

Documentação de Arquitetura de Software

*Arquitetura de Software -
Atividade Supervisionada 3*

Prof. Gilmar Ferreira Arantes

Alunos: João Gabriel Cavalcante, Jair Souza Meira
Rodrigues, Leonardo Moreira de Araújo, Matheus Franco
Cascão, Vitor Paulo Eterno Godoi

2025

INF

INSTITUTO DE
INFORMÁTICA



Sumário

1. [Por que precisamos documentar?](#)
2. [Como devemos estruturar a documentação?](#)
3. [Como devemos visualizar?](#)
4. [Como escrevemos e gerenciamos a documentação?](#)



**Por que precisamos
documentar ?**

Por que precisamos documentar ?

Por que documentar?

- A arquitetura de software define a estrutura e o comportamento de um sistema;
- Sua documentação é essencial para garantir comunicação, manutenção e evolução do projeto.

Comunicação e Alinhamento:

- Alinhar as **expectativas** entre os stakeholders (desenvolvedores, clientes, gestores, etc...);
- Facilitar o **entendimento técnico** e **não técnico** do sistema;
- Reduzir **mal-entendidos** e **ruídos** de comunicação;
- Servir como **ponto de referência** comum para todos os envolvidos.

Suporte ao Ciclo de Vida:

- Guiar o **desenvolvimento**, a **manutenção** e a **evolução**;
- Acelerar o **onboarding** de novos membros;
- Ajudar na **depuração** e **localização** de problemas;
- Registrar **decisões arquiteturais** e suas **justificativas**.



Por que precisamos documentar ?

Garantia de Qualidade:

- Permitir o **planejamento de atributos** como segurança, escalabilidade e desempenho;
- Auxiliar na **definição de estratégias de testes** para a aplicação;
- Servir como suporte para a **validação técnica** do sistema;
- Reduzir **riscos técnicos** com um planejamento mais claro e preciso.

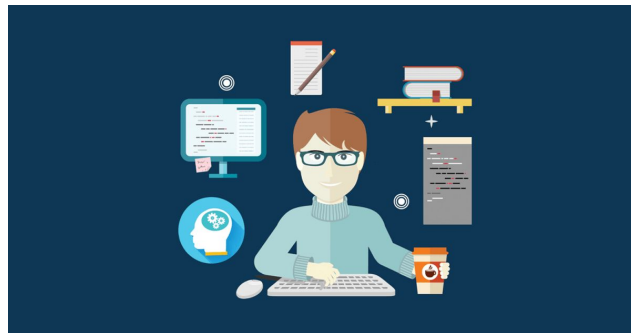
Atende a Requisitos Legais:


- Necessária para **projetos com regulamentações** (ex: saúde, finanças);
- Facilitar **auditorias externas** e **processos de certificação**;
- Evidenciar **conformidades** com padrões técnicos e legais.

Por que precisamos documentar ?

Longevidade do Sistema:

- Facilitar a **análise de impacto** de mudanças;
- Manter a **integridade arquitetural** com a evolução do sistema;
- Preservar **conhecimento essencial** em caso de rotatividade na equipe;
- Reduzir **dependências** de conhecimento tácito.





**Como devemos
estruturar a
documentação ?**

Como devemos estruturar a documentação?

O que é?

- Modelo organizacional utilizado para registrar, apresentar e manter as informações sobre a arquitetura de um sistema.
- Define quais seções, tipos de conteúdo e nível de detalhamento devem ser incluídos na documentação.
- Funciona como um guia para garantir consistência, clareza e completude na comunicação da arquitetura.

Objetivo de Estruturar a Documentação de Arquitetura

- Registrar decisões arquiteturais de forma clara, rastreável e justificável.
- Facilitar o entendimento técnico da arquitetura por todos os stakeholders (desenvolvedores, arquitetos, gestores).
- Garantir a continuidade do conhecimento arquitetural, mesmo com trocas de pessoal.
- Servir como referência para manutenção, evolução e auditorias técnicas.

Como devemos estruturar a documentação?

O que deve conter uma boa documentação de arquitetura

- Objetivos e contexto da arquitetura.
- Decisões arquiteturais, com justificativas e trade-offs.
- Visões e modelos arquiteturais (estrutural, comportamental, implantação).
- Padrões, restrições e diretrizes técnicas.
- Aspectos de qualidade (desempenho, segurança, escalabilidade etc.).
- Glossário, referências e anexos úteis.



Como devemos estruturar a documentação?

Modelo arc42 – Estrutura Prática e Reutilizável

Visão Geral

- Criado por Peter Hruschka e Gernot Starke, baseado em décadas de experiência real.
- Modelo aberto, orientado à prática.
- Proporciona uma estrutura modular e reutilizável para documentar arquiteturas.
- Facilmente adaptável a projetos ágeis ou tradicionais.
- Utilizado amplamente por arquitetos em ambientes ágeis, tradicionais e regulados.



arc

42

Como devemos estruturar a documentação?

Estrutura Sugerida (Template arc42)

Introdução e Metas -> Contexto da documentação e objetivos da arquitetura.

Restrições -> Técnicas, organizacionais e regulatórias.

Visão Geral -> Explicação do sistema como um todo.

Contexto -> Interfaces externas e interações com o ambiente.

Building Blocks (Visão Estrutural) -> Componentes, camadas e relações internas.

Runtime (Visão Comportamental) -> Interações em tempo de execução (cenários típicos).

Deployment -> Estrutura física de implantação (infraestrutura).

Decisões Arquiteturais -> Registros formais de decisões e alternativas consideradas.

Padrões e Diretrizes -> Convenções, práticas e tecnologias utilizadas.

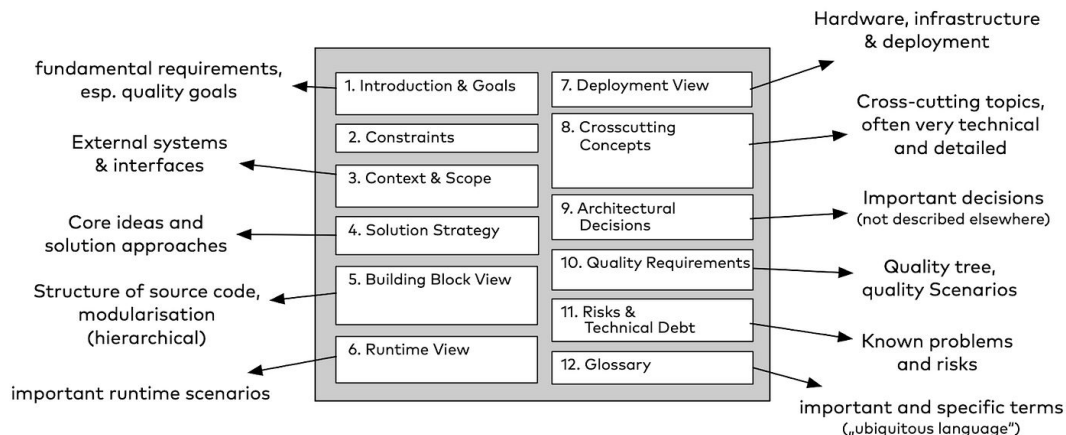
Como devemos estruturar a documentação?

Estrutura Sugerida (Template arc42) - Parte 2

Qualidades Técnicas -> Como a arquitetura trata atributos como segurança, desempenho etc.

Riscos Técnicos -> Pontos críticos e planos de mitigação.

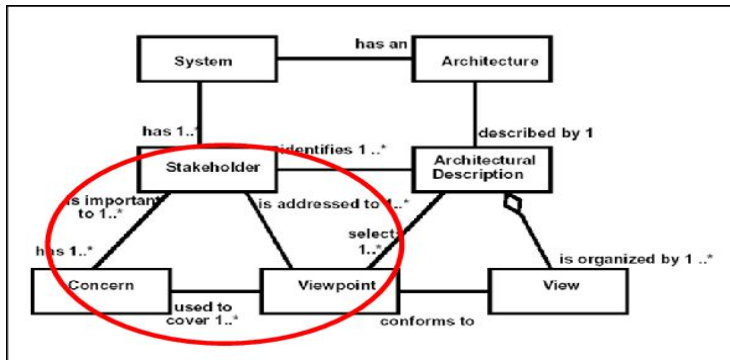
Glossário e Referências -> Termos específicos, fontes, links úteis.



Como devemos estruturar a documentação?

IEEE 1471 - Recommended Practice for Architectural Description of Software-Intensive Systems

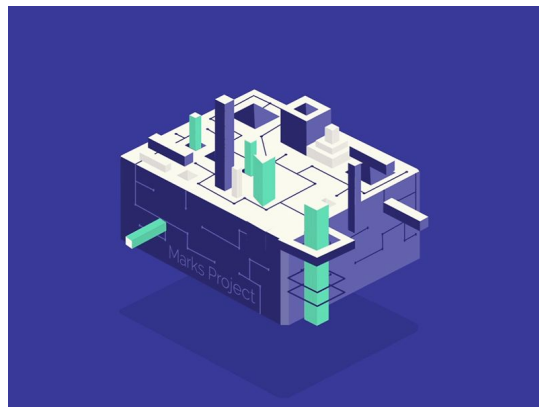
- Norma internacional para descrição de arquiteturas de sistemas.
- Foca na separação de visões para diferentes stakeholders.
- Principais conceitos:
 - Stakeholders e suas preocupações;
 - Viewpoints (pontos de vista);
 - Views (visões resultantes).
- Uso recomendado: ambientes regulados, grandes organizações, governança de TI.



Como devemos estruturar a documentação?

Outras alternativas para estruturar a documentação da Arquitetura

- Documenting Software Architecture (SEI)
- Software Architecture for Developers (Simon Brown)
- C4 Model (Context, Container, Component, Code)
- TOGAF (The Open Group Architecture Framework)
- SAD (Software Architecture Document - RUP/UML)





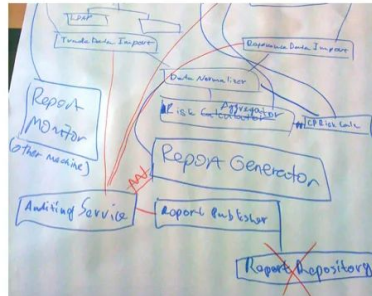
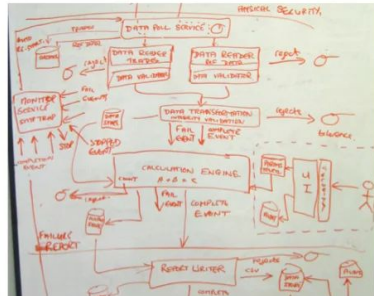
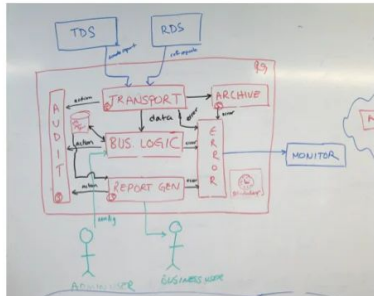
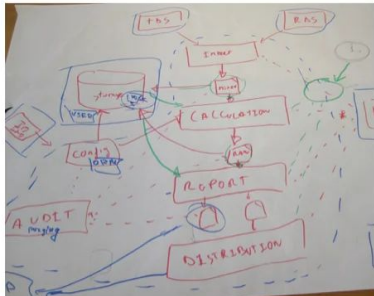
**Como devemos
visualizar?**

Por que é importante visualizar a arquitetura de software?

Visualizar ≠ só desenhar

- Ajudar stakeholders a entenderem o sistema.
- Facilitar a comunicação entre equipes (devs, POs, QA, etc).
- Suportar decisões técnicas e garantir alinhamento com os requisitos.
- O código não conta toda a história: visão arquitetural complementa.

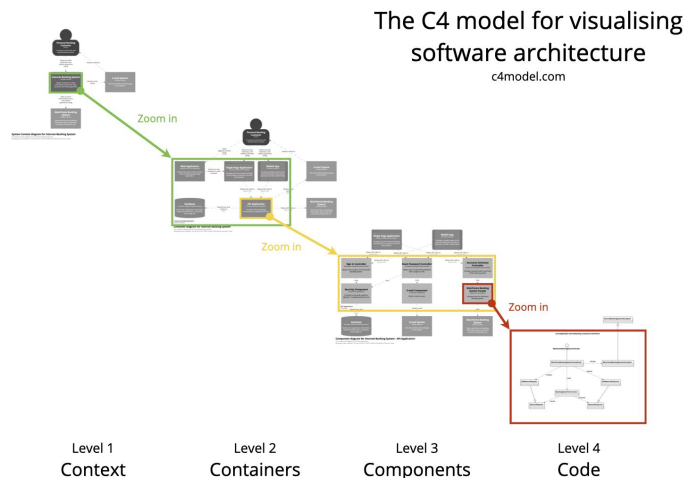
Exemplos de diagramas ambíguos



C4 Model – Visão geral

C4 = Contexto + Containers + Componentes + Código

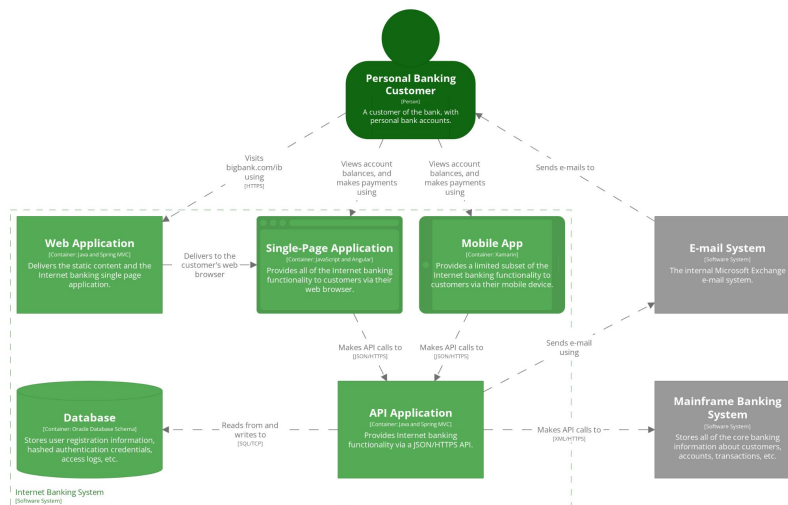
- Criado por Simon Brown.
- Abordagem baseada em abstrações e independente de notação.
- Ideal para documentar e comunicar a estrutura estática de sistemas de software.
- Foco em diferentes níveis de detalhe, de acordo com o público.



C4 Model

O que é um Container no C4?

- Uma **unidade executável ou implantável**.
- Pode ser frontend, backend, mobile app, banco de dados, etc.
- Ajuda a entender **como os blocos principais do sistema** interagem.



[Container] Internet Banking System
The container diagram for the Internet Banking System. Diagram created with Structurizr.
Wednesday, March 22, 2023 at 8:16 AM Coordinated Universal Time

C4 Model - Exemplos de Containers

Exemplos típicos:

- App Web: Node.js, Rails, ASP.NET...
- App Mobile: Android, iOS...
- Banco de Dados: MySQL, MongoDB...
- Armazenamento: Amazon S3, Azure Blob...
- Script ou função: Bash script, Lambda...


C4 Model

Diagrama de Contexto (C4)

- Mostra o **sistema como um todo** e como ele se conecta com usuários e sistemas externos.
- Ideal para comunicação com stakeholders não técnicos.

Diagrama de Componentes (C4)

- Mostra **módulos/componentes** internos e como se relacionam.
- Ajuda a entender **responsabilidades, dependências** e estrutura interna de um container.



**Como escrevemos e
gerenciamos a
documentação?**

Como devemos escrever a documentação de Arquitetura de Software?

Referência:

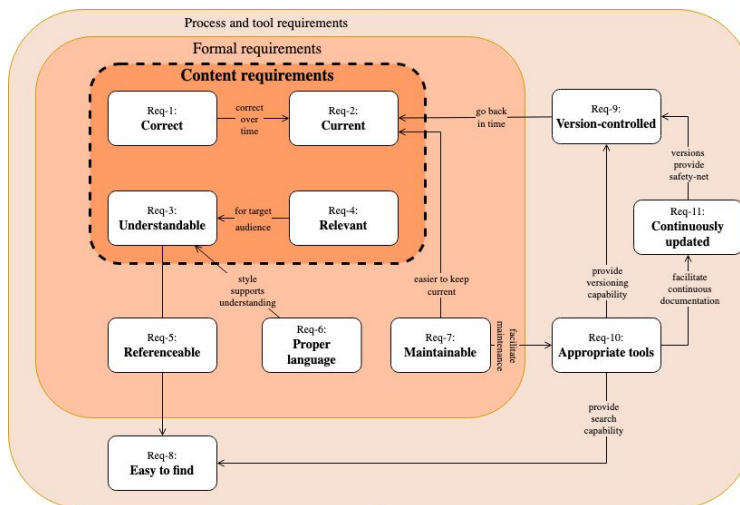
- Para que a documentação não seja feita de qualquer forma, ela deve ser estruturada com base em padrões bem definidos e deve **cumprir requisitos**. Para isso, uma ótima referência bibliográfica é o livro “Principles of technical documentation”, [Starke, 2022].



“Principles of technical documentation”

Requisitos são subdivididos em:

- **Requisitos de Processos e Ferramentas:** Req8, Req9, Req10, Req11
- **Requisitos Formais:** Req5, Req6, Req7
- **Requisitos de Conteúdo:** Req1, Req2, Req3, Req4



“Principles of technical documentation”

- **Requisitos de Conteúdo**

- Req-1: Correto - A documentação precisa ser precisa e livre de erros. Documentação errada geralmente é pior do que nenhuma documentação
- Req-2: Atual - A documentação atual precisa estar correta ao longo do tempo, refletindo as mudanças realizadas no código, infraestrutura ou interfaces do sistema.
- Req-3: Compreensível - A documentação precisa ser entendida pelo público-alvo.
- Req-4: Relevante - Com relação à estrutura, forma e conteúdo, a documentação deve ser relevante para as tarefas de seu público.

- **Requisitos Formais**

- Req-5: Referenciável - Use um esquema de numeração consistente para títulos, diagramas e tabelas.
- Req-6: Linguagem adequada - Use linguagem adequada, ortografia e gramática corretas, voz ativa, declarações positivas e frases curtas.
- Req-7: Manutenível - A manutenibilidade é essencial para manter a documentação atualizada

“Principles of technical documentation”

- **Requisitos de Processos e Ferramentas**

- Req-8: Fácil de encontrar - A documentação em si deve ser fácil de encontrar sempre que necessário. Seu conteúdo deve ser facilmente navegável e pesquisável
- Req-9: Versão controlada - Conforme o sistema evolui, sua documentação também evoluirá, sem perder seu histórico.
- Req-10: Ferramentas apropriadas - Foco no conteúdo, reduza o tempo necessário para configuração de ferramentas.
- Req-11: Atualizado continuamente - Crie o hábito de manter e expandir a documentação com cada mudança relevante em seu sistema.

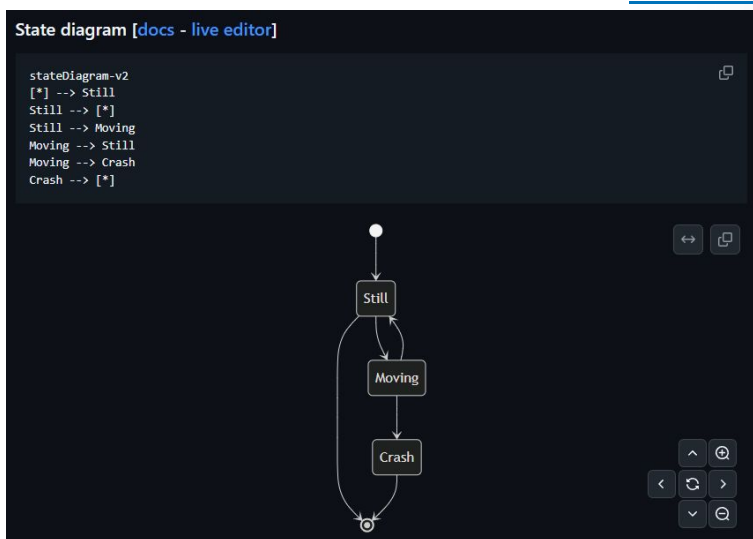
Docs as Code (DaC)

DaC

- Atender a alguns desses requisitos leva à conclusão de que devemos tratar os documentos como código.
- “Documentação como Código” significa que seu processo de documentação se beneficia das mesmas práticas que você usa para desenvolver software de sucesso.



Ref.: [Mermaid](#)



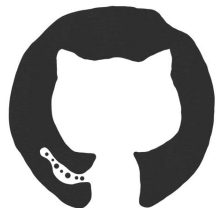
Práticas Recomendadas para Gerenciamento de Conteúdo

Versionamento de Conteúdo:

- Utilize sistemas de controle de versão (e.g., Git), conforme práticas de rastreabilidade e auditoria recomendadas pelo IEEE Std 828-2012, para gerenciar alterações e manter históricos confiáveis.

Separação Estrutural de Camadas:

- Adote a separação entre conteúdo, configuração e apresentação, alinhada com princípios de modularidade do IEEE Std 1471-2000, para facilitar manutenção e reutilização.



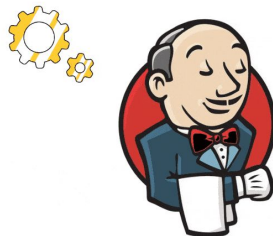
Práticas Recomendadas para Gerenciamento de Conteúdo

Automação de Processos (CI/CD):

- Implemente pipelines de integração e entrega contínua, conforme diretrizes de automação do IEEE Std 12207-2017, para compilação, validação, verificação e publicação eficientes.

Separação Estrutural de Camadas:

- Promova a reutilização de materiais, seguindo práticas de economia de recursos do IEEE Std 1517-2010, para reduzir duplicações e otimizar fluxos de trabalho.



Práticas Recomendadas para Gerenciamento de Conteúdo

Flexibilidade na Escolha de Ferramentas:

- Utilize ambientes de desenvolvimento integrado (IDEs) de preferência pessoal, respeitando recomendações de ergonomia e produtividade do IEEE Std 1063-2001, para criação de conteúdo.



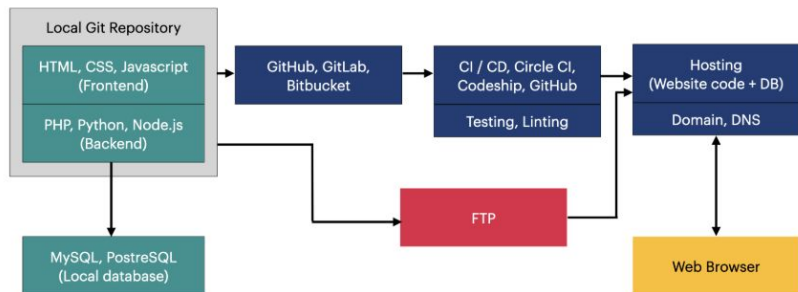
Resultados das Práticas de Gerenciamento de Conteúdo (IEEE)

- **Documentos Modulares:** Subdocumentos organizados (IEEE Std 1016-2009).
- **Conteúdo Personalizado:** Adaptado às partes interessadas (IEEE Std 12207-2017).
- **Imagens Referenciadas:** Sem incorporação, para eficiência (IEEE Std 1517-2010).
- **Docs como Código:** Documentação tratada como código-fonte (IEEE Std 26515-2018).
- **Formato Legível:** Estrutura para máquinas (IEEE Std 1421-2002).
- **Revisões via Git:** Pull requests e versionamento (IEEE Std 828-2012).
- **Multi-Formato:** Exportação para HTML5, PDF, DocBook, Confluence (IEEE Std 1471-2000).



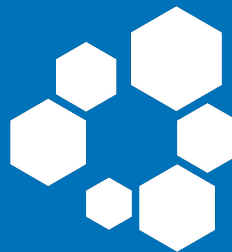
Dev Workflow: O que é e Importância

- Organizar tarefas
- Automatizar processos
- Garantir qualidade
- Acelerar entregas
- Facilitar colaboração



FIM!
OBRIGADO!!

INF
INSTITUTO DE
INFORMÁTICA



UFG
UNIVERSIDADE
FEDERAL DE GOIÁS