Resenha Martin Fowler – Projeto de Software – Lucas Felipe Barbosa

Microsserviços é um tipo de arquitetura de software relativamente recente, que basicamente busca dividir as diferentes partes da aplicação em componentes menores, chamados de serviços, que são independentes. Diferentemente do modelo monolítico, cada um desses serviços pode possuir características distintas como o banco de dados e até mesmo a linguagem de programação.

Uma das vantagens do modelo de microsserviços é que ele busca trazer uma ideia de produto à aplicação, e não de projeto. Trazendo a ideia de produto, a equipe responsável por cada serviço fica inteiramente responsável por ele durante todo o seu ciclo de vida. Enquanto que no modo tradicional de desenvolver software existem os diferentes times, por exemplo os desenvolvedores, os UI/UX, os testers, etc, que trabalham com demandas de todo o monólito, na arquitetura de microsserviços cada um dos serviços possui pessoas pré-designadas a trabalharem com o serviço ou serviços. Dessa forma, ocorre uma maior personalização e um sentimento de pertencimento do serviço pelos times.

Apesar de ser “Microsserviços”, os serviços não necessariamente são tão micro assim. Para definir a quantidade máxima de pessoas envolvidas em um único serviço a Amazon tem a ideia de que um time deve poder ser alimentado por no máximo 2 pizzas, o que daria aproximadamente 12 pessoas por serviço. Isso também não impede que um grupo de 6 pessoas por exemplo gerencie 6 serviços, conforme exposto no artigo de Martin Fowler.

Desenvolvidos para suportarem bem as falhas, uma aplicação que é feita com a arquitetura de microsserviços, por mais que venha a ter alguma parte de seu sistema comprometida, essa parte não influenciará comprometendo todo o resto do sistema, já que devido à característica compartimentalizada desse tipo de arquitetura, cada serviço da aplicação é independente e não influencia no funcionamento dos outros serviços.

Apesar de ser uma arquitetura revolucionadora, nem sempre os microsserviços são a melhor solução. Trabalhar com compartimentalização das diferentes partes da aplicação envolve um nível de complexidade, de orçamento e de recursos envolvidos superior ao do modelo monolítico, e as vezes nem se faz necessário complicar algo que pode ser feito da maneira mais tradicional. Além disso, caso não seja feito um planejamento bem feito e uma transição adequada, corre-se o risco de levar a complexidade que estava no bloco monolítico para dentro dos serviços e para as conexões entre eles, o que não é a proposta dessa arquitetura.