

Desenvolvimento Web II

Aula 01 - Arquitetura de Aplicações Web

Prof. Fabricio Bizotto

Instituto Federal Catarinense
fabricio.bizotto@ifc.edu.br

Ciência da Computação
11 de fevereiro de 2025

1 Introdução a Arquitetura de Software

- Conceitos
- Modelos Arquiteturais
 - Monolito
 - Arquitetura em N camadas (N-tier)
 - Microserviços

2 Material Complementar

- Quiz

3 Referências

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.

Características

- **Acoplamento forte:** todas as partes do sistema dependem umas das outras.
- **Deploy Único:** o sistema é implantado como uma única unidade.
- **Escalabilidade vertical::** para escalar, é necessário replicar o monólito inteiro.

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.
- **Menor complexidade operacional:** menos componentes para gerenciar.

Desvantagens

- **Manutenção:** À medida que o sistema cresce, o código pode se tornar complexo e difícil de entender.
- **Escalabilidade limitada:** Não é possível escalar partes específicas do sistema.
- **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.
- **Falhas:** Um erro em uma parte do sistema pode afetar todo o monólito.

Casos de Uso

- Aplicações pequenas ou de baixa complexidade.
- Projetos com equipes pequenas e prazos curtos.
- Sistemas que não exigem alta escalabilidade ou flexibilidade.

Principais Arquiteturas de Software

Monolito - Representação

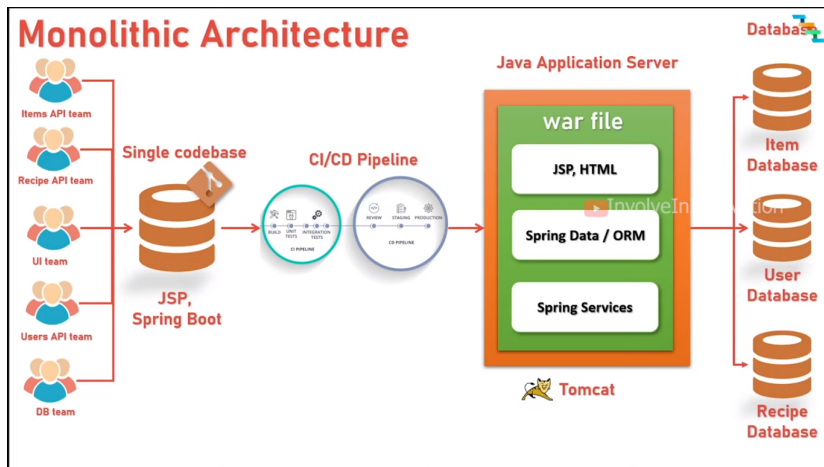


Figura: Arquitetura Monolítica.

Modelos Arquiteturais

N camadas

N-tier

Principais Arquiteturas de Software

Arquitetura em N camadas (N-tier) - Definição

A arquitetura em camadas organiza o sistema em camadas lógicas, onde cada camada tem uma responsabilidade específica. Cada camada é responsável por uma parte específica da aplicação.

- **Camada Fechada:** só pode se comunicar com as camadas adjacentes.
- **Camada Aberta:** posso pular ela (Opcional)
- Neste exemplo, como a camada está aberta, podemos ir da camada de negócio para a camada de persistência sem passar pela camada de serviço.

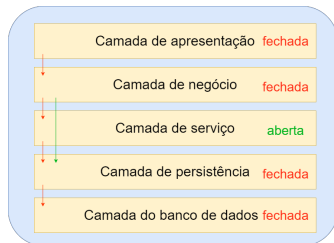


Figura: Arquitetura em camadas - Fluxo.

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

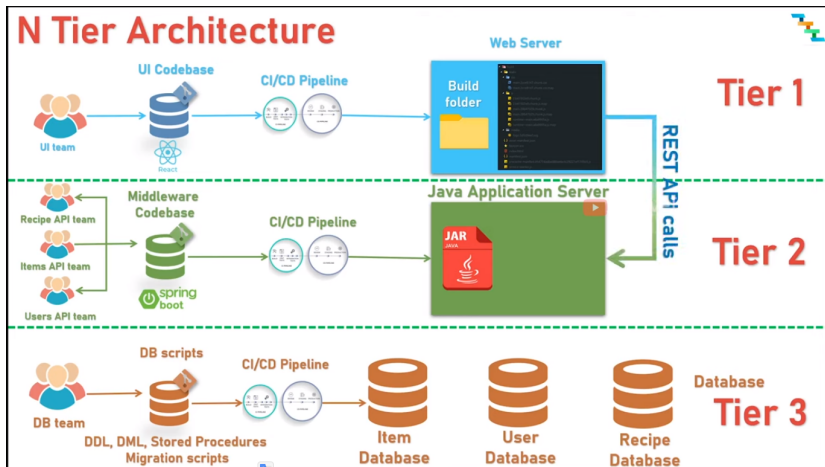


Figura: Arquitetura em N camadas (N-tier).

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.

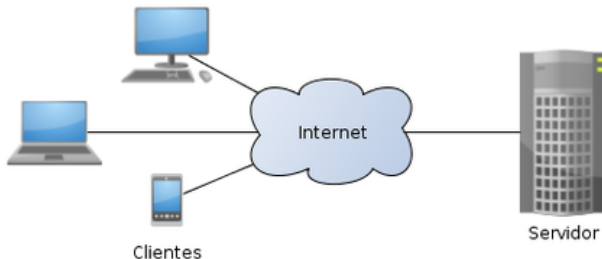


Figura: Arquitetura Cliente-Servidor.

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**.
- 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**, geralmente usando protocolos como HTTP ou mensageria.

Vantagens

- **Flexibilidade:** Facilita a adoção de novas tecnologias e a evolução do sistema.
- **Resiliência:** Falhas em um serviço não afetam necessariamente outros serviços.
- **Escalabilidade:** Permite escalar apenas os serviços que precisam de mais recursos.

Desvantagens

- **Complexidade:** Requer ferramentas para gerenciar deploy, monitoramento e comunicação entre serviços.
- **Latência:** A comunicação entre serviços pode introduzir atrasos.
- **Consistência:** Manter a consistência de dados entre serviços pode ser desafiador.
- **Gerenciamento:** O gerenciamento de uma arquitetura de microserviços (governança) é mais complexo, pois existem mais componentes para gerenciar.

Principais Arquiteturas de Software

Microserviços - Representação

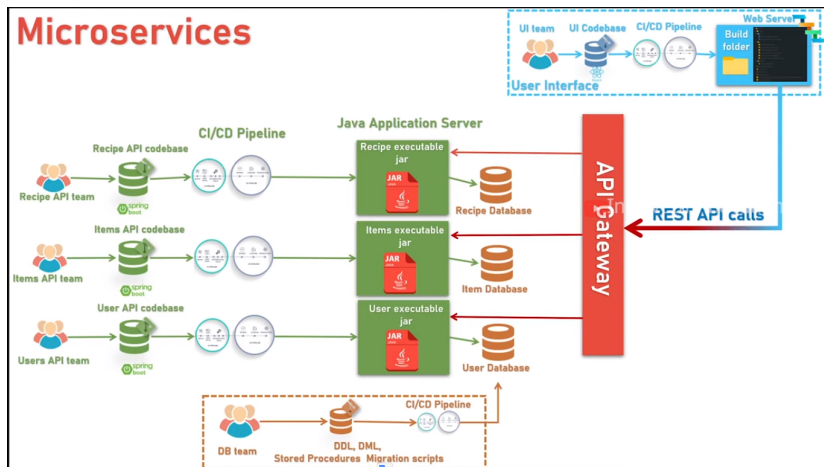


Figura: Arquitetura de Microserviços.

Casos de Uso

- Sistemas grandes e complexos com equipes distribuídas.
- Aplicações que exigem alta escalabilidade e disponibilidade.
- Projetos que precisam evoluir rapidamente e adotar novas tecnologias.

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**.
- **Monolitos.**
Podcast **Hipsters Ponto Tech**.

Recapitulando

QUIZ

Vamos praticar um pouco o que vimos até agora?

QUIZ - Desenvolvimento Web - Arquitetura de Software

Referências