Containerização e orquestração de serviços e aplicações Docker

Fábio Rico Maio 48286
Departamento de Informática
Universidade da Beira Interior
Covilhã, Portugal
fabio.rico.maio@ubi.pt

Simão Almeida Fraga 47831
Departamento de Informática
Universidade da Beira Interior
Covilhã, Portugal
simao.fraga@ubi.pt

Abstract - O Docker, uma plataforma open-source, revolucionou o desenvolvimento, implementação e gestão de aplicações nos últimos anos. Este relatório explora o mundo da containerização e orquestração de serviços, com especial ênfase no papel crucial do Docker. Ao examinar os componentes essenciais e como estes se comparam com máquinas virtuais, investigamos como os containers proporcionam uma encapsulação leve, portátil e eficiente de aplicações e suas dependências. Destacando benefícios como isolamento, portabilidade e facilidade de uso, sublinhamos o impacto transformador do Docker nos fluxos de trabalho de desenvolvimento..

I. Introdução

Nos últimos anos, testemunhamos uma revolução significativa na forma como as aplicações são desenvolvidas, empacotadas e implantadas em ambientes de produção. Essa transformação é impulsionada em grande parte pela proliferação da tecnologia de containerização, com o Docker à frente dessa revolução.

Este relatório tem como objetivo explorar em profundidade o mundo da containerização e da orquestração de serviços e aplicações, com foco no papel central desempenhado pelo Docker. Investigaremos como o Docker simplificou o desenvolvimento, o teste, a implantação e o gerenciamento de aplicações, além de discutir as práticas recomendadas e os desafios associados a essa tecnologia.

Nos próximos capítulos, vamos abordar aspetos fundamentais, como o que é o Docker e como funciona, os benefícios que ele oferece para as equipes de desenvolvimento e operações, exemplos reais de casos de uso, os desafios enfrentados e as tendências futuras na área de containerização e orquestração.

II. Docker

O Docker é uma plataforma de código aberto que revolucionou a maneira como as equipes de desenvolvimento e operações (DevOps) criam e gerenciam aplicações. Ele introduziu uma abordagem inovadora conhecida como "containerização", que permite empacotar aplicações e suas dependências em containers, oferecendo inúmeras vantagens em termos de eficiência, portabilidade e escalabilidade.

O Docker permite aos programadores empacotar, distribuir e executar aplicações em unidades leves e autónomas. Foi fundado em 2013 e tornou-se uma ferramenta indispensável para equipas de desenvolvimento. [1]

III. Componentes principais

O Docker possui três componentes principais:

A. *Docker Engine*

É o núcleo do sistema e é responsável pela execução dos containers. Funciona como um motor que impulsiona a criação, execução e gestão dos containers. Este é ainda responsável por realizar as interações entre o sistema operativo subjacente e os containers. [1]

B. *Containers*

São ambientes autónomos e isolados que encapsulam aplicações e todas as suas dependências. [1]

C. *Docker Hub*

Repositório de containers que facilita a partilha e distribuições de aplicações. No fundo, este atua como um repositório central para as imagens dos containers. Os utilizam têm a possibilidade de carregar, armazenar e partilhar as suas imagens. [1]

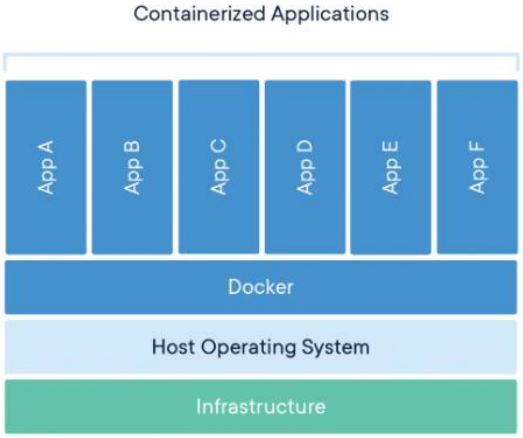

Ilustração 1 - Exemplo de apps em containers

IV. O QUE SÃO OS CONTAINERS?

Os containers são viabilizados pela capacidade de isolamento de processos e virtualização incorporada no kernel do Linux. Essas capacidades, possibilitam que vários componentes de uma aplicação compartilhem os recursos de uma única instância do sistema operativo do host. Isso é semelhante à forma como um hipervisor permite que várias máquinas virtuais (VMs) compartilhem a CPU, memória e outros recursos de um único servidor físico.[2]

Dessa forma, a tecnologia de containers oferece todas as funcionalidades e benefícios das *Virtual Machines*, incluindo isolamento de aplicações, escalabilidade econômica e descartabilidade, com vantagens adicionais significativas:

V. VANTAGENS DE USO DO DOCKER

A. *Peso*

Ao contrário das VMs, os containers não carregam o peso de uma instância completa de sistema operativo e hipervisor. Incluem apenas os processos e dependências do sistema operativo necessários para executar o código. O tamanho dos containers é medido em megabytes (em comparação com gigabytes para algumas VMs), proporcionando um uso mais eficiente da capacidade de hardware e tempos de inicialização mais rápidos.[2]

B. *Isolamento*

Os containers são altamente portáteis. Isso significa que uma aplicação empacotada em um containers pode ser executada de forma consistente em diferentes ambientes, como desenvolvimento, teste e produção, independentemente das diferenças de configuração. [2]

C. *Eficiência*

Os containers são leves e iniciam rapidamente, já que compartilham o kernel do sistema operacional do host, tornando-os mais eficientes em termos de recursos em comparação com as máquinas virtuais, que exigem a virtualização de um sistema operacional completo. [2]

D. *Produtividade*

Aplicações em containers podem ser escritas uma vez e executadas em qualquer lugar. Comparados às VMs, os containers são mais rápidos e fáceis de implantar, provisionar e reiniciar. Isso os torna ideais para uso em pipelines de integração contínua e entrega contínua (CI/CD) e mais adequados para equipes de desenvolvimento que adotam práticas ágeis e DevOps.[2]

E. *Facilidade de uso*

O Docker fornece uma interface simples e poderosa para criar, implantar e gerenciar containers. Ele também possui um repositório central chamado Docker Hub, onde os containers pré-configurados podem ser compartilhados e reutilizados. [2]

O Docker simplifica muito o processo de desenvolvimento, implantação e gerenciamento de aplicações, tornando-o uma escolha valiosa para equipas de desenvolvimento e operações que desejam tornar seus fluxos de trabalho mais ágeis, eficientes e portáteis. Com o Docker, é possível criar, testar e implantar aplicativos de maneira mais rápida e confiável, além de facilitar a manutenção e a escalabilidade dos serviços em containers.

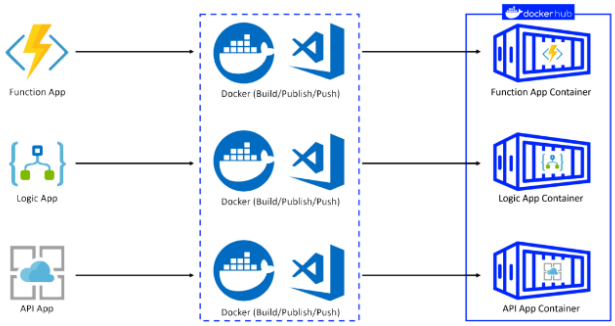

Ilustração 2 - Exemplo dos processos de "containerização"

VI. CASOS DE USO

A. *Implantação continua (CI/CD):*

O Docker agiliza a Implantação Contínua (CI/CD) ao permitir a automação da construção, teste e implantação de containers. Isso acelera os ciclos de desenvolvimento, possibilitando atualizações contínuas e rápidas.[3]

B. *Microserviços:*

Em arquiteturas de microserviços, o Docker simplifica a implantação e gestão de serviços independentes, facilitando a escalabilidade, manutenção e atualização de partes específicas do sistema. [3]

C. *Ambientes de desenvolvimento isolado:*

Desenvolvedores se beneficiam ao criar ambientes isolados usando Docker, garantindo consistência e eliminando problemas de compatibilidade entre máquinas. [3]

D. *Ambientes de Teste:*

O Docker facilita a criação de ambientes de teste replicáveis, assegurando condições consistentes para testes de integração e testes de aceitação do usuário. [3]

E. *Reprodutibilidade e Versionamento:*

A capacidade de versionamento do Docker permite a criação de ambientes precisamente reproduzíveis, simplificando a gestão de versões de software em diferentes estágios do desenvolvimento e produção. [3]

VII. DESAFIOS E SOLUÇÕES

Embora o Docker tenha transformado positivamente a forma como desenvolvemos, implementamos e dimensionamos aplicações, a sua adoção não está isenta de desafios significativos. Nesta seção vamos explorar os desafios chave associados ao uso do Docker e apresentaremos soluções estratégicas para mitigar essas questões.

A. *Monitoramento e Gerenciamento:*

- Desafio: O monitoramento eficiente dos containers em execução e a gestão de sua escalabilidade podem ser desafiadores.[4]
- Solução: Implementar práticas de segurança, como isolamento de recursos, limitação de privilégios,

monitoramento constante e atualizações regulares, para mitigar riscos de segurança. [4]

B. Complexidade na Orquestração:

- Desafio: Orquestrar e gerenciar a execução de vários containers em escala pode se tornar complexo, especialmente em ambientes distribuídos. [4]
- Solução: Utilizar ferramentas de orquestração, como Kubernetes, Swarm ou Docker Compose, para simplificar e automatizar tarefas de implantação, escalabilidade e monitoramento. [4]

Entender estes desafios e implementar soluções eficazes é crucial para garantir uma implementação bem-sucedida e segura da tecnologia Docker em ambientes complexos e em constante evolução.

VIII. Tendências Futuras

A rápida evolução do ecossistema Docker e as demandas crescentes na área de desenvolvimento de software indicam um horizonte empolgante de inovações e aprimoramentos. Nesta seção, exploraremos as tendências emergentes que moldarão o futuro da containerização e orquestração de serviços e aplicações, destacando áreas de desenvolvimento promissoras e a integração do Docker com tecnologias em ascensão.

A. Inteligência Artificial e Machine Learning:

- Tendência: A integração do Docker com ferramentas de Inteligência Artificial e Machine Learning está cada vez mais proeminente, oferecendo recursos avançados de automação e otimização. [4]
- Futuro: Prevê-se que a capacidade de otimizar automaticamente a alocação de recursos, identificar padrões de desempenho e facilitar práticas de Machine Learning em ambientes de containers seja um ponto focal para inovações futuras. [4]

B. Edge Computing:

- Tendência: O Docker está desempenhando um papel essencial em ambientes de Edge Computing, onde a computação ocorre mais próxima dos dispositivos finais. [4]
- Futuro: À medida que a computação de borda continua a crescer, antecipa-se uma maior integração do Docker em soluções destinadas a ambientes descentralizados e com baixa latência. [4]

IX. Conclusão

Em conclusão, exploramos a revolução que o Docker trouxe ao desenvolvimento e à implantação de aplicações, concentrando-nos na containerização e orquestração de serviços. O Docker não apenas simplificou drasticamente os processos de desenvolvimento, teste e implementação, mas também proporcionou eficiência, portabilidade e escalabilidade para equipes de desenvolvimento e operações.

Este relatório serviu como uma jornada de descoberta do impacto transformador do Docker na tecnologia da informação. Ao abraçar as melhores práticas, superar desafios e permanecer atento às tendências futuras, as equipes podem continuar a aproveitar ao máximo essa ferramenta inovadora, impulsionando a eficiência, agilidade e confiabilidade em ambientes de desenvolvimento e produção. É evidente que o Docker continuará a desempenhar um papel crucial na evolução do cenário tecnológico, moldando a forma como concebemos, implementamos e escalamos aplicações no mundo digital em constante evolução.

X. Referências

[1] Tiago M. C. Simões, “Desenvolvimento de Software para a Nuvem - Módulo 2,” hackmd.io, https://hackmd.io/@UBI/S1VA58Kzp#31-Docker , 2023.

[2] IBM, “What is Docker?”, IBM., https://www.ibm.com/topics/docker , 2023.

[3] Tech With Tim, “An Introduction To Docker For Beginners,” Tech With Tim, https://www.youtube.com/watch?v=ZiwvZenOzto , 2020.

[4] OpenAI, "ChatGPT: A Large-Scale Generative Language Model", OpenAI, 2022