**UNIVERSIDADE SENAI JOINVILLE**

**Felipe Rafael Rothbarth**

**OT08 - Apresentação do projeto**

**JOINVILLE**

**2024**

**Fichamento - OT08 - Apresentação do projeto**

### **Como estruturar um software?**

A estruturação de um software envolve diversas etapas, desde a concepção inicial até a implementação, manutenção e evolução. Uma abordagem típica inclui:

1. Levantamento de Requisitos: Identificar as necessidades do cliente ou usuário.
2. Análise de Requisitos: Detalhar e priorizar os requisitos identificados.
3. Design de Software: Criar a arquitetura e o design detalhado do sistema.
4. Desenvolvimento: Escrever o código-fonte e implementar as funcionalidades.
5. Testes: Verificar se o software atende aos requisitos e está livre de erros.
6. Implantação: Disponibilizar o software para o uso.
7. Manutenção: Corrigir erros, adaptar-se a novos requisitos ou melhorar o desempenho.

### **Quais são os componentes de software?**

Os componentes de software são partes modulares de um sistema, cada uma desempenhando uma função específica. Os principais componentes incluem:

1. Módulos: Segmentos do software que contêm funcionalidades específicas.
2. Classes/Objetos: Estruturas que encapsulam dados e comportamento em linguagens orientadas a objetos.
3. Interfaces: Definem como os módulos ou componentes interagem entre si.
4. Bibliotecas: Conjunto de funções e rotinas reutilizáveis.
5. Banco de Dados: Onde as informações e dados são armazenados e gerenciados.
6. APIs (Interfaces de Programação de Aplicações): Permitem que diferentes softwares se comuniquem.

### **Quais são os tipos de arquitetura de software?**

Os principais tipos de arquitetura de software incluem:

1. Monolítica: Todo o software é implementado em um único códigobase, com todas as funcionalidades integradas.
2. Cliente-Servidor: O software é dividido entre um servidor que fornece serviços e clientes que os utilizam.
3. Microservices: Funcionalidades são divididas em serviços independentes que se comunicam entre si.
4. SOA (Arquitetura Orientada a Serviços): Similar aos microservices, mas focado em serviços que oferecem funcionalidades específicas.
5. Event-Driven: Baseado em eventos que disparam ações específicas no software.
6. Arquitetura em Camadas (Layered Architecture): O software é dividido em camadas como apresentação, lógica de negócios e acesso a dados.

### **Quais são as 3 camadas de software?**

As 3 camadas de software mais comuns na arquitetura em camadas são:

1. Camada de Apresentação (Presentation Layer): Interface do usuário e interação. Exemplo: HTML/CSS, aplicações desktop.
2. Camada de Negócios (Business Layer): Processamento da lógica de negócios. Exemplo: Regras de negócios, algoritmos de processamento.
3. Camada de Dados (Data Layer): Gerenciamento e acesso a dados. Exemplo: Bancos de dados, serviços de persistência.

### **Quais são os tipos de software?**

Os principais tipos de software incluem:

1. Software de Sistema: Inclui sistemas operacionais, drivers, utilitários.
2. Software de Aplicação: Software usado pelos usuários para realizar tarefas específicas, como editores de texto, navegadores de internet.
3. Software de Programação: Ferramentas que ajudam os desenvolvedores a escrever código, como IDEs, compiladores.
4. Software Embarcado: Software desenvolvido para dispositivos específicos, como firmware em eletrodomésticos.
5. Software de Entretenimento: Jogos, mídias interativas.
6. Software de Rede: Gerenciamento de redes e comunicação, como servidores web, DNS.

### **O que é um Plano de Projeto de Software?**

Um Plano de Projeto de Software é um documento que descreve como um projeto de desenvolvimento de software será executado, controlado e concluído. Ele inclui:

* Objetivos e escopo: O que o projeto pretende alcançar.
* Cronograma: Linhas do tempo para cada fase do projeto.
* Recursos: Quais recursos (humanos, técnicos, financeiros) serão necessários.
* Riscos: Identificação de riscos potenciais e planos de mitigação.
* Plano de comunicação: Como as partes interessadas serão mantidas informadas.
* Plano de qualidade: Como a qualidade do software será assegurada.

### **Quais são os 3 principais padrões da arquitetura web?**

Os 3 principais padrões da arquitetura web são:

1. MVC (Model-View-Controller):
   * Model: Gerencia os dados e a lógica de negócios.
   * View: Apresenta os dados para o usuário.
   * Controller: Interage com o Model para atualizar o View.
2. REST (Representational State Transfer):
   * Arquitetura que usa HTTP para criar serviços web escaláveis e interoperáveis. Segue princípios como uso de verbos HTTP (GET, POST, PUT, DELETE) e recursos (URLs).
3. SPA (Single Page Application):
   * Arquitetura onde a aplicação web carrega uma única página HTML e atualiza dinamicamente o conteúdo conforme a interação do usuário. Exemplos incluem frameworks como Angular, React e Vue.js.

**REFERÊNCIAS BIBLIOGRÁFICAS**

SOMMERVILLE, Ian. Software Engineering. 10. ed. Boston: Pearson, 2015.

GAMMA, Erich; HELM, Richard; JOHNSON, Ralph; VLISSIDES, John. Design

Patterns: Elements of Reusable Object-Oriented Software. Boston: Addison-Wesley, 1994.

MARTIN, Robert C. Clean Architecture: A Craftsman's Guide to Software Structure and Design. Boston: Prentice Hall, 2017.

FOWLER, Martin. Patterns of Enterprise Application Architecture. Boston: Addison-Wesley, 2002.

Vue.js. Documentation. Disponível em: https://vuejs.org/v2/guide/ Acesso em: 08 ago. 2024.

Express.js. Express - Node.js web application framework. Disponível em:<https://expressjs.com/>. Acesso em: 08 ago. 2024.