Resenha “Microservices” - Pedro Henrique Maia Alves

O que são microserviços? É um estilo arquitetural de desenvolvimento onde existe um conjunto de serviços pequenos modulares e limitados, onde cada um executa seu próprio processo e se comunicam através de APIs HTTPS. Diferem-se dos sistemas monolíticos pois são modulares, autônomos e podem possuir diversidade tecnológica. Enquanto um monólito coloca todas suas funcionalidades em apenas um código e escala isto replicando em diversos servidores, os microserviços desenvolvem separadamente suas funcionalidades e distribuem aos servidores somente aquilo que é necessário no momento atual.

Algumas características destacadas no artigo são principais para o entendimento deste conceito, embora não se pode afirmar que exista uma definição formal do que é arquitetura de microserviços, é esperado que sistemas que adotam esta arquitetura apresente no mínimo algum destes conceitos :

- **Componentização via serviço** - Cada módulo do sistema corresponde a um serviço, e este é independente com o seu próprio ciclo de vida (deploy, escalabilidade, atualização etc.), reduzindo o acoplamento do sistema.

- **Organização por capacidades de negócio** - Os serviços devem refletir as funcionalidades do sistema (ex: Folha de pagamento), ao invés de camadas técnicas como (UI/UX, DBA, DEV), promovendo times multifuncionais e melhor alinhados, como sugere a Lei de Conways.

- **Produto, não projeto** - Cada serviço construído pelo time, é tratado como um produto contínuo, mantido e evoluído pelo mesmo time ao longo do tempo, ou seja, não existe um produto “terminado” ao fim de um projeto.

- **Smart endpoints e dum pipes** - A lógica deve centralizar nos endpoints , enquanto a comunicação deve ser simples e leve através de HTTPs e API RESTs por exemplo.

- **Governança descentralizada** - Liberdade para cada time escolher a melhor tecnologia como linguagens, frameworks e banco de dados conforme o contexto do serviço, evitando padrões rígidos, engessados e únicos.

- **Gerenciamento de dados descentralizados** - Cada serviço controla seu próprio armazenamento, preservando a autonomia e evitando um banco de dados único e compartilhado por todos os serviços daquele projeto.

- **Automação de infraestrutura CI/CD** - Testes, builds, deploys e monitoramento são altamente automatizados para lidar com a multiplicidade de serviços.

- **Design para falhas** - A arquitetura deve ser tolerante e deve esperar as falhas, um monitoramento robusto, dashboards, logs são fundamentais para avaliar o funcionamento do sistema.

- **Design evolutivo** - A arquitetura deve ser tolerante a mudanças e deve suportar refatorações frequente de códigos, versionamento e divisão gradual de monólitos, facilitando que serviços apareçam ou saiam conforme as necessidades do cliente.

Quais são as vantagens e desvantagens da adoção da arquitetura de microserviços? Dentre elas, está a definição clara entre módulos, o que facilita a divisão do trabalho em equipes grandes e torna o desenvolvimento mais organizado. Os deploys independentes são outra vantagem perceptível pois cada serviço pode ser atualizado ou corrigido sem a necessidade de interromper todo o sistema, reduzindo falhas globais. Além disso, a diversidade tecnológica é um avanço para o projeto pois cada microserviço pode ser desenvolvido com a linguagem, framework ou BD que melhor se adapta ao contexto, evitando que todos trabalhem com uma solução única.

Porém adotar microserviços também traz alguns desafios. A comunicação por exemplo entre serviços é feita de maneira externa e remota o que a torna disposta a falhas e lenta quando comparada a um sistema monolítico. Além disso, o fato de cada serviço manter sua própria base de dados implica em consistência eventual, o que dificulta a manutenção de transações fortes e imediatas entre diferentes partes do sistema. Há ainda a questão da complexidade operacional: gerenciar dezenas ou até centenas de serviços exige uma infraestrutura robusta e uma maturidade em práticas de DevOps que nem todas as equipes possuem. Portanto Fowler alerta para o chamado “microservices premium” onde organizações adotam esta arquitetura sem avaliar os custos adicionais e os riscos que podem surgir.

Devemos ou não adotar os microserviços? Fowler recomenda iniciar com cautela, “você não deve começar com uma arquitetura de microsserviços. Em vez disso, comece com um monólito, mantenha-o modular e divida-o em microsserviços quando o monólito se tornar um problema.” pois refatorar um sistema distribuído, é muito mais complexo que modularizar um monólito. Além disso, micro serviços exigem infraestrutura, testes e desenvolvedores dispostos a adotar esta complexidade.

No meu dia a dia como estudante é perceptível alguns conceitos de microserviços que afetam o desenvolvimento como: Dividir o trabalho em partes menores e

modulares, separação de responsabilidades e camadas bem definidas e times multifuncionais, estas características estão presente em nossos trabalhos interdisciplinares, embora geralmente não adotamos diferentes bancos de dados e após a escolha de um framework e linguagem, damos continuidade no projeto utilizando apenas elas.