Laboratório de Sistemas Computacionais Complexos

2020/03/06 - AULA 02

https://uclab.xyz/sistemas-complexos-aula02

Renato Cordeiro Ferreira
renatocf@ime.usp.br

Thatiane de Oliveira Rosa
thatiane@ime.usp.br

João Francisco Daniel
joaofran@ime.usp.br

Alfredo Goldman
gold@ime.usp.br

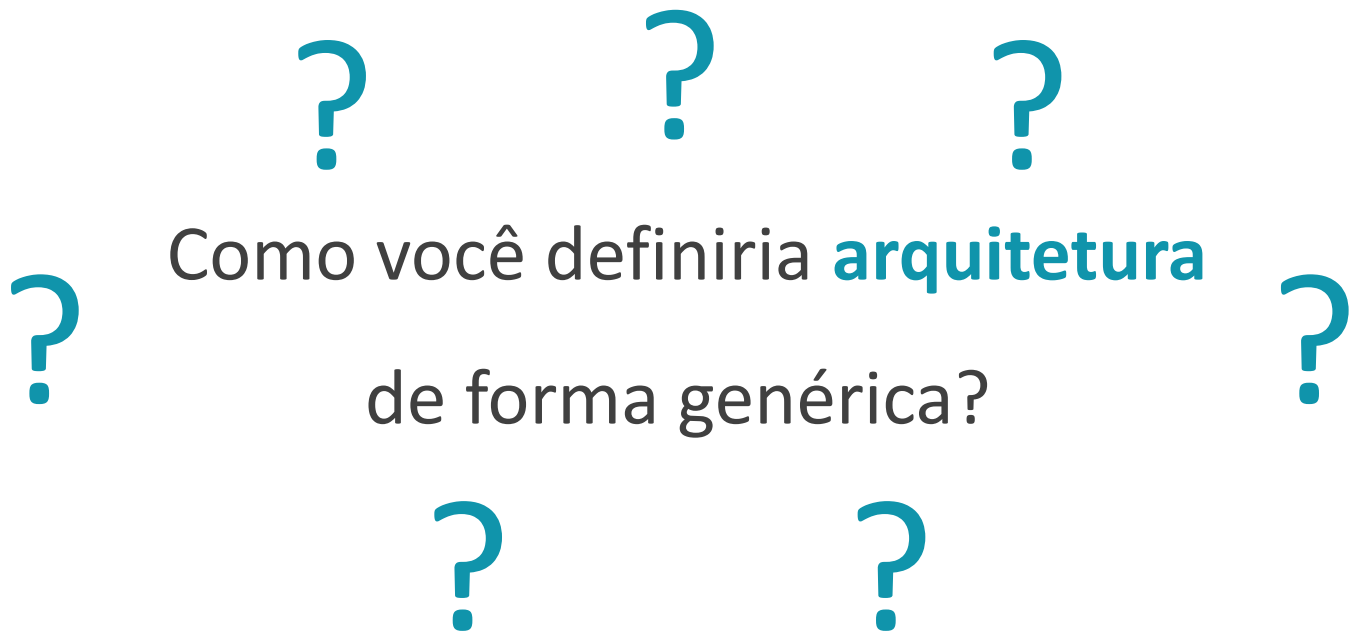
Agenda

Tema da aula:
Arquitetura de Software

1. Warm-up da disciplina
2. O que é arquitetura de software
3. Importância da arquitetura de software
4. Alguns estilos arquiteturais

Warm-up

O que é arquitetura de software?



O que é arquitetura de software?

Organização **hierárquica** dos elementos de um software, que ilustra a sua **estrutura** e forma de **interação**, e especifica **fundamentos**, **propriedades**, **regras** e **restrições** que guiam o projeto e evolução do software.

PRESSMAN, 2002; GARLAND; ANTHONY, 2003; ISO/IEC/IEEE, 2011; GARLAN, 2014

O que é arquitetura de software?

Toda arquitetura é um projeto, mas nem todo projeto é uma arquitetura.

Arquitetura representa as **decisões significativas** de projeto que **moldam** um **sistema**, onde o **significativo é medido** pelo **custo da mudança**.

BOOCH, 2006

O que é arquitetura de software?

Tem o objetivo **facilitar** o **desenvolvimento**, a **implantação**, a **operação** e a **manutenção** do software.

MARTIN, 2007

Importância da arquitetura de software

Se bem definida:

- Facilita o **gerenciamento** da **complexidade**
- Auxilia a **evitar problemas** durante o processo de desenvolvimento
- Contribui na **compreensão**, **reutilização**, **desenvolvimento**, **análise**, **evolução** e **manutenção** do software

HOFMEISTER, et al. 2000; Paul et al., 2002; GARLAND; ANTHONY, 2003; GARLAN, 2014; CERVANTES E KAZMAN, 2016

Importância da arquitetura de software

Fator crítico de sucesso para o projeto.

Fundamental para **relacionar** as **características** do software com a sua **implementação**.

Ajudar a **satisfazer requisitos não funcionais** e de **qualidade**, como:

- Desempenho
- Confiabilidade
- Portabilidade
- Escalabilidade e
- Interoperabilidade

Importância da arquitetura de software

É comum desenvolver software sem uma arquitetura formal ou com uma arquitetura confusa e mal definida

- Resultado:
  - Módulos e código-fonte desorganizados, sem papéis e responsabilidades definidas e com relacionamentos confusos entre si
  - Conhecido com ***"Big ball of mud"***

MARTIN, 2017

Importância da arquitetura de software

Sem uma arquitetura formal **é difícil**:

- Determinar **características arquiteturais** da aplicação
- **Responder perguntas básicas** sobre implantação e manutenção:
  - A arquitetura é dimensionada?
  - Quais são as características de desempenho do software?
  - Com que facilidade o software responderá à mudança?
  - Quais são as características de implantação do software?
  - Quão responsiva é a arquitetura?

MARTIN, 2017

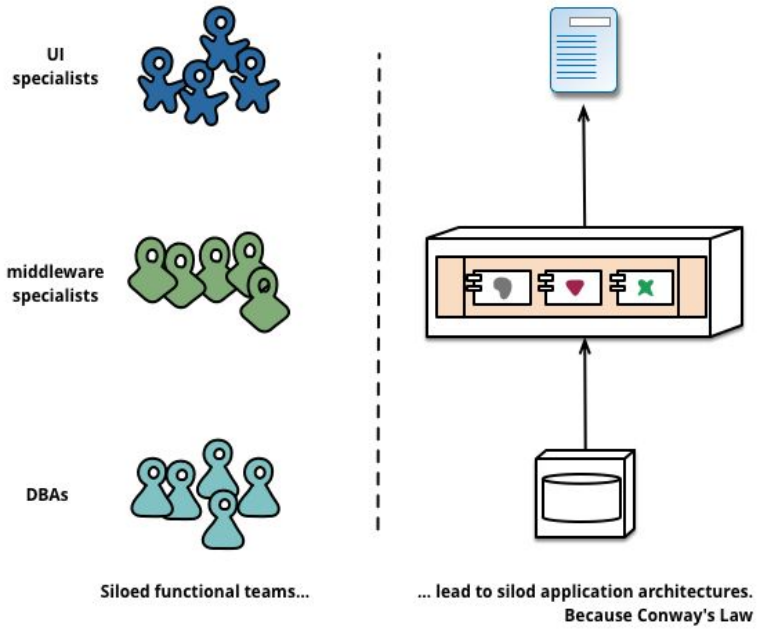
Alguns estilos arquiteturais

1. Arquitetura em camadas
   Layered architecture
2. Arquitetura dirigida a eventos
   Event-driven architecture
3. Arquitetura orientada a serviços
   Service-oriented architecture
4. Arquitetura de microsserviços
   Microservices architecture

Lei de Conway

Qualquer organização que projete um sistema produzirá um **design cuja estrutura** é uma **cópia da estrutura de comunicação da organização**

Melvyn Conway, 1967



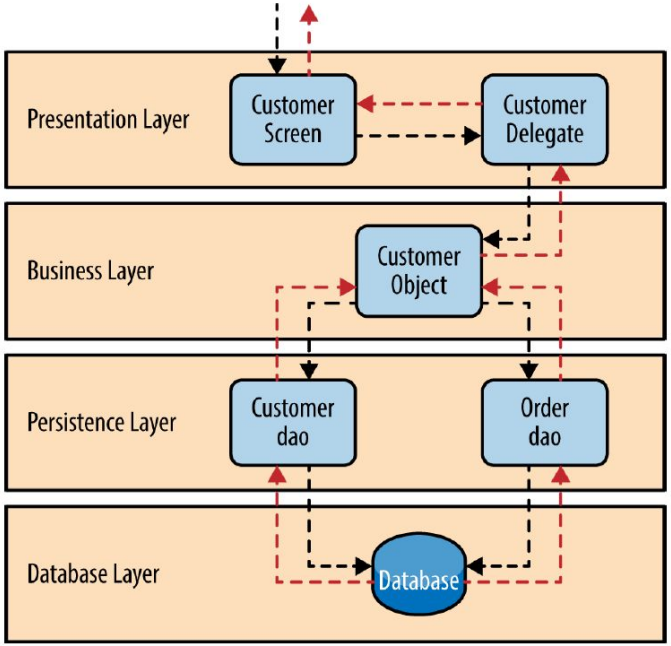
Arquitetura em camadas

Padrão **mais comum e difundido**.

Cada camada desempenha uma **função específica**.

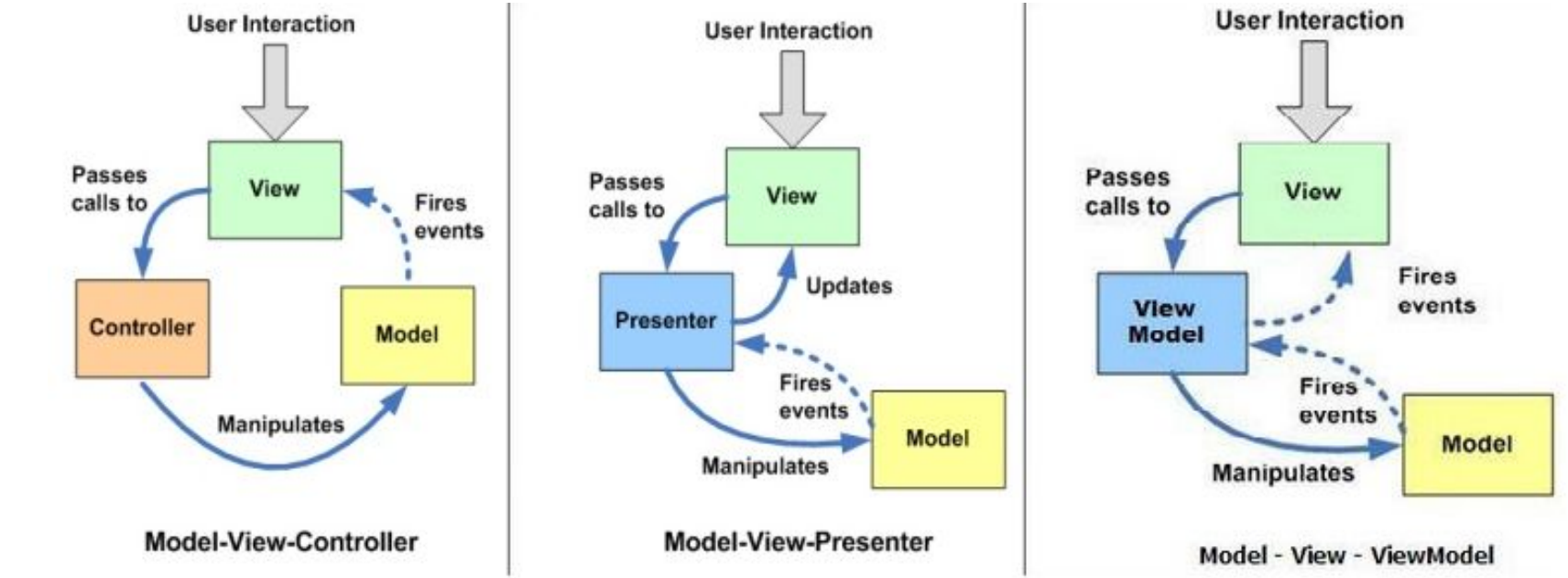
Algumas camadas comuns:

- Apresentação
- Lógica de negócio
- Persistência
- Banco de dados

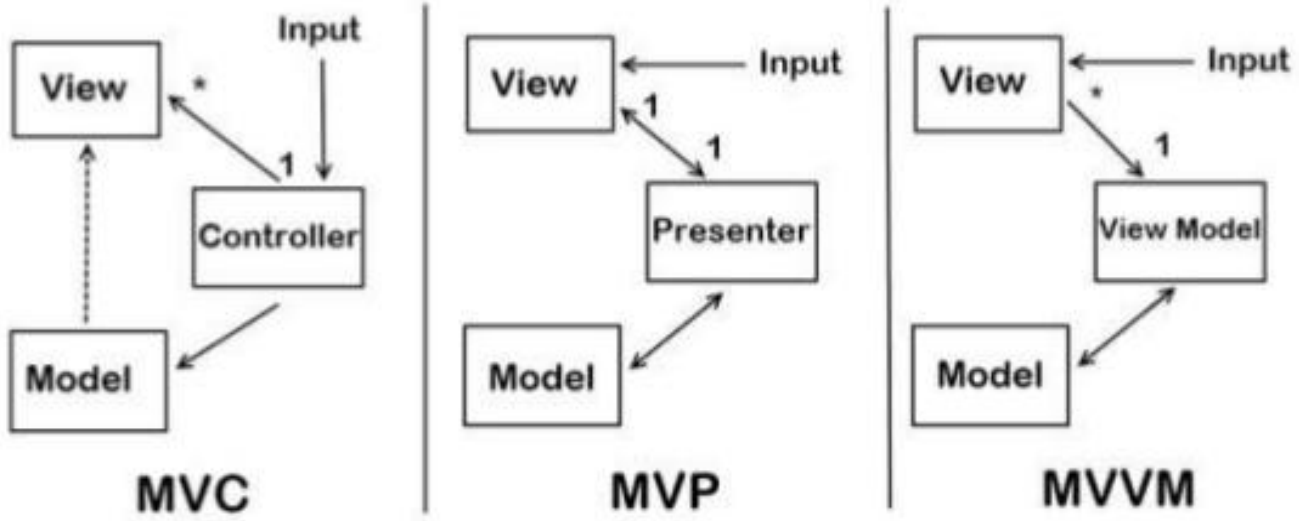


MARTIN, 2017

Arquitetura em camadas



Arquitetura em camadas

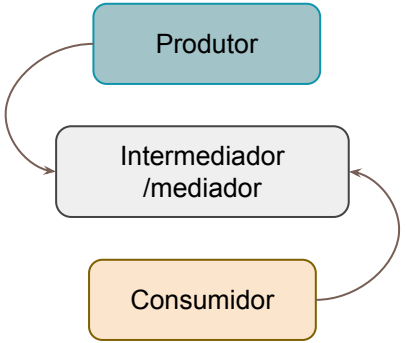


Arquitetura dirigida a eventos

Padrão popular de **arquitetura assíncrona e distribuída**.

Voltado para desenvolvimento de software **escalonável**, **adaptável**, **grandes** e **complexos**.

Seus componentes têm **propósito único**, são **altamente desacoplados** e recebem e processam **eventos** de **maneira assíncrona**.

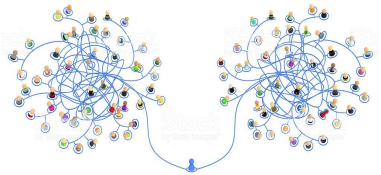


Arquitetura dirigida a eventos

O que é um evento?

- **Depende do projeto**, pode ser o cadastro de um novo usuário, a inscrição em uma conferência, a compra de um produto, o desbloqueio de uma nova fase em um jogo...
- **Ação significante para a empresa**
- Seu mapeamento e definição estão **fortemente relacionados ao negócio**
- Fato ocorrido no **passado**

Arquitetura dirigida a eventos



- **Orquestrador**:
  - Adotada quando é necessário centralizar a orquestração de várias etapas de um evento (execução em paralelo ou em sequência)
- **Broker**:
  - Adotada quando pode-se encadear os eventos sem a necessidade de um orquestrador

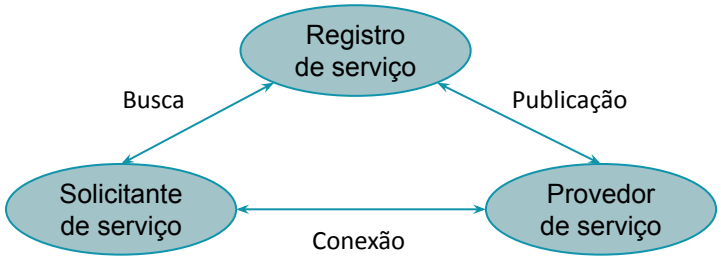
Arquitetura orientada a serviços

SOA (*Service Oriented Architecture*) – Arquitetura Orientada a Serviços:

- Propõe que **vários sistemas colaborem** para fornecer um conjunto final de recursos, ou seja, as funcionalidades do software são disponibilizadas em forma de **serviços**
- Surgiu com o objetivo de:
  - Promover a **reutilização** do software
  - Facilitar a **integração** de novas funcionalidades em sistemas legados

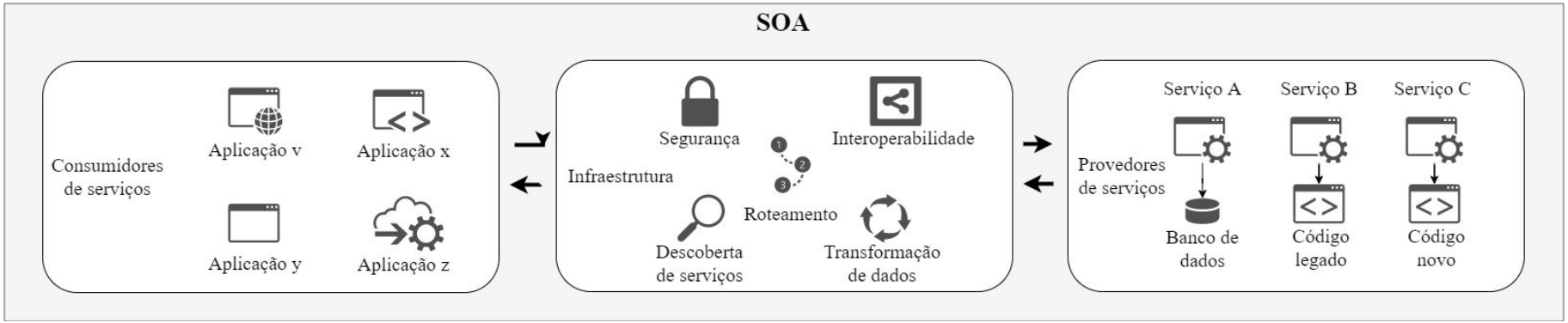
Arquitetura orientada a serviços

SOA é uma abordagem **cliente/servidor** em que uma aplicação é composta por usuários (clientes) e provedores (servidores) de serviços de software, que enfatiza o **acoplamento flexível entre os componentes** e o **uso de interfaces independentes**



NATIS; SCHULTE, 2003; BIANCO et al., 2007

Arquitetura orientada a serviços



Enterprise Service Bus (ESB)

"Smart pipes, dumb endpoints"

Arquitetura orientada a serviços

Pontos fortes
Reutilização de serviços
Agilidade nos negócios
Processos de negócio otimizados
Maior escalabilidade
Interoperabilidade
Integração com sistemas legados

Pontos fracos
Granularidade do serviço
Conexão e integração de serviços
Processo de desenvolvimento complexo
Desempenho insatisfatório
Teste complexos
Baixo nível de confiabilidade
Complexidade de interoperabilidade
Menor segurança

Arquitetura de microsserviços

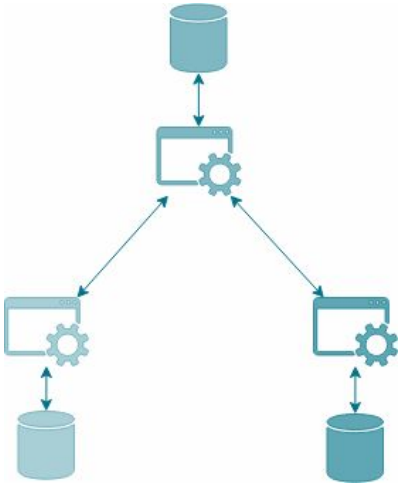
Arquitetura de microsserviços:

Microsserviços são serviços **pequenos**, **autônomos** e que **trabalham em conjunto**

NEWMAN, 2015

[…] propõe o desenvolvimento de softwares a partir de um conjunto de **serviços pequenos** e **isolados**, onde cada um possui seus dados, é **independentemente isolado**, **escalável** e **resiliente** a **falhas**. Os serviços se integram uns aos outros para formar um **sistema coeso e flexível**

BÓNER, 2016



Licença

Estes slides são concedidos sob uma Licença Creative Commons. Sob as seguintes condições:

Atribuição, Uso Não-Comercial e Compartilhamento pela mesma Licença.

Mais detalhes sobre essa licença em: creativecommons.org/licenses/by-nc-sa/3.0/

Créditos

Imagens usadas nesta apresentação são provenientes de: freepik.com

Laboratório de Sistemas Computacionais Complexos

2020/03/06 - AULA 02

https://uclab.xyz/sistemas-complexos-aula02

Renato Cordeiro Ferreira
renatocf@ime.usp.br

Thatiane de Oliveira Rosa
thatiane@ime.usp.br

João Francisco Daniel
joaofran@ime.usp.br

Alfredo Goldman
gold@ime.usp.br