

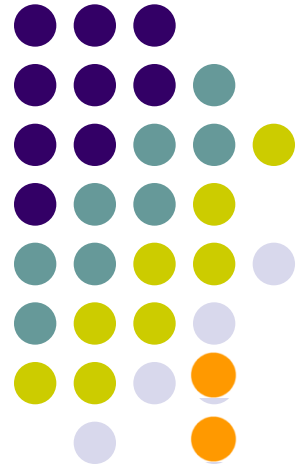
Arquitetura de Software

Microserviços

Microservices, API e API First Design

José Motta Lopes

josemotta@bamplicom



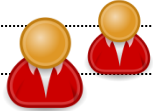


Agenda

- **Pérolas de Jeff Bezos**
- **Efeito Ringelmann**
- **Arquiteturas Monolíticas e Microserviços**
- **Lei de Conway**
- **Organização de Apps Monolíticas e Microserviços**
- **Automação da Infraestrutura em Microserviços**
- **Vantagens de Arquiteturas Monolítica e Microserviço**
- **Aplicação Netflix**
- **API: definição, fases de adoção e ciclo de vida**
- **API First Design**
- **OpenAPI e Swagger**
- **Exercício**



Últimos 50 anos de TI



Interface HM

Processamento

Sistema Arquivos



1970

1980

1990

2000

Terminal

Mainframe

Micro

Disco

Micro

Arquivo

Cliente

Servidor

SQL

Arquivo

- ✓ Mainframes
- ✓ Superminis
- ✓ Porta Serial

- ✓ Ethernet

- ✓ IBM PC
- ✓ MS DOS
- ✓ MS Windows

- ✓ LAN
- ✓ ISO/OSI

- ✓ dBase
- ✓ Novell
- ✓ Linux
- ✓ Open Source
- ✓ Apache
- ✓ Netscape

- ✓ IEEE 802.3

- ✓ Docker
- ✓ Big Data
- ✓ NOSQL
- ✓ API
- ✓ Github

- ✓ Wireless

2002

Jeff Bezos

CEO Amazon

“email mandato”



Jeff Bezos – Amazon 2002



- Os times irão expor dados e funcionalidades através interfaces de serviço.
- As equipes devem se comunicar entre si através des interfaces.
- Não haverá outra forma permitida de comunicação entre processos: sem link direto, sem leitura de dados ou memória compartilhada de outros times, sem porta-dos-fundos. A única comunicação permitida são chamadas à interface de serviços, através da rede.
- Não importa qual tecnologia as equipes usem.
- Todas as interfaces de serviço, sem exceção, devem ser projetadas para serem externalizáveis. O time deve planejar e projetar para expor a interface para desenvolvedores externos.



“Quem não fizer isso será demitido. Tenham um bom dia”



Jeff Bezos – Amazon 2002



A visão descentralizada da empresa na “Regra das duas pizzas”:

- Quando for reunir a equipe ou convidar para a próxima reunião, considere quantas pessoas você pode alimentar com duas pizzas – essa é quantidade de pessoas que devem ser convidadas.



Amazon 2018

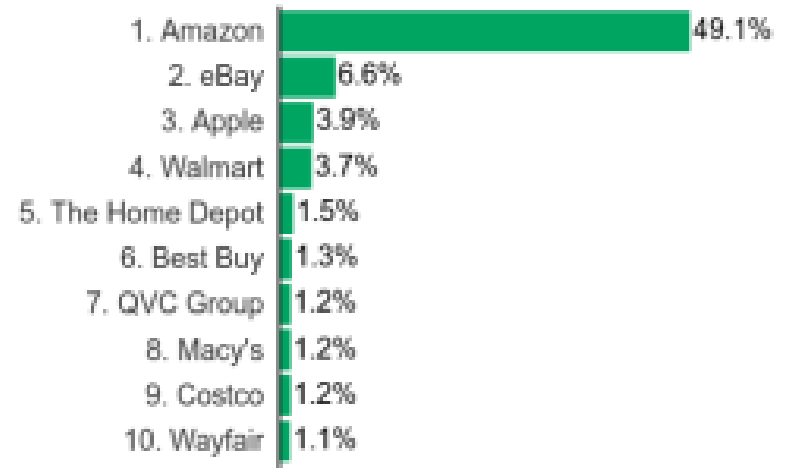


VAREJO EUA

- **\$258.22 Bilhões** em vendas
- **49.1%** de gastos do varejo **online**
- **5%** de todas as vendas no varejo

“Mais pessoas estarão gastando dinheiro online na Amazon do que em todos os outros concorrentes – combinados”

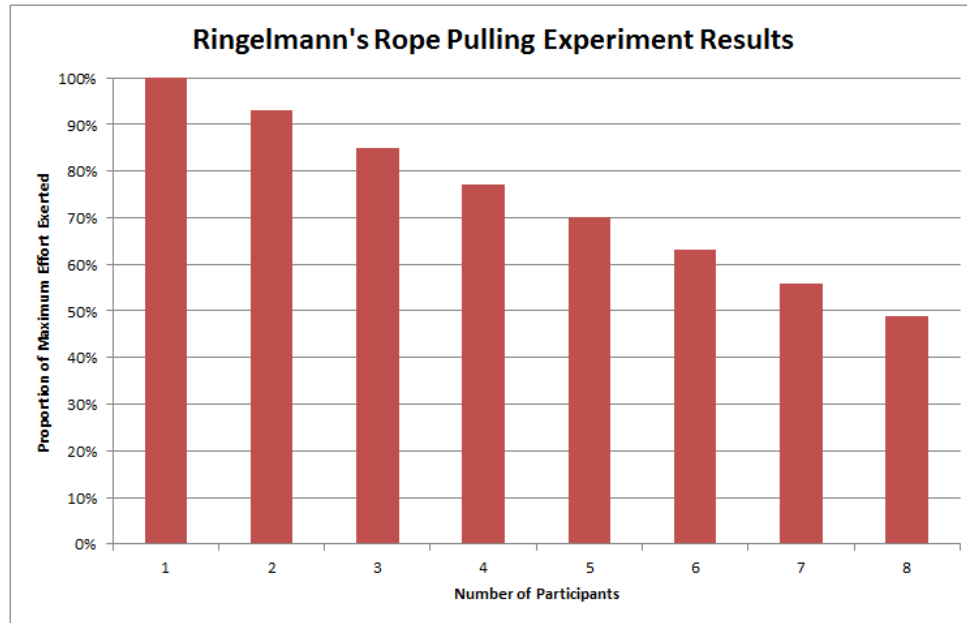
Top 10 U.S. companies based on % of e-commerce sales



Source: eMarketer, July 2018

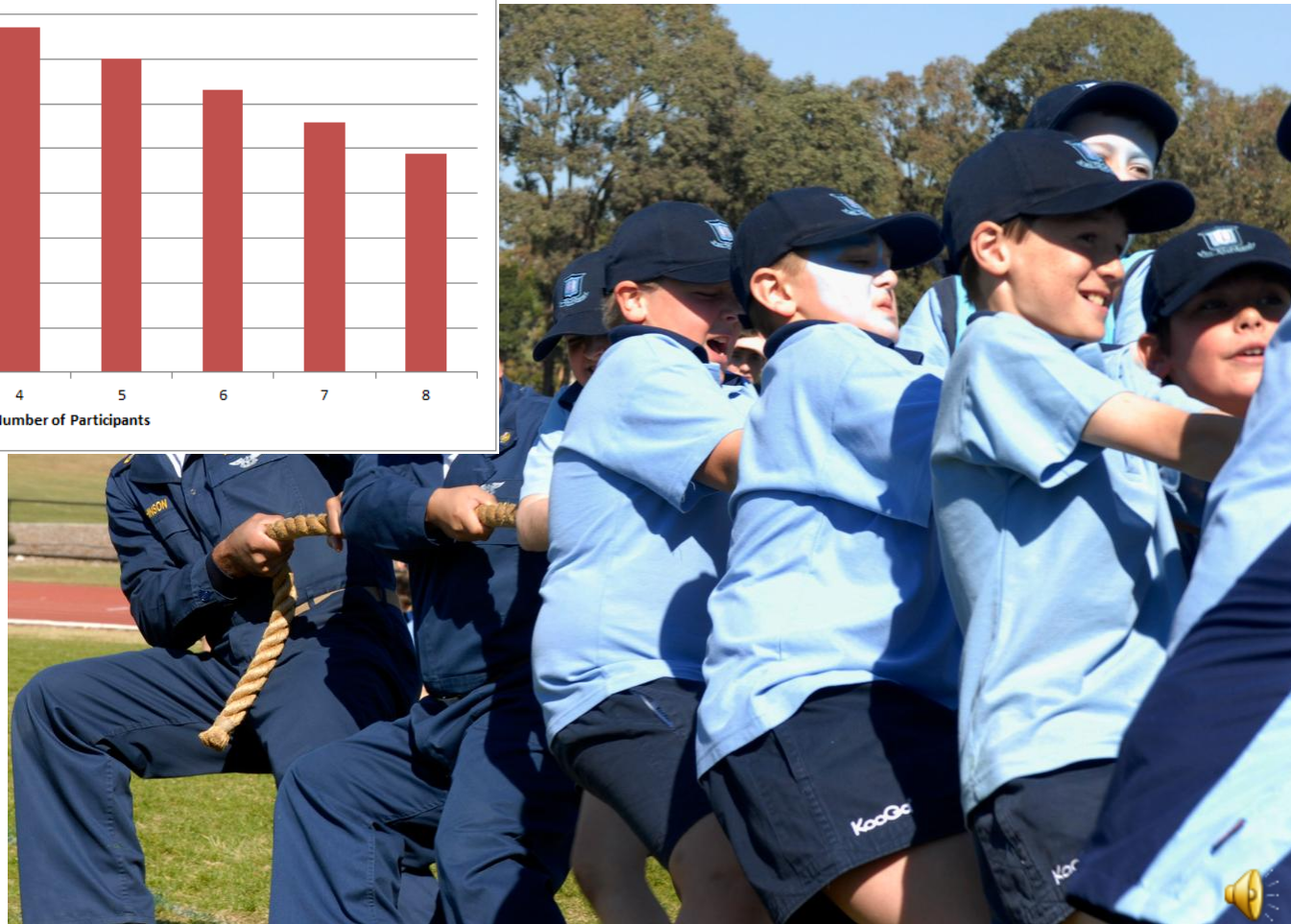


Efeito Ringelmann



Preguiça Social

É a tendência dos membros de um grupo se tornarem cada vez menos produtivos, à medida que o tamanho do grupo cresce.



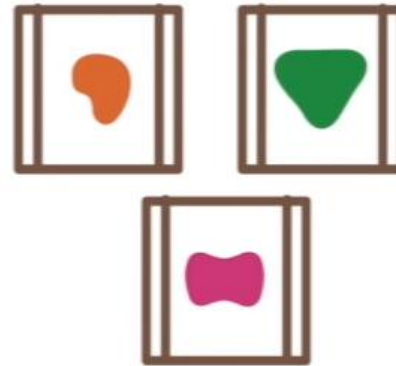
Arquitecturas



Monolítica



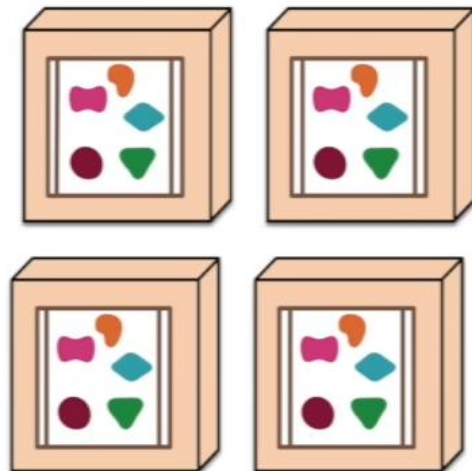
Microservice



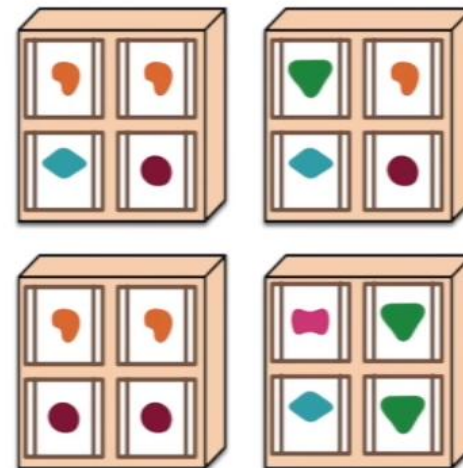
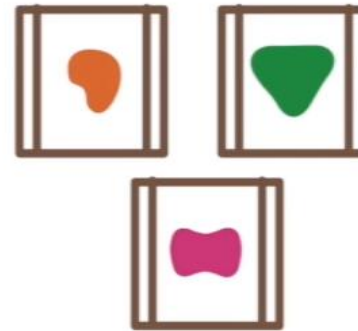
Escalando as aplicações



Monolítica

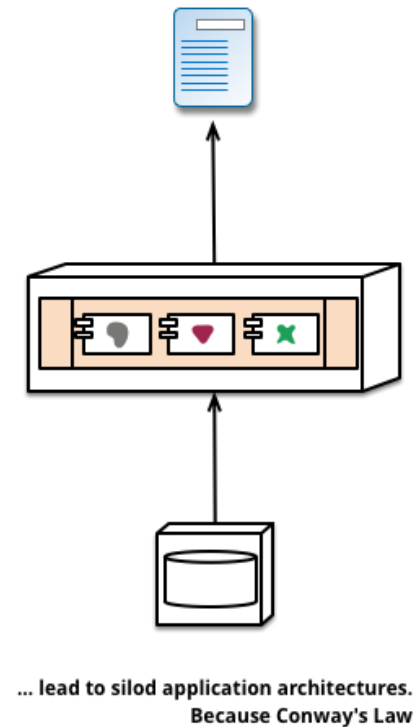
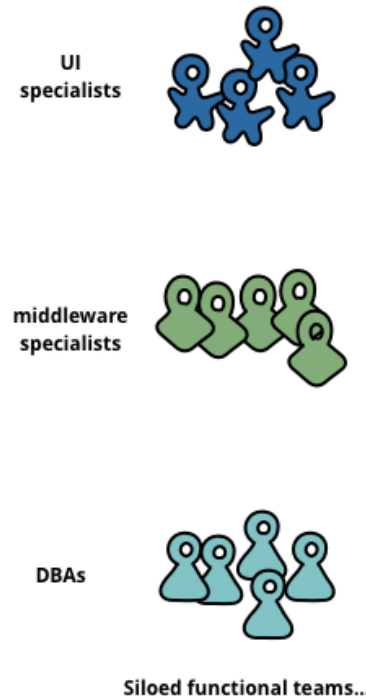


Microservice



Organização monolítica

- Dividem a aplicação focando na tecnologia e organizam equipes de acordo com a capacidade de negócios.
- Uma mudança simples pode envolver vários times, exigindo mais tempo para aprovação de orçamentos.
- As equipes resolvem o conflito com o menor dos males, adaptando a lógica para qualquer aplicação que tenham acesso.



Lei de Conway



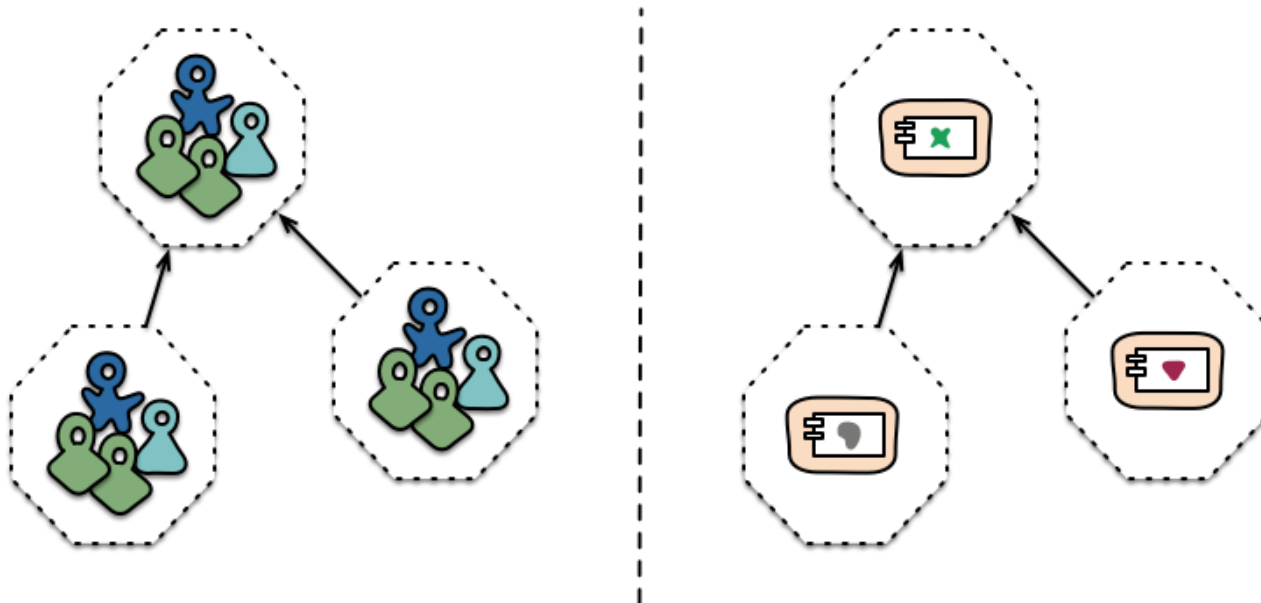
- **As organizações que projetam sistemas estão restritas a produzir projetos que refletem as estruturas de comunicação dessas organizações.**
- **Qualquer peça complexa de software reflete a estrutura organizacional que a produziu.**



Organização microservice



- A aplicação é dividida de forma diferente, decomposta em serviços, organizados em torno da capacidade dos negócios.
- Times multidisciplinares incluem a gama completa das habilidades necessárias para se desenvolver o projeto.



Cross-functional teams...

... organised around capabilities
Because Conway's Law



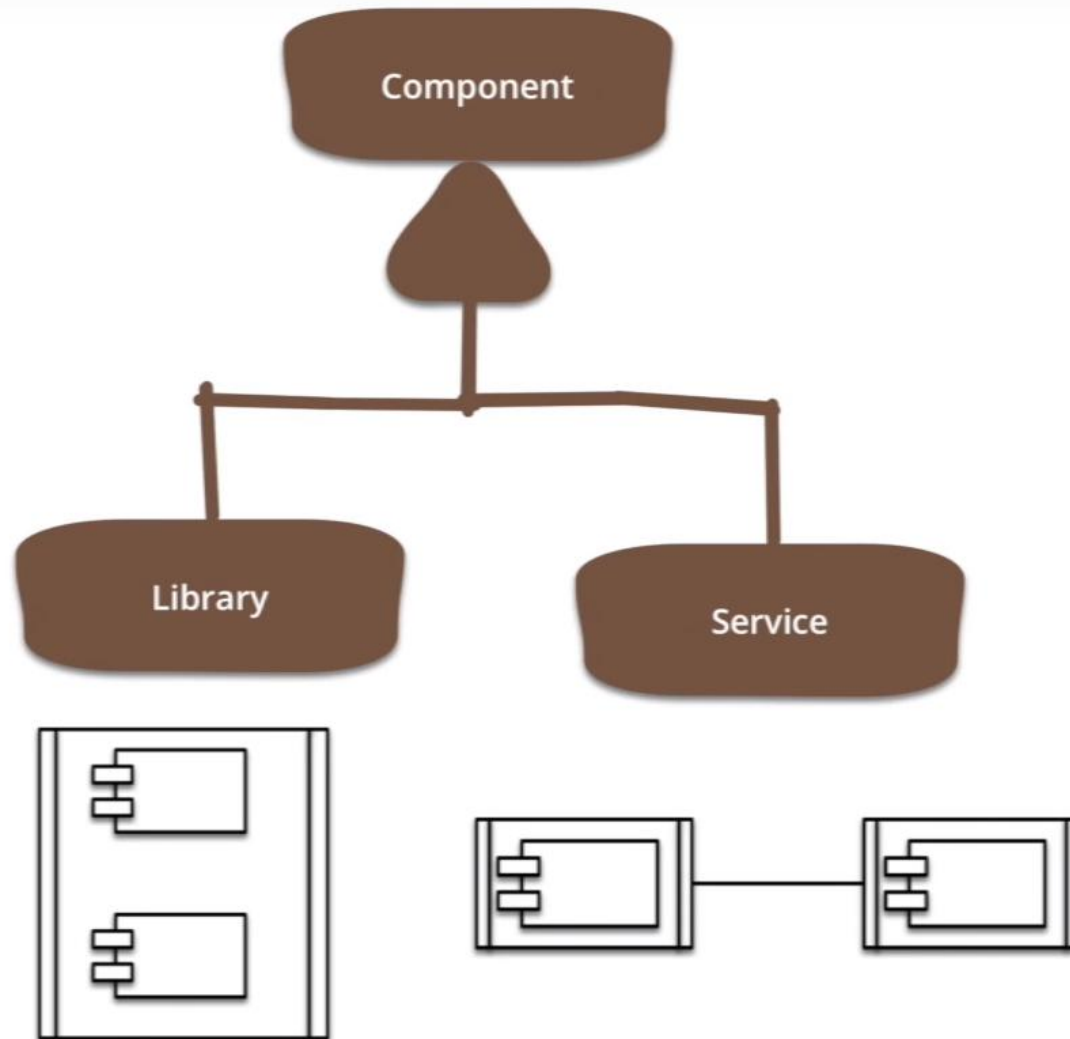
Organização microservice



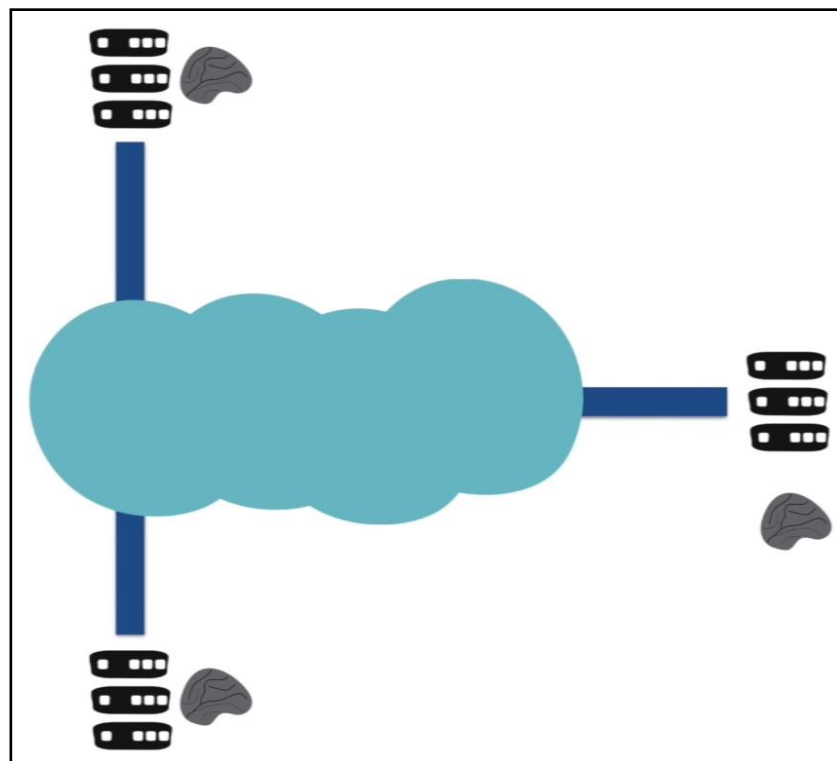
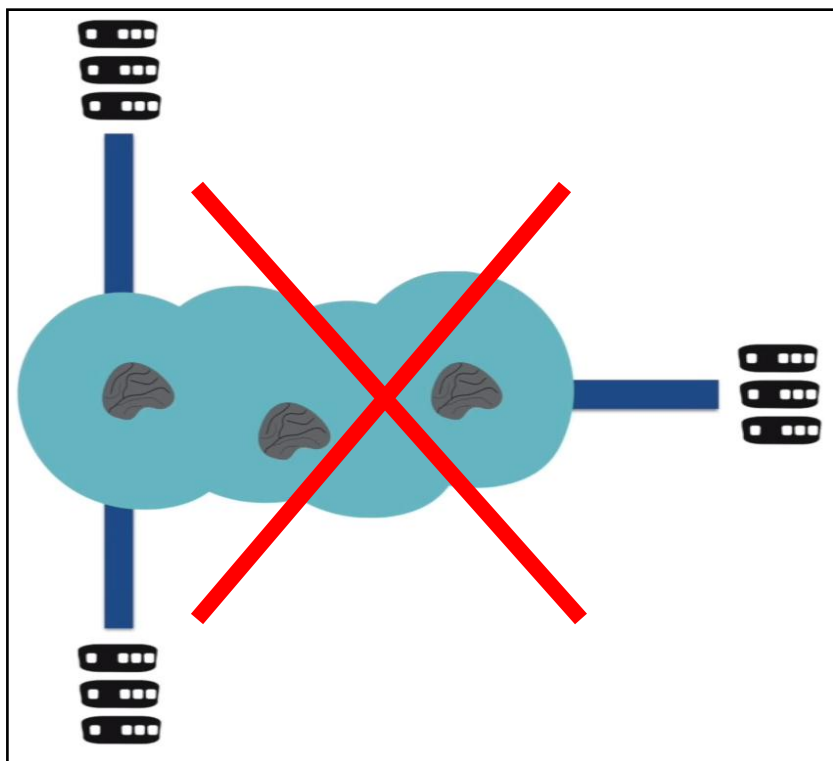
- Na aplicação monolítica, costuma-se usar a noção de que um projeto está sendo feito. Ao final, a equipe de desenvolvimento entrega o software para a manutenção e se desmantela.
- Nos microservices, a equipe de desenvolvimento deve assumir o produto em todo o seu ciclo de vida. No jargão Amazon seria **“you build it, you run it!”**
- Assim, os desenvolvedores experimentam a rotina dos usuários
- Como o software auxilia os usuários a melhorar os negócios?
- A maior granularidade dos serviços facilita esse relacionamento



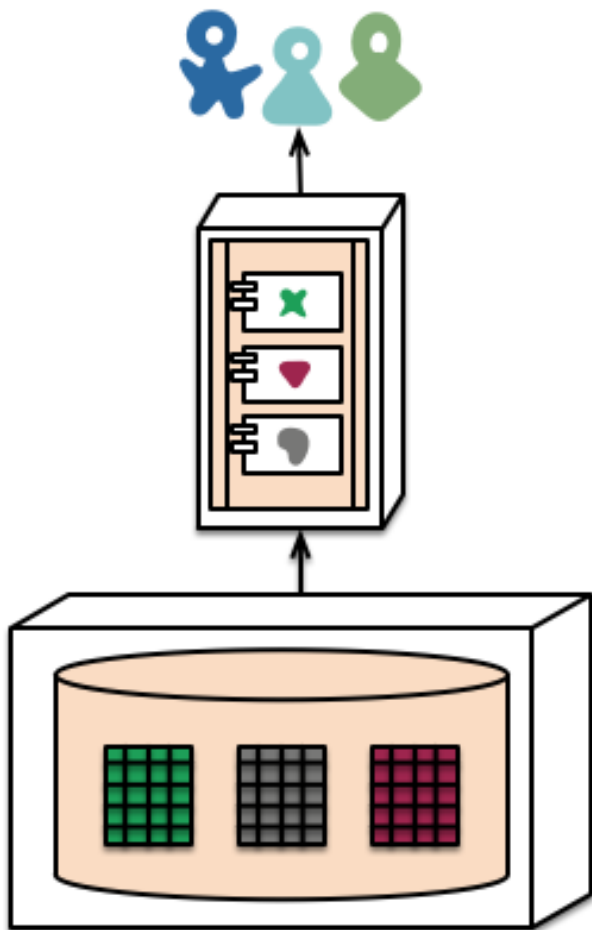
Componentes



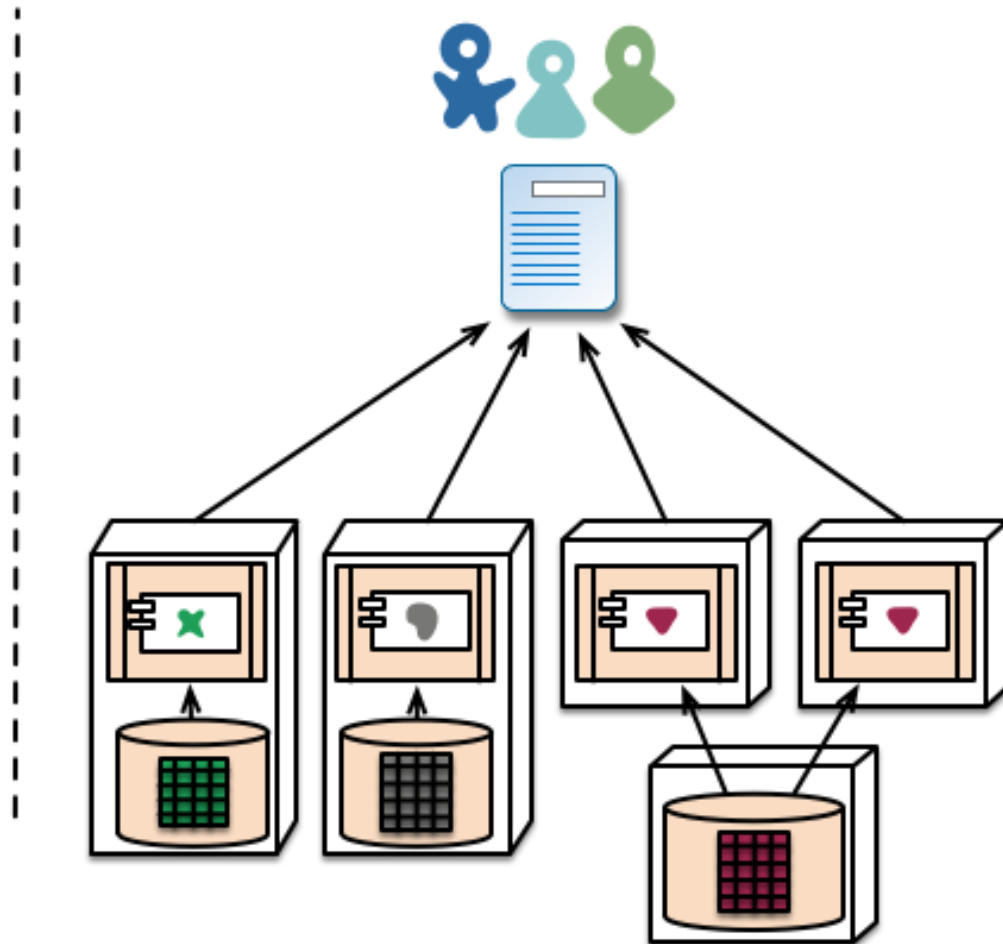
Inteligência nas pontas



Descentralização de Dados

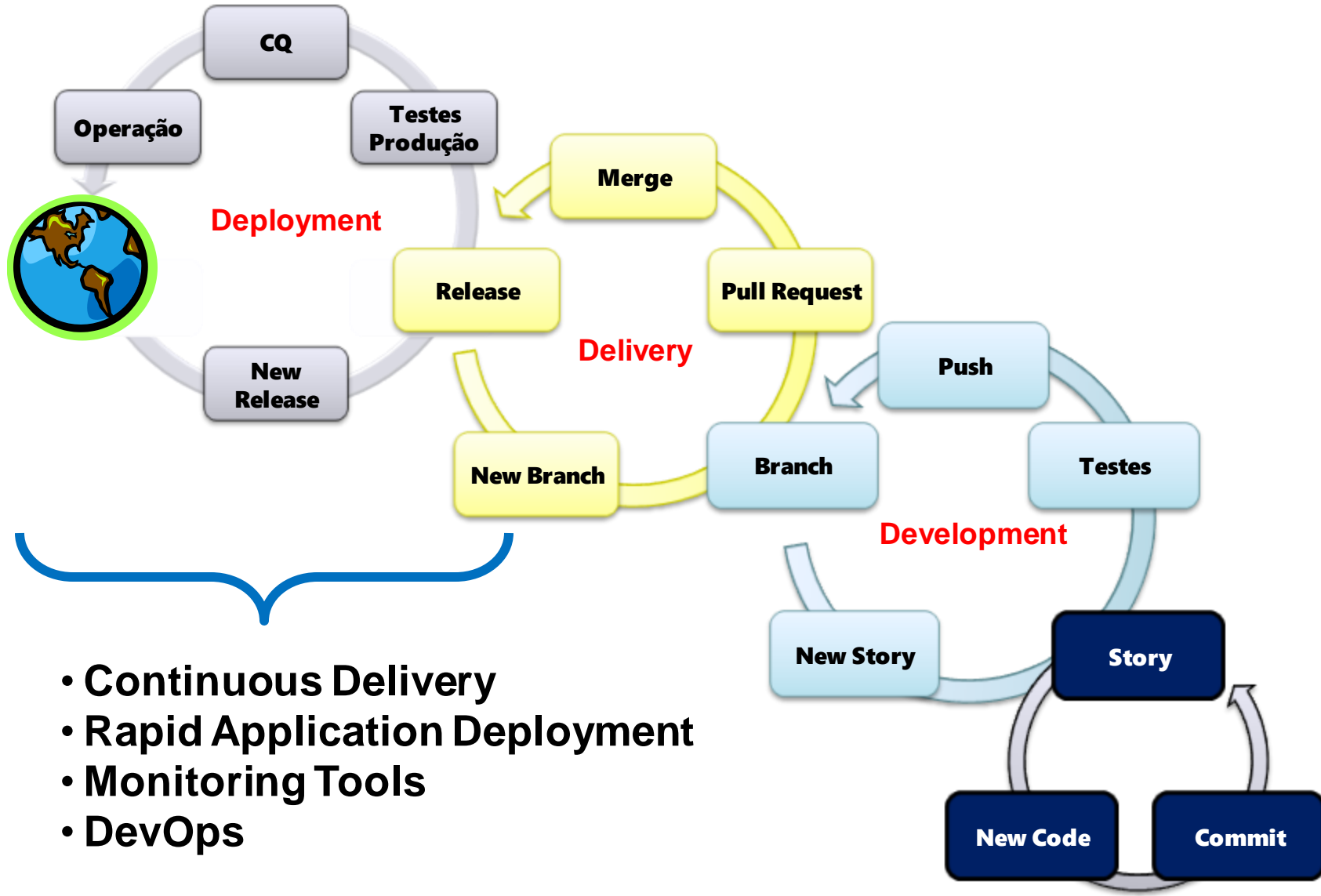


monolith - single database



microservices - application databases

Automação infraestrutura



Projetado para falhar



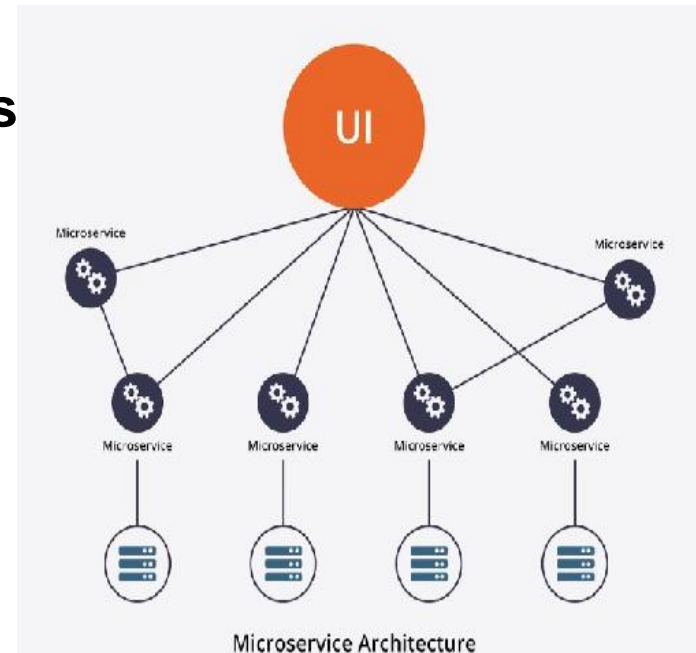
Netflix Chaos Monkey



Microservices



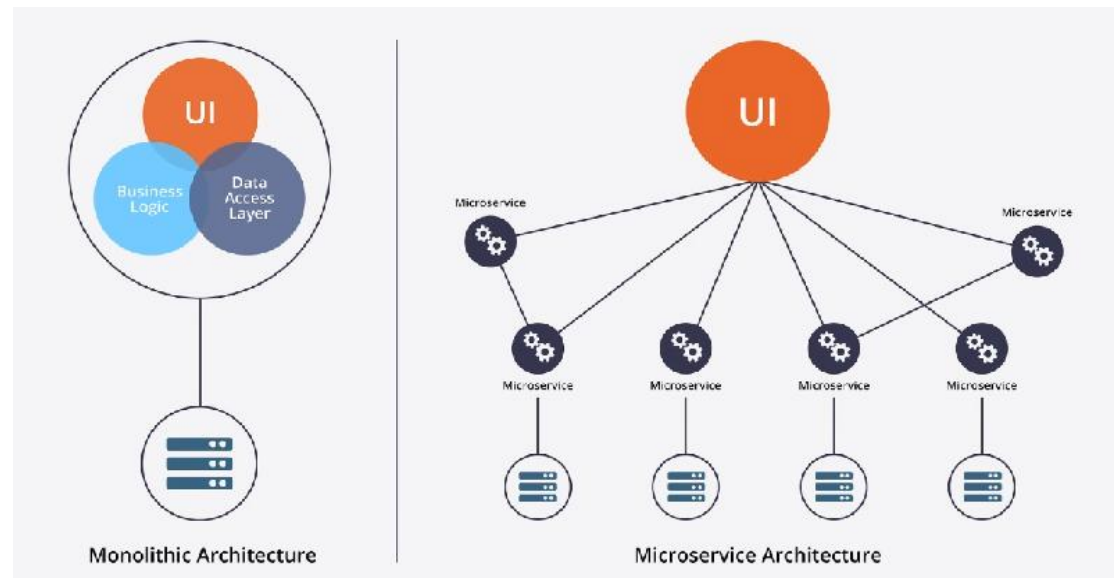
- ***Microservices*** é um estilo de arquitetura com aplicações compostas de serviços ou módulos *loosely coupled*.
- Em vez de um programa grande, diversas aplicações pequenas. Uma pessoa basta para entender o microservice.
- Cada *microservice* tem um *API endpoint* com funções de lógica de negócio bem definidas, protocolos REST ou http.
- Se presta à *continuous integration & continuous deployment* para produção e gerenciamento de aplicações complexas.



Monolítico: vantagens



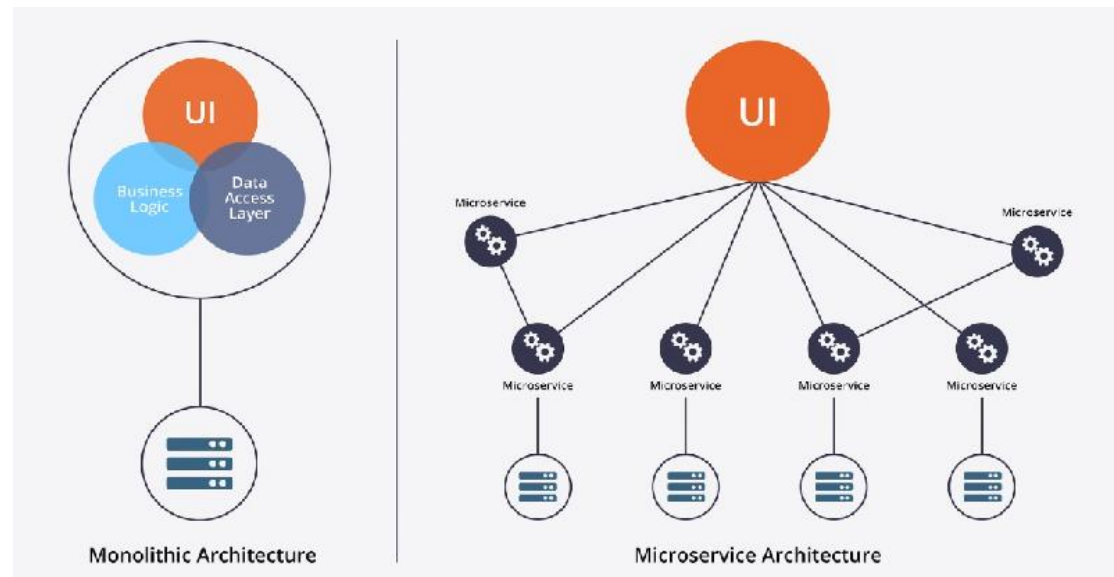
- Simplicidade
- Consistência
- Refactoring entre módulos





Microservices: vantagens

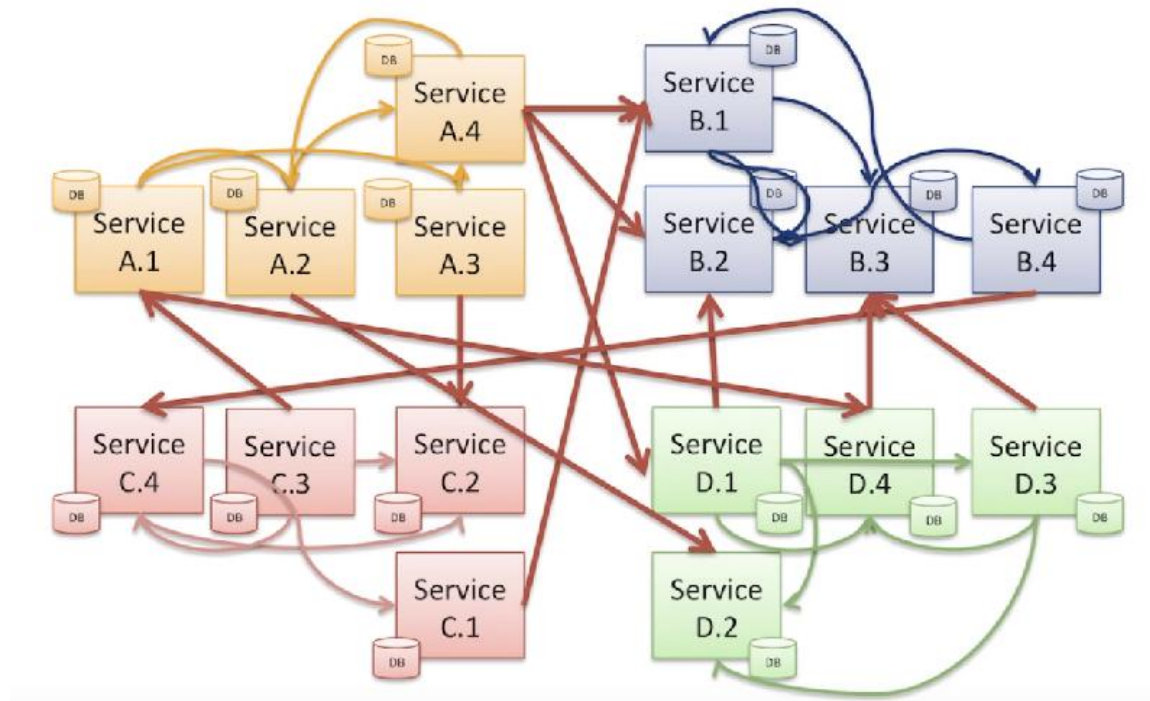
- Deployment parcial
- Maior disponibilidade
- Independem da linguagem/plataforma utilizada
- Iterações rápidas no ciclo de P&D
- Times pequenos decidem pelo microservice
- Isolação de falhas e degradação parcial
- Combinam com containers
- Modularidade
- **ESCALÁVEIS**



Microservices: desvantagens



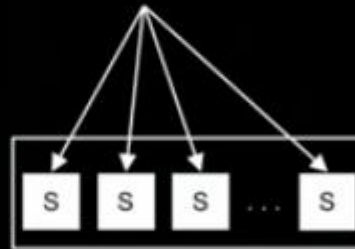
- Networking mais complexo
- Overhead
 - DB
 - Servidores



Microservices



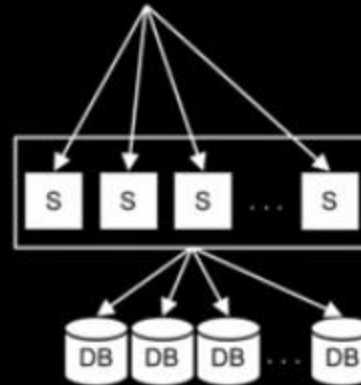
Microservices are an abstraction



Microservices



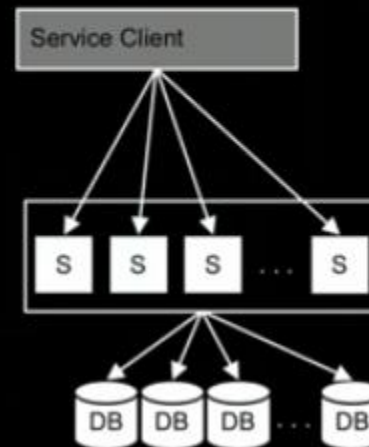
Microservices are an abstraction



Microservices



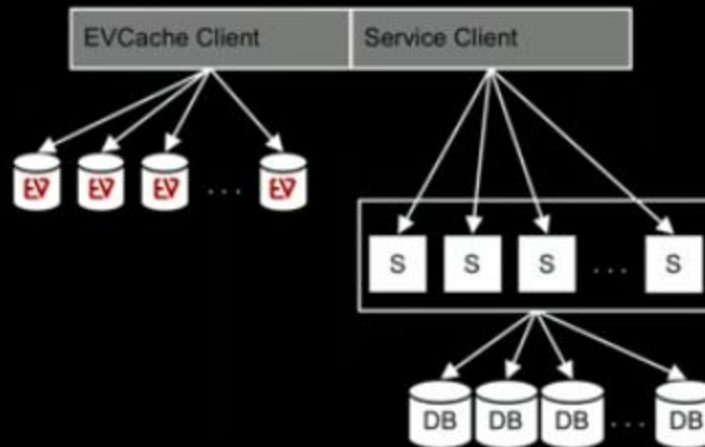
Microservices are an abstraction



Microservices



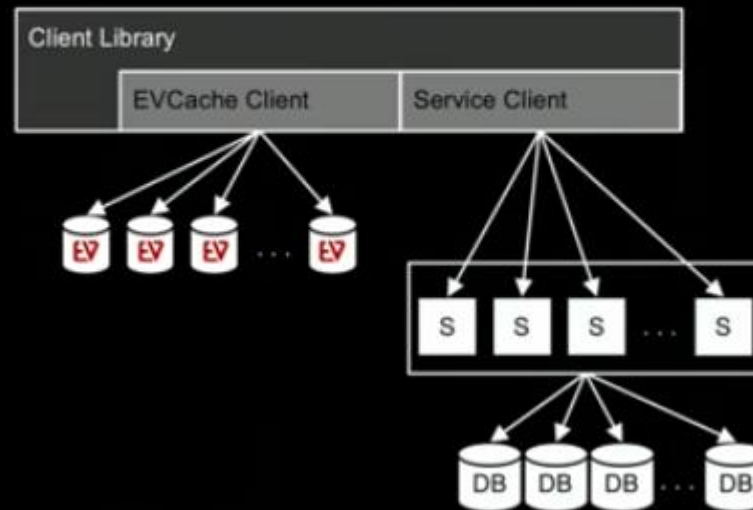
Microservices are an abstraction



Microservices



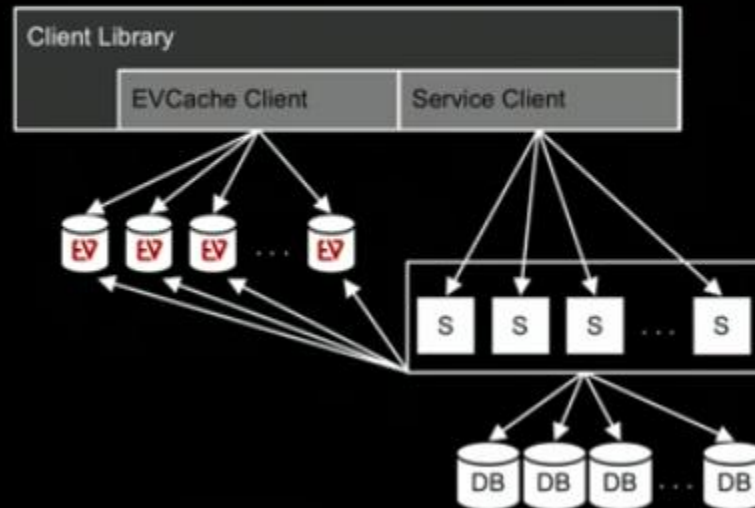
Microservices are an abstraction



Microservices



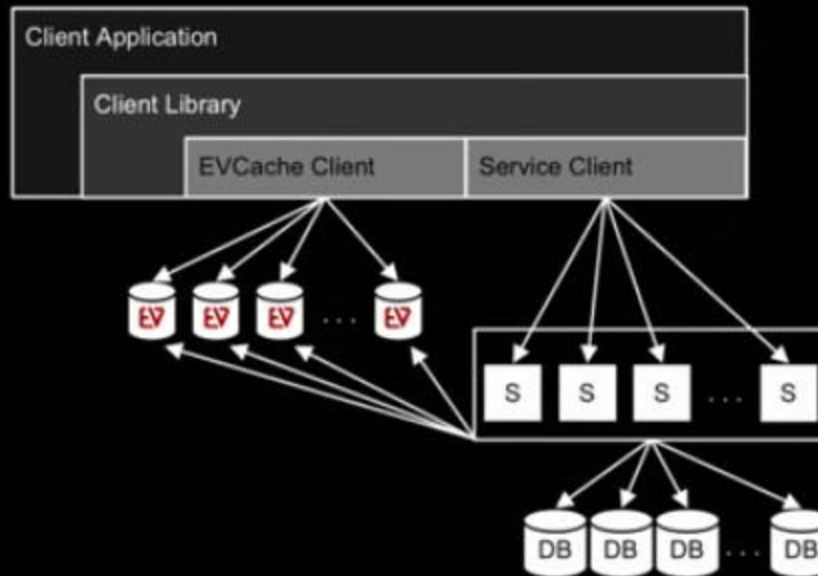
Microservices are an abstraction



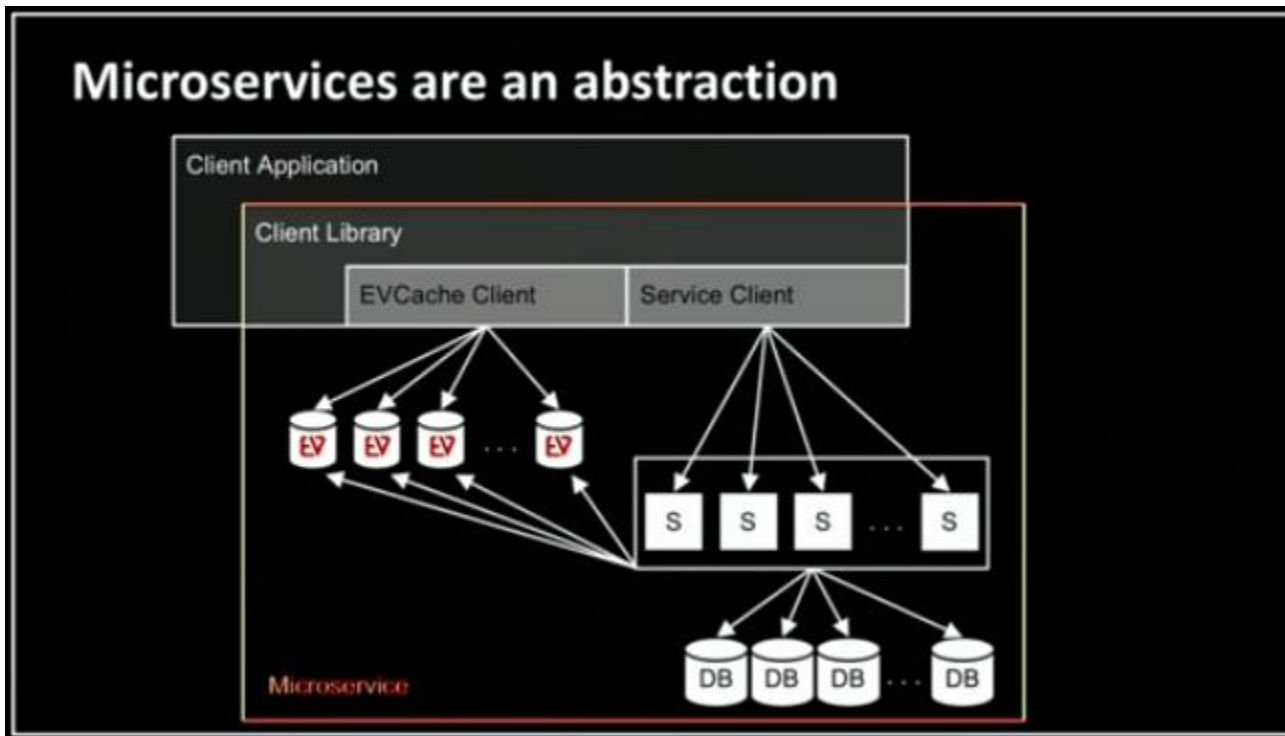
Microservices



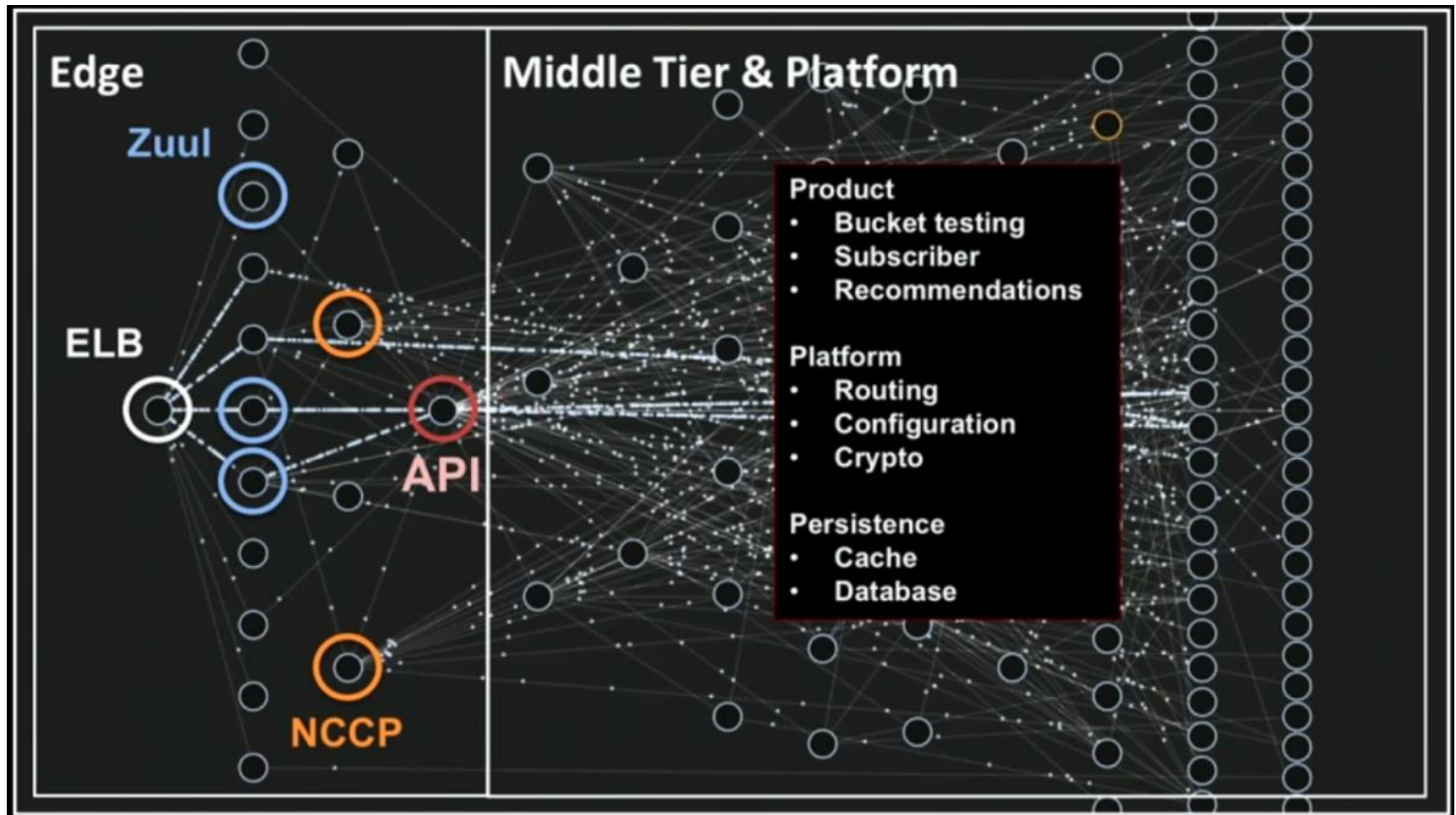
Microservices are an abstraction



Microservices



Aplicação Netflix

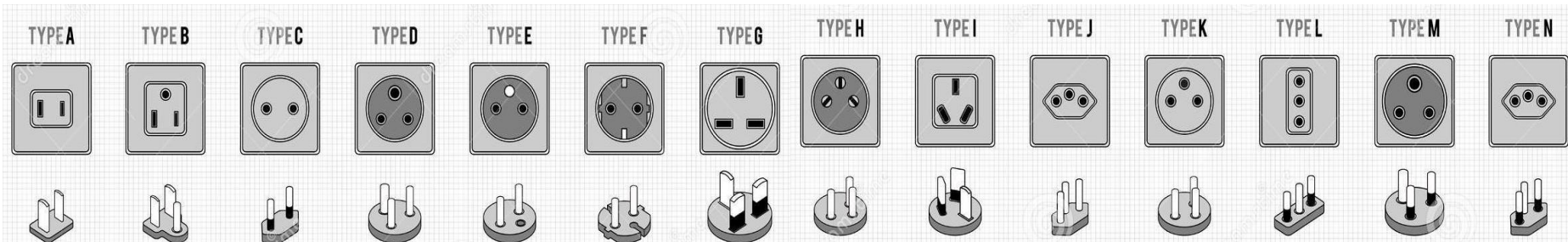


API



Application Programming Interface

- API é um contrato que especifica dados e operações.
- Integra toda a equipe:
 - Arquiteto projetista foca em valor, reuso e inovação.
 - Desenvolvedores da aplicação consomem a API.
 - Desenvolvedores do serviço implementam a API.
- API se abstrai da implementação
- Pode haver várias implementações de uma mesma API.
- Protocolos http, REST são os mais usados.



API: Fases de adoção



API: private / public / partner

1. **Adhoc**: adoção limitada, sem documentos formais.
2. **Desenvolvimento**: documentos como referência.
3. **Habilitada**: apps ativadas e consumidas com orientação.
4. **Ativa**: apps capazes de se auto ativar (orientação mínima).
5. **Comunidade**: API compartilhada e treinada sem orientação.

API: Ciclo de vida



Evitar estagnação, suportar toda a jornada de consumo

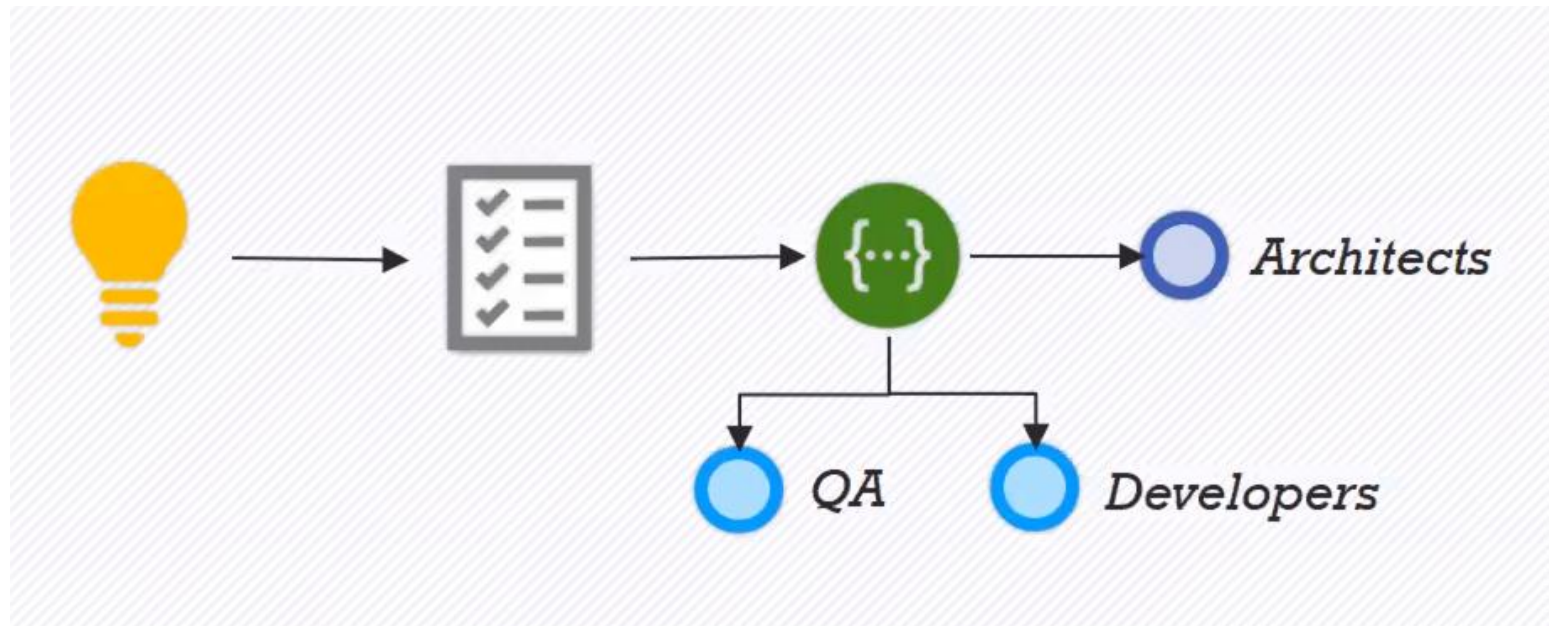
- **Descoberta**: identificar as potencialidades e aptidões.
- **Mapeamento**: propor soluções usando a capacidade da API.
- **Exploração**: consumo na fase de protótipo.
- **Ativação**: registro para acesso nas fases de *test & stage*.
- **Integração**: desenvolvedores consomem API via código.
- **Certificação**: aprovação da API de produção.
- **Monitoramento**: monitoração uso e gargalos de produção.
- **Melhoria**: como melhor atender aos clientes?
- **Atualização**: conscientização de uso e notificação melhorias.



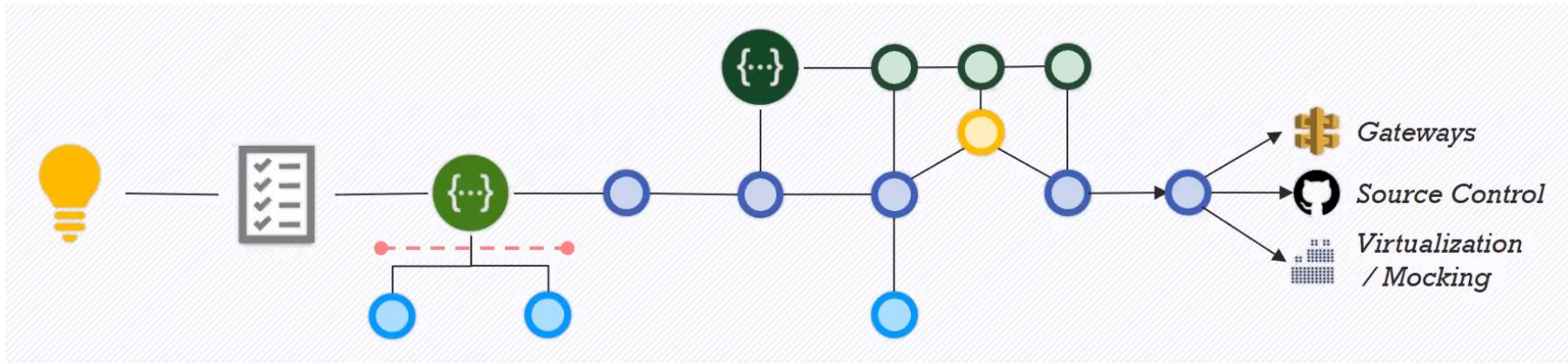
OpenAPI e Swagger

- **Cria contrato RESTful automaticamente**
- **Usado por milhões de desenvolvedores e empresas**
- **Mapeia recursos e operações associadas**
- **Formato do arquivo compatível humano & máquina**
- **SwaggerHub**
 - **Serviço web**
 - **Documenta e gerencia versões da API**
 - **Gera códigos automáticos a partir da API**
 - **Compartilha a biblioteca de API corporativa**

API First Design



API First Design





Exercício

Vamos utilizar o **Github** e o **Pivotal Tracker** neste curso.
Para ter acesso de escrita aos recursos, será preciso se autenticar.
Caso ainda não tenha, favor abrir uma conta gratuita em ambos os sites.

<https://github.com/>
<https://www.pivotaltracker.com>

Repositório Github:

<https://github.com/bamplifier/mba33>

Projeto Pivotal Tracker:

<https://www.pivotaltracker.com/n/projects/2181753>

GIT (instalar programa):

<https://git-scm.com/downloads>

GUI RÁPIDO:

What is Github?

<https://guides.github.com/activities/hello-world/>

Writing Stories

<https://vimeo.com/118871271>

Github Flow



Workflow & Git commands

