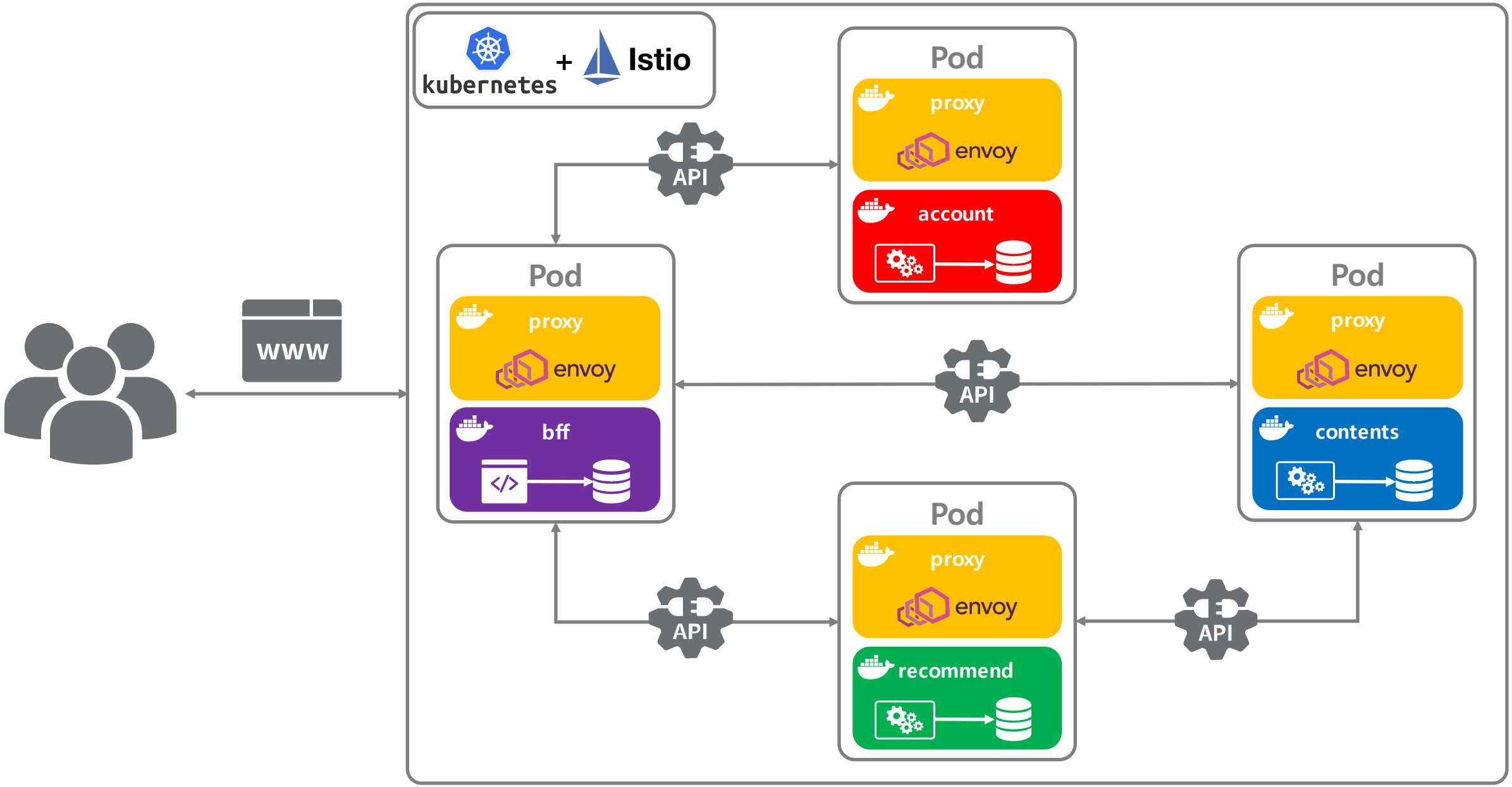
코미디


러브 액츄얼리

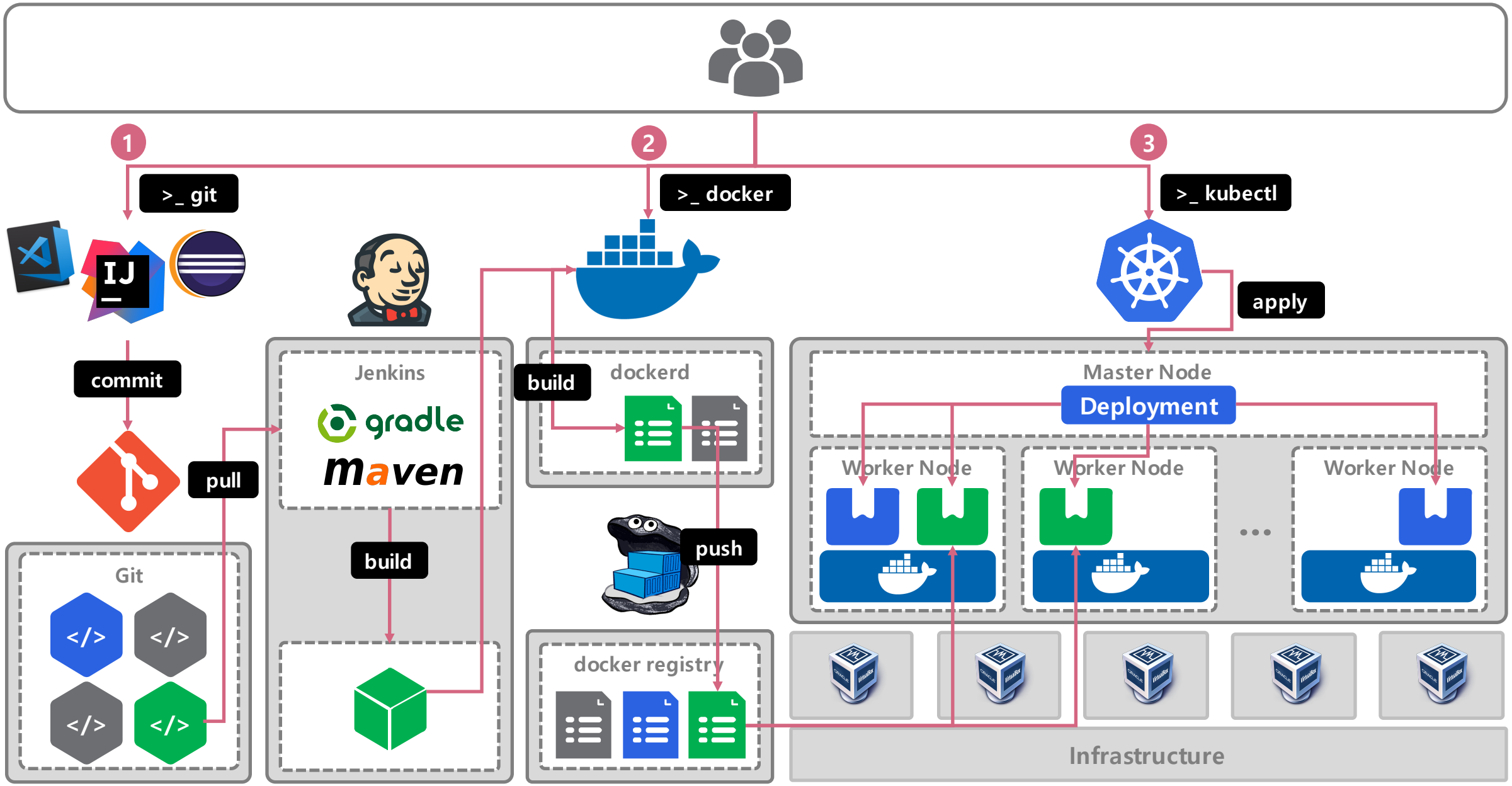

킹스맨


섹스 앤 더 시티 2

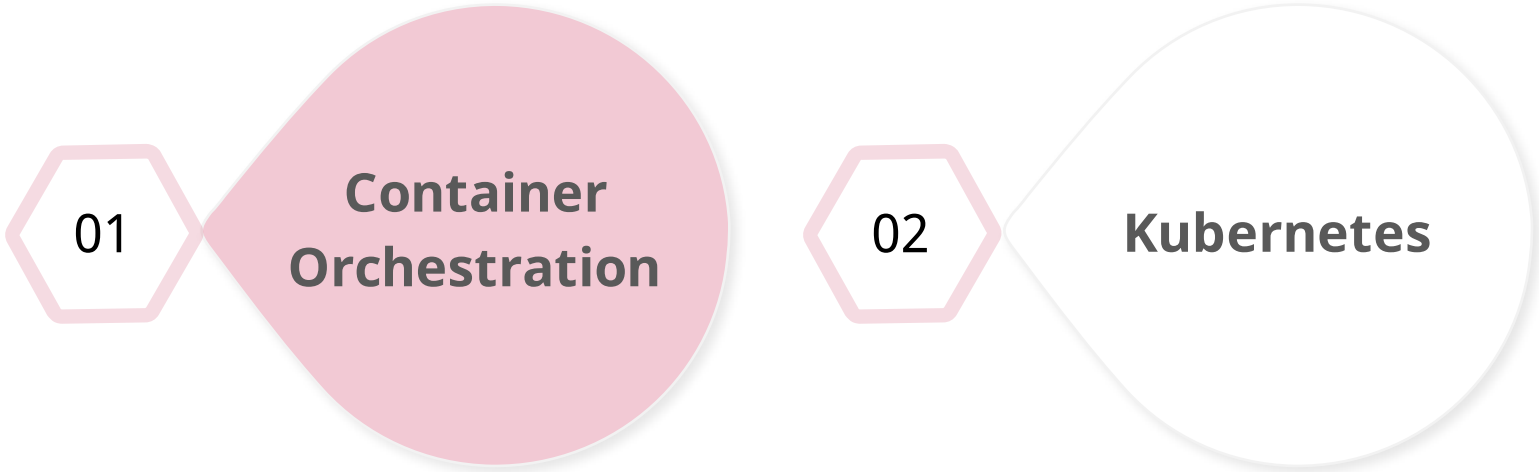

JIM CARREY
트루먼쇼



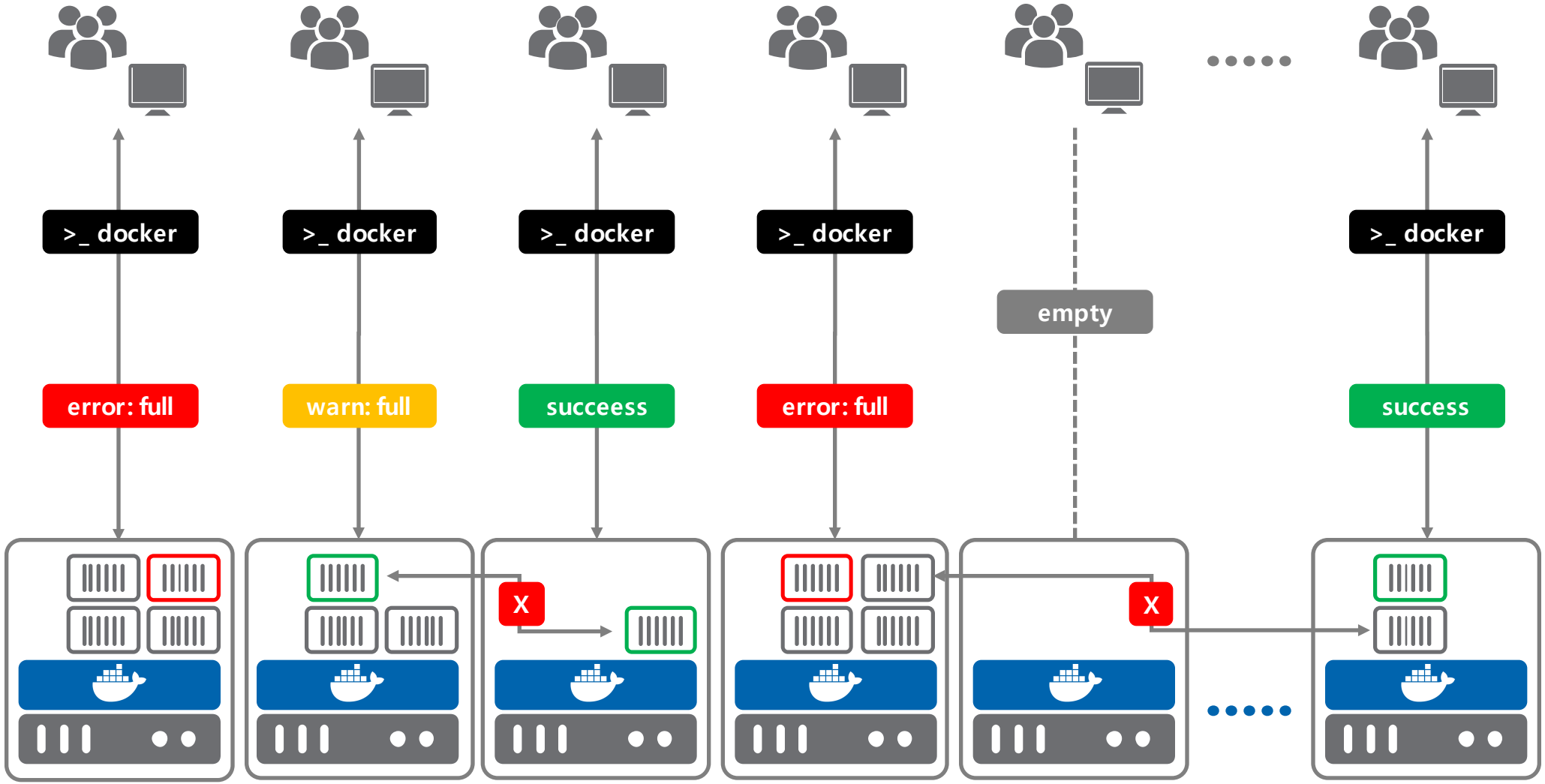
PART 01. Overview



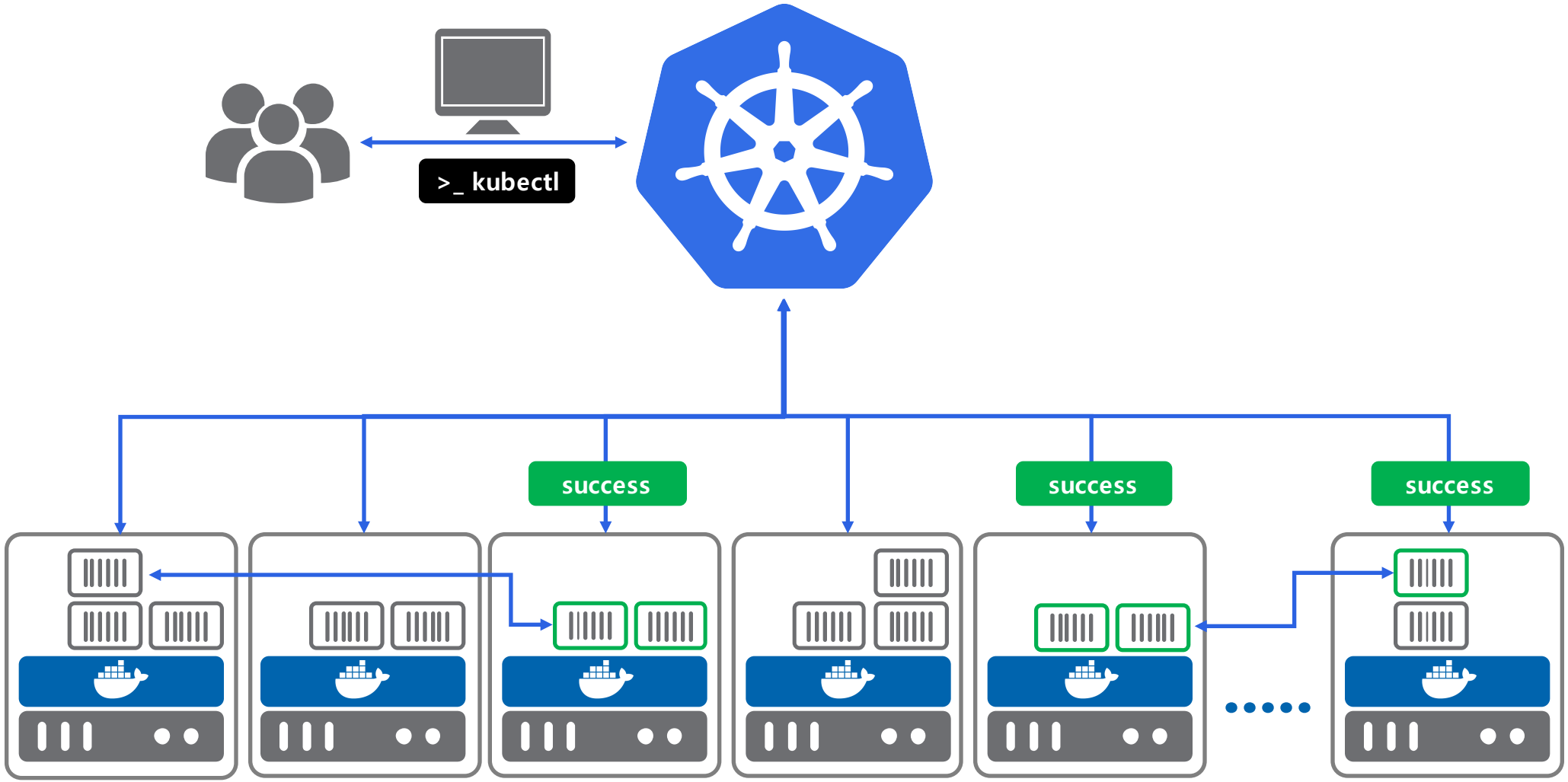
PART 02. What is Kubernetes



Docker Cluster 환경에서 효율적으로 운영할 수 있는가?



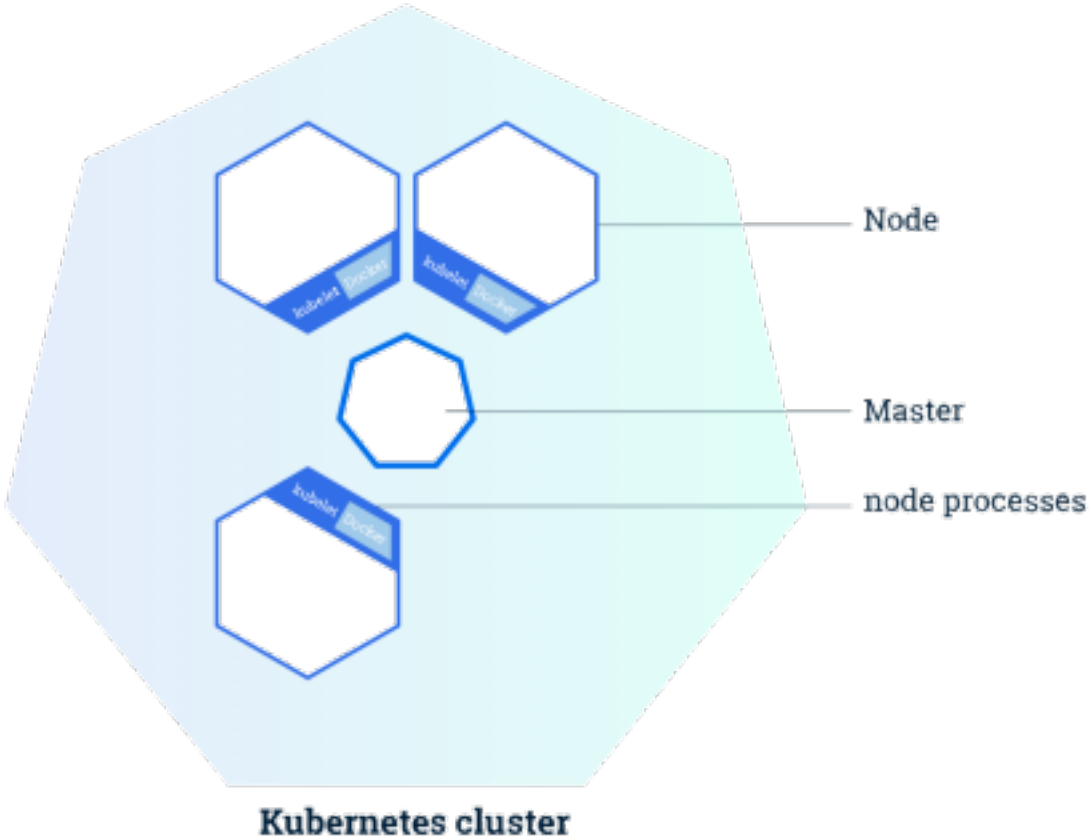
Docker Cluster를 중앙에서 조정할 수 있는 무엇이 필요하다**!**



PART 02. What is Kubernetes

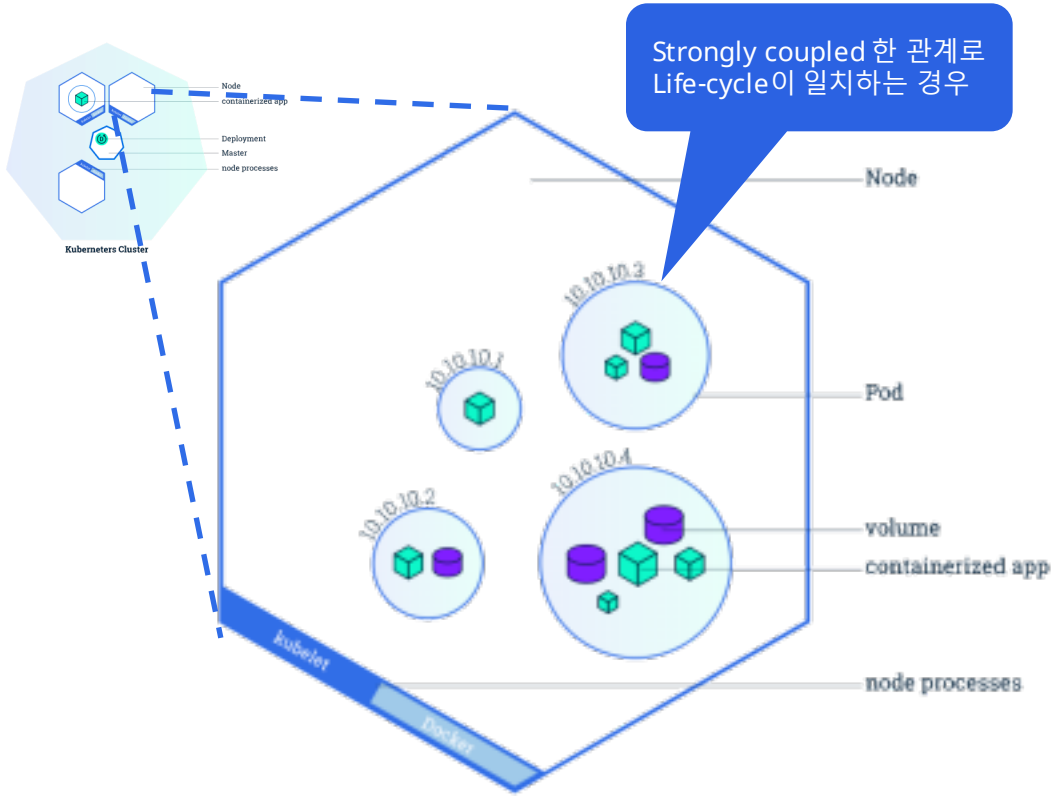
Kubernetes is an open-source system for **automating deployment, scaling, and management of containerized applications.**

Kubernetes Architecture



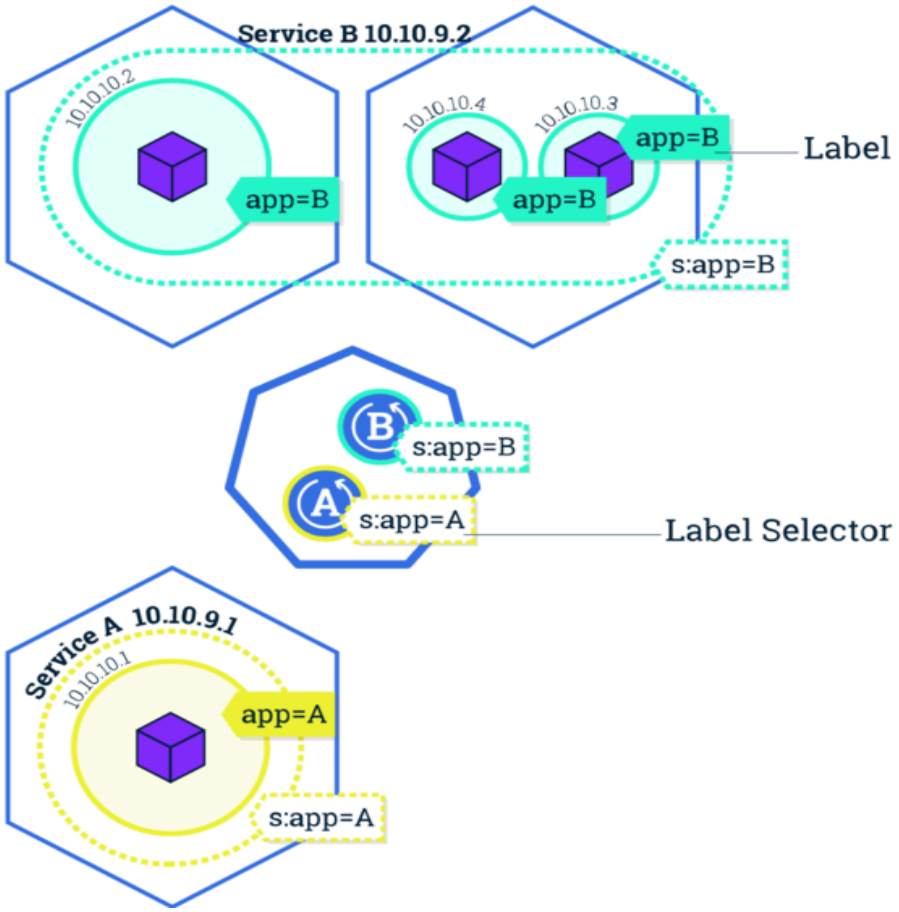
- **Kubernetes Cluster**
  - 하나의 클러스터 안에 Master와 여러개의 Node로 구성
- **Master**
  - 노드를 관리하는 주체
  - 노드의 글로벌 이벤트를 감지하고 응답하는 등 의사결정을 수행
- **Kube API Server**
  - API 개체에 대한 유효성을 검사, 구성
- **Controller Manager**
  - 핵심 Kubernetes Controller가 실행되는 프로세스
  - 실제 노드에 배포될 Pod, Endpoint 등을 생성, 업데이트, 삭제하는 매니저 역할
  - Deployment, StatefulSets 등 기능에 따라 다수의 컨트롤러가 존재
- **Scheduler**
  - 노드에 아직 할당되지 않은 Pod들을 감시하고 실행될 노드를 선택
- **etcd**
  - Kubernetes의 저장소로 활용

Kubernetes Architecture



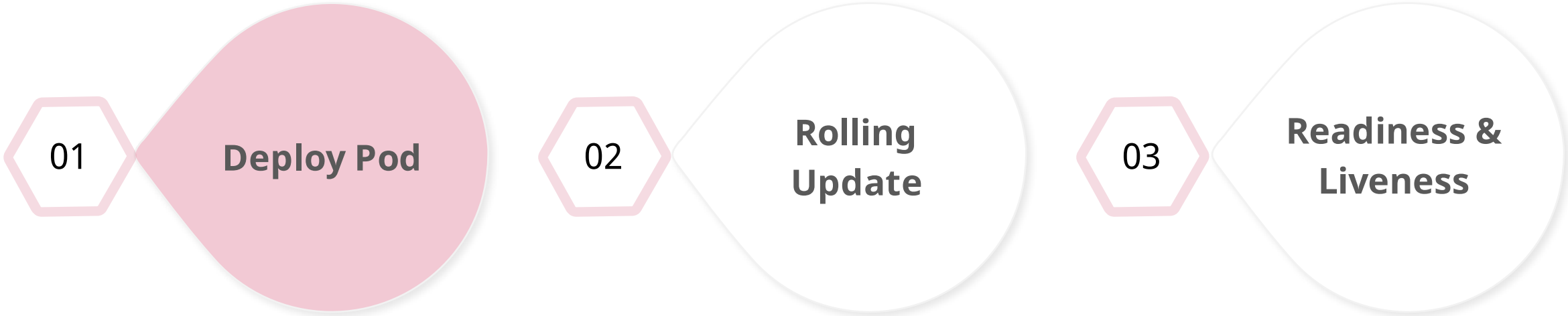
- **Node**
  - VM 또는 실제 머신을 의미
  - kubelet이라는 에이전트를 통해 마스터와 통신
  - 실제 컨테이너인 Pod가 생성되는 곳
- **Pod**
  - 컨테이너화된 App.이 배포되는 컴포넌트
  - 복수개의 컨테이너로 구성할 수도 있음
  - Scaling, Replication 의 단위
- **Deployment**
  - 애플리케이션의 배포/삭제, Scale out의 역할
  - Deployment를 생성하면 Pod, ReplicaSets를 함께 생성
  - Pod이 생성되면서 컨테이너화된 애플리케이션들이 배포되고, ReplicaSets은 애플리케이션의 Replica수를 지속적으로 모니터링하고 유지시킴

Kubernetes Architecture



- Service
  - Pod의 논리적 집합과 액세스 정책을 정의하는 추상화 개념
  - 서비스가 타겟으로 하는 Pod의 집합은 Label Selector에 의해 결정
  - 로드밸런싱, 서비스 디스커버리 등 제공
  - 서비스를 통해 외부에서 접속하거나 클러스터 내부에서만 접근하도록 설정 가능
- Label
  - Pod와 같은 Object에 설정되는 key/value 쌍의 별칭
- Selector
  - 타 Object를 찾기 위한 별칭 검색어
  - 해당 서비스에서 관리하는 Pod의 목록(실제 Pod의 엔드포인트)을 맵핑하고 있는 Endpoint 객체가 자동으로 생성
  - Pod이 죽으면 자동으로 Endpoint에서 제거되고, Selector와 매칭되는 새로운 Pod가 생기면 자동으로 추가됨

PART 03. Deploy on K8S



K8S Cluster 정보 확인하기

1. K8S Cluster 환경에 제대로 접속되었는지 확인

```
$ kubectl cluster-info
```

2. 사용할 Namespace 제대로 지정되었는지 확인

```
$ kubectl config get-contexts
```

bff-service 배포하기

1. Deployment를 배포하기 위한 **bff-service-deployment.yaml**을 작성

```
apiVersion: apps/v1beta2 # for versions before 1.8.0 use apps/v1beta1
kind: Deployment
metadata:
  name: bff-service-deployment
  labels:
    app: bff-service
spec:
  replicas: 1
  selector:
    matchLabels:
      app: bff-service
  template:
    metadata:
          labels:
        app: bff-service
        spec:
      containers:
      - name: bff-service
        image: dtlabs/bff-service:1.0
              ports:
        - containerPort: 8080
        imagePullPolicy: Always
        env:
        - name: SPRING_PROFILES_ACTIVE
          value: k8s
            resources:
               requests:
                  memory: "256Mi"
                  cpu: "50m"
               limits:
                  memory:  "1Gi"
                  cpu: "500m"
```

bff-service 배포하기

2. bff-service-deployment.yaml을 작성한 경로로 이동

```
$ cd $(bff-service-deployment.yaml이 작성된 경로)
```

3. Deployment Object 생성: ***--record*** 옵션을 통해 이력 관리 가능

```
$ #kubectl apply –f $(filename)
$ kubectl apply –f bff-service-deployment.yaml --record
```

4. Deployment Object 조회: Deployment, Pod 조회

```
$ kubectl get deploy,po
```

bff-service 외부에 노출하기

1. 배포한 bff-service를 외부에 노출하기 위한 **bff-service.yaml**을 작성

```
apiVersion: v1
kind: Service
metadata:
  name: bff-service
spec:
  ports:
  - name: “http”
    port: 8080
    targetPort: 8080
  selector:
    app: bff-service
  type: NodePort
```

2. bff-service.yaml을 작성한 경로로 이동

```
$ cd $(bff-service.yaml이 작성된 경로)
```

3. Service Object 생성

```
$ #kubectl apply –f $(filename)
$ kubectl apply –f  bff-service.yaml
```

4. Service Object 조회

```
$ kubectl get svc
```

5. bff-service 접속

PART 03. Deploy on K8S

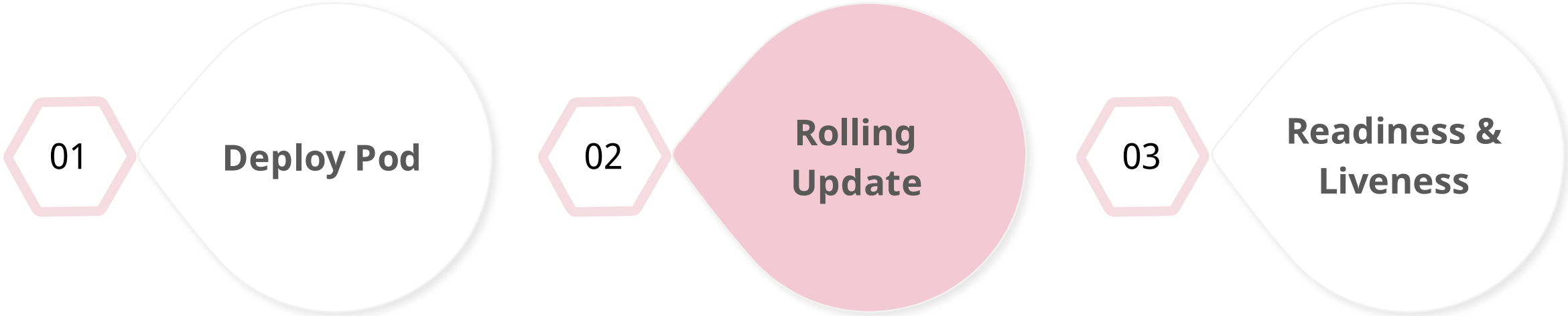


Image Update 하기

1. bff-service Image를 2.0 버전으로 Update

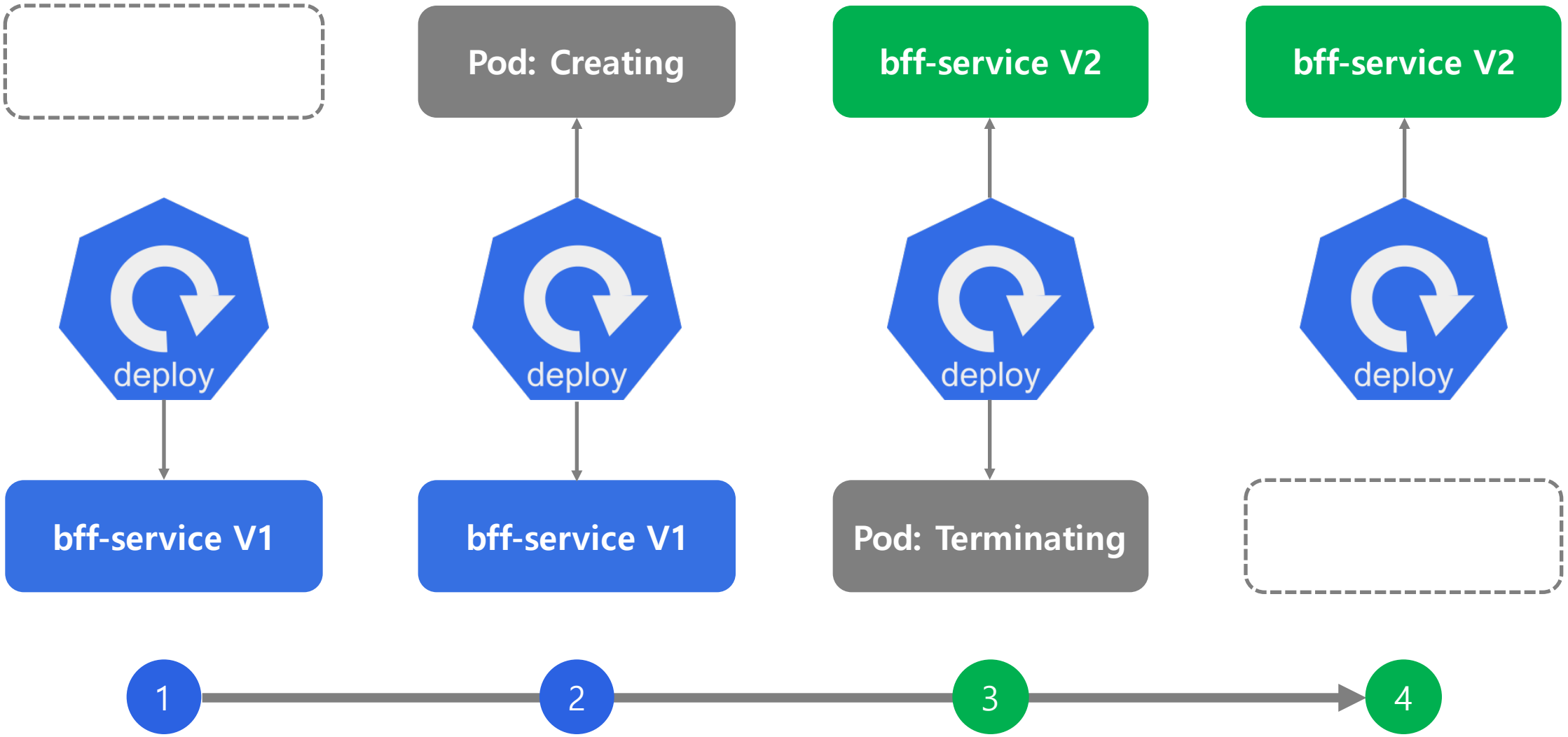
```
$ # kubectl set image deployment/$(deployment_name) $(container_name)=$(image_name)
$ kubectl set image deployment/bff-service-deployment bff-service=dtlabs/bff-service:2.0 --record
```

2. Image가 Update되어 Pod이 새로 생성됨을 확인

```
$ kubectl describe deployment bff-service-deployment
```

3. bff-service에 다시 접속하기

Image Update 하기



Rollback 하기

1. Deployment Object의 Histroy 확인

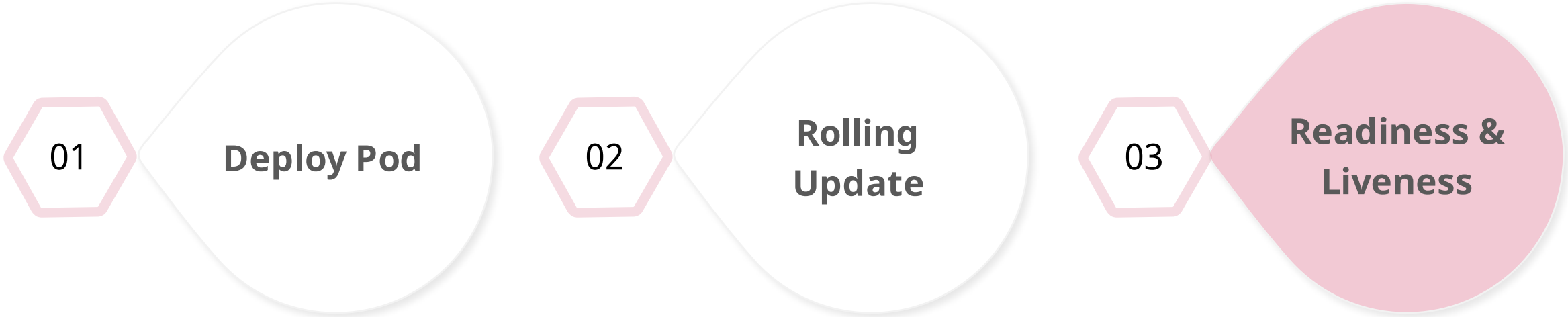
```
$ # kubectl rollout history deployment/$(deployment_name)
$ kubectl rollout history deployment/bff-service-deployment
```

2. 확인한 History의 Revision을 통해 Rollback 수행: ***--to-revision*** 옵션으로 Revision 지정 가능

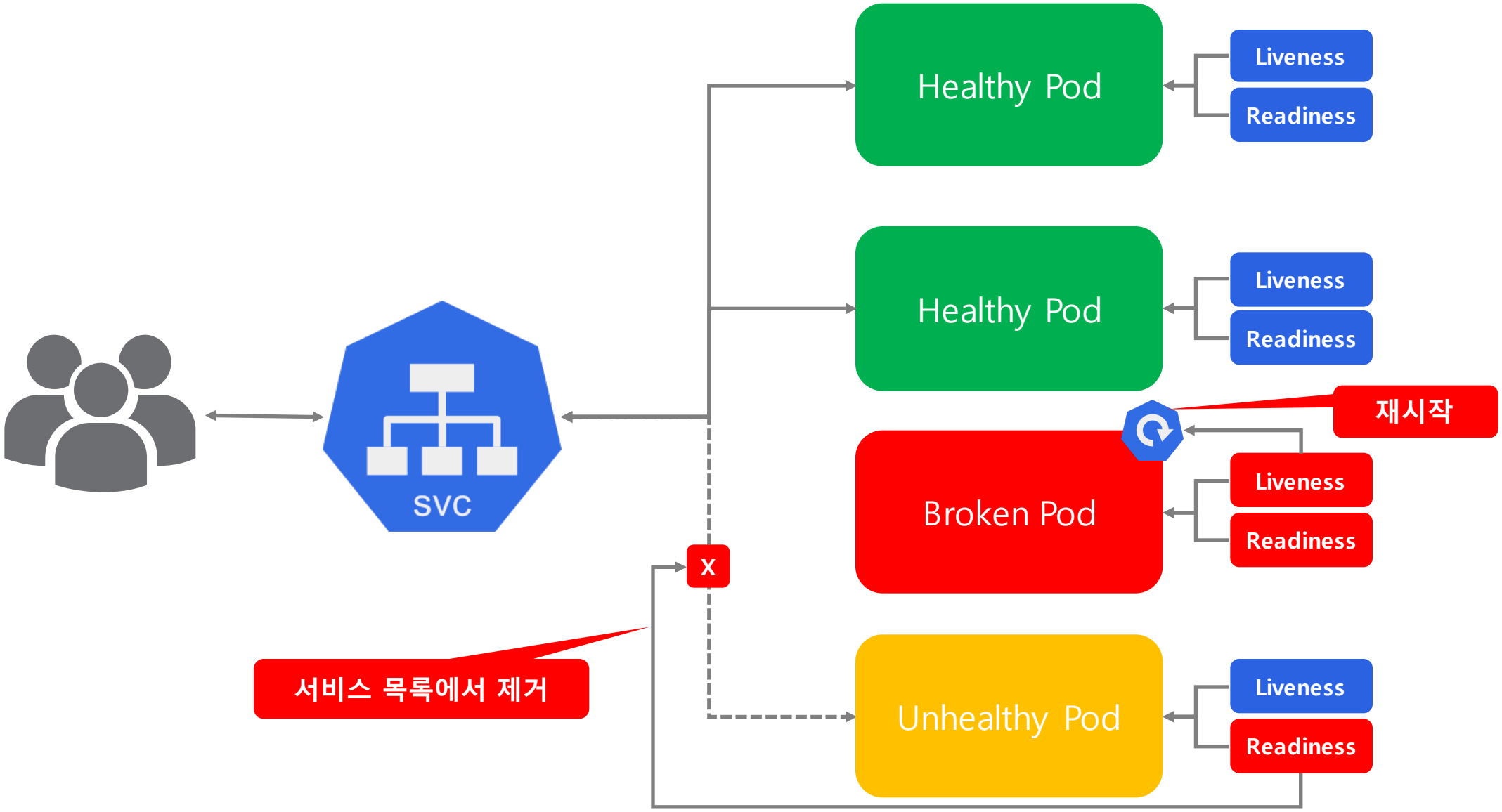
```
$ # kubectl rollout undo deployment/$(deployment_name) --to-revision=$(revision)
$ kubectl rollout undo deployment/bff-service-deployment --to-revision=1
```

3. bff-service로 다시 접속해 보기

PART 03. Deploy on K8S



V2를 서비스 중단 없이 사용자에게 제공하려면?



readinessProbe & livenessProbe 적용하기

1. bff-service-deployment.yaml에 readinessProbe & livenessProbe 주석 제거

```
containers:
- name: bff-service

                ... 생략 ...

  readinessProbe:
    httpGet:
      path: /v1/promotions
      port: 8080
    initialDelaySeconds: 30
    periodSeconds: 10
  livenessProbe:
    exec:
      command:
      - cat
      - bff-service
    initialDelaySeconds: 60
    periodSeconds: 5
```

2. Deployment Object 재배포

```
$ kubectl apply –f  bff-service-deployment.yaml
```

3. bff-service V2로 Image Update

```
$ kubectl set image deployment/bff-service-deployment  bff-service=dtlabs/bff-service:2.0  --record
```

4. bff-service 접속

어벤져스:인피니티 워

새로운 조합을 이룬 어벤져스, 역대 최강 빌런 타노스에 맞서 세계의 운명이 걸린 인피니티스톤을 향한 모한 대결이 펼쳐진다! 마블의 클라이맥스를 목격하라!

재생 ▶ 내 동영상 목록 +

액션 모험

코미디

감사합니다