



INSTITUTO FEDERAL
Triângulo Mineiro
Campus Uberlândia Centro

Projeto Backend

Microserviços e NoSQL

Introdução

Prof. Lucas Montanheiro
lucasmontanheiro@iftm.edu.br

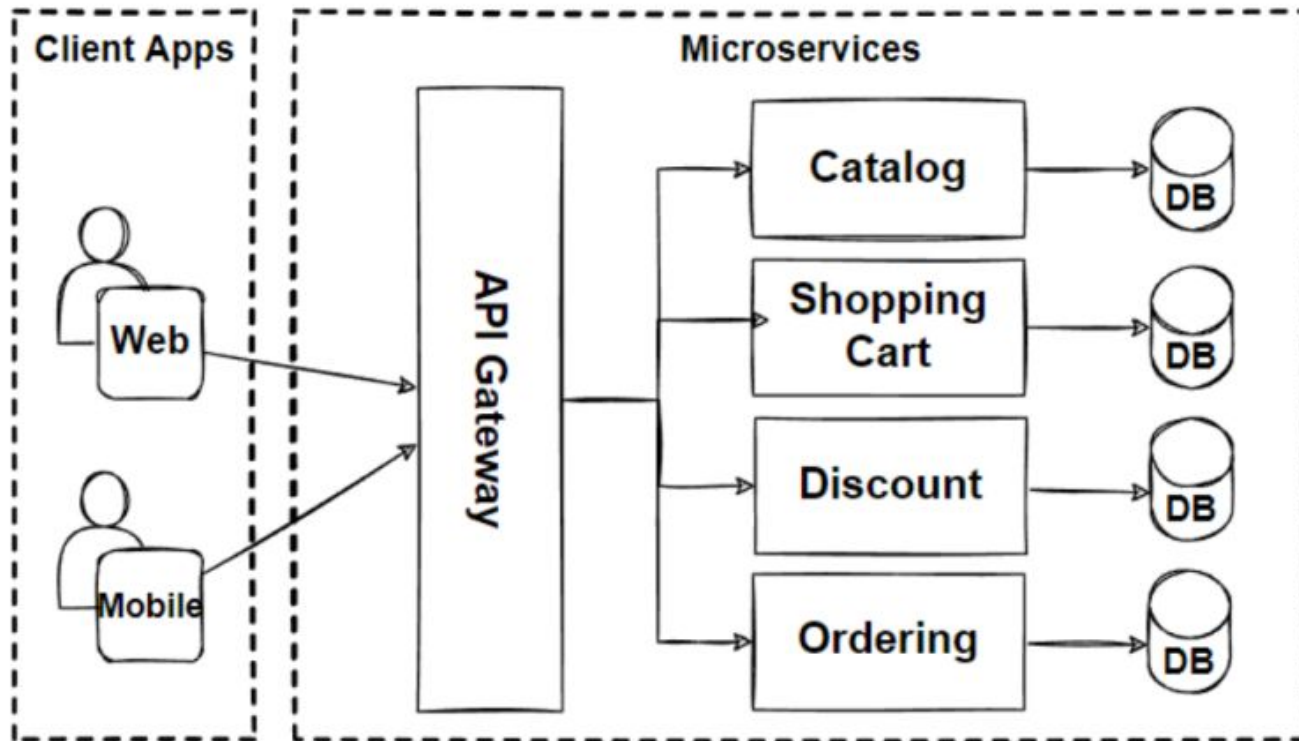
Definição de Microserviços

- É uma abordagem arquitetônica e organizacional de desenvolvimento de aplicações no qual o software consiste em pequenos serviços independentes que se comunicam usando APIs bem definidas;
 - Coleção de pequenos serviços autônomos;
 - Cada serviço é independente e deve implementar uma única funcionalidade;
 - Esses serviços se comunicam por meio de APIs, permitindo a construção de sistemas complexos e escaláveis;

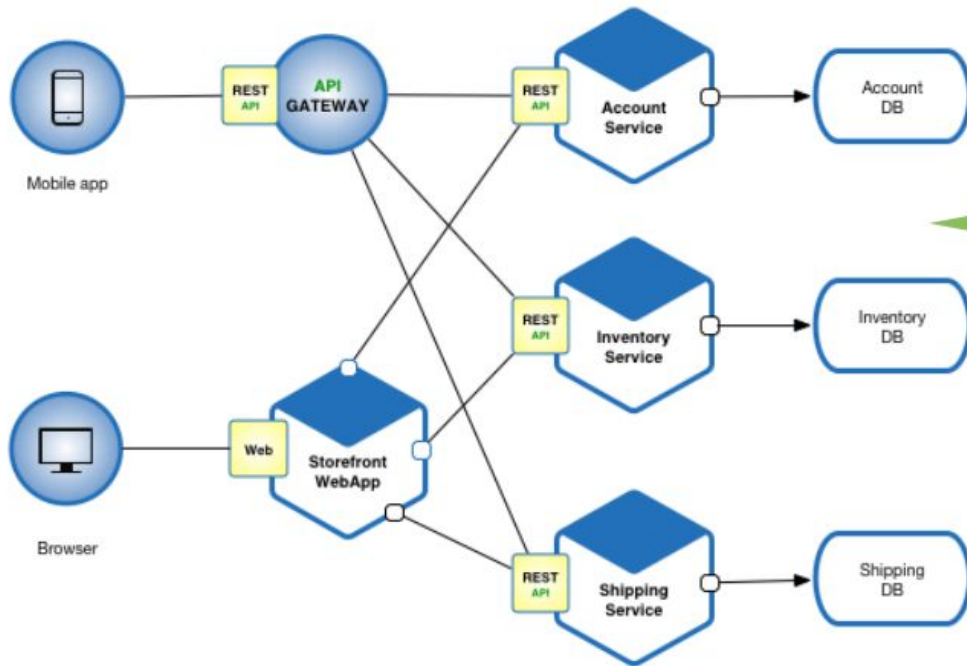
Definição de Microserviços

- A arquitetura de microserviços enfatiza a modularidade, o desacoplamento e a capacidade de implantação e atualização contínua de serviços individuais!

Arquitetura Simples de Microserviços



Arquitetura Simples de Microsserviços



microservices.io

ESTILO ARQUITETURAL
QUE ESTRUTURA UMA
APLICAÇÃO COMO UM
CONJUNTO DE SERVIÇOS
FRACAMENTE
ACOPLADOS,
ORGANIZADOS A PARTIR
DA LÓGICA DE NEGÓCIO

Qual o problema do monolito?



**NEGÓCIOS PRECISAM
INOVAR MAIS RÁPIDO!**

=

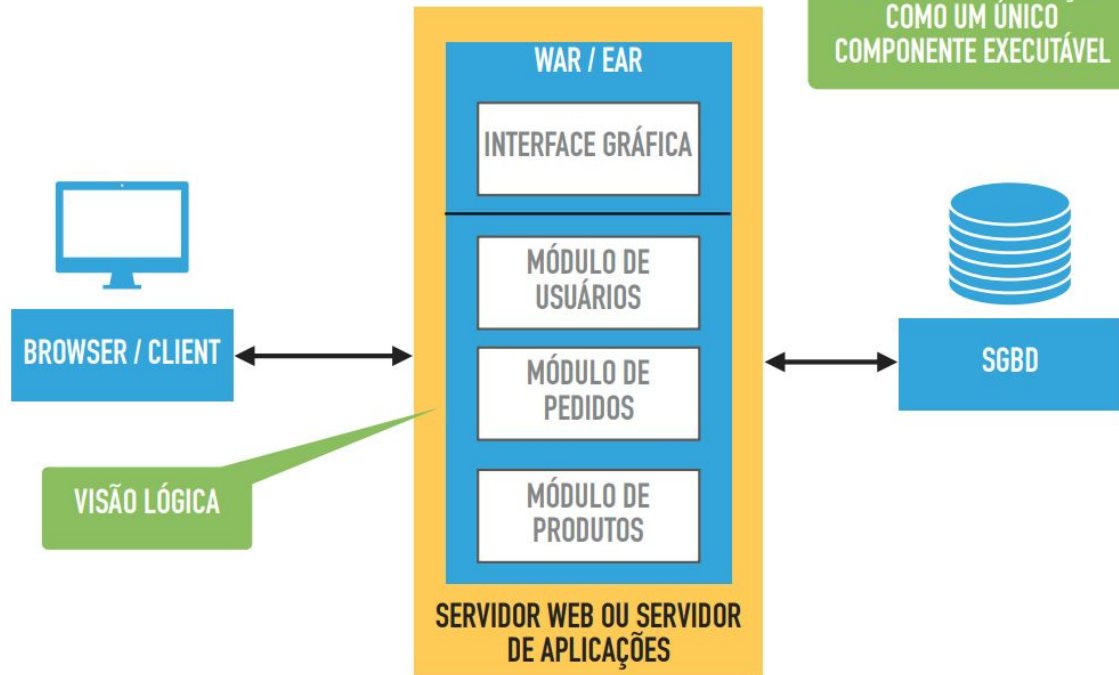
**SOFTWARES PRECISAM
SER CONSTRUÍDOS
MAIS RÁPIDO!**

**REDUÇÃO DO TEMPO DE
ENTREGA**

**AUMENTO DA FREQUÊNCIA
DE IMPLANTAÇÃO**

Qual o problema do monolito?

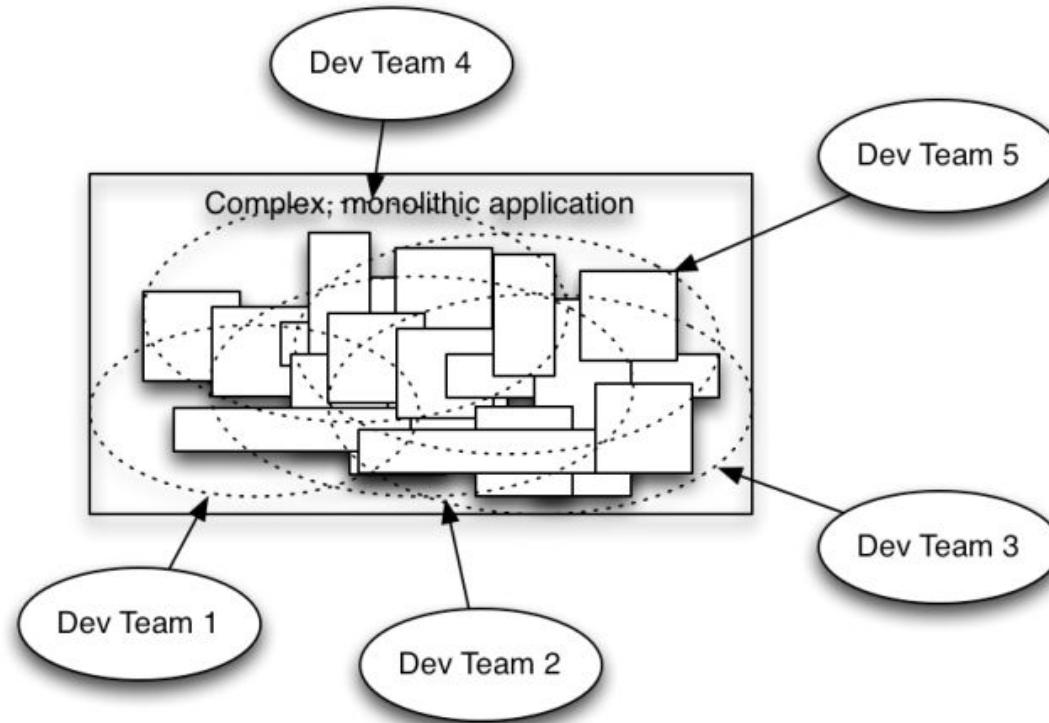
TRADICIONAL: ARQUITETURA MONOLÍTICA



Todos trabalhando juntos...



E a Aplicação segue crescendo...



E então temos um caos!

- Desenvolvimento e implantação ágil se torna impossível;
- Tecnologias usadas começam a ficar obsoletas, mas...
- Refatorar não é mais viável!



Características dos Microserviços

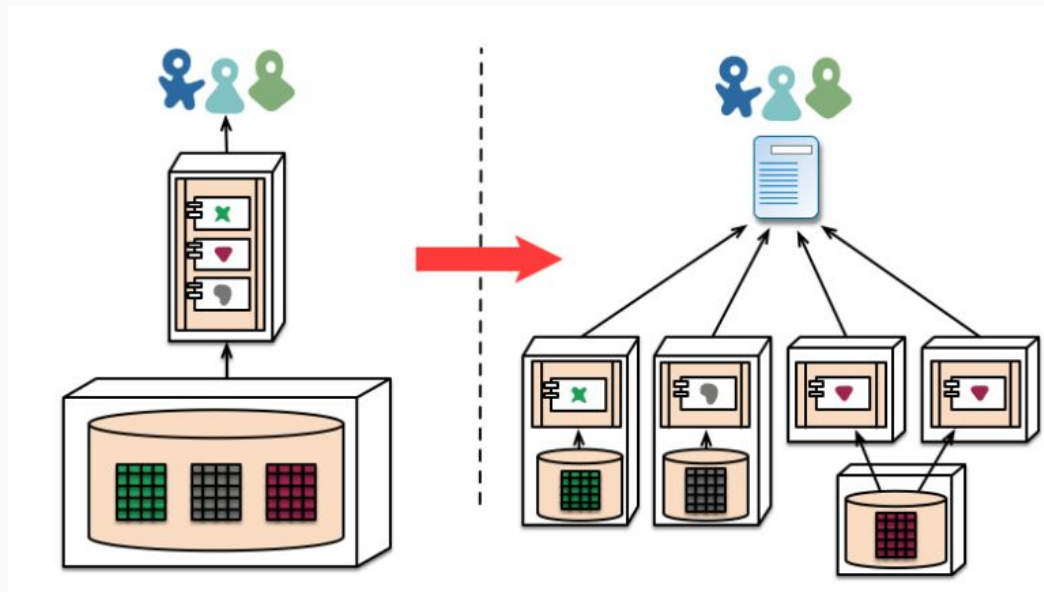
- Multi Linguagem: Java, C#, Python, NodeJs...
- Facilmente escalável;
- Fácil implementação (ou não);
- Fácil refatoração;
- Comunicação síncrona e assíncrona entre serviços.

Características dos Microserviços

Componentização

Independência:

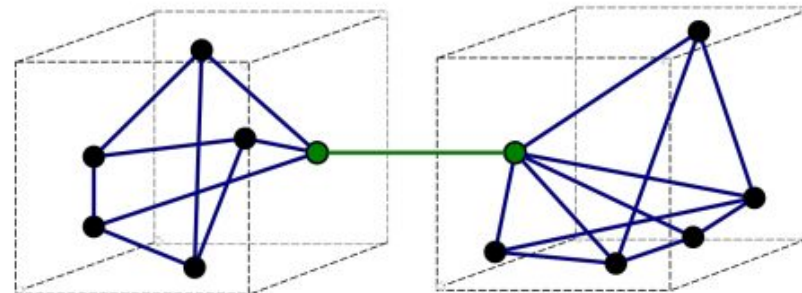
- Os microserviços dividem um sistema complexo em componentes independentes;
- Cada microserviço é uma unidade autônoma, capaz de operar e evoluir independente dos outros serviços.



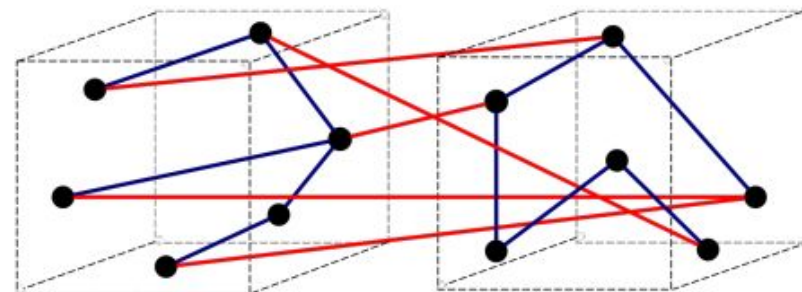
Características dos Microserviços

Desacoplamento:

- Os microsserviços são desacoplados. As mudanças em um serviço não afetam os outros;
- Isso permite que as equipes trabalhem de forma independente em serviços diferentes.



a) Good (loose coupling, high cohesion)

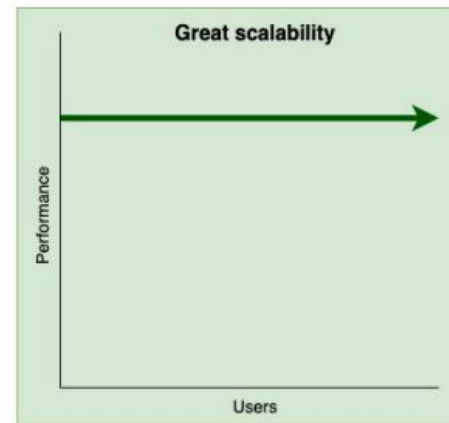
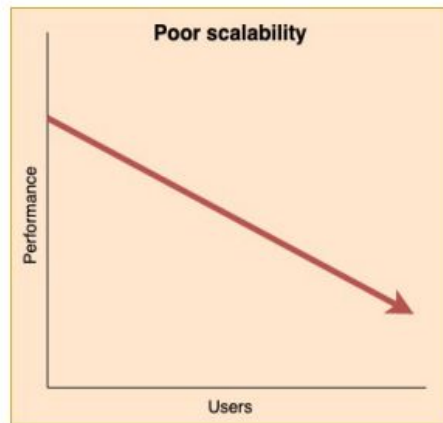


b) Bad (high coupling, low cohesion)

Características dos Microsserviços

Escalabilidade Granular:

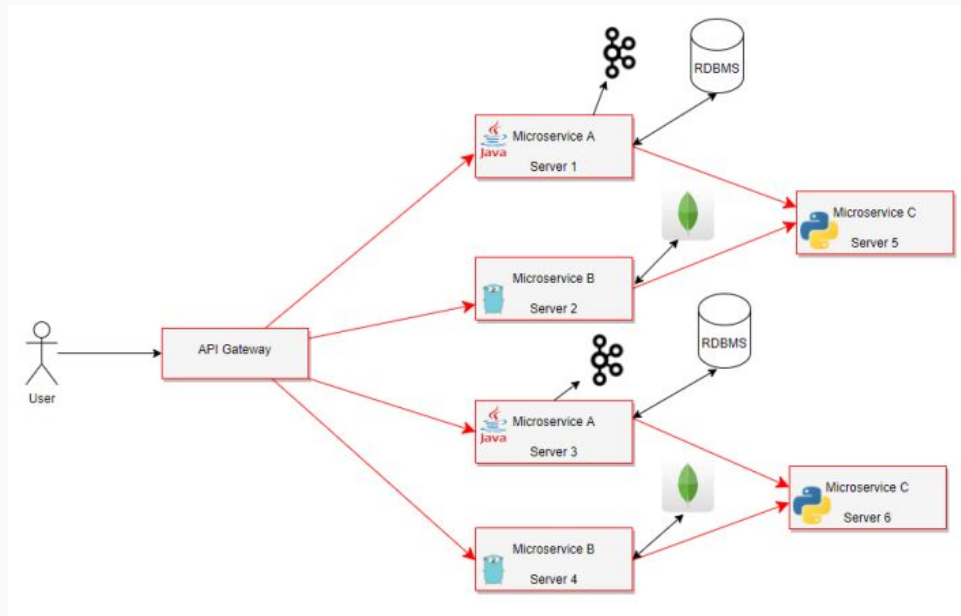
- Cada serviço pode ser escalado de forma independente, permitindo a alocação precisa de recursos onde são mais necessários;
- Isso é especialmente útil para lidar com cargas de trabalho variáveis em partes específicas do sistema.



Características dos Microserviços

Tecnologias Diversificadas:

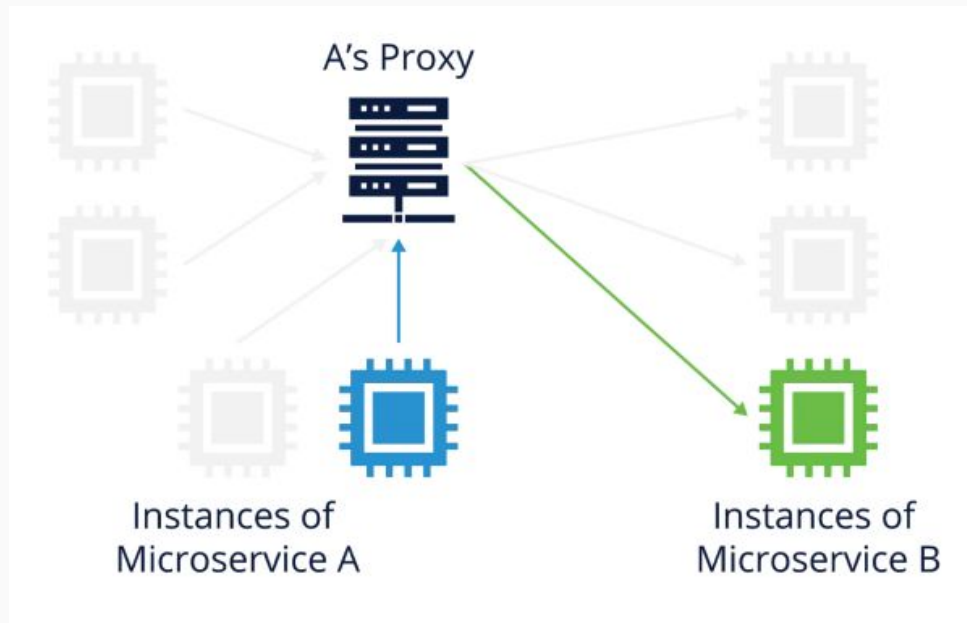
- Cada serviço pode ser desenvolvido usando a tecnologia mais adequada para a funcionalidade que ele oferece;
- Isso possibilita a escolha das melhores ferramentas e linguagens para cada serviço.



Características dos Microserviços

Resiliência e Tolerância a Falhas:

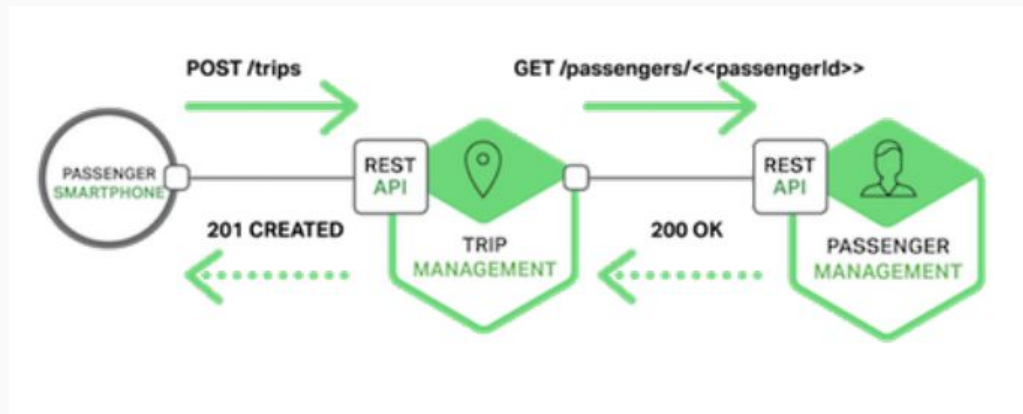
- Se um serviço falhar, não afetará necessariamente o sistema inteiro;
- Redundância e escalabilidade ajudam a manter a disponibilidade do sistema mesmo em caso de falhas.



Características dos Microserviços

Comunicação via APIs:

- Os microsserviços se comunicam entre si por meio de APIs bem definidas, geralmente usando protocolos como HTTP/REST ou mensagens assíncronas.



Vantagens dos Microsserviços

- Permite a integração e implantação contínua de aplicações grandes e complexas;
- Melhor estabilidade - serviços menores são mais fáceis de testar;
- Melhor “implantabilidade” - serviços podem ser implantados de maneira independente;
- Permite a organização do desenvolvimento em múltiplos times com responsabilidades específicas e independentes.

Vantagens dos Microsserviços

- Cada microsserviço é relativamente pequeno - fácil de entender e mais rápido de manipular na IDE;
- A aplicação inicia mais rápido, o que diminui o tempo de implantação e torna o desenvolvimento mais produtivo;

Vantagens dos Microsserviços

- Melhor isolamento de falha: se há vazamento de memória em um serviço, só ele será afetado. Os outros continuarão a responder as requisições normalmente.
- Elimina o comprometimento a longo prazo com um stack de tecnologias. Quando um novo serviço é desenvolvido, ele pode utilizar novas tecnologias. Reescrever um serviço existente também torna-se viável.

Desvantagens dos Microsserviços

- Desenvolvedores precisam lidar com a complexidade adicional de criar um sistema distribuído;
- Ferramentas/IDEs são orientadas a construir aplicações monolíticas e não proveem suporte explícito para aplicações distribuídas;
- Testes integrados são mais difíceis de executar (são muitos serviços/componentes);

Desvantagens dos Microserviços

- Desenvolvedores precisam implementar um mecanismo de comunicação inter-serviços;
- Implementar casos de uso que contenham com múltiplos serviços sem usar transações distribuídas é difícil;
- Implementar casos de uso que contenham múltiplos serviços requer uma coordenação cuidadosa entre os times;

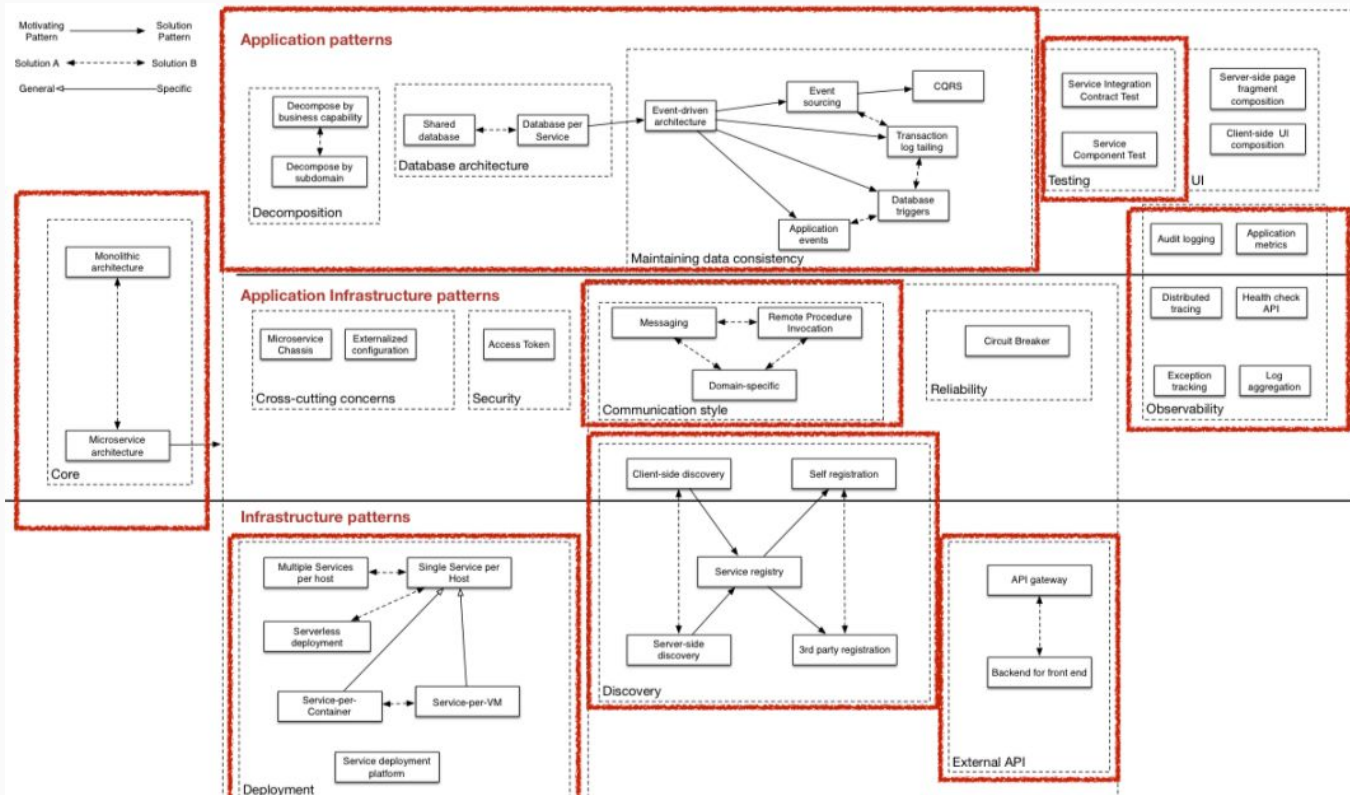
Desvantagens dos Microsserviços

- Maior complexidade de implantação e configuração.
- Aumento do consumo de memória. A arquitetura de microsserviços substitui as N instâncias monolíticas por $N \times M$ instâncias de serviços. Cada uma roda na sua própria JVM (ou equivalente), o que é necessário para isolar as instâncias.

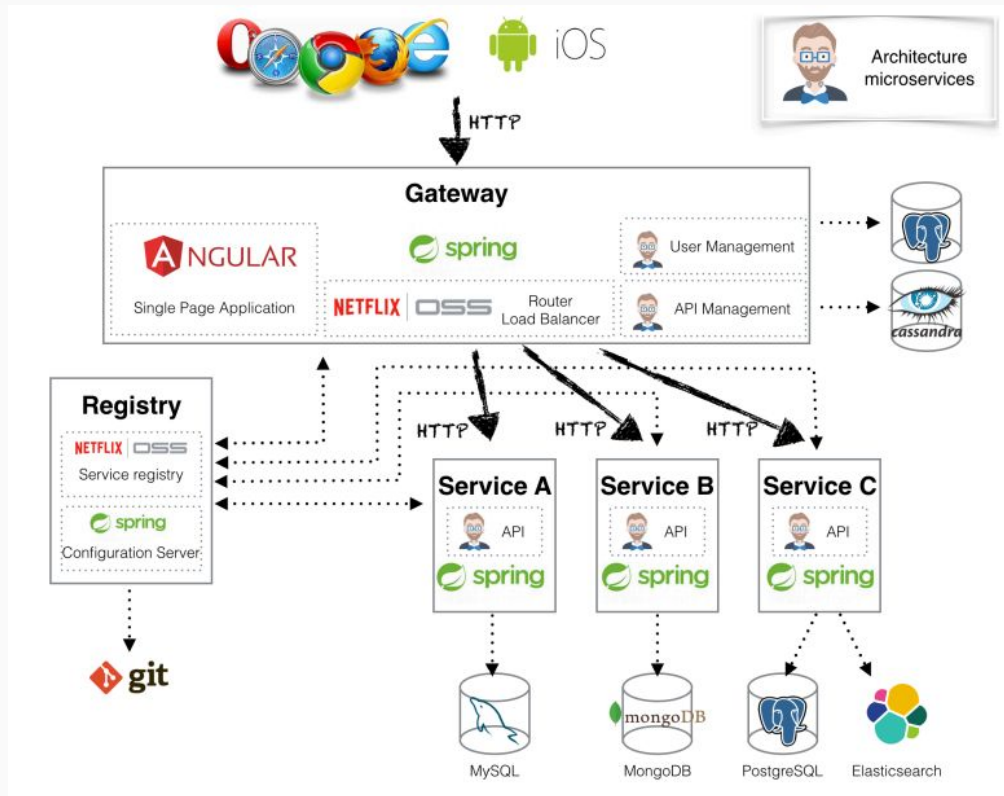
Para mitigar as desvantagens...



Padrões de Microserviços



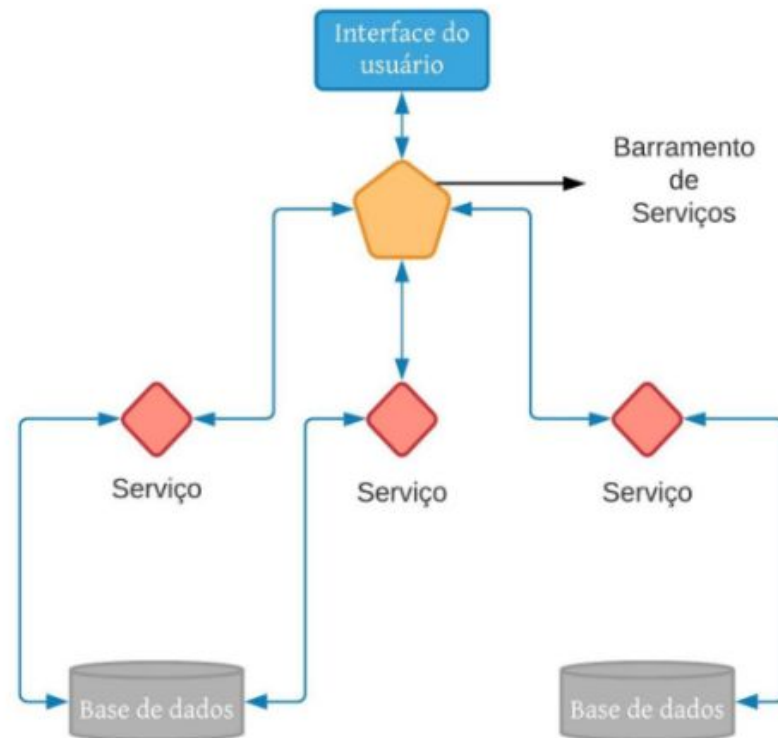
Exemplo de Stack de Tecnologia



Antecessor dos Microserviços

SOA: Arquitetura Orientada a Serviços

- Actor Model;
- Orientado a Objeto;



Microserviços vs SOA

	Microserviço	SOA
Comunicação entre serviços	Comunicação direta entre os serviços utilizando protocolos como REST ou gRPC	Barramento de serviço único que utiliza protocolos como SOAP
Dados	Organização, Esquema e Dados únicos por serviço	Organização, Esquema e Dados compartilhados por toda a aplicação
Utilização	Serviços pequenos e simples	Grandes aplicações monolíticas

O Ecossistema dos Microserviços

- Uma abordagem para engenharia de *software* aplicada ao ecossistema de microserviços foi apresentada por Susan J. Fowler, em seu livro *Production-Ready Microservices*;
- O ecossistema é dividido em quatro camadas interligadas:



1. Hardware

- São os computadores e equipamentos nos quais os microsserviços são armazenados e executados;
- Estes servidores podem estar dentro da empresa, ou ainda pertencerem a provedores de infraestrutura em nuvem:
 - Amazon Web Services (AWS),
 - Google Cloud Platform,
 - Windows Azure.
- Tecnologias de containerização e clusterização;
- Monitoramento e logging: eventuais problemas podem ser rapidamente identificados e resolvidos;

2. Comunicação

- Engloba todo o conjunto responsável por permitir a interação entre os microsserviços na aplicação;
- É composta por elementos chave, como:
 - Endpoints de API,
 - Service Discovery,
 - Service Registry,
 - Load Balancers.

2. Comunicação

Endpoints de API:

- Endpoints são os pontos pelos quais toda a comunicação externa acontece;
- É nesses endpoints que o microserviço envia e recebe informações de outros microserviços ou aplicações:

2. Comunicação

Service Discovery e Service Registry:

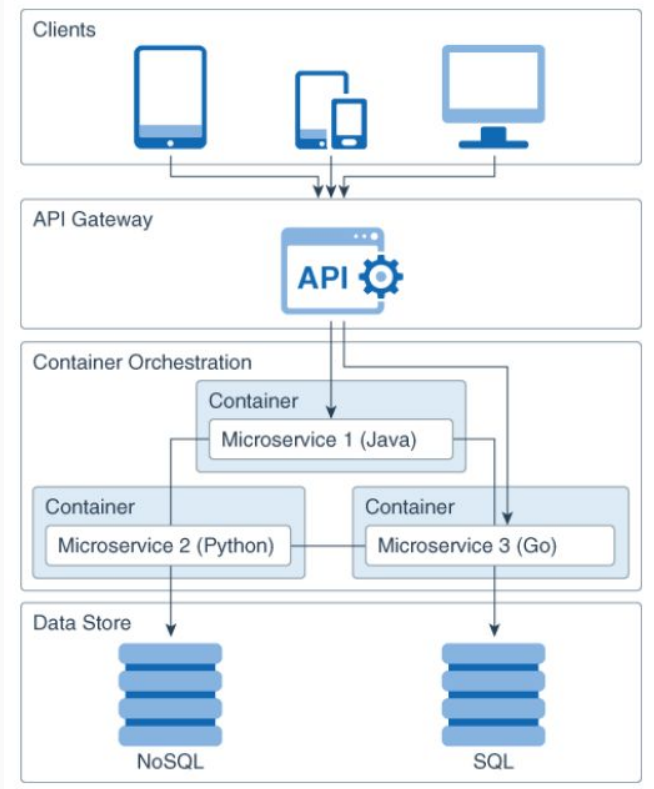
- Configurações de microsserviços mudam dinamicamente com as características de escalabilidade:
 - É necessário ter mecanismos que mantenha a localização atualizada de cada serviço;

2. Comunicação

Load Balancer

- Responsáveis por rotear as requisições que vêm dos clientes para as instâncias do microsserviço;
 - Garante que nenhum servidor seja sobrecarregado, maximizando a velocidade e a capacidade de execução;

2. Comunicação



3. Plataforma de Aplicações

- Engloba todas as ferramentas que são independentes de um microsserviço;
- Repositórios de código e versionamento:
 - GitHub,
 - GitLab,
 - Bitbucket.
- Builds centralizados e automatizados com integração e deploys contínuos;
- Logs centralizados com monitoramento a nível de microsserviço: entender eventuais problemas;

4. Microsserviços

- Compostos apenas pelos códigos e pelas configurações, que permitem a entrega das funcionalidades;
- Tornar os microsserviços totalmente isolados dos demais componentes;
 - Não podem ter contato com as especificidades de *hardware*, service discovery, balanceamento de carga e processos de deploy;

Avanço da Tecnologia

