

# Fundamentos de DevOps

<http://linkedin.com/in/guijac>

Aula 07 - Orquestração de Contêineres

---

**Prof. Esp. Guilherme Jorge Aragão da Cruz**

 [guilherme.cruz@alumni.usp.br](mailto:guilherme.cruz@alumni.usp.br)

 [linkedin.com/in/guijac](http://linkedin.com/in/guijac)

# Roteiro

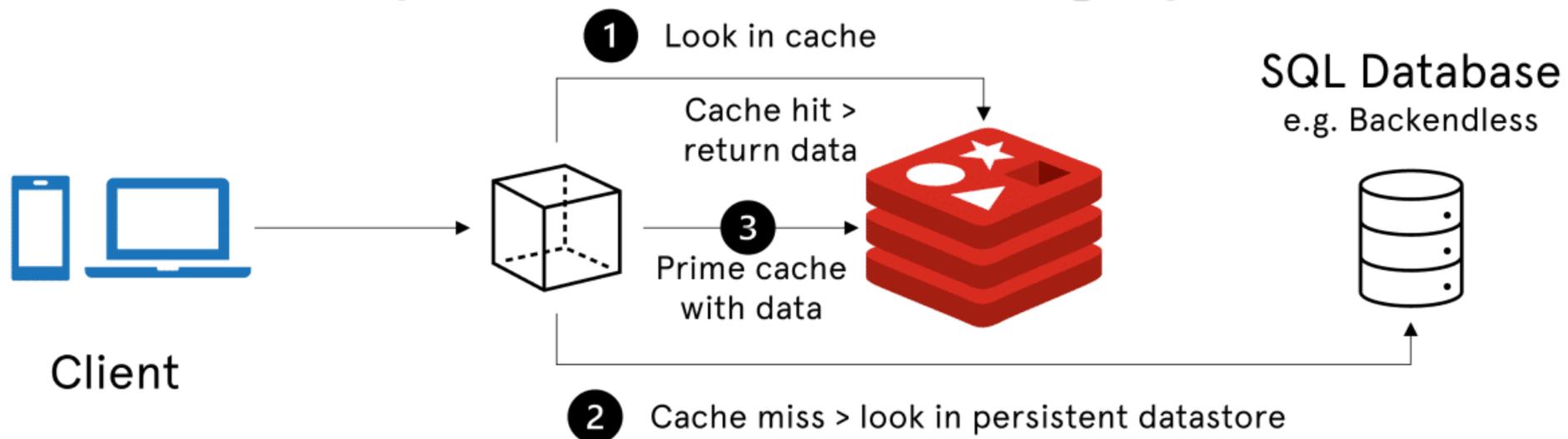
---

- REDIS;
- Orquestração de Contêineres:
  - Definição;
  - Motivações;
  - Principais Características.
- Docker Compose;
- Kubernetes;
- Referências Bibliográficas.

# REDIS

- Acrônimo de **RE**mote **D**ictionary **S**erver;
- Trabalha com armazenamento de estrutura de dados de chave-valor na memória, possuindo uma boa performance, sendo utilizado principalmente para cache de dados.

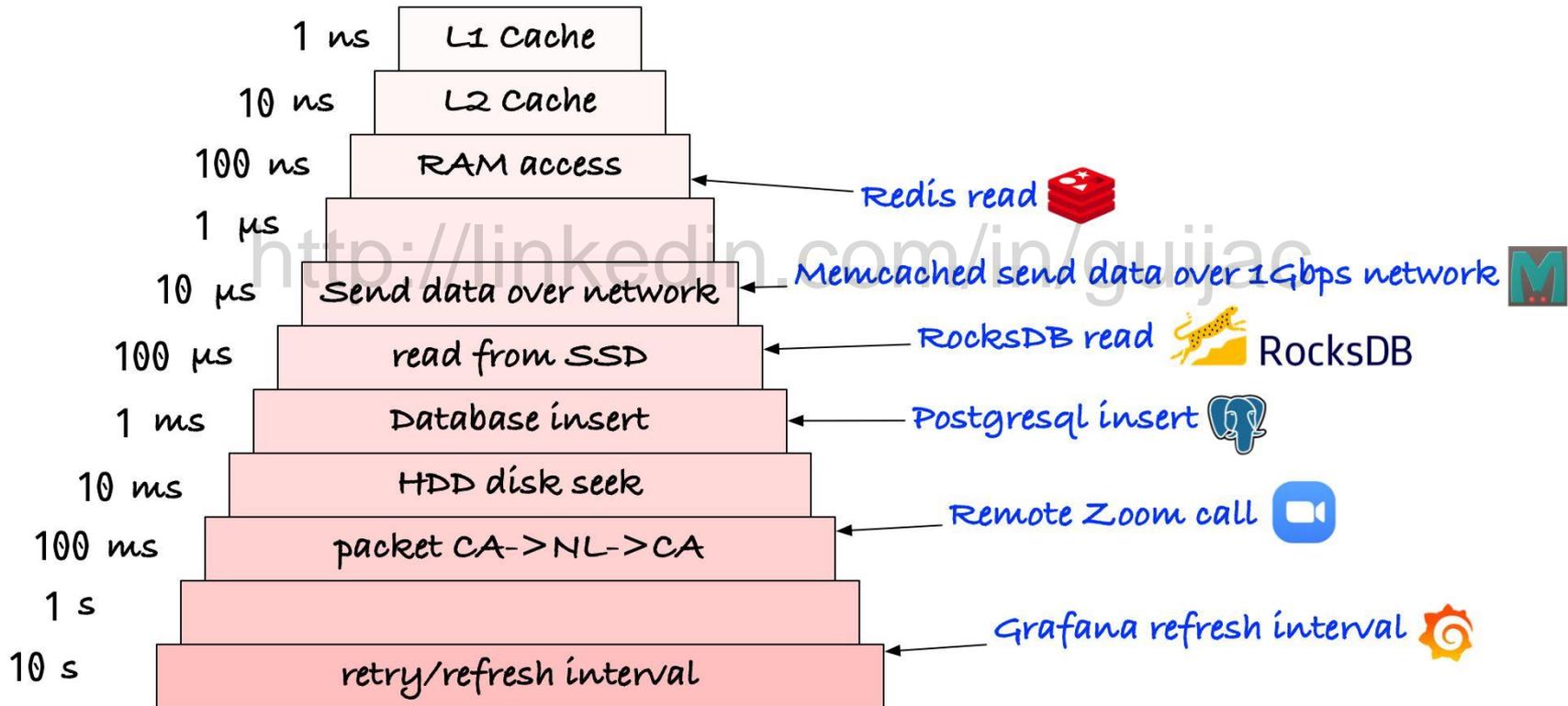
## How Redis is typically used



Fonte: [Redis: What It Is, What It Does, and Why You Should Care | Backendless](https://www.backendless.com/blog/redis-what-it-is-what-it-does-and-why-you-should-care)

# REDIS

## Latency Numbers You Should Know



Fonte: [EP22: Latency numbers you should know. Also... - by Alex Xu \(bytebytego.com\)](#)

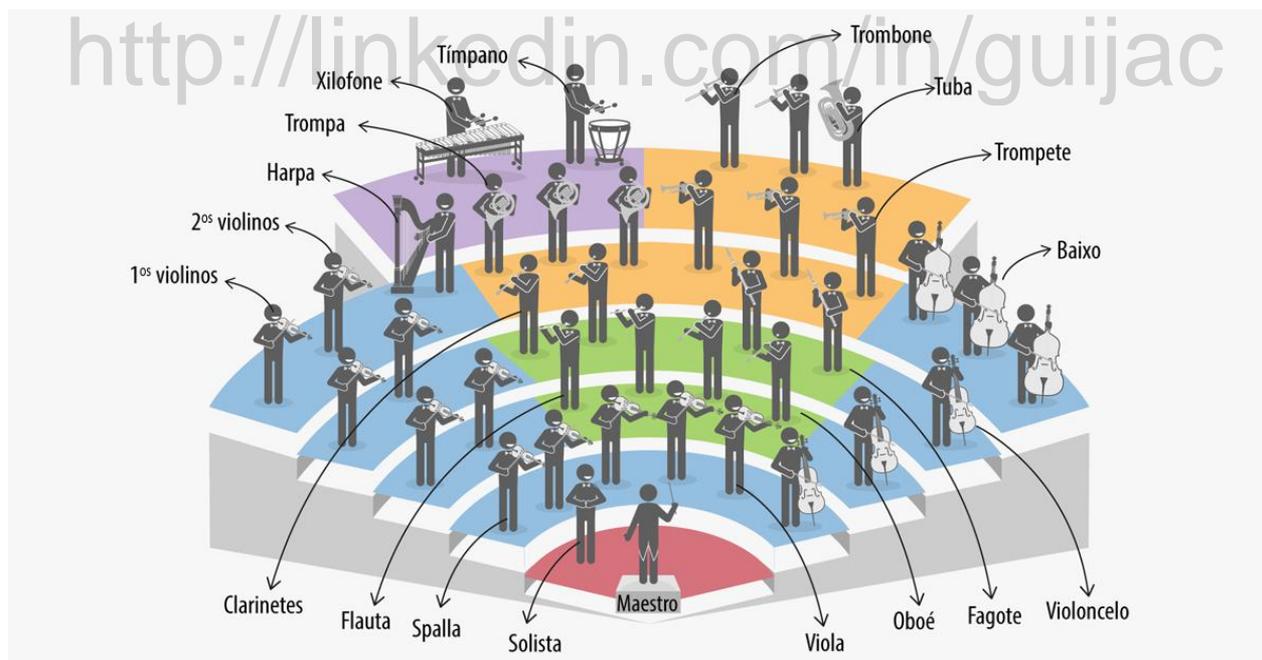
1 ns = 10<sup>-9</sup> seconds | 1 us = 10<sup>-6</sup> seconds = 1.000 ns | 1 ms = 10<sup>-3</sup> seconds = 1.000 us = 1.000.000 ns

# Orquestração?

---

<http://linkedin.com/in/guijac>

# Orquestração?

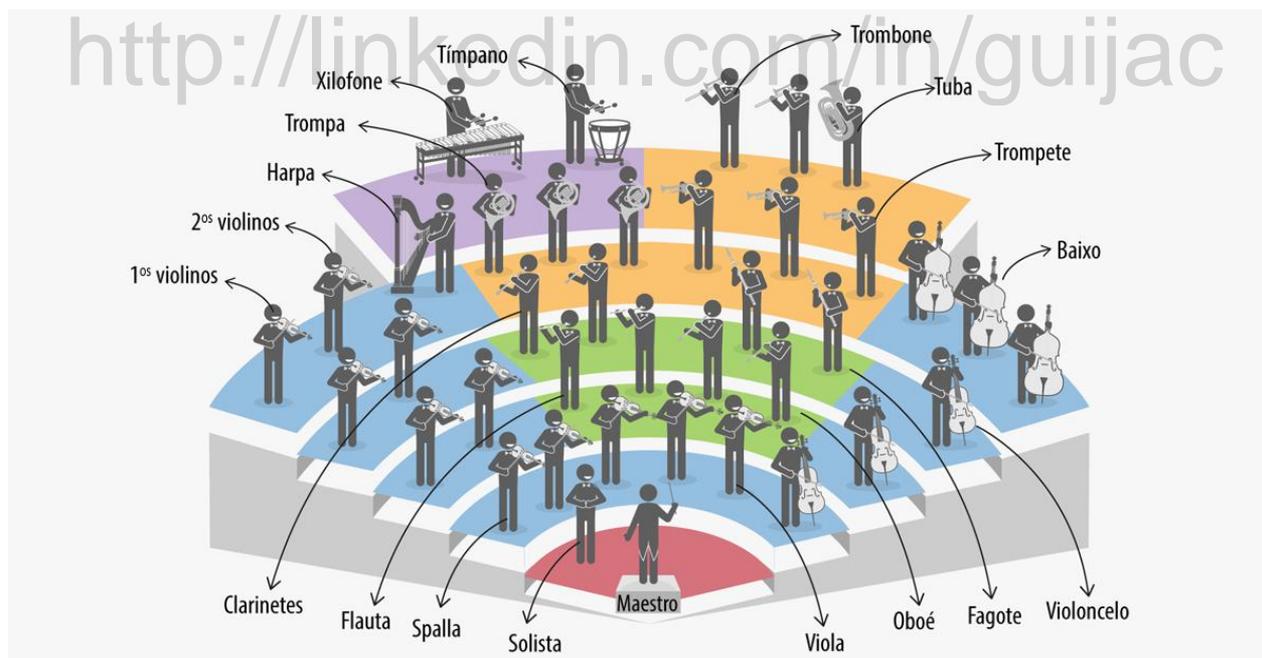


Fonte: [O que é uma Orquestra Sinfônica? | Souza Lima](#)

# Orquestração?

“ Ato de compor ou adaptar uma melodia para uma orquestra, organizado em instrumentos para executar cada parte, podendo se unir com outros instrumentos. Coordenar de forma harmônica.”

SOUZA LIMA - CONSERVATÓRIO E FACULDADE DE MÚSICA (2025)

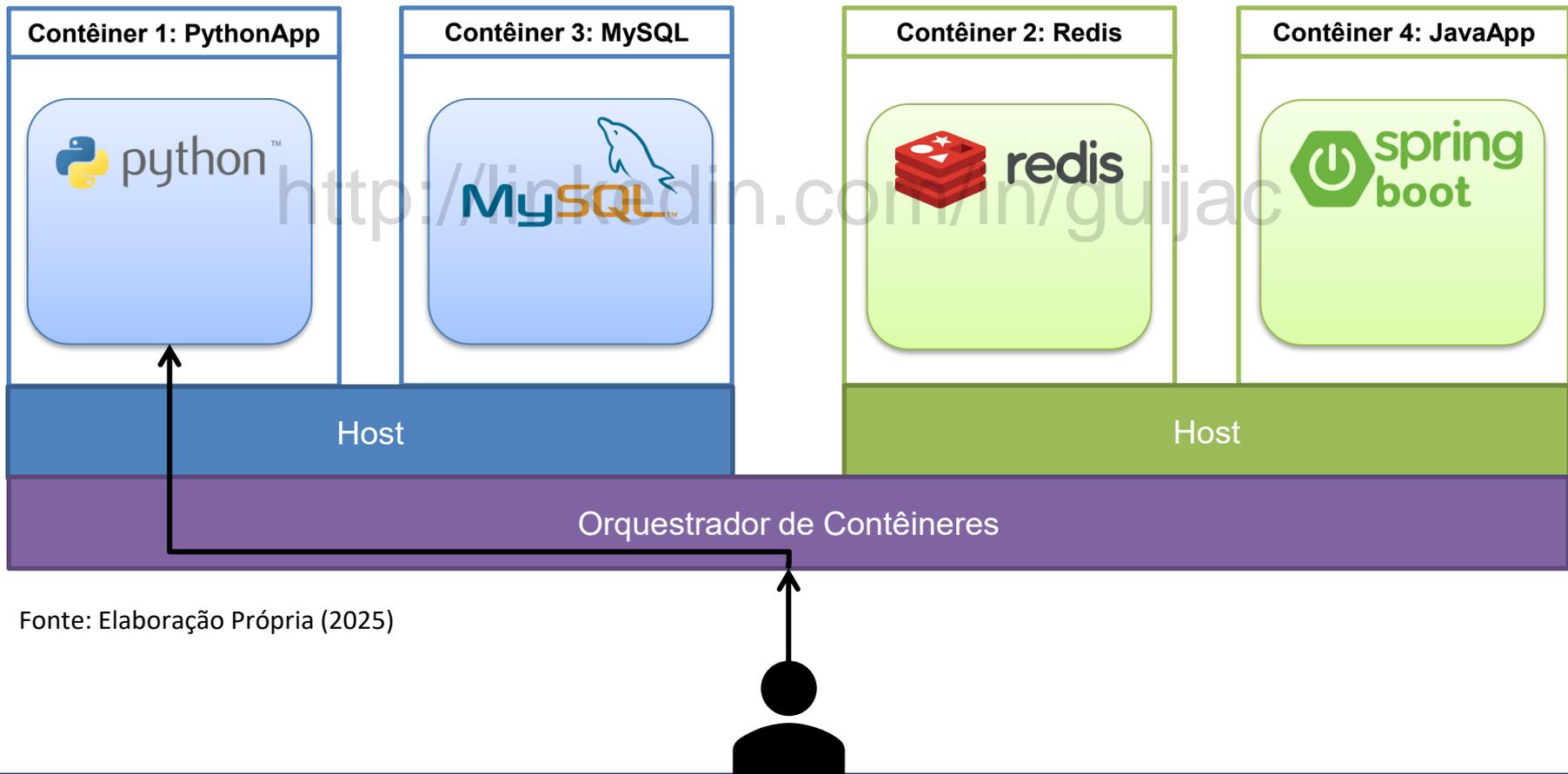


Fonte: [O que é uma Orquestra Sinfônica? | Souza Lima](#)

# Orquestração de Contêineres: Definição

“ Processo que automatiza e gerencia um grande número de contêineres e como eles interagem entre si. ”

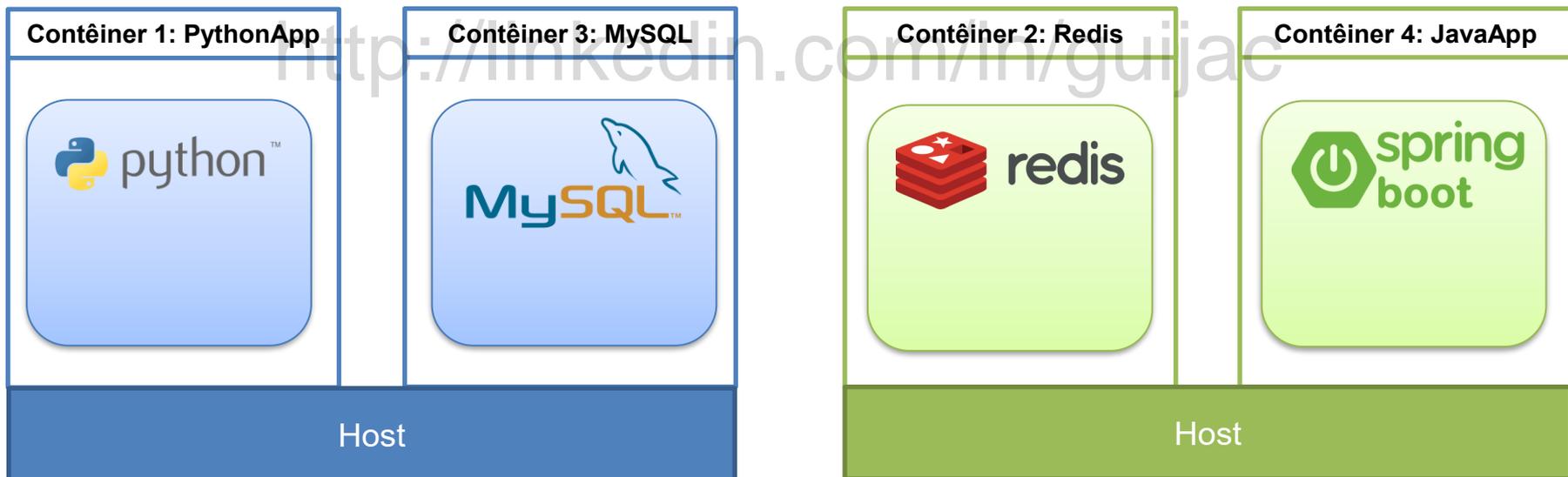
MICROSOFT (2025)



Fonte: Elaboração Própria (2025)

# Orquestração de Contêineres: Motivações

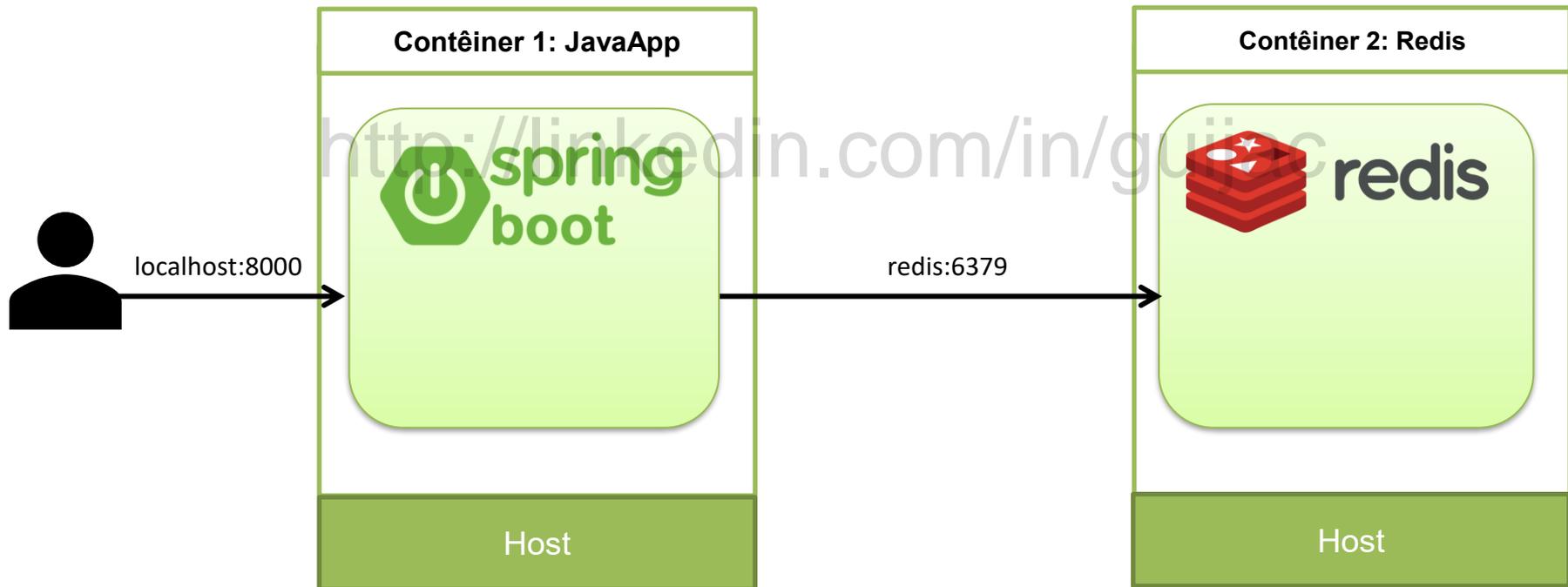
- Em um ambiente que faz uso massivo de contêineres chegaremos na ordem de centenas ou mesmo milhares de componentes para serem gerenciados.



Fonte: Elaboração Própria (2025)

# Orquestração de Contêineres: Motivações

- Em grande parte dos casos, contêineres também possuem dependências entre si.



Fonte: Elaboração Própria (2025)

## Orquestração de Contêineres: Principais Características

---

“*Monitora a integridade do ambiente, inspecionando falhas de contêineres e reiniciando automaticamente;*  
*Habilita contêineres para localizarem uns aos outros automaticamente, mesmo quando eles trocam de computadores host e mudam seus endereços IP, através da **Descoberta de Serviços**;*  
*Gerencia a **Configuração** das aplicações, com base no contêiner em que ela será executada;*  
*Realiza o **Balanceamento de Carga e Roteamento de Tráfego.**”*

MICROSOFT e RED HAT (2025)

# Docker Compose

---

```
version: "3.9"
services:
  redis:
    image: redis:alpine
    ports:
      - "6379:6379"
    healthcheck:
      test: ["CMD", "redis-cli", "ping"] # Verifica se o serviço está saudável.
      interval: 5s # Intervalo entre as execuções do healthcheck.
      timeout: 3s # Tempo máximo para o healthcheck responder.
      retries: 10 # Número de tentativas antes de considerar o serviço como "unhealthy".
  app:
    build: # o mesmo que um "docker build".
      context: . # localização do Dockerfile.
    ports:
      - "8080:8080"
    environment:
      SPRING_DATA_REDIS_HOST: redis
      SPRING_DATA_REDIS_PORT: 6379
    depends_on:
      redis:
        condition: service_healthy
```

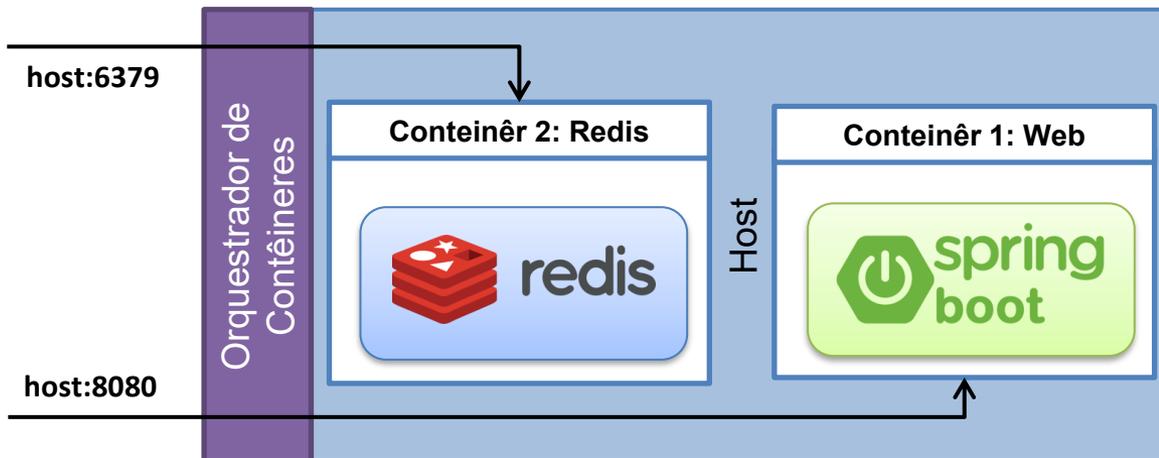
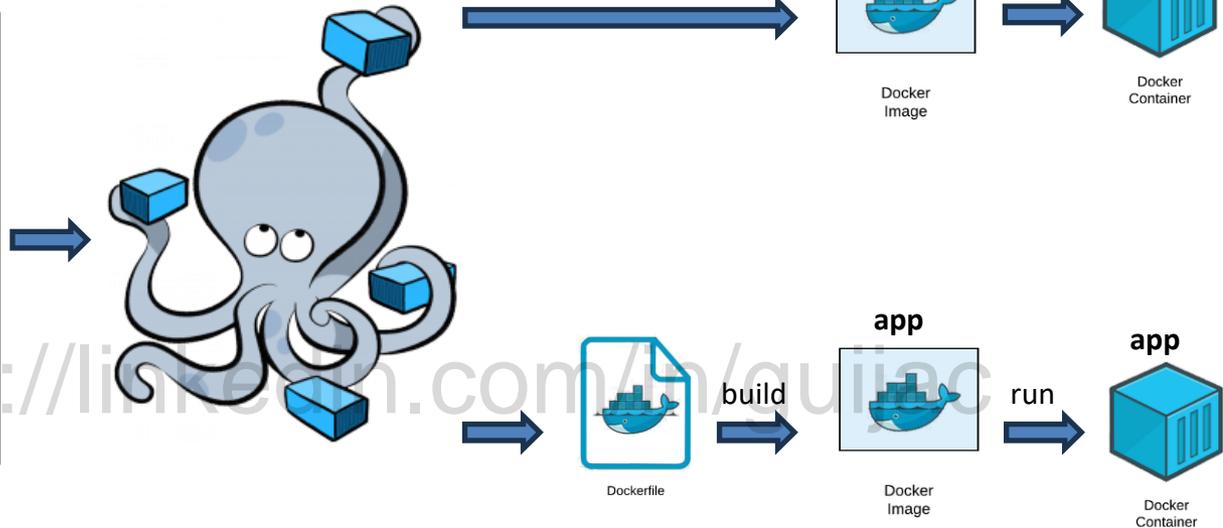
Exemplo de arquivo docker-compose.yml para uso de uma aplicação Java com dependência do Redis.

```
docker-compose up # cria e inicializa os contêineres definidos em um arquivo docker-compose.yml
docker-compose down # paralisa e remove todos os contêineres e seus componentes como, imagem e volume.
```

# Docker Compose

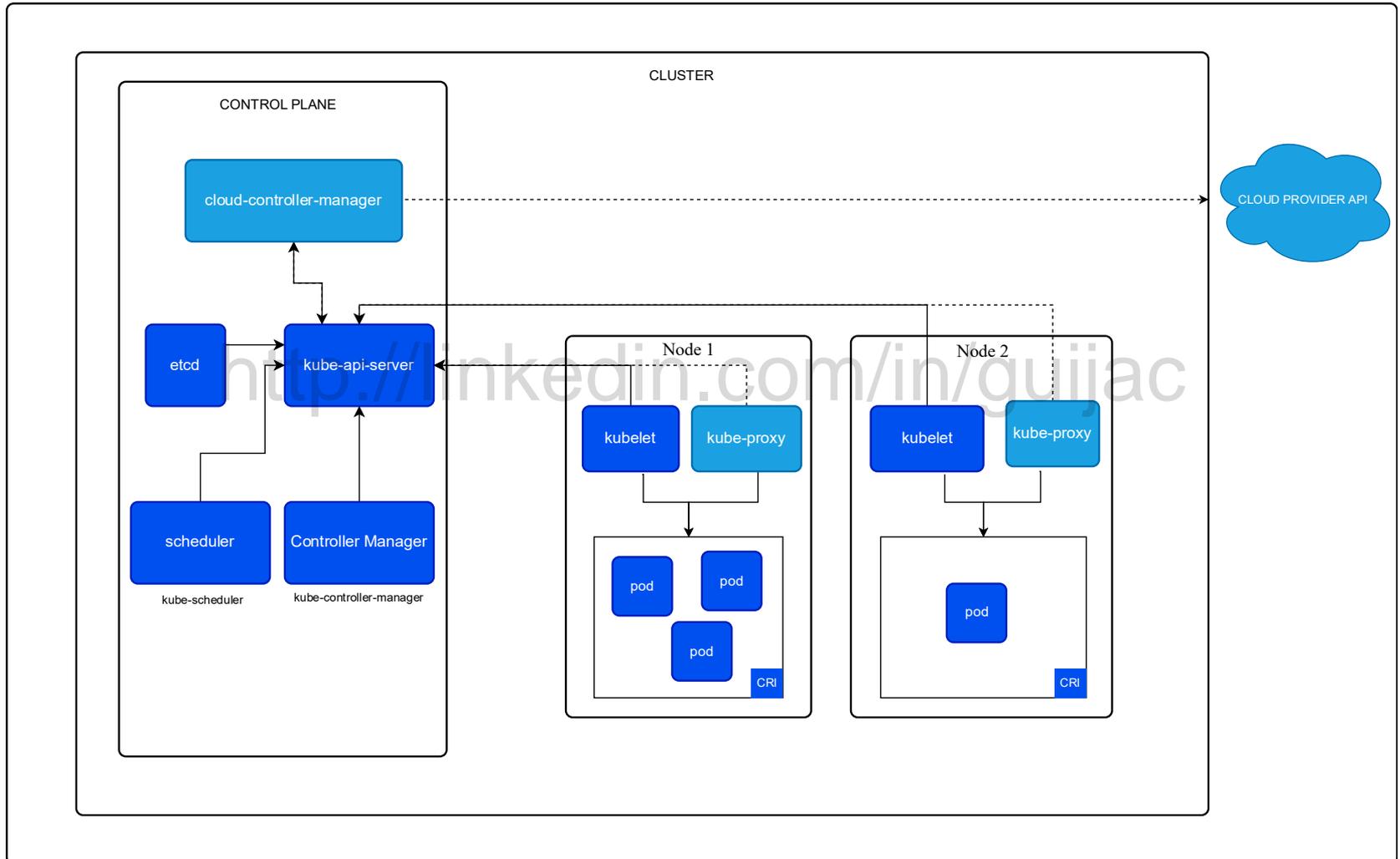
docker-compose.yml.

```
version: "3.9"
services:
  redis:
    image: redis:alpine
    ports:
      - "6379:6379"
    healthcheck:
      test: ["CMD", "redis-cli", "ping"]
      interval: 5s
      timeout: 3s
      retries: 10
  app:
    build:
      context:
      ports:
        - "8080:8080"
    environment:
      SPRING_DATA_REDIS_HOST: redis
      SPRING_DATA_REDIS_PORT: 6379
    depends_on:
      redis:
        condition: service_healthy
```



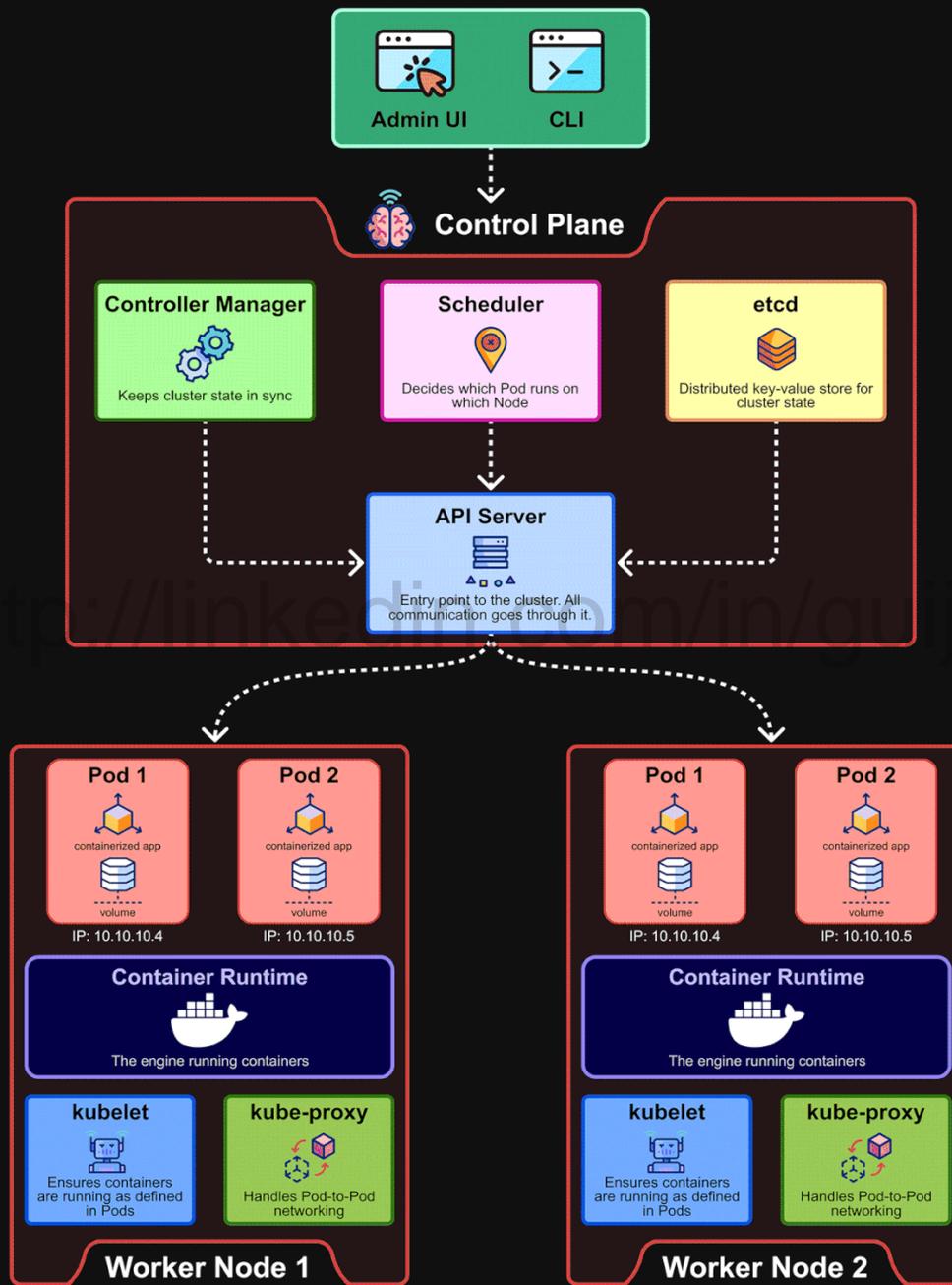
Fonte: Elaboração Própria (2025)

# Kubernetes



Fonte: [Cluster Architecture | Kubernetes](#)

# Kubernetes Explained



# Referências Bibliográficas

---

DOCKER. **Awesome Compose**. Disponível em <https://github.com/docker/awesome-compose>. Acesso em 18 jan 2025;

DOCKER. **Compose file version 3 reference**. Disponível em <https://docs.docker.com/compose/compose-file/compose-file-v3/>. Acesso em 18 jan 2025;

MICROSOFT. **Visão geral da Orquestração de Contêineres do Windows**. Disponível em <https://learn.microsoft.com/pt-br/virtualization/windowscontainers/about/overview-container-orchestrators>. Acesso em 18 jan 2025;

RED HAT. **O que é orquestração de containers?** Disponível em <https://www.redhat.com/pt-br/topics/containers/what-is-container-orchestration>. Acesso em 18 jan 2025;

RISSETO, Fabrício. **ESBs, o que são, do que se alimentam**. Disponível em <http://www.fabriciorissetto.com/blog/ESBs/>. Acesso em 18 jan 2025;

WALLEN, Jack. **Simplifying the mystery: When to use docker, docker-compose, docker swarm and Kubernetes**. Disponível em <https://www.techrepublic.com/article/simplifying-the-mystery-when-to-use-docker-docker-compose-and-kubernetes/>. Acesso em 18 jan 2025.

# Por hoje (de teoria!) é só!

Container Orchestration Software  
(Docker, Openshift & Kubernetes)

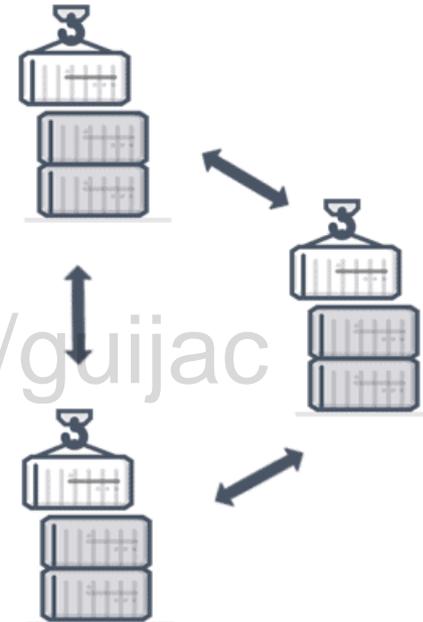


Automate:

- Configuration
- Provisioning
- Availability
- Scaling
- Security
- Resource allocation
- Load balancing
- Health monitoring



Application Environment  
w/ Multiple Containers



Fonte: <http://linkedin.com/in/guijac> [What is Container Orchestration? Definition & Related FAQs | Avi Networks](http://linkedin.com/in/guijac)

**Prof. Esp. Guilherme Jorge Aragão da Cruz**

guilherme.cruz@alumni.usp.br

linkedin.com/in/guijac

# Licença

- Este conteúdo está licenciado sob a Licença Creative Commons Atribuição-NãoComercial-Compartilhalgual 4.0 Internacional (CC BY-NC-SA 4.0).
- Todos os direitos autorais sobre este conteúdo pertencem ao autor, e este material não pode ser usado comercialmente sem autorização expressa.
- Material elaborado no contexto da disciplina **Fundamentos de DevOps do Centro Universitário Senac**, para fins educacionais.
- As referências a marcas, produtos e tecnologias têm caráter exclusivamente educacional, não havendo vínculo institucional ou comercial com as organizações citadas.
- Para ver o texto completo da licença, acesse <https://creativecommons.org/licenses/by-nc-sa/4.0/legalcode>.

---

**Prof. Esp. Guilherme Jorge Aragão da Cruz**

 guilherme.cruz@alumni.usp.br

 linkedin.com/in/guijac