

# ARQUITETURA DE SOFTWARE

João Choma Neto

[joao.choma@unicesumar.edu.br](mailto:joao.choma@unicesumar.edu.br)

<https://github.com/JoaoChoma/arquitetura-software>

Unicesumar – Maringá



# O que veremos nesse semestre?

- Conceitos de Arquitetura de software
- Decisões sobre projeto de arquitetura de software
- Visões de arquitetura
- Padrões

# O que veremos nesse semestre?

- Arquitetura de aplicações
- Sistemas de processamento de transações
- Sistemas de processamento de linguagens

# O que veremos nesse semestre?

- Arquitetura de aplicações
- Sistemas de processamento de transações
- Sistemas de processamento de linguagens

# Avaliação

- *1º Bimestre*

*1,0 - Atividade de Estudo Programada.*

*1,0 - Prova Integrada.*

*8,0 - Avaliação Prática e Teórica*

# Avaliação

- **2º Bimestre**

*1,0 - Atividade de Estudo Programada.*

*1,0 - Prova Integrada.*

*8,0 - Avaliação Prática e Teórica*

# Referências utilizadas

## • Bibliografia básica

- *PRESSMAN, Roger. Engenharia de software - 8 / 2016 Porto Alegre AMGH 2016*
- *BOOCH, Grady; RUMBAUGH, James; JACOBSON, Ivar; SILVA, Fábio Freitas da; MACHADO, Cristina de Amorim. UML: guia do usuário - 2. ed. rev. e atu / 2012 Rio de Janeiro: Elsevier, 2012.*
- *PFLEEGER, Shari Lawrence. Engenharia de software : teoria e prática [recurso eletrônico] - 2. ed. / 2004 São Paulo: Pearson, 2004.*

Ver com a galera o que eles já estudaram



SENTA  
QUE LÁ  
VEM  
HISTÓRIA



**AH NÃO**

# Linguagens mais utilizadas atualmente

Classificação	Linguagem de programação	Quota de mercado	Tendência
1	Python	27.61%	-2.8%
2	Java	17.64%	-0.7 %
3	JavaScript	9.21%	+0.4 %
4	C#	7.79%	+0.8 %
5	C/C++	7.01%	+0.4 %



## History of Popular Programming Languages: A timeline



# Arquitetura de um software

- O projeto de arquitetura de software abrange
  - concepção inicial do sistema até sua implementação
  - Manutenção
  - Evolução e atualizações ao longo do tempo

# Um projeto

- Fornece uma estrutura organizacional para o sistema
- Definições de:
  - Definindo os componentes principais
  - Responsabilidades dos componentes
  - Interação entre componentes

# Uma das razões do projeto

- **COMUNICAÇÃO**

- A definição do projeto permite gerenciar a complexidade do sistema
- Permite que desenvolvedores e projetistas compreendam e trabalhem em partes específicas do sistema de forma isolada
- Permite o desenvolvimento serializado e paralelo, ao mesmo tempo em que mantêm uma visão do conjunto coeso.

# Como assim comunicação?

- Artefato de comunicação entre todos os stakeholders envolvidos no projeto:
  - Desenvolvedores
  - Gerentes de projeto
  - Analistas de negócios
  - Scrum master
  - PO
  - Clientes

## Como assim comunicação?

- Uma arquitetura bem definida facilita o entendimento comum dos objetivos do sistema, suas capacidades, limitações
- Ajuda a alinhar as expectativas
- Ajuda a reduzir mal-entendidos



## ESCOLHAS

---

- Antes de codificar e implementar detalhes específicos, o projeto de arquitetura permite que as equipes avaliem diferentes abordagens e tomem decisões informadas sobre as melhores estratégias



# ESCOLHAS

- Tecnologias
- padrões de projeto
- APIs
- Requisitos conflitantes
- Desempenho
- Segurança
- Escalabilidade
- Custo.

# RESULTADOS

- A arquitetura influencia diretamente atributos de qualidade do software
- Desempenho, confiabilidade, usabilidade, e segurança

# RESULTADOS

- Um projeto de arquitetura garante que esses atributos sejam considerados e otimizados desde o início
- A IDEIA é que o produto final consiga atender as expectativas dos usuários e stakeholders

TA, MAS E DAI?



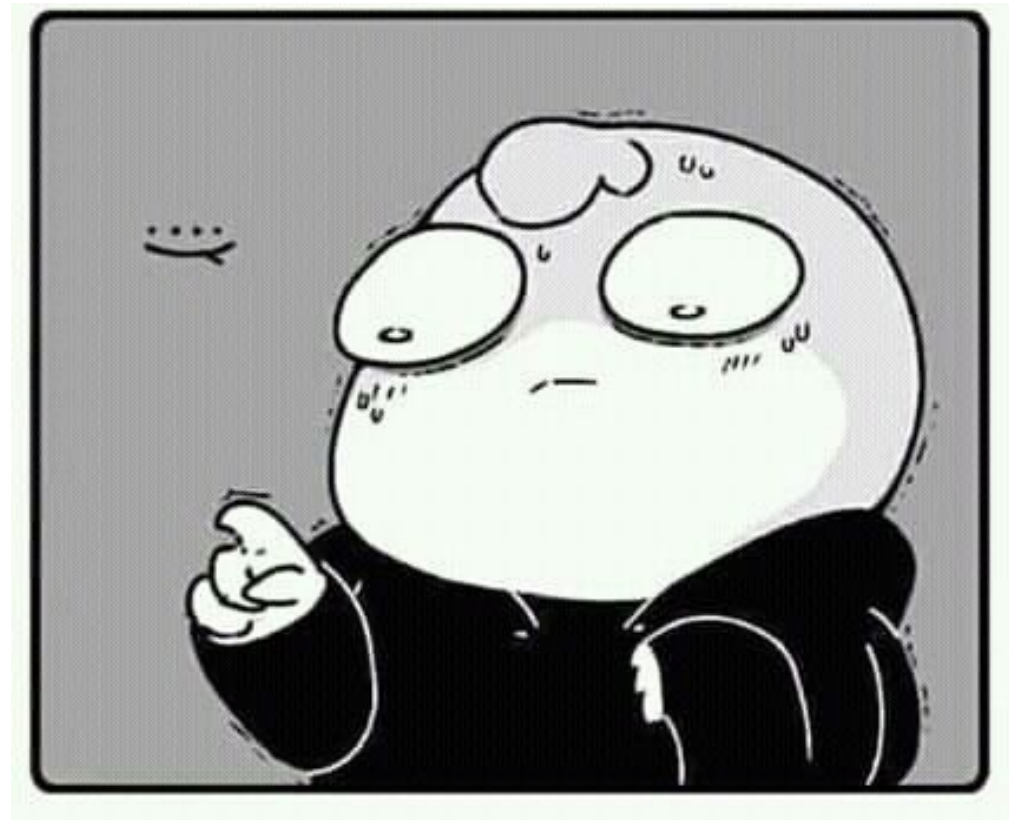
UM  
ESTUDO  
DE CASO



**AH NÃO**

- Você está desenvolvendo um sistema de e-commerce para uma loja que vende produtos diversos online.
- O sistema precisa gerenciar produtos, pedidos, pagamentos e notificações para os usuários.

# MOMENTO DE ESCOLHA





- **Aplicando o Padrão MVC**

- **Model:**

- Classes que representam entidades como Produto, Pedido, e Pagamento
- Estas classes contêm a lógica para acessar os dados (por exemplo, banco de dados) e manipular as informações dos produtos, pedidos e pagamentos

- **Aplicando o Padrão MVC**

- **View:**

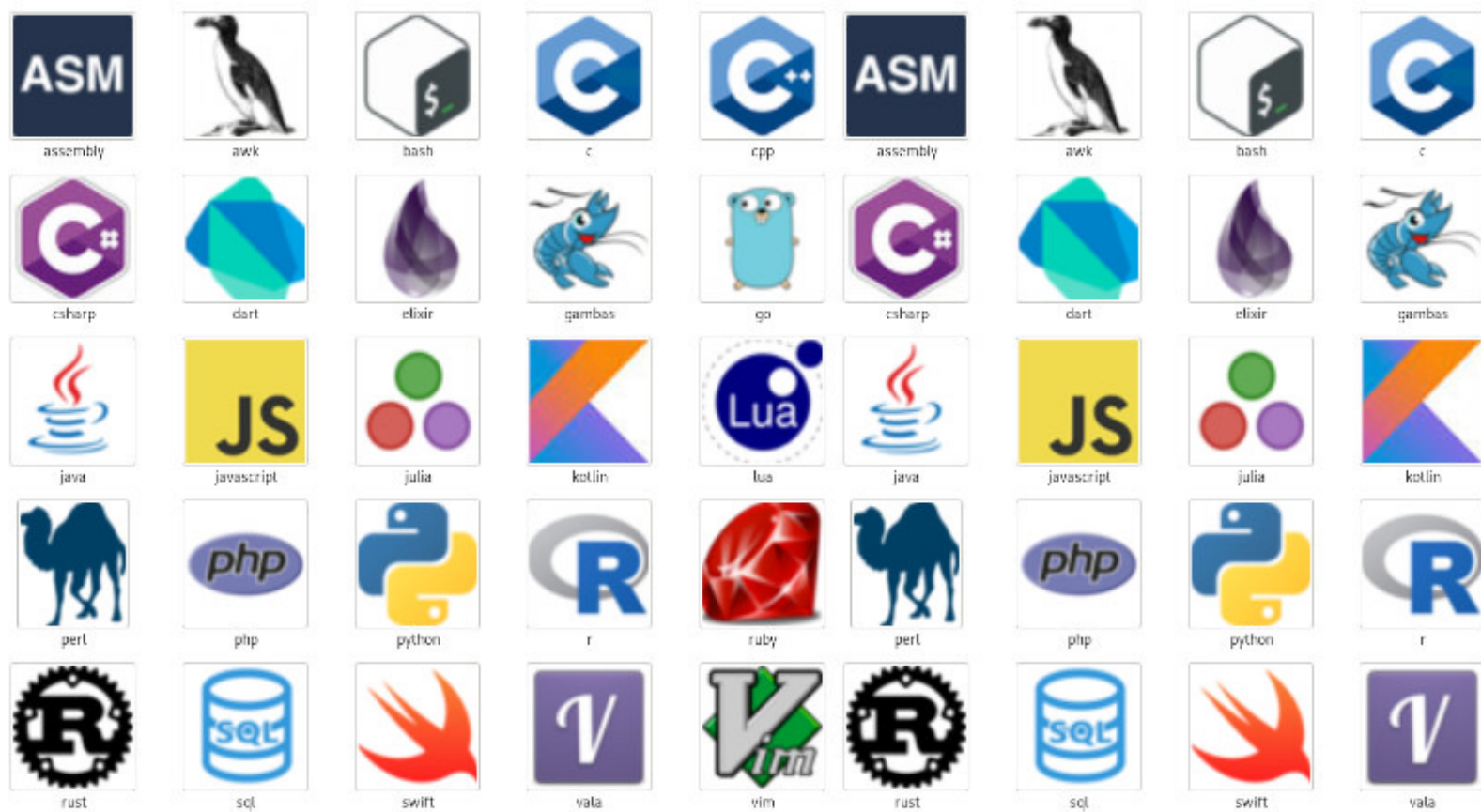
- Interfaces de usuário que exibem a informação ao cliente
- Criação de páginas web para cada classe de modelo
- Criação de páginas web para listar produtos, um carrinho de compras, e formulários para entrada de pagamento

- **Aplicando o Padrão MVC**

- **Controller:**

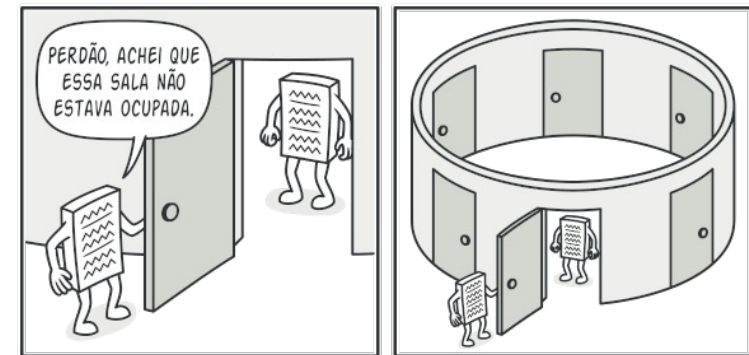
- Componentes que processam as ações do usuário
  - Adicionar um produto ao carrinho
  - Realizar um pedido
- Interação com o Model para atualizar os dados
- Interação com a View apropriada para resposta ao usuário

# Qual tecnologia posso usar?



Para banco de dados tem algo?

- Singleton é um padrão de projeto de software. Este padrão garante a existência de apenas uma instância de uma classe, mantendo um ponto global de acesso ao seu objeto.



E para meios de pagamento?

- **Padrão Factory Method para Criação de Pagamentos**
- Considerando a necessidade de processar diferentes tipos de pagamentos (cartão de crédito, PayPal, boleto), você utiliza o padrão Factory Method
- Isso permite que o sistema crie objetos de Pagamento específicos para cada tipo de pagamento, sem acoplar o código aos classes concretas de pagamento

- Vai, fala como .....
- Como vou fazer esse projeto arquitetural?



# O que é uma decisão de projeto?

- Decisões de projeto referem-se às escolhas feitas durante o desenvolvimento de software
- Decisões afetam:
  - Estrutura do código
  - Comportamento do sistemas
  - Funcionalidades do sistema

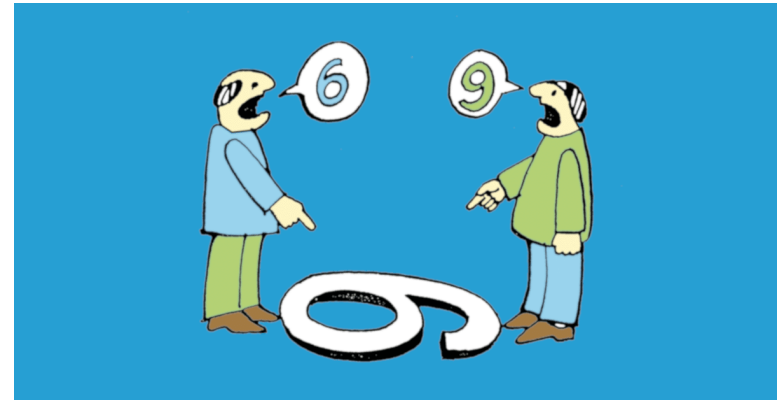


# O que é uma decisão de projeto?

- Estas decisões abrangem:
  - Seleção de tecnologias e frameworks
  - Definição de estruturas de dados
  - Algoritmos
  - Padrões de projeto
  - Componentes

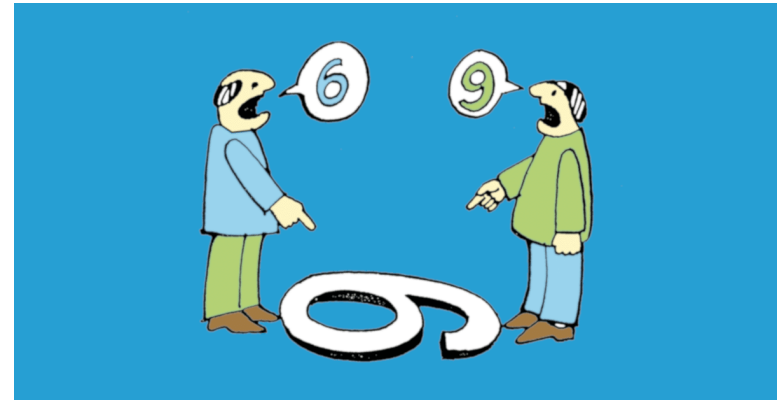
Como eu vejo essa arquitetura?

- Visões de arquitetura são representações ou perspectivas específicas do sistema que destacam certos aspectos ou componentes da arquitetura



Como eu vejo essa arquitetura?

- A visão busca facilitar o entendimento e a comunicação da estrutura e do funcionamento do sistema entre todos os stakeholders



# Visões

- **Visão Lógica:** Mostra a organização funcional do sistema, como os principais componentes e serviços são organizados e interagem para realizar a funcionalidade do sistema
- **Visão de Desenvolvimento:** Foca na estrutura do software no ambiente de desenvolvimento, incluindo módulos, pacotes e camadas de software

# Visões

- **Visão de Processo:** Descreve os processos ou threads que executam no sistema e suas interações
- **Visão Física:** Também conhecida como visão de implantação, mostra como o software é mapeado em hardware ou em outros sistemas

# Padrões

- Padrões de arquitetura são soluções reutilizáveis para problemas comuns de design de software
- Padrões de arquitetura fornecem um modelo ou template que pode ser adaptado para resolver um problema de design em vários contextos

## Exemplo de padrões

- **Padrão MVC (Model-View-Controller):** Separa a lógica de negócios, a interface do usuário e a entrada do usuário em três componentes distintos, facilitando a manutenção e a escalabilidade.

## Exemplo de padrões

- **Padrões de Criação:** Relacionados à criação de objetos ou componentes do sistema, como Singleton e Factory.
- **Padrões Estruturais:** Lidam com a composição ou estrutura de classes e objetos, como Adapter e Composite.



## Exemplo de padrões

- **Padrões Comportamentais:** Focam em como os objetos e classes interagem e distribuem responsabilidades, como Observer e Strategy.
- **Padrões GRASP:** Princípios gerais de design orientado a objetos que guiam responsabilidades de classes.

# ARQUITETURA DE SOFTWARE

João Choma Neto

[joao.choma@unicesumar.edu.br](mailto:joao.choma@unicesumar.edu.br)

<https://github.com/JoaoChoma/arquitetura-software>

Unicesumar – Maringá

