Microservices

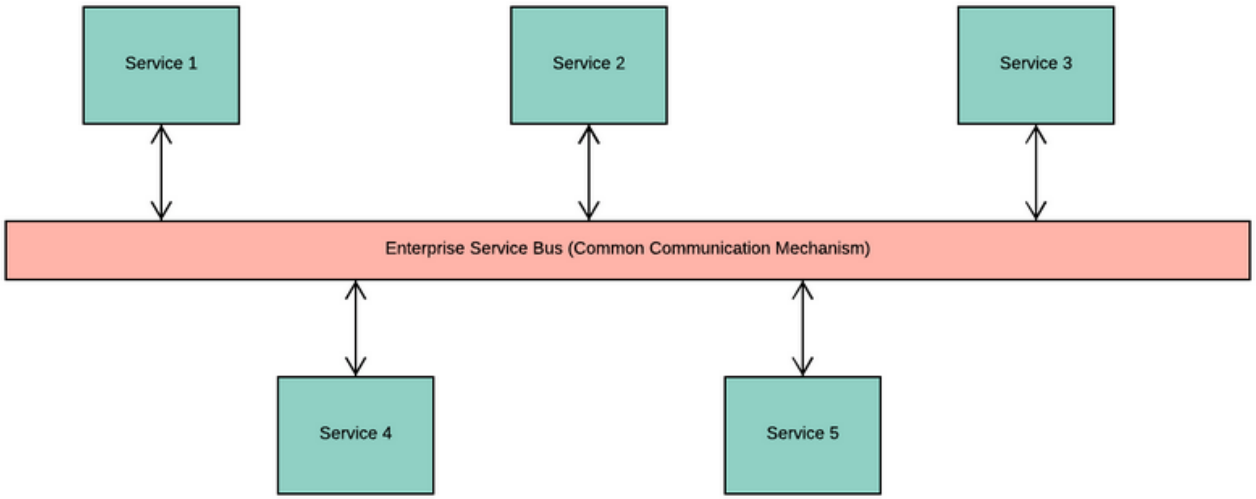
Tiago Heinrich

UniSociesc Joinville

14/05/2020

Arquitetura orientada a serviços

- *Service Oriented Architecture* (SOA), 2000
- Design de software em que os serviços são fornecidos aos outros componentes
- Diferentes serviços podem ser usados em conjunto para fornecer a funcionalidade de um *software*

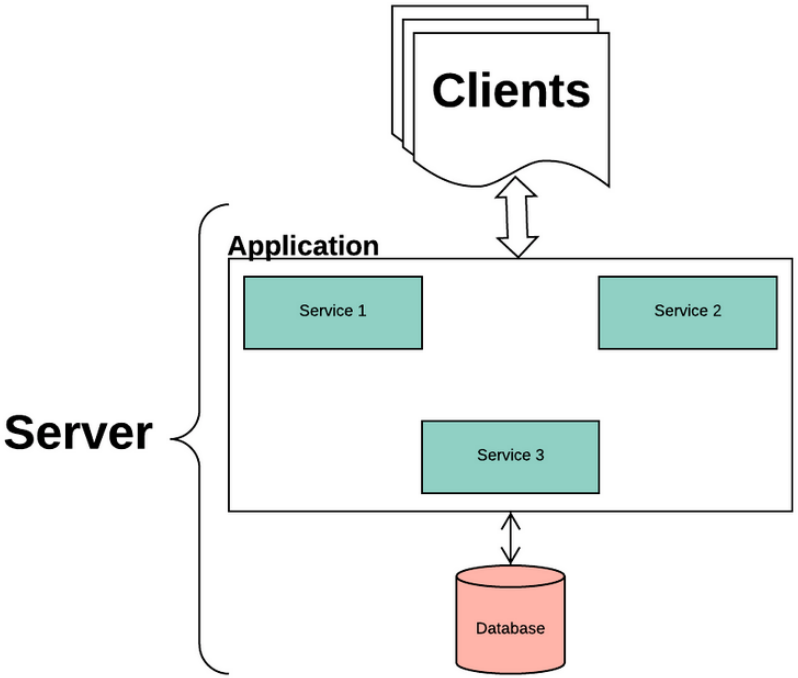


Arquitetura orientada a serviços

- Preferido para produtos de software em larga escala
- Foca na integração de vários serviços em um único aplicativo
- Refere a aplicação monolítica. Isso significa que você possui uma única camada de aplicativo que contém sua interface com o usuário ou camada de apresentação, lógica de negócios e camada de banco de dados, tudo integrado em uma única plataforma

Arquitetura monolítica

- Aplicação monolítica descreve uma única aplicação



- criação de conta
- exibição de catálogo de produtos
- criação e validação de seu carrinho de compras
- geração de fatura
- confirmação de pedido
- mecanismo de pagamento

- Todos esses serviços são executados em uma única camada de aplicativo

Arquitetura monolítica

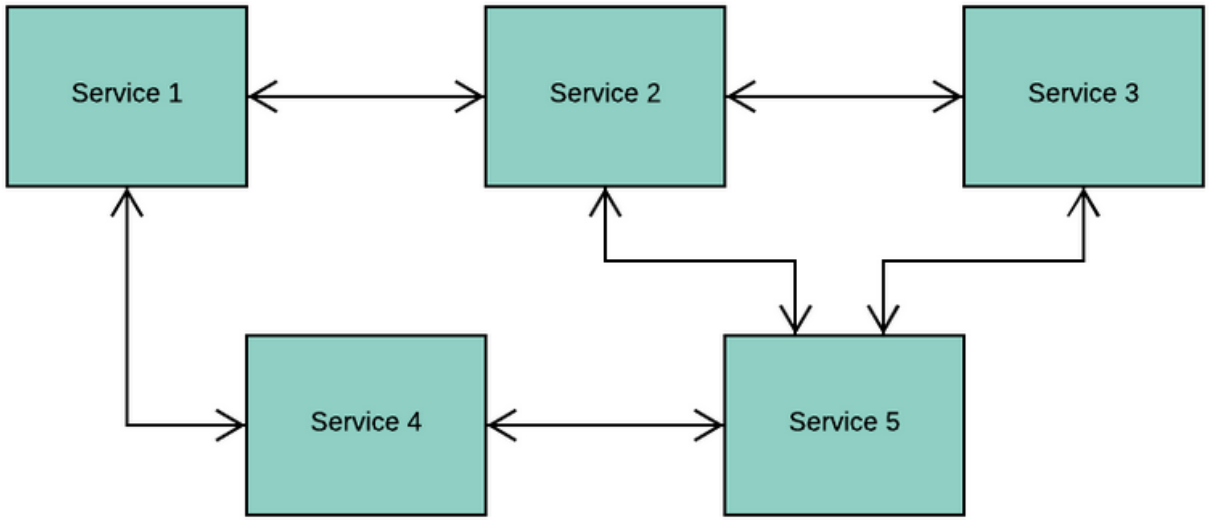
- Crescimento da complexidade ao longo do tempo
- Funções são interdependentes e entrelaçadas
- Escalabilidade é limitada
- Falta de flexibilidade (amarrados à tecnologia)
- Dificuldades para realizar alterações em produção
  - Mudança requer que os desenvolvedores reconstruam a totalidade do aplicativo

Microservices

- Microsserviços é uma técnica de desenvolvimento de software
- Variante do estilo estrutural da arquitetura orientada a serviços (SOA)
- Não existe uma definição única:
  - Arquitetura são implementáveis independentemente
  - Organizado em torno dos recursos
  - Pode ser implementado usando diferentes linguagens de programação, bancos de dados, ambiente de *hardware* e *software*
  - Geralmente pequenos, habilitados para mensagens, desenvolvidos de forma autônoma e descentralizados

Microservices

- Objetivo em modularizar o aplicativo
- Dividindo serviços independentes em módulos menores
- Sendo implementados, dimensionados e até mantidos independentemente de outros serviços existentes

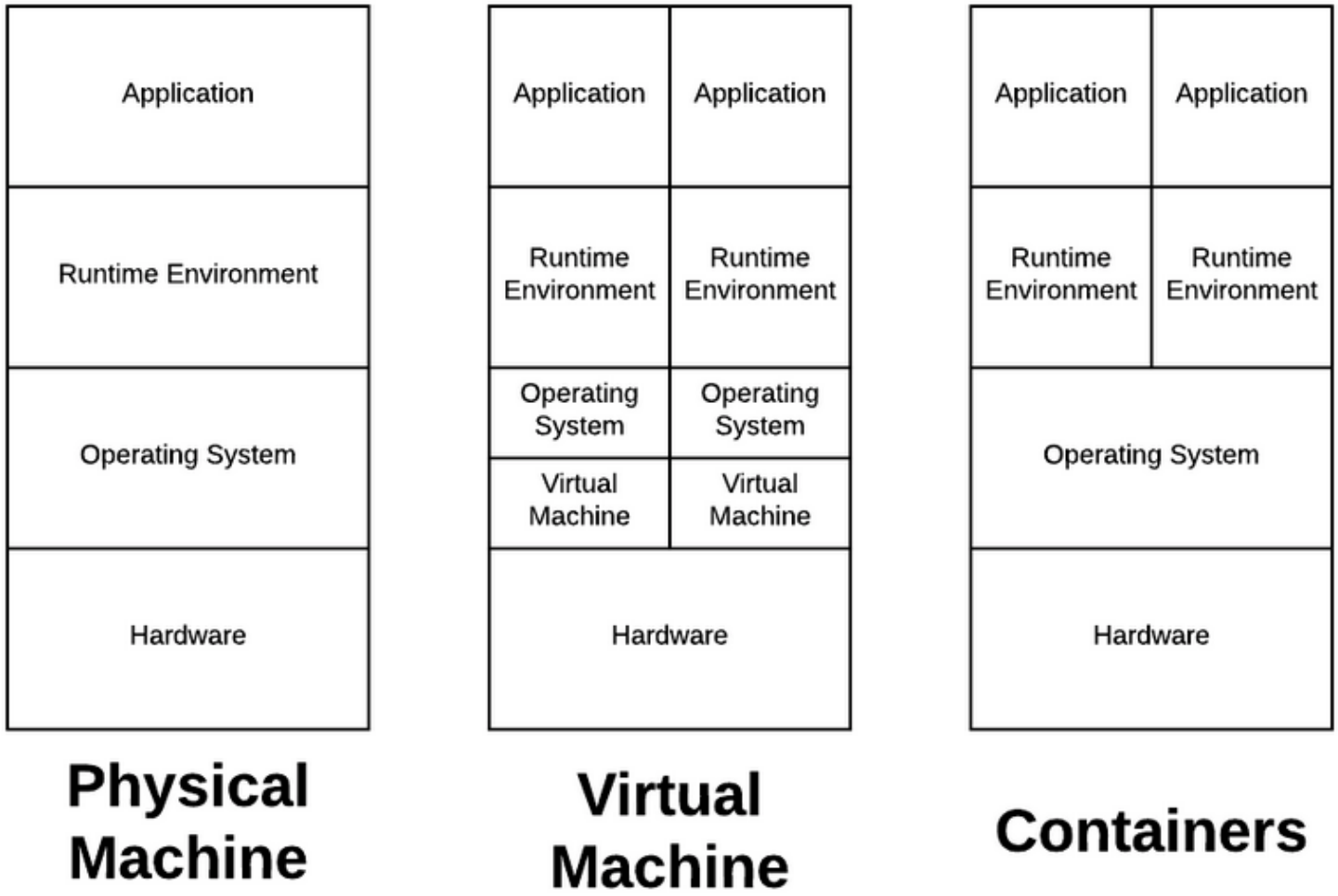


Microservices

- Separação de preocupações e garante o desenvolvimento ágil de *software*
- Reduz a complexidade (serviços e desenvolvimento)
- Reduz o risco, permitindo a implantação em blocos
- Manutenção fácil e flexível
- Permite manter modelos de dados separados para cada um dos serviços fornecidos

Microservices

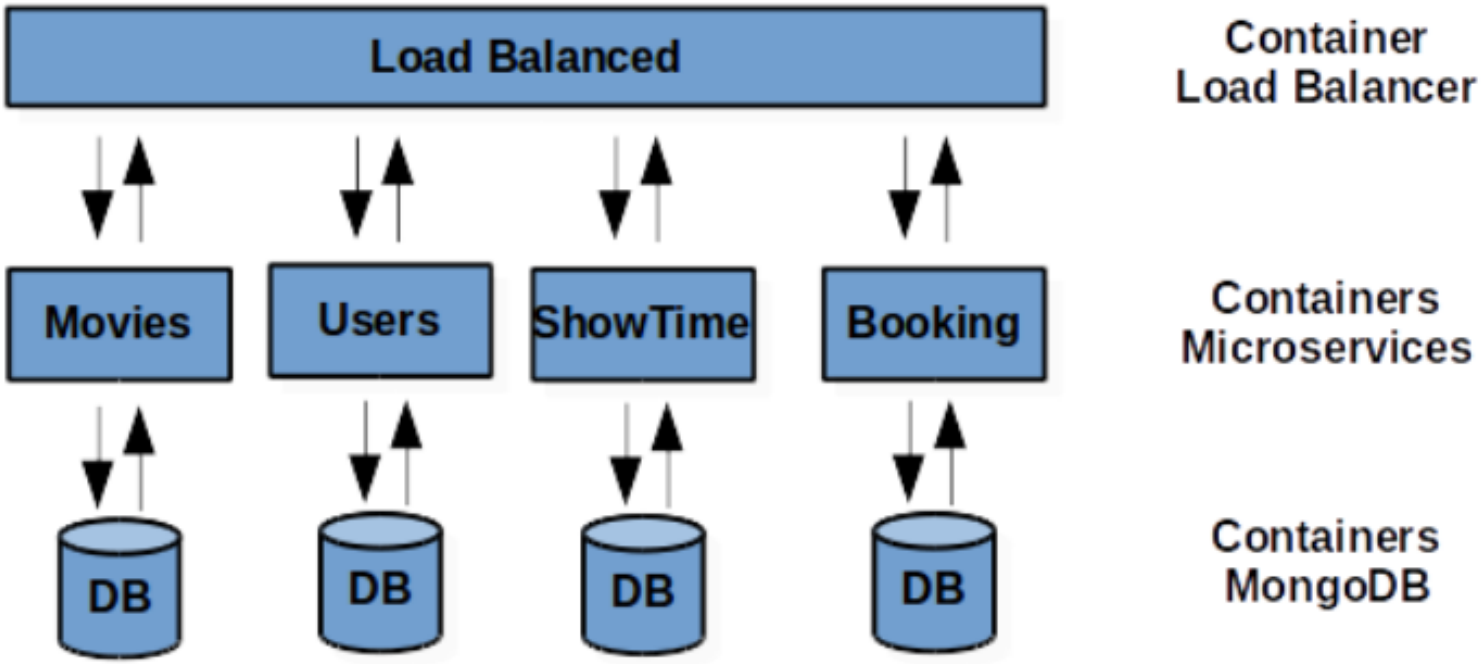
- Evoluímos do uso da virtualização de hardware para os `containerization`



Microservices

- Acessível e custos reduzidos
- Você pode executar vários contêineres em uma máquina física ou até mesmo executá-lo em uma única VM
- O Docker é a plataforma líder mundial em contêiner de software. Ele encapsula seu microsserviço no que chamamos de `Docker container`

Docker


Load Balanced
Movies
Users
ShowTime
Booking
DB
DB
DB
DB
Container
Load Balancer
Containers
Microservices
Containers
MongoDB

Prática Microservices

Docker Compose

- Compose é uma ferramenta para definir e executar aplicativos em vários contêineres
- O arquivo YAML configura os serviços do seu aplicativo
  1. Defina o ambiente do seu aplicativo com um Dockerfile
  2. Defina os serviços que compõem seu aplicativo no docker-compose.yml
  3. Execute o docker-componha e o `Compose` inicia e executa todo o aplicativo

Kubernetes

Kubernetes

- Kubernetes se torna cada vez mais popular como solução de orquestração de contêineres
- Docker e Kubernetes não são concorrentes diretos
- O Docker é uma plataforma de contêiner e o Kubernetes é um orquestrador de contêineres para plataformas de contêineres como o Docker
- Dominação atual do mercado. Atualmente, 30% das empresas usam o Docker

Kubernetes

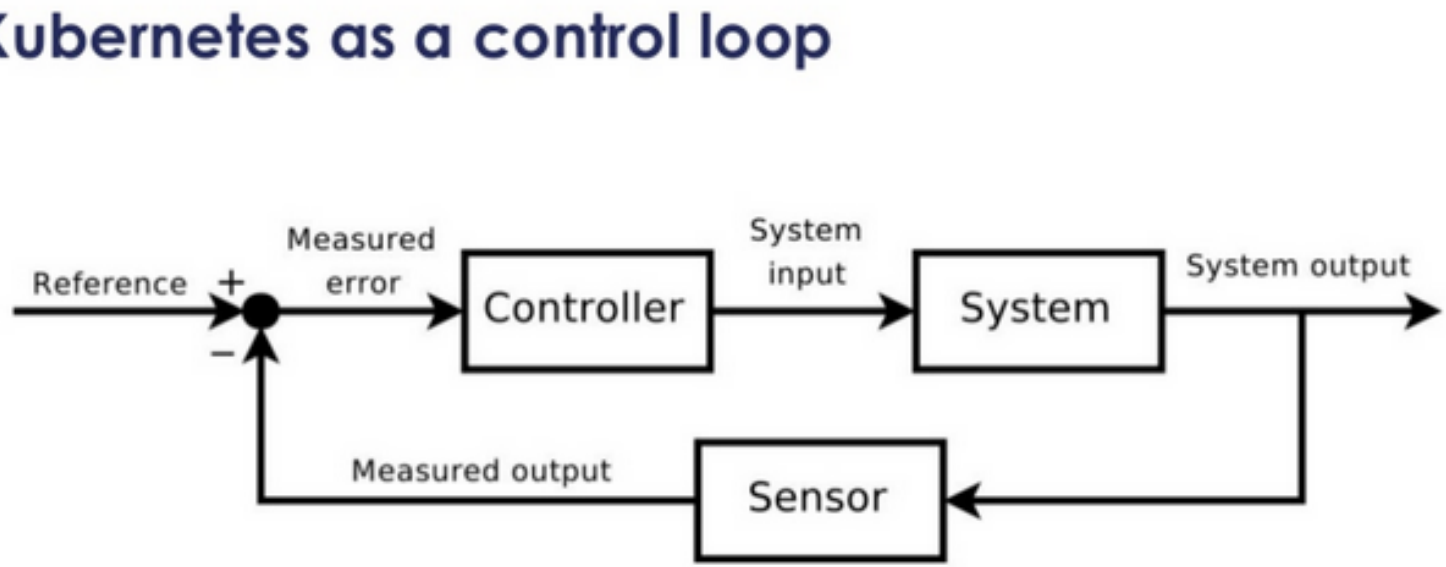
- Docker Engine:
  1. Dockerfile (define o necessário para executar)
  2. Docker Image (é o componente estático portátil que é executado)
- O Docker forneceu um padrão aberto para empacotar e distribuir aplicativos em contêineres
- Como todos esses contêineres seriam coordenados/ programados/ atualizados?

Kubernetes

- Orquestração: Kubernetes, Mesos, e Docker Swarm
- Kubernetes foi desenvolvido pela Google (open source)
- Automatizando, implantação, agendamento e dimensionamento de contêineres
- Com suporte ao Docker

Kubernetes

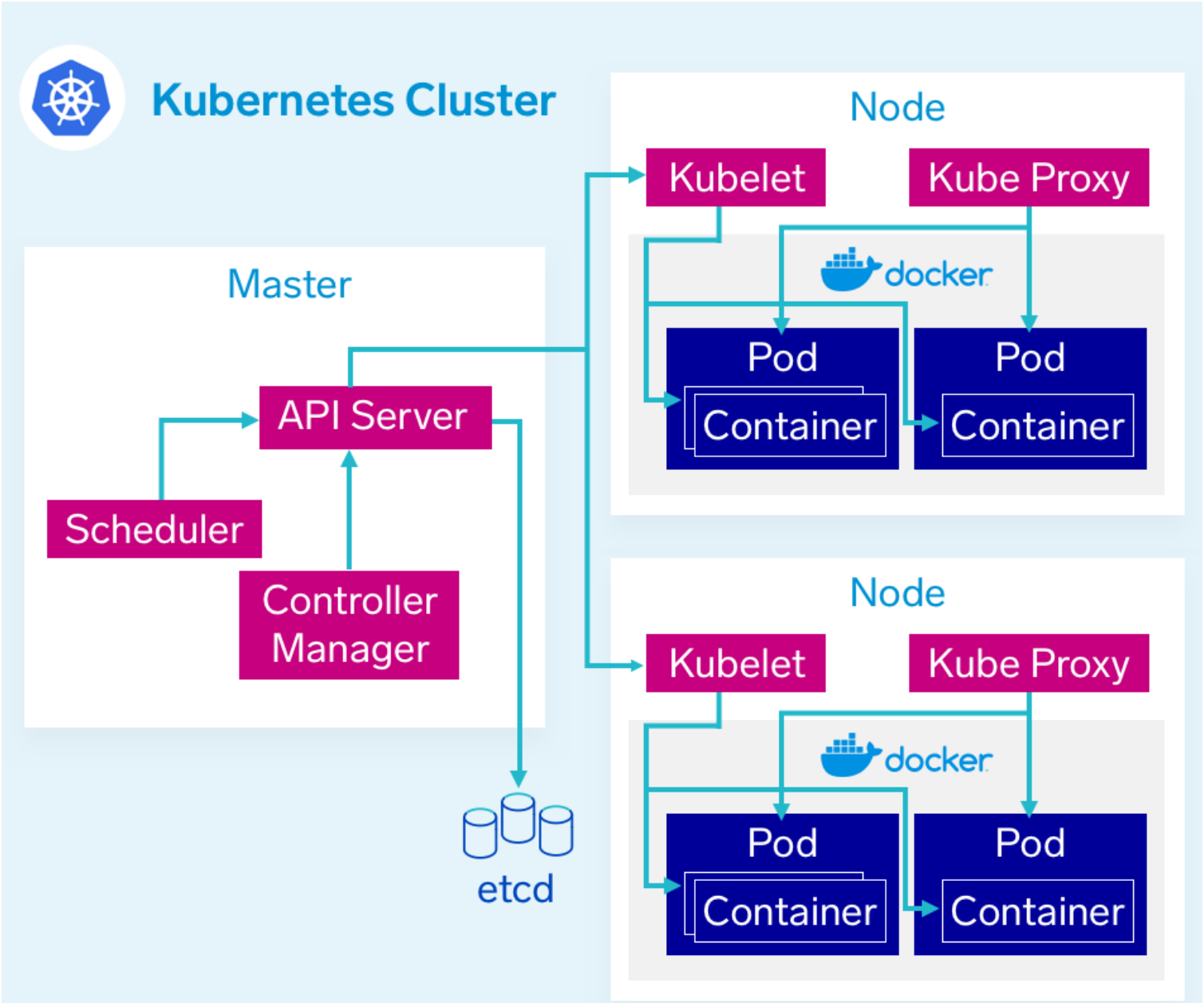
- Como o Kubernetes funciona?
  - Declare como você deseja que seu sistema seja (images)
  - Kubernetes faz isso acontecer



Kubernetes

- Arquitetura Kubernetes
  - Componentes que não se importam um com o outro
  - Todos os componentes se comunicam através do servidor da API
  - Componentes operam suas próprias funções e expõem as métricas que podemos coletar
  - The Control Plane (Master)
  - `Nodes` – onde os pods são agendados
  - `Pods` – contém os contêineres

Kubernetes


Kubernetes Cluster
Master
API Server
Scheduler
Controller Manager
etcd
Node
Kubelet
Kube Proxy
docker
Pod
Container
Pod
Container
Node
Kubelet
Kube Proxy
docker
Pod
Container
Pod
Container

Kubernetes

- `Master`: O orquestrador
- `Nodes`: Onde os contêineres são implementados para execução. Os `Nodes` são a infraestrutura física em que seu aplicativo é executado
- `Pods`:
  - O recurso de nível mais baixo no cluster Kubernetes
  - Um pod é composto de um ou mais contêineres, mas geralmente apenas um único contêiner
  - Os limites para os `pods` definem quais recursos, CPU e memória, eles precisam executar