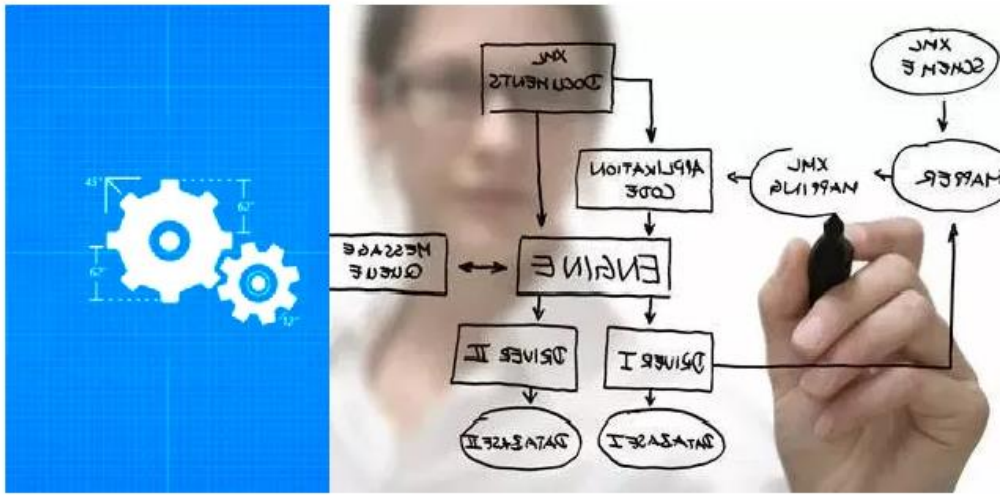
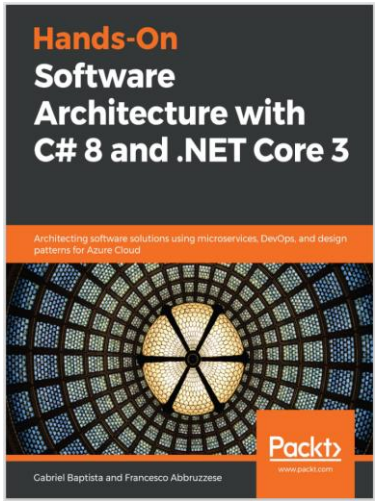
Arquitetura de Software

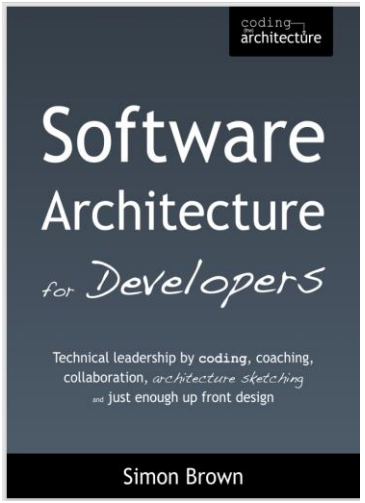
Unidade 1 - Introdução

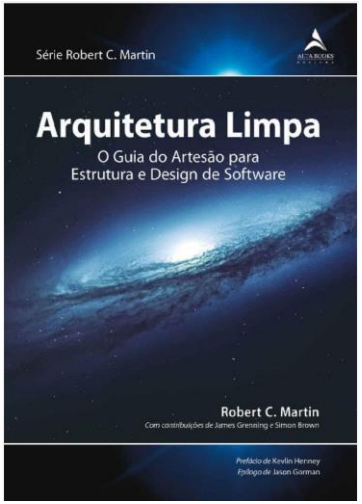


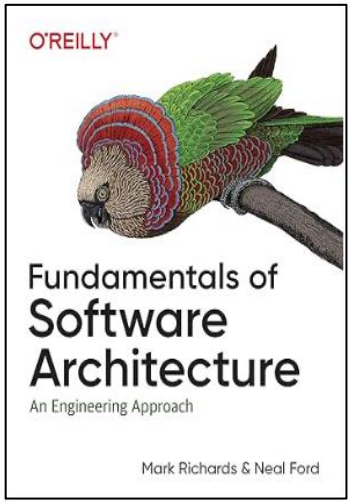
Prof. Aparecido V. de Freitas
Doutor em Engenharia
da Computação pela EPUSP
aparecido.freitas@online.uscs.edu.br
aparecidovfreitas@gmail.com

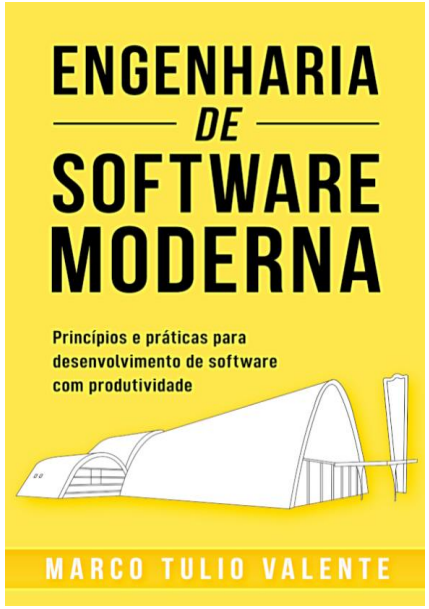
Bibliografia

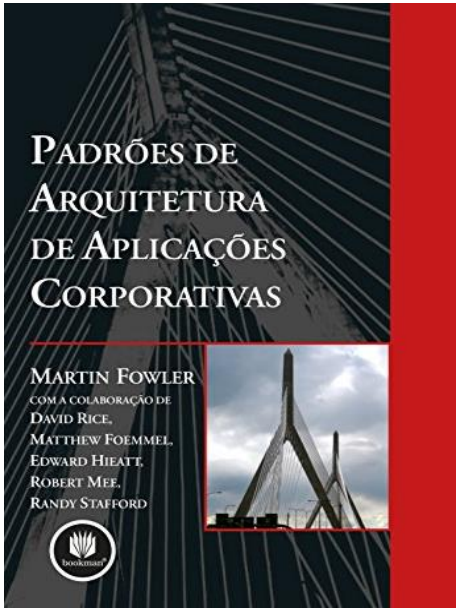


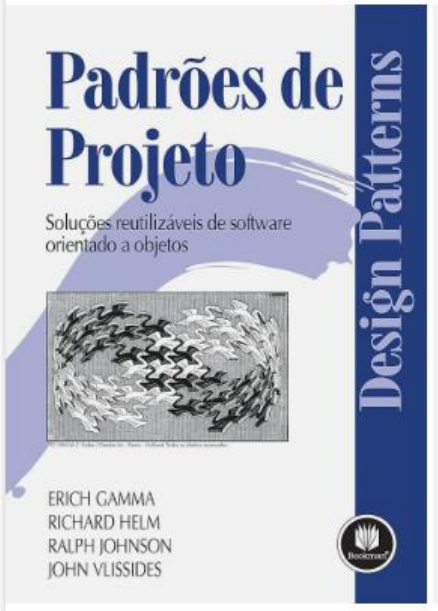












Introdução

- A **Arquitetura** de **Software** é composta por alguns **princípios**;
- Embora haja ferramentas que auxiliem na concepção da arquitetura, sem o emprego desses **princípios** não se desenhará com **qualidade** a arquitetura do software;

O que se pode pensar dessa imagem?

O que se pode pensar dessa imagem?

- Quarto desorganizado!
- Quarto desarrumado!
- Muitos objetos espalhados pelo quarto!

Algumas questões

- Você acharia alguma roupa no quarto ?
- Você acharia facilmente algum objeto no quarto?
- É fácil mudar o layout do quarto?
- Você se sentiria bem nesse quarto?

Uma outra visão

- Quarto organizado;
- Objetos em seus lugares corretos;
- Ambiente confortável.

Qual a diferença dos dois ambientes?

Caos x Ambiente Perfeito !

Ambiente Organizado

- Você acharia alguma roupa no quarto ?
- Você acharia facilmente algum objeto no quarto?
- É fácil mudar o layout do quarto?
- Você se sentiria bem nesse quarto?

Outra questão

- Se num ambiente físico tão simples (um quarto) pode-se facilmente se instalar um **caos**, imagine como seria num ambiente de desenvolvimento de software?

No desenvolvimento de software, essa situação pode ocorrer (caos), porém as consequências são muito mais drásticas ! ! !

O trabalho do Desenvolvedor de Software termina quando o produto é entregue ao usuário ?

Ciclo de Vida do Software

- O **software** não termina quando é entregue ao usuário;

- Após a entrega, se inicia a fase de **manutenção** e **suporte**, que é crucial para garantir que o software continue a funcionar corretamente e atenda às **necessidades** do **usuário**;

Ciclo de Vida do Software

- Esta fase pode incluir **correção** de bugs, atualizações, melhorias e adaptação a novos requisitos ou tecnologias, adaptação para **escalabilidade**;

- Portanto, o desenvolvimento de software é um processo **contínuo** que vai além da entrega inicial.

Será que é fácil efetuar manutenção em softwares desorganizados?

Que problemas usualmente se encontra ao se fazer manutenção em um sistema desorganizado?

Software desorganizado
Observações frequentes

- Dificuldade para se **localizar** código;
- Partes internas difíceis de serem **alteradas**;
- Alterando-se alguma parte, **quebram-se** outras;
- Desenvolvedor inseguro e com **estresse**;
- Atribui-se falta de **qualidade** ao sistema de software;
- Atribui-se **incompetência** do desenvolvedor;
- Cliente passa a ter **desconfiança** da equipe de software;
- **Credibilidade** do desenvolvedor é afetada;
- **Custos** podem afetar o projeto;
- **Prazos** podem afetar o projeto;
- Mudanças de negócio não absorvidas pelo software;
- Há probabilidade de se **redesenvolver** o software.

Como desenvolver softwares organizados?

Diretriz Básica

- “A única constante de um software é a sua permanente mudança”;

(Martin Fowler)

Diretriz Básica

- O software deve ser organizado, estruturado, para **absorver** **mudanças**;
- Software não deve ser desenhado para permanecer **estático** ao longo do tempo;
- Desenvolvedor experiente emprega essa diretriz no desenvolvimento de software;
- **Software foi feito para ser alterado**;
- Software deve fornecer valor agregado ao cliente para o negócio atual e deve se **adequar** às futuras mudanças tecnológicas e corporativas.

Software para absorver mudanças

- Mudança de paradigma: Focar em **Qualidade** de Software;
- A Engenharia de Software trata a Qualidade focando conceitos de **Organização**, **Práticas** e **Diretrizes** de desenvolvimento de software;
- Esses conceitos são tratados na área de **Arquitetura de Software**;

O que é Arquitetura de Software?

Arquitetura de Software

- Corresponde à **estrutura** que abrange os **componentes** do software, as propriedades externamente visíveis desses componentes e as relações entre eles;
- Um **componente** pode ser um módulo de programa, uma classe orientada a objetos ou pode ainda ser estendido para abranger um banco de dados, etc;
- As **propriedades** dos componentes são características necessárias para o entendimento de como eles interagem com outros componentes;
- As **relações** podem ser, por exemplo, uma chamada de **módulo**, uma conexão a um banco de dados, etc.

Estilos de Arquitetura

- Arquitetos da Construção Civil usam diversos estilos arquitetônicos para a construção de um casa;

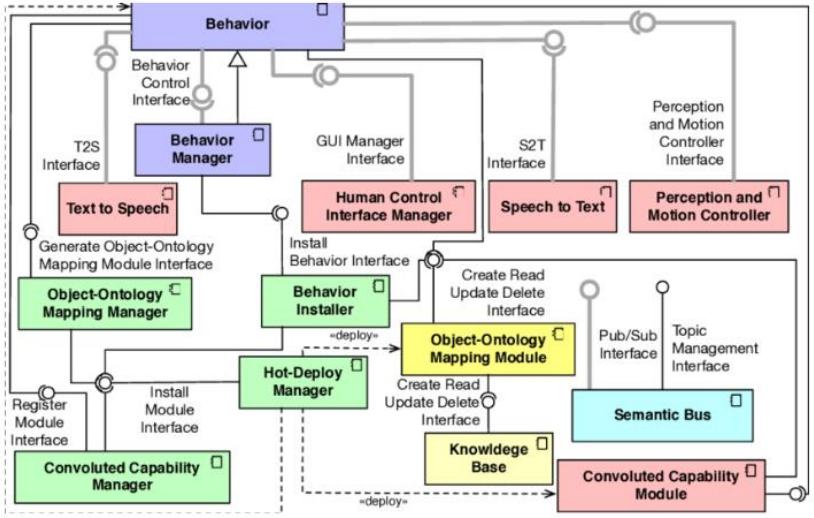
- Por exemplo, estilo Colonial, Estilo Contemporâneo, Estilo Clássico, etc

Arquitetura de Software

- A **Arquitetura de Software** é um conjunto de **estruturas** necessárias para desenvolver, manter e operacionalizar de um sistema de software;

- Corresponde à uma **representação** de **alto nível** do sistema que proporciona uma visão abstrata de suas partes e de como elas interagem;

- A **Arquitetura de Software** lida com as decisões de projeto que afetam a estrutura e o comportamento de um sistema de software.



Arquitetura de Software

- A arquitetura de software lida com as decisões de projeto que afetam a estrutura e o comportamento de um sistema, incluindo:

1. **Componentes de Software**: As diferentes partes que compõem o sistema, como módulos, classes ou pacotes, e suas responsabilidades.
2. **Relacionamentos**: Como esses componentes interagem ou se comunicam entre si.
3. **Restrições**: As regras e diretrizes que os desenvolvedores devem seguir ao construir o sistema, incluindo padrões de design e princípios de programação.
4. **Qualidades do Sistema**: Características como desempenho, segurança, escalabilidade e manutenibilidade.

Por que devemos estudar Arquitetura de Software ?

Importância da Arquitetura de Software

- **Compreensão e Comunicação**: Facilita a compreensão do sistema como um todo, ajudando as equipes a visualizar e discutir suas estruturas e comportamentos.
- **Decisões de Design**: Ajuda a determinar como o sistema será construído e quais tecnologias serão usadas.
- **Manutenção e Evolução**: Uma boa arquitetura torna mais fácil manter e evoluir o software ao longo do tempo.
- **Reuso**: Componentes bem projetados podem ser reutilizados em diferentes sistemas.

Arquiteturas de Software

- Seguem abaixo as principais arquiteturas de software:
  - Arquitetura Monolítica;
  - Arquitetura Cliente Servidor;
  - Arquitetura MVC (Model-View-Controller);
  - Arquitetura em Camadas (N-Tier);
  - Arquitetura Orientada a Serviços;
  - Arquitetura de Microsserviços;
  - Arquitetura Baseada em Componentes;
  - Arquitetura Serverless;
  - Arquitetura Orientada a Eventos.