PUC Minas

Laboratório de Aplicações Móveis e Distribuídas

Padrão de Projeto Model-View-Controller (MVC)

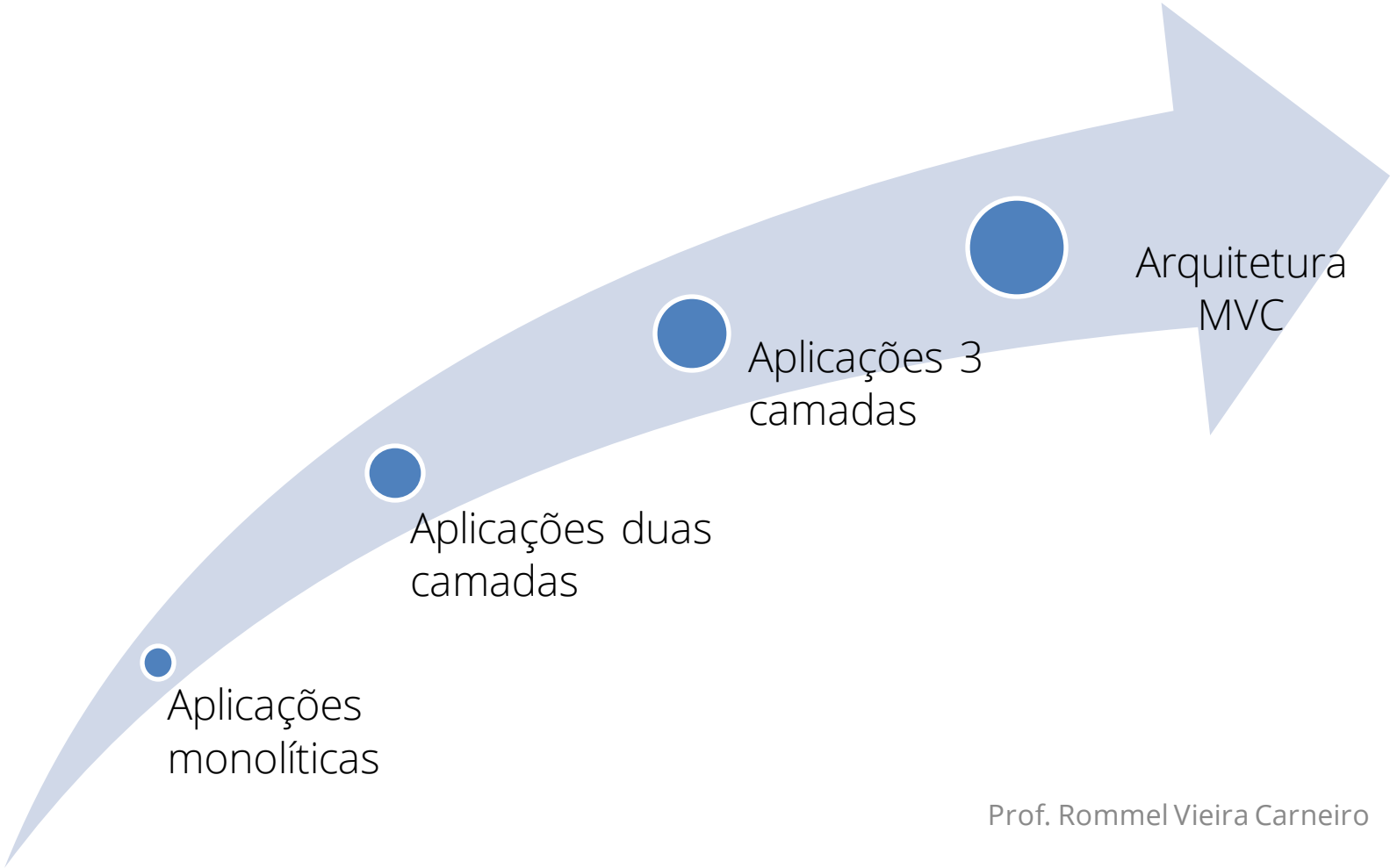
Prof. Rommel Vieira Carneiro

Tópicos

- Padrões de Projeto
  - Model-View-Controller (MVC) e variações
- Conceitos e Modelos de Programação
  - Responsive Web Design (RWD)
  - Single Page Applications (SPA)
  - Progressive Web Apps (PWA)
- Frameworks de Front End
  - Principais Frameworks
  - Comparação entre Frameworks

Model-View-Controller (MVC)

Evolução da arquitetura das aplicações



Model-View-Controller (MVC)

MVC é um padrão arquitetural muito aplicado em soluções Web que separa a aplicação em três camadas com tipos de componentes distintos:

Model | View | Controller

Vantagens

- Separação de responsabilidades
- Facilidade na manutenção da aplicação
- Baixo acoplamento e alta coesão entre componentes
- Reaproveitamento de código
- Facilita a divisão da equipe/tarefas
- Escalabilidade

Model-View-Controller (MVC)

MVC é um padrão arquitetural muito aplicado em soluções Web que separa a aplicação em três camadas com tipos de componentes distintos:

Model | **View** | **Controller**

Componentes da camada Model

- Modelam e representa os dados da aplicação
- Encapsulam as regras de negócios inerente aos dados
- Realizam o acesso ao banco de dados ou repositório

Model-View-Controller (MVC)

MVC é um padrão arquitetural muito aplicado em soluções Web que separa a aplicação em três camadas com tipos de componentes distintos:

Model | **View** | **Controller**

Componentes da Camada View

- Interage com o Controller e com o Model
- Apresenta os dados da aplicação na interface
- Utiliza padrões Web e mecanismo de templates

Model-View-Controller (MVC)

MVC é um padrão arquitetural muito aplicado em soluções Web que separa a aplicação em três camadas com tipos de componentes distintos:

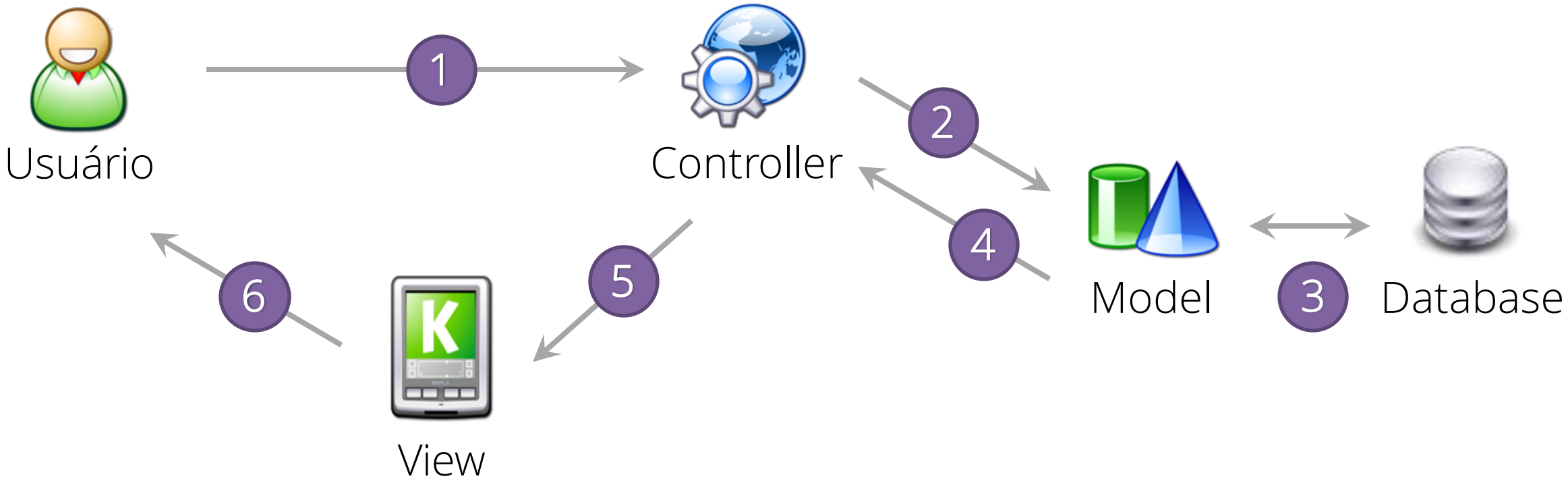
Model | **View** | **Controller**

Componentes da camada Controller

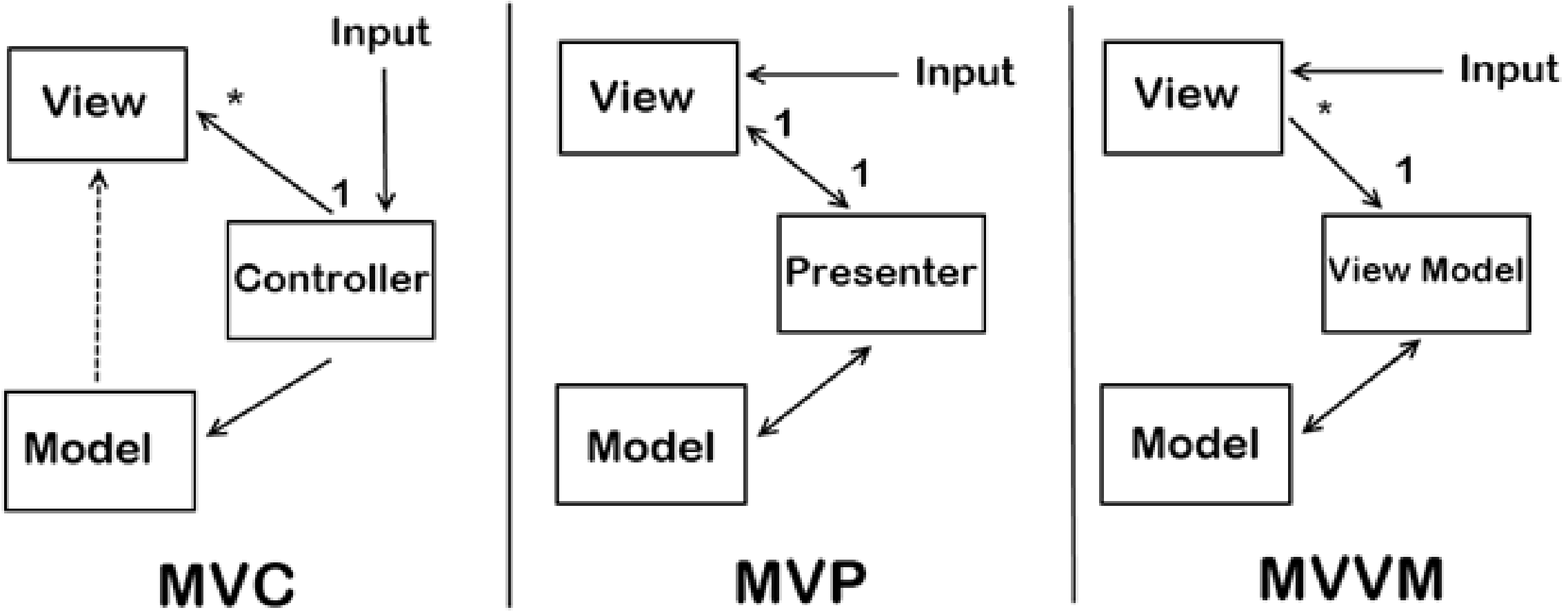
- Recebe solicitações do usuário da aplicação
- Controla o fluxo da aplicação disparando ações
- Realiza a comunicação entre usuário e componentes

Model-View-Controller (MVC)

Fluxo de Dados



Outras abordagens – MV*



Fonte: MVVM Compared To MVC and MVP - http://geekswithblogs.net/dlussier/archive/2009/11/21/136454.aspx

Outras abordagens – Model-View-Presenter (MVP)

Criada nos anos 90 pela Taligent (IBM), o padrão Model-View-Presenter (MVP) é uma derivação do MVC que tem foco na apresentação.

Características Principais

- Separação forte Apresentação e Gerenciamento de Dados
- Simplificação do testes unitários (View ⬅➡ Presenter)

Fonte: MVP - The Taligent Programming Model for C++ and Java - http://www.wildcrest.com/Potel/Portfolio/mvp.pdf

Outras abordagens – Model-View-ViewModel (MVVM)

O Model-View-ViewModel (MVVM) é derivado do MVP e foi criado por arquitetos de software da Microsoft.

Características Principais

- Conceitualmente idêntico ao MVP
- Foco na implementação de aplicação para a plataforma Windows Presentation Foundation (WPF) e Microsoft Silverlight

Outras abordagens – Model-View-ViewModel (MVVM)

ViewModel

- Atua como Model da View, mantendo o estado dos dados
- Não conhece a estrutura da View a qual está associada

View

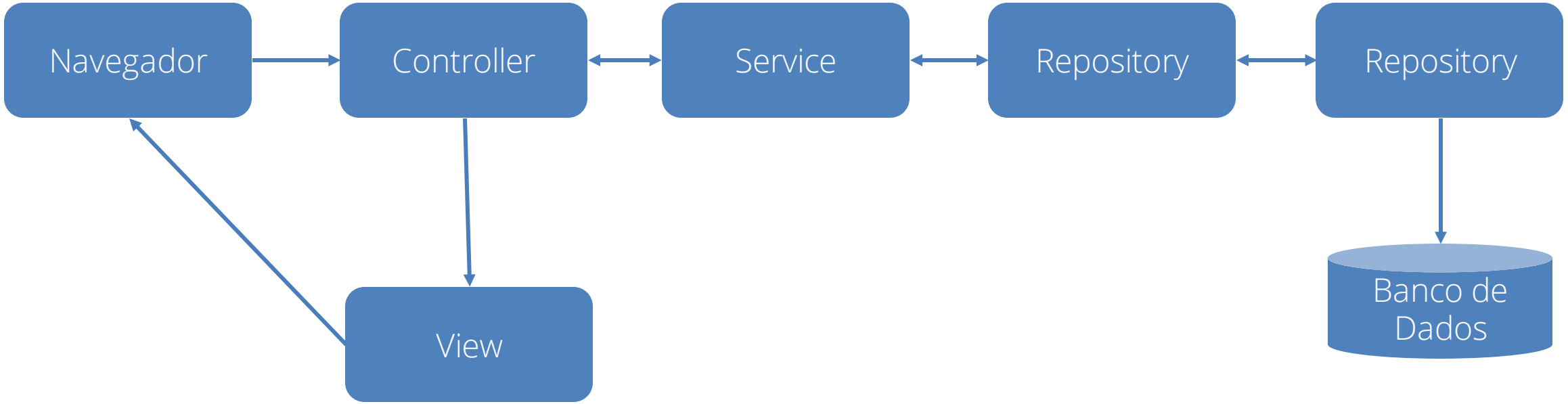
- Se liga ao ViewModel por uma propriedade DataContext

Model

- Encapsula a lógica de negócios e os dados
- Não faz referências diretas à View ou à ViewModel

Fonte: Entendendo o Pattern Model View ViewModel MVVM

Aplicação MVC com o Java Spring Boot

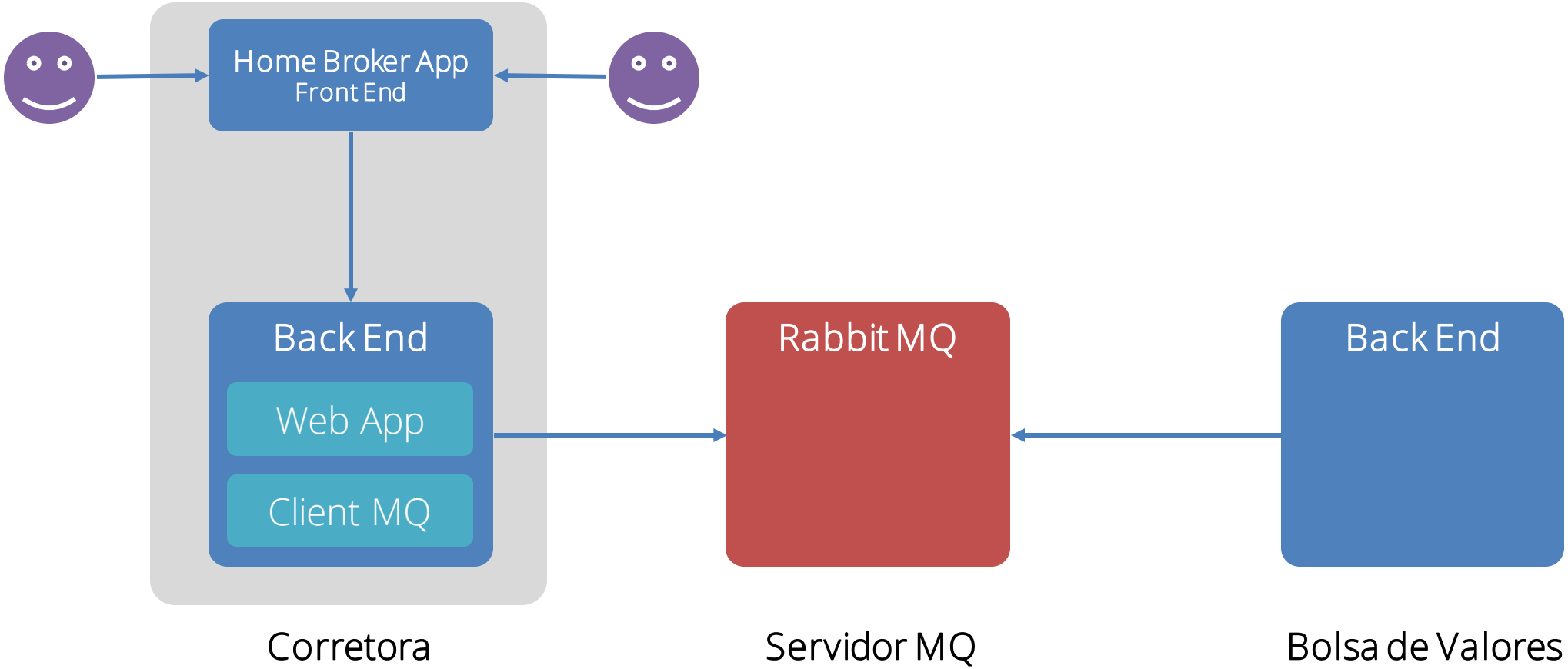


Aplicação MVC com o Java Spring Boot

Passo a passo

1. O controlador do Spring MVC recebe uma solicitação HTTP do cliente (navegador).
2. O controlador do Spring MVC processa a solicitação e envia essa solicitação para a camada de serviço.
3. A camada de serviço é responsável por conter a lógica de negócios da aplicação.
4. A camada de repositório é responsável por interagir com bancos de dados para salvar e recuperar dados da aplicação.
5. O controlador do Spring MVC retorna a visualização (JSP ou Thymeleaf) para ser renderizada no navegador.

Trabalho MVC – Home Broker



Trabalho MVC – Home Broker

O trabalho consite na construção de uma aplicação completa que oferece uma interface de homebroker e operacionaliza a negociação de ativos em uma bolsa de valores:

A aplicação consiste em três módulos principais:

1. **Corretora:**
    1. Uma aplicação web que proporciona uma interface para os usuários realizarem login.
    2. Após o login, os usuários podem visualizar os ativos disponíveis para negociação.
    3. Este módulo é responsável por permitir que os usuários enviem ordens de compra e venda para o servidor de mensageria (Servidor MQ).
2. **Servidor MQ:**
    1. Recebe todas as requisições provenientes do módulo Corretora.
    2. Interage com o módulo da Bolsa de Valores para comunicar os pedidos de transações.
    3. Responsável por processar e rotear as ordens recebidas para a Bolsa de Valores e retornar as transações realizadas para que sejam repassadas à Corretora.
3. **Bolsa de Valores:**
    1. Possui um livro de ofertas e armazena todas as transações realizadas.
    2. Interage com o módulo do Servidor MQ para receber pedidos de compra e venda.
    3. Processa os pedidos recebidos, executa as transações correspondentes e retorna os resultados ao Servidor MQ para serem repassados à Corretora.

Este sistema proporciona aos usuários a capacidade de visualizar e negociar ativos na bolsa de valores de forma eficiente, com a corretora atuando como intermediária entre os usuários e a bolsa de valores. O Servidor MQ desempenha um papel crucial na comunicação entre a Corretora e a Bolsa de Valores, garantindo uma troca eficaz de informações e a execução precisa das transações.

Prof. Rommel Vieira Carneiro

Trabalho MVC – Home Broker

- Curso App MVC (Lucas Ângelo)
- Repositório de roteiro da disciplina

Obrigado!

Prof. Rommel Vieira Carneiro