09

Para saber mais: Docker e Microserviços

Sobre Microserviços

Já ouviu falar de Microserviços? Se já ouviu, pode pular a introdução abaixo e ir diretamente para a parte "Docker e Microserviços", senão continue comigo.

Uma forma de desenvolver uma aplicação é colocar todas as funcionalidades em um único "lugar". Ou seja, a aplicação roda em uma única instância (ou servidor) que possui todas as funcionalidades. Isso talvez é a forma mais simples de criar uma aplicação (também a mais natural) mas quando a base de código cresce alguns problemas podem aparecer.

Por exemplo, qualquer atualização ou bug fix necessita parar todo o sistema, buildar o sistema todo e subir novamente. Isso pode ficar demorado e lento. Em geral, quanto maior a base de código mais difícil será manter ela mesmo com uma boa cobertura de testes e as desvantagens não param por ai. Outro problema é se alguma funcionalidade possuir um gargalo no desempenho o sistema todo será afetado. Não é raro de ver sistemas onde relatórios só devem ser gerados à noite para não afetar o desempenho de outras funcionalidades. Outro problema comum é como ciclos de testes e build demorados (falta de agilidade no desenvolvimento), problemas no monitoramento da aplicação ou falta de escalabilidade. Enfim, o sistema se torna um legado pesado onde nenhum desenvolvedor gostaria de colocar a mão no fogo.

Arquitetura de Microservicos

Então a idéia é fugir desse tipo de arquitetura monolítica monstruosa e dividir ela em pequenos pedaços. Cada pedaço possui uma funcionalidade bem definida e roda como se fosse um "mini sistema" isolado. Ou seja, em vez de termos uma única aplicação enorme teremos várias instâncias menores que dividem e coordenam o trabalho. Essas instâncias são chamadas de Microserviços.

Agora fica mais fácil monitorar cada serviço específico, atualizá-lo ou escalá-lo pois a base de código é muito menor, e assim o deploy e o teste serão mais rápido. Podemos agora achar soluções específicas para esse serviço sem precisar alterar os demais. Outra vantagem é que um desenvolvedor novo não precisa conhecer o sistema todo para alterar uma funcionalidade, basta ele focar na funcionalidade desse microservico.

Importante também é que um microserviço seja acessível remotamente, normalmente usando o protocolo HTTP trocando mensagens JSON ou XML, mas nada impede que outro protocolo seja usado. Um microserviço pode usar outros serviços para coordenar o trabalho.

Repare que isso é uma outra abordagem arquitetural bem diferente do monolítico e por isso também é chamado de *arquitetura de microserviços*.

Por fim, uma arquitetura de Microserviços tem um grau de complexidade muito alta se comparada com uma arquitetura monolítica. Aliás, há aqueles profissionais que indicam partir para uma arquitetura monolítica primeiro e mudar para uma baseada em microserviços depois (http://sdtimes.com/martin-fowler-monolithic-apps-first-microservices-later/), quando estritamente necessário.

Docker e Microserviços

Trabalhar com uma arquitetura de microserviços gera a necessidade de publicar o serviço de maneira rápida, leve, isolada e vimos que o Docker possui exatamente essas características! Com Docker e Docker Compose podemos criar um ambiente ideal para a publicação destes serviços.

O Docker é uma ótima opção para rodar os microserviços pelo fato de isolar os containers. Essa utilização de containers para serviços individuais faz com que seja muito simples gerenciar e atualizar esses serviços, de maneira automatizada e rápida.

Docker Swarm

Ok, tudo bem até aqui. Agora vou ter vários serviços rodando usando o Docker. E para facilitar a criação desses containers já aprendemos usar o Docker Compose que sabe subir e orquestrar vários containers. O Docker Compose é a ferramenta ideal para coordenar a criação dos containers, no entanto para melhorar a escalabilidade e desempenho pode ser necessário criar muito mais containers para um serviço específico. Em outras palavras, agora gostaríamos de criar *muitos* containers aproveitando *várias* máquinas (virtuais ou físicas)! Ou seja, pode ser que um microserviços fique rodando em 20 containers usando três máquinas físicas diferentes. Como podemos facilmente subir e parar esses containers? Repare que o Docker Compose não é para isso e por isso existe uma outra ferramenta que se chama o Docker Swarm (que não faz parte do escopo desse curso).

Docker Swarm facilita a criação e administração de um cluster de containers.