1. 쿠버네티스 소개

도커

- **이미지**: 애플리케이션과 해당 환경을 패키지화 한 것
- **레지스트리**: 도커 이미지를 저장하고 다른 사람이나 컴퓨터 간에 해당 이미지를 쉽게 공유할 수 있는 저장소
  - 대표적으로 도커 허브가 있다.
- **컨테이너**: 도커 기반 컨테이너 이미지에서 생성된 일반적인 리눅스 컨테이너를 의미
  - 호스트와 호스트에서 실행중인 다른 프로세스와 완전히 격리되어 있다.
  - 할당된 리소스 양만 액세스하고 사용할 수 있다.

그럼 기존 가상머신과 차이?

이미지 레이어

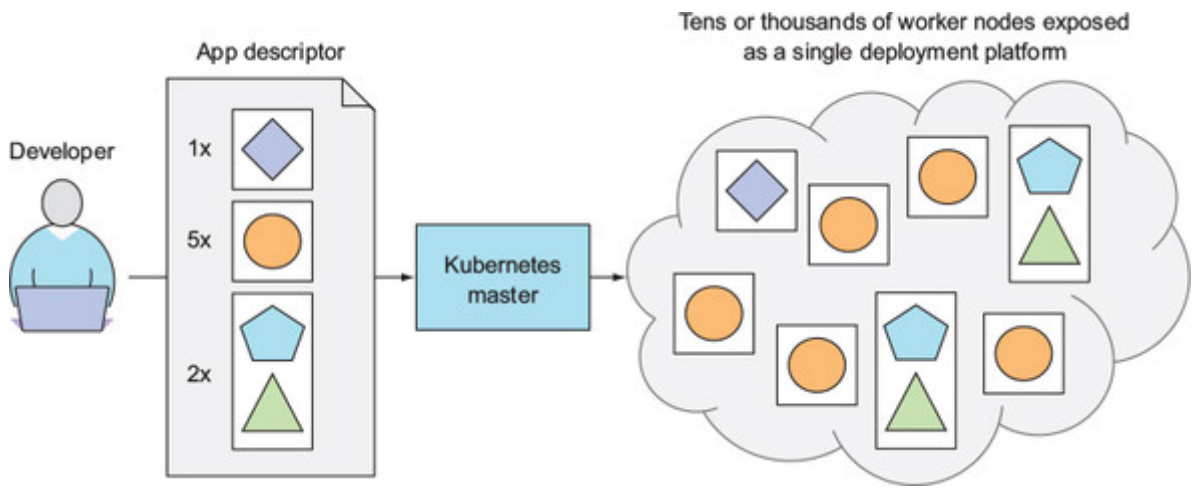
- 도커 이미지는 여러 이미지 레이어로 구성된다.
- 2개의 다른 이미지일지라도 기본 이미지로 동일한 부모 이미지를 사용할 수 있다. 이런 식으로 공유되는 이미지를 같이 활용할 수 있는 것.
- 공유되는 이미지가 있다면 이미 전송하거나, 받은 이미지가 존재한다면 다시 전송하거나 받을 필요가 없어서 배포 및 다운로드 순서가 빨라지게 된다.
- 각 레이어는 동일 호스트에 한 번만 저장되어 공유 및 활용되어진다.

쿠버네티스

배포 가능한 애플리케이션 구성요소의 수가 많아짐에 따라 모든 구성요소의 관리가 더 어려워짐.

- 엄청난 규모의 배포관리를 해야하는 솔루션이 필요.

구성요소



마스터 노드와 워커노드로 구성

- 애플리케이션 매니페스트를 마스터에 게시하면 쿠버네티스는 해당 애플리케이션을 워크 노드 클러스터에 배포

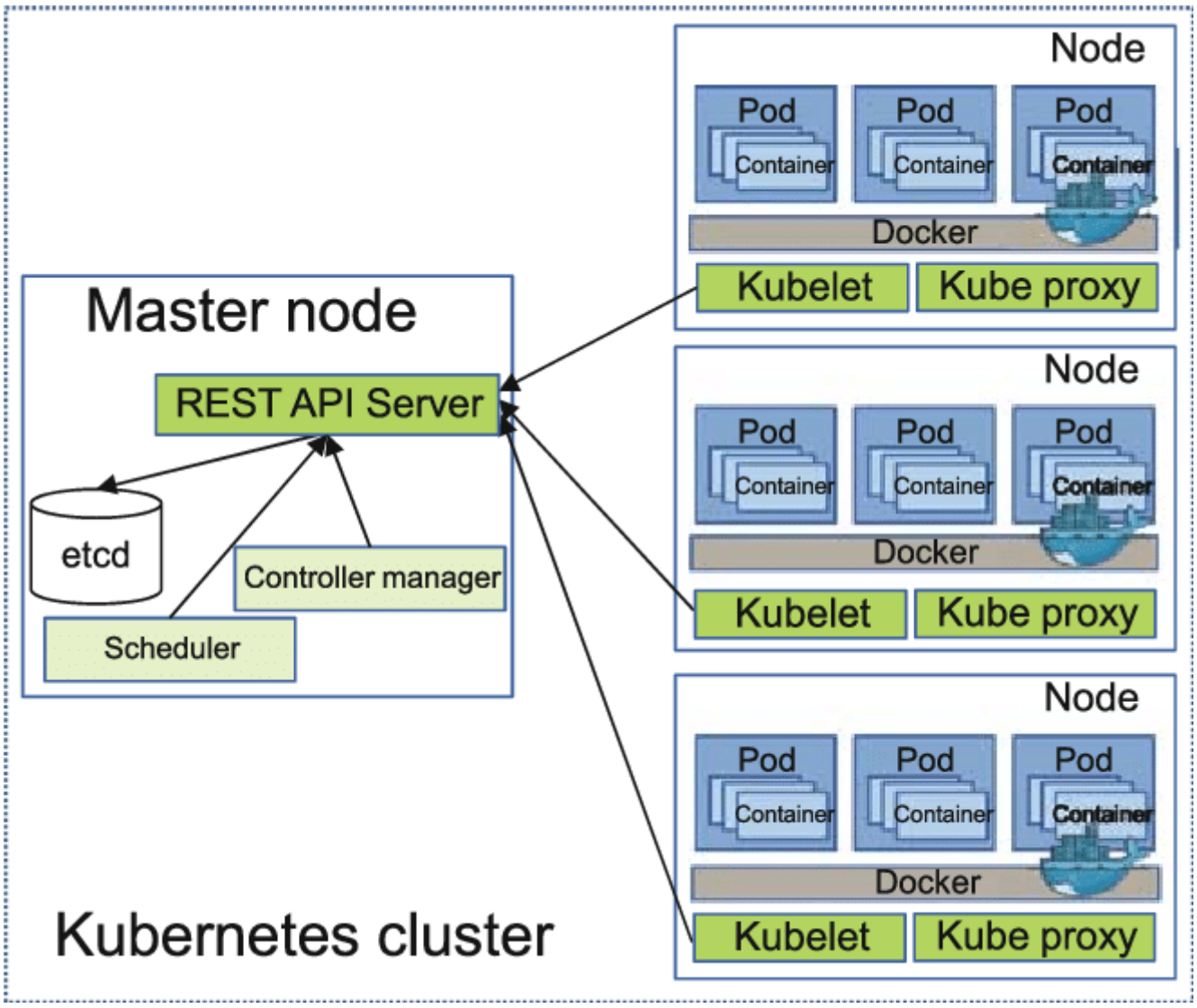
개발자는 애플리케이션 핵심기능에만 집중!

애플리케이션 개발자가 특정 인프라 관련 서비스를 애플리케이션에 구현하지 않아도 된다.

- 서비스 디스커버리, 스케일링, 로드밸런싱, 자가 치유 등이 포함 → 서비스 코드에 구현하지 않아도 된다.

애플리케이션 구현에만 집중할 수 있게 만들어준다.

쿠버네티스 클러스터 아키텍처



마스터 (컨트롤 플레인)

- 쿠버네티스 API 서버: 사용자, 컨트롤 플레인 구성요소와 통신
- 스케줄러: 애플리케이션의 배포 담당
- 컨트롤러 매니저: 구성 요소 복제본, 워커 노드 추적, 노드장애 처리 등 클러스터 단 기능 수행
- etcd: 클러스터 구성 저장하는 분산 데이터 저장소

워커 노드

- 컨테이너 런타임: 도커, rkt 같은 컨테이너 실행요소
- kubelet: API 서버와 통신하고 노드의 컨테이너 관리
- kube-proxy: 애플리케이션 구성요소 간 네트워크 트래픽을 로드밸런싱

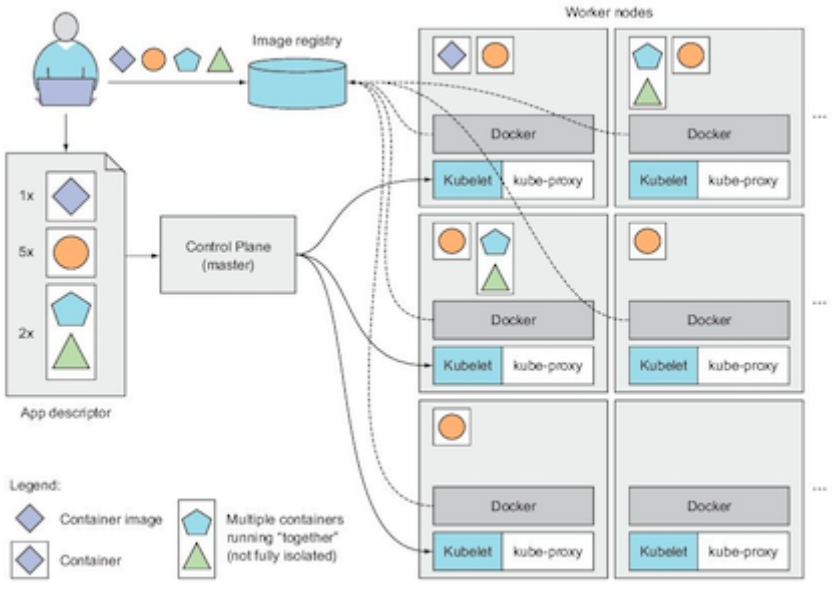


Figure 1.10 A basic overview of the Kubernetes architecture and an application running on top of it