Desenvolvimento Web II

Aula 01 - Introdução

Prof. Fabricio Bizotto

Instituto Federal Catarinense

fabricio.bizotto@ifc.edu.br

Ciência da Computação

6 de janeiro de 2024

Roteiro

Conceitos

A arquitetura da aplicação descreve a estrutura interna e interações entre seus componentes. A arquitetura de uma aplicação é composta por:

- **Componentes**: partes que compõem a aplicação. Exemplos: cliente, servidor, banco de dados, etc.
- **Conectores**: mecanismos que permitem a comunicação entre os componentes. Exemplos: protocolos de comunicação, APIs, etc.
- **Restrições**: regras que definem como os componentes e conectores podem interagir. Exemplos: autenticação, autorização, etc.

Tomada de Decisão

Escolher a arquitetura correta para uma aplicação é uma das decisões mais importantes que um arquiteto de software deve tomar.

Modelos Arquiteturais

Monolito

Monolithic

Principais Arquiteturas de Software

Monolito - Definição

Abordagem tradicional no desenvolvimento de software na qual todos os componentes de uma aplicação são combinados em uma única unidade totalmente integrada. A aplicação é implantada como uma única base de código que contém todas as funcionalidades.

Vantagens

- **Simplicidade da Arquitetura**: não existem muitas camadas e componentes para gerenciar. É mais fácil para começar.
- **Tecnologias**: usar uma única linguagem de programação e tecnologias para desenvolver a aplicação pode facilitar o entendimento da equipe.
- **Fluxo de implantação**: o ‘deploy‘ é simples de fazer e gerenciar. Não há necessidade de implantar vários componentes separadamente.

Desvantagens

- **Acoplamento forte**: a aplicação é uma única unidade totalmente integrada, o que significa que qualquer alteração em um componente pode afetar outros componentes da aplicação.
- **Escalabilidade limitada**:a aplicação é implantada como uma única unidade, o que significa que todos os componentes da aplicação devem ser escalados juntos horizontalmente.
- **Implantação única**: a aplicação é implantada como uma única unidade, o que significa que todos os componentes da aplicação devem ser implantados juntos. Qualquer alteração, por menor que seja, requer a implantação de toda a aplicação.
- **Tecnologias limitadas**: todos os componentes da aplicação devem ser desenvolvidos usando a mesma linguagem de programação e tecnologias.

Principais Arquiteturas de Software

Monolito - Representação

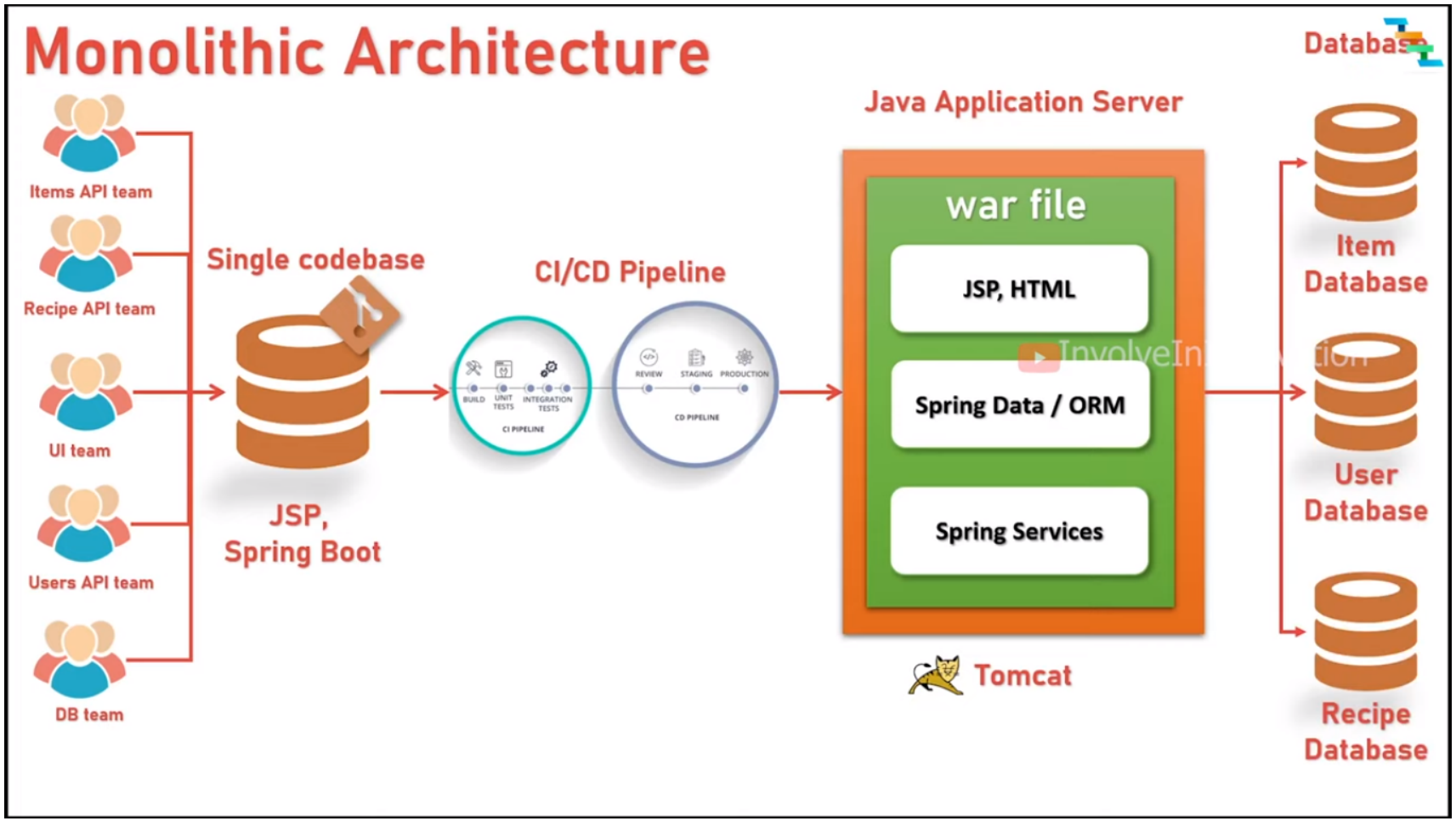

Figura: Arquitetura Monolítica.

- **Escalabilidade horizontal**: adicionar mais instâncias de um componente.
- **Escalabilidade vertical**: adicionar mais recursos (CPU, memória, etc) a um componente.

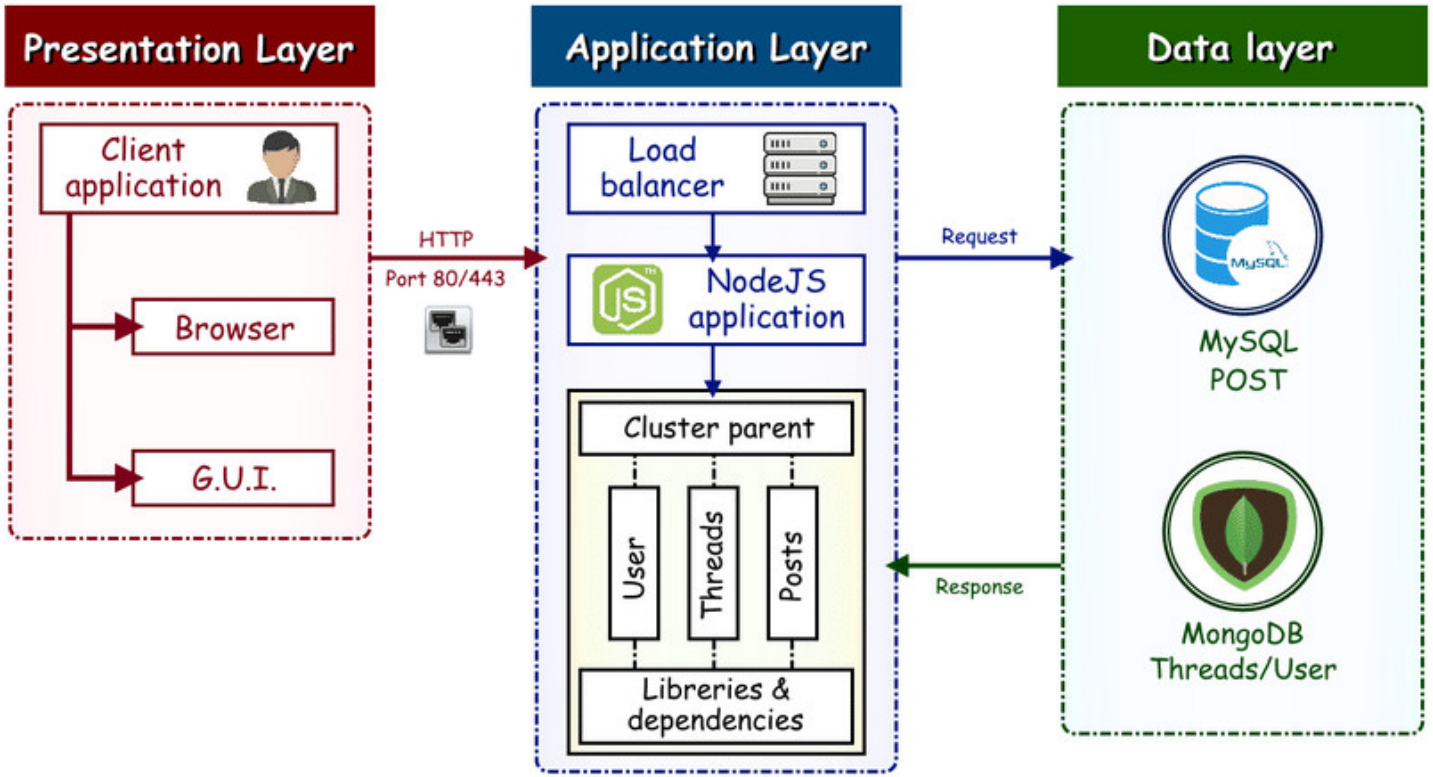

Figura: Arquitetura Monolítica com *Load Balancer*.

Modelos Arquiteturais

N camadas

N-tier

Principais Arquiteturas de Software

Arquitetura em N camadas (N-tier) - Definição

A arquitetura em N camadas é um padrão de arquitetura de software no qual a aplicação é dividida em camadas lógicas ou físicas.

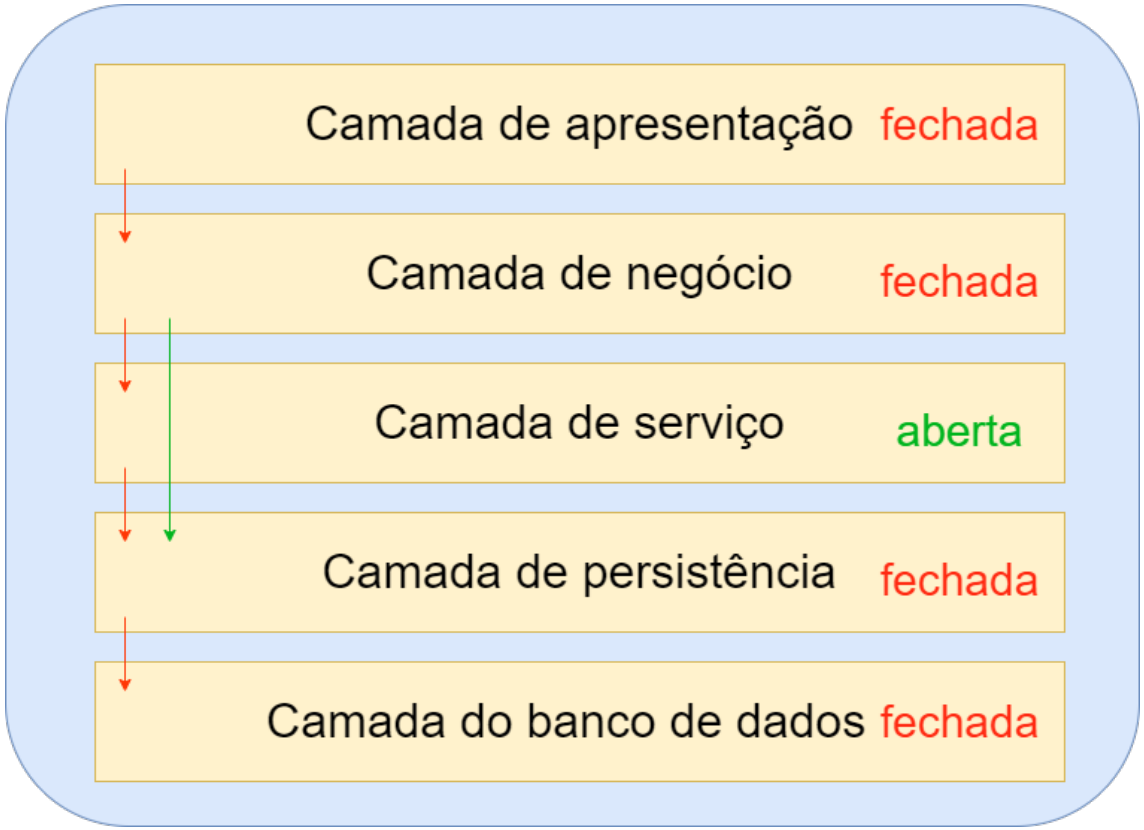

Figura: Arquitetura em camadas - Fluxo.

Vantagens

- **Separação de Responsabilidades**: A separação clara das responsabilidades em diferentes camadas (como apresentação, lógica de negócios e acesso a dados) facilita a manutenção e a evolução do sistema.
- **Escalabilidade**: A escalabilidade é facilitada, pois cada camada pode ser dimensionada independentemente das outras, permitindo a otimização de recursos.
- **Facilidade de Testes**: Cada camada pode ser testada separadamente, o que simplifica os testes unitários e facilita a identificação e correção de falhas.
- **Manutenção**: Alterações em uma camada específica não devem afetar as outras, tornando a manutenção mais simples e menos propensa a efeitos colaterais indesejados.

Desvantagens

- **Complexidade Inicial**: A implementação de uma arquitetura em camadas pode ser mais complexa inicialmente, especialmente para projetos pequenos ou simples.
- **Comunicação entre camadas**: A comunicação entre camadas pode resultar em algum overhead, especialmente em sistemas distribuídos, o que pode impactar o desempenho. Alguns exemplos são latência, serialização e desserialização de dados, etc.
- **Duplicação de Lógica**: Pode ocorrer uma duplicação de lógica entre as camadas, o que pode levar a inconsistências se não for gerenciado adequadamente.

Principais Arquiteturas de Software

Arquitetura em N camadas (N-tier) - Representação

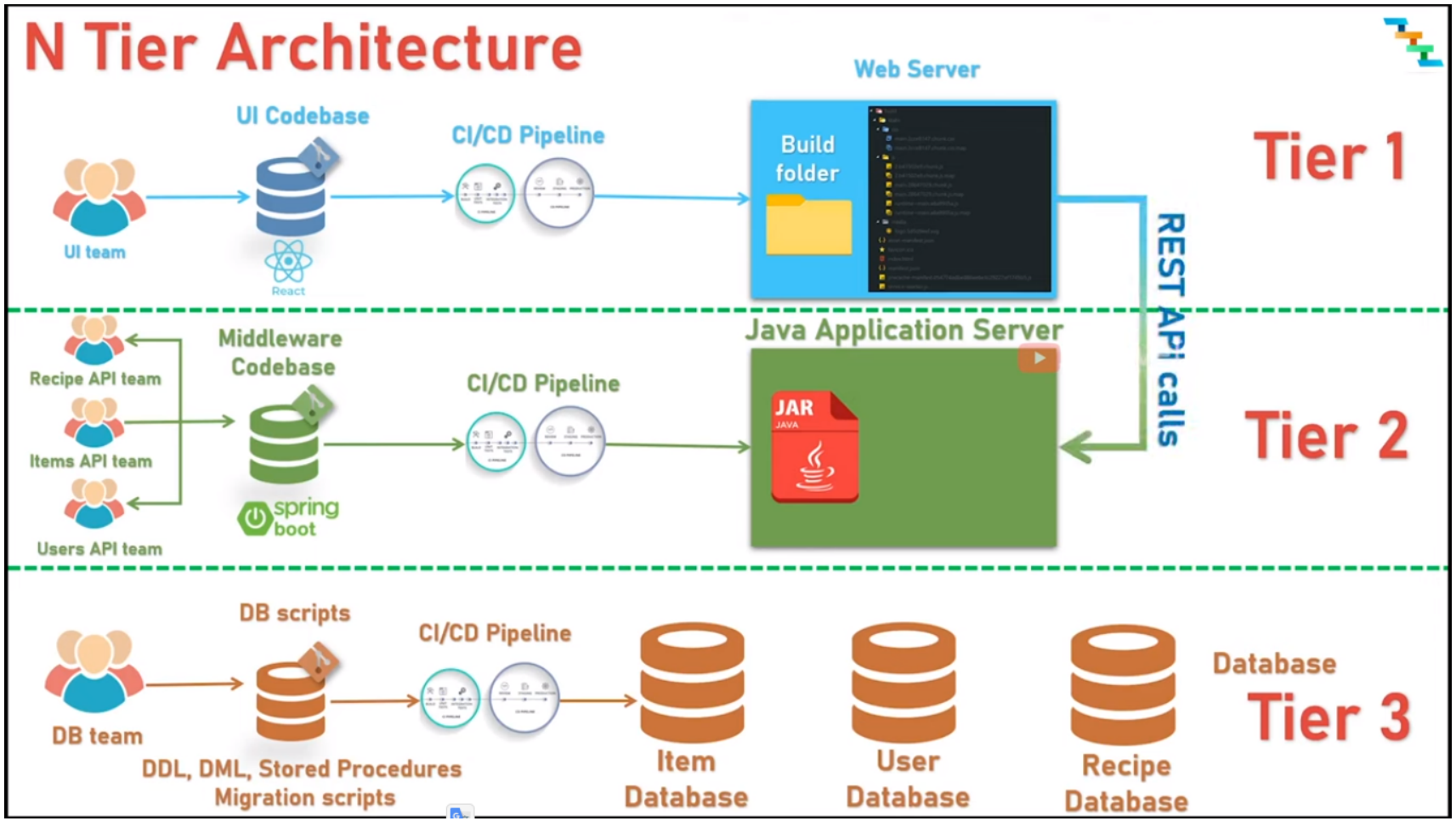

Figura: Arquitetura em N camadas (N-tier).

Principais Arquiteturas de Software

Arquitetura em 2 camadas - Cliente-Servidor

A arquitetura de software em duas camadas, também conhecida como arquitetura cliente-servidor, é um modelo simples no qual a lógica de aplicação é dividida em duas partes principais: a camada de apresentação (cliente) e a camada de dados (servidor). Essa arquitetura é bastante direta e é frequentemente utilizada em aplicações pequenas e simples.

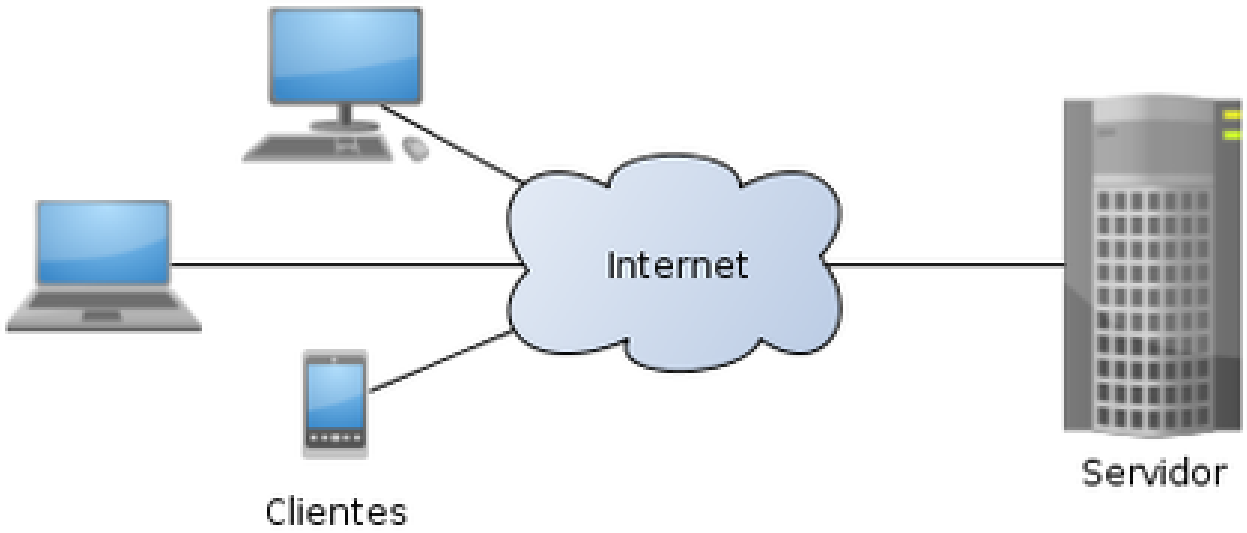

Figura: Arquitetura Cliente-Servidor.

Modelos Arquiteturais

Microserviços

Microservices

Principais Arquiteturas de Software

Microserviços - Definição

- A arquitetura de microsserviços é uma abordagem arquitetônica e organizacional do desenvolvimento de software na qual o software consiste em pequenos serviços independentes.
- É similar à arquitetura orientada a serviços (SOA), mas com algumas diferenças importantes, tais como, por exemplo, o tamanho dos serviços e a forma como eles se comunicam.
- Esses serviços são mantidos por pequenas equipes autossuficientes.
- Cada serviço é desenvolvido, implantado e gerenciado de forma independente.
- Cada serviço é responsável por uma única funcionalidade.
- Os serviços se comunicam entre si através de APIs bem definidas.

Vantagens

- **Separação de Responsabilidades**: Tudo é desenvolvido através de pequenas unidades de código e publicado em processos de deploy automatizados e independentes.
- **Tecnologias**: Essa abordagem permite que cada serviço seja desenvolvido usando a *stack* mais adequadas para o problema que está sendo resolvido.
- **Aplicabilidade**: em aplicações de grande porte e complexas que precisam ser escaladas rapidamente e com equipes distribuídas.

Desvantagens

- **Complexidade**: A complexidade de uma arquitetura de microsserviços é maior do que a de uma arquitetura monolítica, pois existem mais componentes para gerenciar.
- **Comunicação**: A comunicação entre os serviços pode resultar em algum overhead, especialmente em sistemas distribuídos, o que pode impactar o desempenho.
- **Testes**: Os testes de integração são mais complexos, pois envolvem a comunicação entre os serviços.
- **Gerenciamento**: O gerenciamento de uma arquitetura de microsserviços (governança) é mais complexo, pois existem mais componentes para gerenciar.

Principais Arquiteturas de Software

Microserviços - Representação

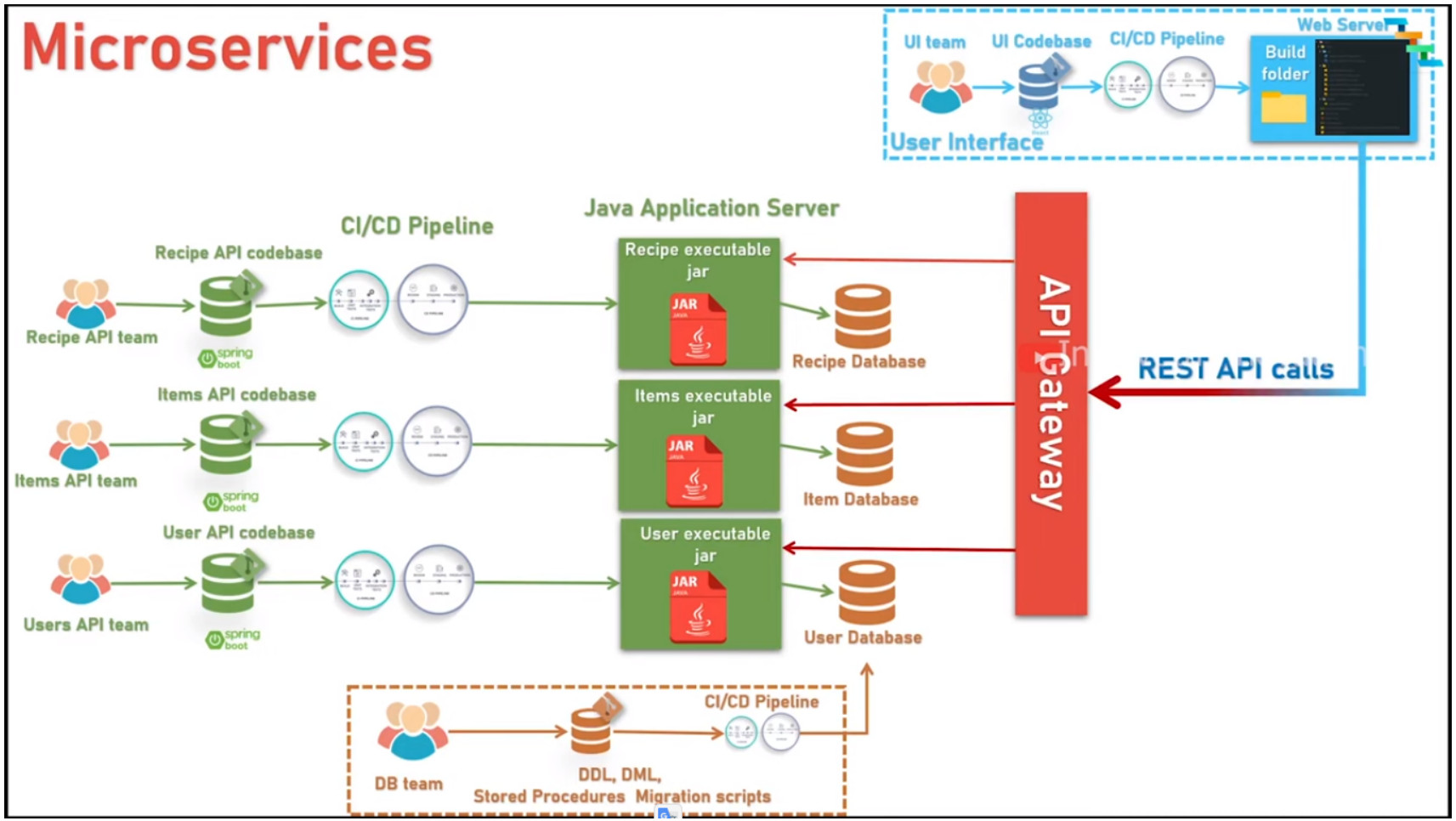

Figura: Arquitetura de Microsserviços.

Principais Arquiteturas de Software

Monolito vs. Microsserviços

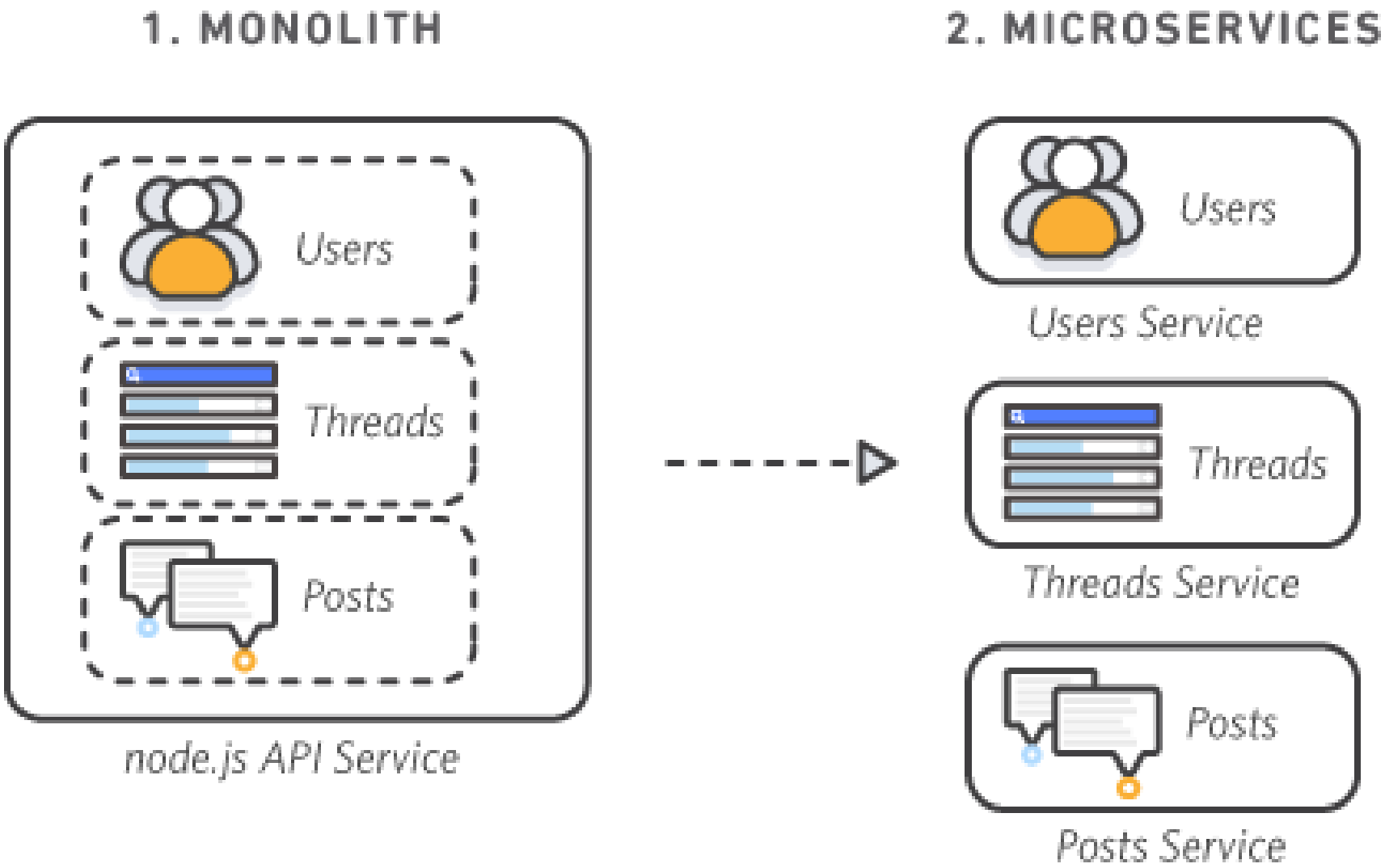

Figura: Monolito vs. Microsserviços.

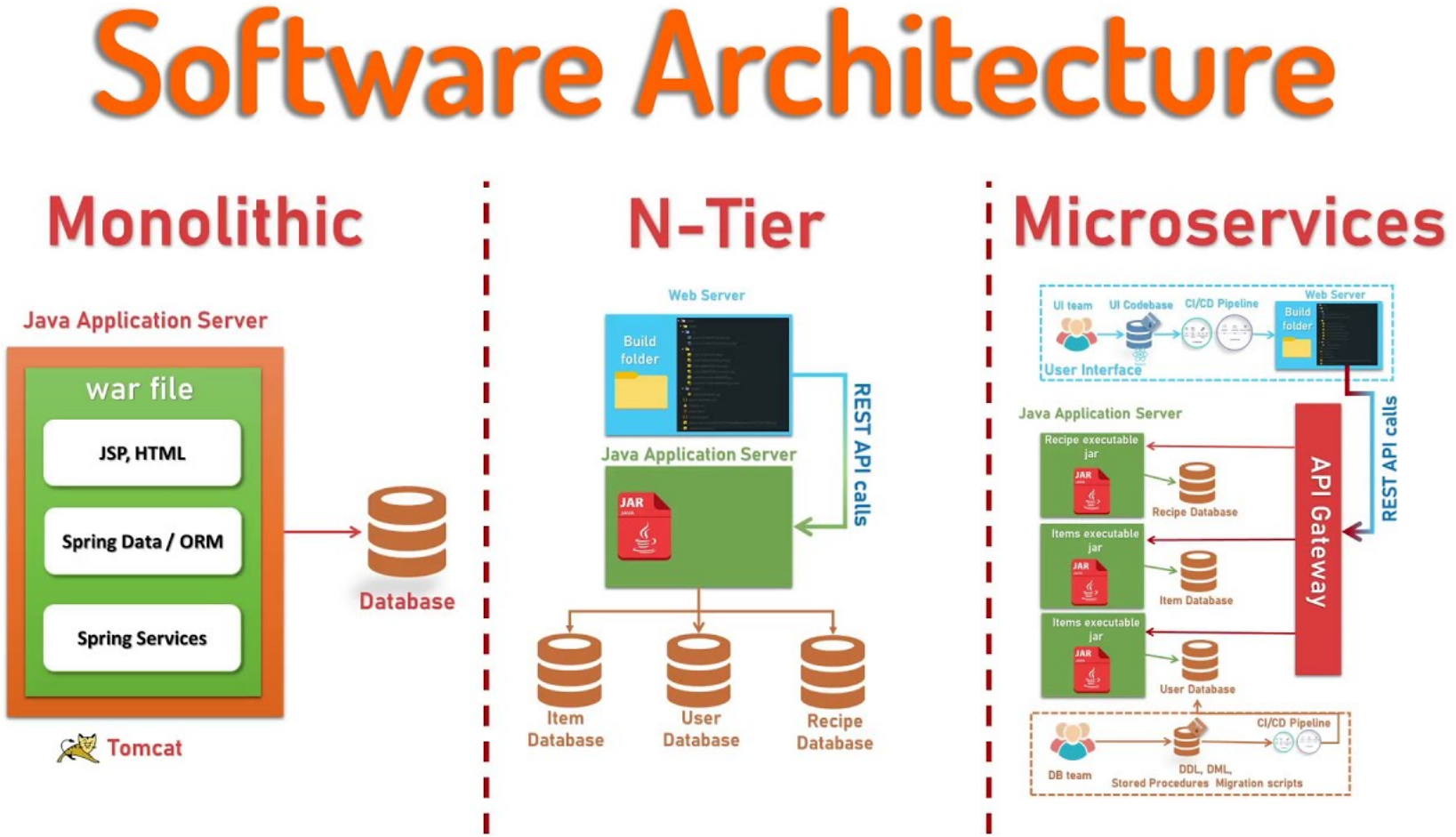

Figura: Monolito, N Camares e Microsserviços.

Material Complementar

Vídeos, Podcasts, Livros, etc

- **Aplicação Monolítica // Dicionário do Programador.**
  Canal **Código Fonte TV**.
- **Arquitetura de Software: Monolítica x SOA x Microserviços.**
  Canal **Marcos Dósea**.
- **Microservices // Dicionário do Programador.**
  Canal **Código Fonte TV**.
- **Microservices na prática.**
  Canal **Full Cycle**.
- **Microservices na prática.**
  Podcast Hipsters Ponto Tech **Monolitos**.