STEPHANY DE OLIVEIRA SOUSA MILHOMEM 4/12/2025

Documentação de Arquitetura de Software

STEPHANY DE OLIVEIRA SOUSA MILHOMEM 4/13/25 2:14PM

O que é documentação de Arquitetura de Software?

A **documentação de arquitetura de software** é o registro estruturado das decisões arquiteturais, componentes do sistema, suas interações e justificativas técnicas. Essa documentação descreve, de maneira clara e acessível, como o sistema é organizado e quais são as decisões que sustentam sua estrutura. Ela pode incluir:

- Diagramas;
- Textos explicativos;
- Modelos;
- Artefatos.

♡ 0 💬 0

STEPHANY DE OLIVEIRA SOUSA MILHOMEM 4/13/25 2:37PM

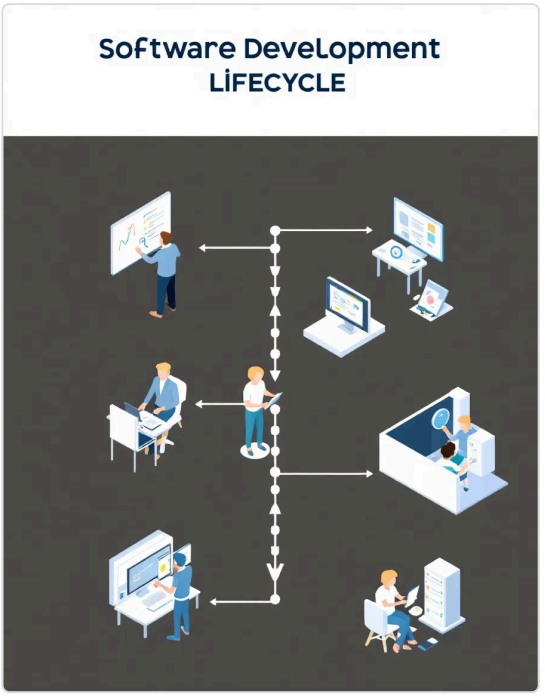
Quando e Por Que Documentar a Arquitetura?

Quando documentar?

- No início do projeto (para alinhar expectativas).
- Durante decisões críticas (ex.: escolha de banco de dados).
- Após mudanças significativas (evolução do sistema).

Por que documentar?

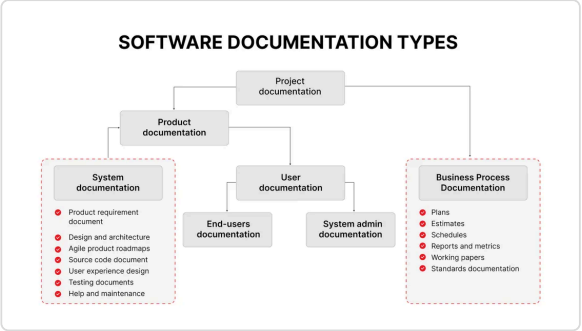
- **Reduzir retrabalho** (evitar decisões ruins por falta de contexto).
- **Otimizar onboarding** de novos membros da equipe.
- **Cumprir normas** (em projetos regulados, como saúde ou finanças).
- **Facilitar auditorias** e revisões técnicas.



♡ 0 💬 0

ALINE AYUMI SOMA HAMANO 4/13/25 1:20AM

Elementos Essenciais da Documentação



♡ 0 💬 0

ALINE NUNES DOS SANTOS 4/13/25 1:05AM

Boas Práticas na Escrita da Documentação

Objetivo:
Garantir que a documentação seja clara, precisa e útil para todos os stakeholders, permitindo a fácil compreensão e comunicação das decisões arquiteturais.

Pontos fundamentais:

- **Clareza e Objetividade:**
  Use uma linguagem simples e direta, evitando jargões desnecessários.
  Explique termos e conceitos técnicos quando necessário, lembrando sempre de quem será o público-alvo (técnico, gestor, cliente, etc.).
- **Consistência:**
  Mantenha um padrão de escrita e formatação ao longo de toda a documentação.
  Use terminologias consistentes para evitar ambiguidades e erros de interpretação.
- **Estrutura e Organização:**
  Organize a documentação em seções bem definidas que facilitem a localização das informações (por exemplo, visão geral, detalhes técnicos, diagramas, etc.).
  Utilize índices, sumários e links internos para permitir uma navegação rápida e intuitiva.
- **Detalhamento Adequado:**
  Evite exageros; a documentação deve conter o que é necessário para esclarecer e respaldar a arquitetura, sem se tornar excessivamente verbosa.
  Considere incluir exemplos práticos, explicações de decisões e referências a padrões ou modelos utilizados.
- **Revisão e Validação:**
  Periodicamente revise o conteúdo para assegurar que está atualizado com as mudanças na arquitetura.
  Estabeleça um processo de revisão com colegas e stakeholders para identificar e corrigir possíveis inconsistências ou erros.

♡0 ◯0

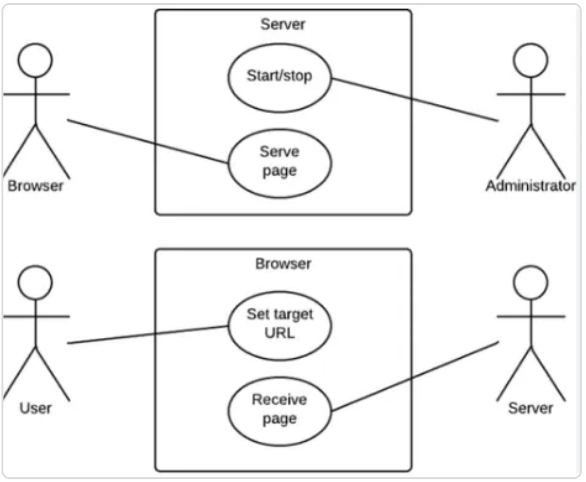
ALINE NUNES DOS SANTOS 4/13/25 1:09AM

Ferramentas e Técnicas de Apoio à Documentação

Objetivo:
Utilizar ferramentas e técnicas que facilitam a criação, visualização e manutenção dos artefatos arquiteturais.

Principais pontos a serem considerados:

- **Ferramentas de Modelagem e Diagramação:**
  Diagramas Estruturais e Comportamentais:
  Utilize diagramas como UML (diagramas de classes, de sequência, de atividades) para representar a estrutura e o fluxo do sistema. Outras abordagens, como BPMN para processos de negócio, também podem ser bastante úteis.
  Ferramentas Gráficas:
  Ferramentas como o **Microsoft Visio**, **draw.io**, **Lucidchart**, ou mesmo plataformas específicas como o **Enterprise Architect** e o **Archi** auxiliam na criação de diagramas padronizados e de fácil compreensão pelos stakeholders.
- **Integração com Outros Sistemas:**
  Documentação Automatizada e Integrada:
  Muitas equipes utilizam ferramentas que se integram ao controle de versões (como o **Git**) para manter a documentação atualizada juntamente com o código-fonte, reduzindo discrepâncias entre o que está implementado e o que foi documentado.
  Sistemas de Gestão de Documentação:
  Plataformas como o **Confluence**, **SharePoint** ou outras ferramentas de gerenciamento colaborativo ajudam a centralizar e versionar a documentação, permitindo acesso rápido e atualização contínua.
- **Técnicas de Modelagem:**
  Modelos Conceituais e Lógicos:
  Esquematizar a estrutura conceitual com diagramas que destacam entidades, relações e fluxos, e depois detalhar com diagramas mais técnicos, pode facilitar a transição de ideias para a implementação.
  Abordagens Visuais e Iterativas:
  O uso de técnicas como mapas mentais para organizar ideias ou modelos iterativos que evoluem conforme o projeto avança pode tornar o processo de documentação mais dinâmico e adaptável.
- **Integração com Metodologias Ágeis:**
  Documentação "Viva":
  Em contextos ágeis, recomenda-se a atualização constante da documentação



para refletir as mudanças no projeto. Ferramentas colaborativas permitem que os membros da equipe contribuam em tempo real, facilitando a adaptação e o refinamento do documento arquitetural.

♡ 0 ◯ 0

ALINE NUNES DOS SANTOS 4/13/25 1:21AM

Manutenção e Atualização da Documentação ao Longo do Ciclo de Vida

Objetivo:
Assegurar que a documentação reflete fielmente as mudanças na arquitetura e permaneça um recurso confiável e atual para todas as equipes envolvidas.

Passo a Passo para uma Manutenção Eficiente:

1. **Definição de um Cronograma de Revisões:**
   Periodicidade: Estabeleça revisões regulares (por exemplo, mensal ou a cada grande iteração) para verificar se a documentação ainda está alinhada com o estado atual do sistema.
   Datas-Chave: Programe revisões após mudanças significativas na arquitetura ou após a implementação de novas funcionalidades.
2. **Integração com Processos de Desenvolvimento:**
   Documentação Viva: Utilize práticas ágeis onde a documentação é atualizada juntamente com o desenvolvimento do sistema, garantindo que alterações sejam refletidas imediatamente.
   Automação e Integração: Considere o uso de ferramentas que se integrem com sistemas de controle de versão (como Git), possibilitando atualizações automatizadas e o rastreamento das mudanças.
3. **Definição de Responsabilidades:**
   Papéis Claros: Determine quem será o responsável por atualizar e revisar a documentação. Pode ser uma equipe dedicada ou a colaboração conjunta dos desenvolvedores, arquitetos e gerentes de projeto.
   Processo de Validação: Implemente um fluxo de revisão colaborativa onde alterações sugeridas na documentação são validadas e aprovadas pelos stakeholders.
4. **Feedback Contínuo:**
   Canal de Comunicação: Estabeleça um canal (como reuniões regulares ou uma ferramenta colaborativa) para receber feedback dos usuários da documentação.

Ajustes Baseados em Uso: Atualize o documento conforme necessário, com base nas críticas e sugestões dos usuários que interagem diretamente com a documentação.

5. **Registro e Histórico de Alterações:**
   Controle de Versão: Utilize sistemas de versionamento para manter um histórico de alterações e possibilitar o retorno a versões anteriores, se necessário.
   Logs de Alterações: Mantenha um registro com as alterações realizadas, justificativas e a data de execução para facilitar a rastreabilidade e auditoria do documento.

♡ 0 ◯ 0

ALINE NUNES DOS SANTOS 4/13/25 1:27AM

Exemplos de Aplicação

Objetivo:
Demonstrar como as boas práticas, ferramentas e processos descritos anteriormente podem ser aplicados em situações reais (ou simuladas), auxiliando na compreensão prática do uso da documentação arquitetural.

Exemplo 1: Sistema de E-commerce

- **Contexto:**
  Imagine um sistema de e-commerce que precise integrar diversas funcionalidades como catálogo de produtos, gerenciamento de pedidos, pagamentos e suporte ao cliente.
- **Aplicação Prática:**
  Boas Práticas:
  A documentação inicia com uma visão geral do sistema, destacando os principais módulos (catálogo, carrinho, pagamento e atendimento). Cada módulo é documentado com diagramas de classe e sequência, utilizando uma linguagem acessível para técnicos e não técnicos.
  Ferramentas e Técnicas:
  São usados diagramas UML para representar as interações entre módulos e ferramentas como draw.io ou Lucidchart para criar fluxogramas dos processos de compra.
  Manutenção:
  A cada nova funcionalidade (por exemplo, a adição de um sistema de recomendações), a equipe atualiza a documentação e mantém um histórico de mudanças utilizando um repositório integrado ao Git e uma plataforma colaborativa como o Confluence.
  Benefícios:
  Facilita a comunicação entre equipes

(desenvolvimento, QA e negócios) e permite uma rápida identificação de impactos em mudanças futuras.

Exemplo 2: Sistema de Gerenciamento Bancário

- **Contexto:**
  Um sistema de gerenciamento bancário para processamento de transações, gestão de contas e serviços financeiros digitais.
- **Aplicação Prática:**
  Boas Práticas:
  A documentação é dividida em várias seções, incluindo visão lógica (estrutura dos dados e fluxos de transações) e visão física (distribuição dos servidores e redes de comunicação). Há detalhamento dos requisitos não funcionais, como segurança e performance, fundamentais para o setor financeiro.
  Ferramentas e Técnicas:
  Utiliza-se o padrão C4 model para representar a arquitetura de sistemas em camadas, diagramas de fluxo de dados para ilustrar as transações bancárias e ferramentas colaborativas para manutenção da documentação.
  Manutenção:
  Com as constantes mudanças regulatórias e a evolução tecnológica, o sistema possui ciclos de revisão periódicos para atualizar tanto a arquitetura quanto a documentação associada. O versionamento é feito por meio de ferramentas como Git, assegurando a rastreabilidade das modificações.
  Benefícios:
  Permite auditar as mudanças e garante que todas as partes interessadas estejam cientes das adaptações necessárias para cumprir com as normas do setor.

♡ 0 ◯ 0

ALINE AYUMI SOMA HAMANO 4/13/25 1:34AM

Decisões Arquiteturais e seus Registros

- Capturar escolhas críticas (ex.: uso de microsserviços vs monolito, banco de dados SQL vs NoSQL).
- Justificativas baseadas em requisitos, trade-offs e restrições.
- **Como**: Padrão de registro (ex.: templates como ADR - Architecture Decision Record).
- **Exemplo de registro (ADR – Architectural Decision Record):**
  Decisão: Utilizar arquitetura de microsserviços

Contexto: O sistema precisa escalar de forma independente por módulo.
Opções Consideradas: Monolito, Microsserviços, Modular Monolith
Consequência: Aumento na complexidade de infraestrutura, necessidade de orquestração
Data: 12/04/2025
Responsável: Arquiteto do projeto

♡ 0 ◯ 0

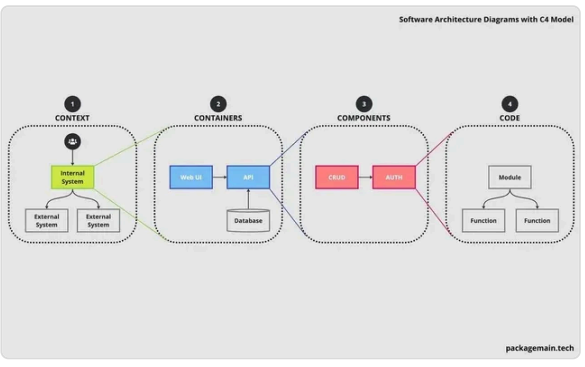
STEPHANY DE OLIVEIRA SOUSA MILHOMEM 4/13/25 2:22PM

Objetivos Principais da Documentação Arquitetural

1. **Comunicação:** É a ferramenta primária para comunicar a arquitetura a todas as partes interessadas (stakeholders), incluindo desenvolvedores, testadores, gerentes de projeto, novos membros da equipe e até mesmo clientes (dependendo do nível de detalhe).
2. **Base para Análise:** Permite que a arquitetura seja revisada e analisada quanto a atributos de qualidade (como desempenho, segurança, escalabilidade, manutenibilidade) antes mesmo de muito código ser escrito.
3. **Guia para Implementação:** Serve como um mapa para as equipes de desenvolvimento, garantindo que o sistema seja construído de acordo com o planejado.
4. **Preservação do Conhecimento:** Captura as decisões importantes e o raciocínio por trás delas, evitando que esse conhecimento se perca com o tempo ou com a saída de pessoas da equipe.
5. **Planejamento da Evolução:** Facilita a manutenção e a evolução futura do sistema, pois fornece um entendimento claro de sua estrutura atual.
6. **Onboarding:** Ajuda novos membros da equipe a entenderem rapidamente a estrutura e os princípios do sistema.

♡ 0 ◯ 0

ALINE LIMA MARTINS COELHO 4/13/25 3:02PM



C4 Model

O C4 Model é uma abordagem para modelar arquiteturas de software, organizada em quatro níveis de abstração: Contexto, Contêiner, Componente e Código. Cada nível oferece uma visão única da arquitetura, desde uma visão geral de alto nível até os detalhes de implementação.

- **Contexto:** delineia o escopo do seu sistema de software e suas interações com usuários e outros sistemas.
- **Contêiner:** se aprofunda nos componentes principais do sistema.
- **Componente:** oferece uma visão mais detalhada dos elementos individuais dentro de cada contêiner
- **Código:** fornece uma visão detalhada dos blocos de construção do sistema, apresentando as classes e interfaces que sustentam os componentes.

♡ 0 ◯ 0

ALINE AYUMI SOMA HAMANO 4/13/25 1:26AM

Stakeholders e suas Preocupações

- **Quem são**: Identifique todas as partes interessadas (ex.: clientes, desenvolvedores, equipe de operações, negócios, reguladores).
- **Preocupações**: Documente os interesses de cada stakeholder (ex.: desempenho, segurança, custos, escalabilidade).
- **Exemplo**:
  Stakeholder: Equipe do financeiro.
  Preocupação: Redução de custos com bom desempenho.

♡ 0 ◯ 0

ALINE AYUMI SOMA HAMANO 4/13/25 1:31AM

Visões e Requisitos Arquiteturais

Diferentes perspectivas para atender a preocupações específicas:

- **Visão Lógica**:
  Componentes, módulos e suas interações (ex.: diagrama de classes, serviços).
- **Visão de Processos**:
  Foca nos processos em execução, concorrência e comunicação
- **Visão Física**:
  Infraestrutura, servidores, redes e deploy (ex.: diagrama de implantação).
- **Visão de Desenvolvimento**:
  Estrutura de código, repositórios e dependências (ex.: módulos do Maven, pacotes npm).

Requisitos Arquiteturais

- **Funcionais**:
  Capacidades que o sistema deve oferecer (ex.: "O sistema deve processar pagamentos via PIX").
- **Não-Funcionais (NFRs)**:
  Qualidades do sistema (ex.: desempenho, disponibilidade, segurança):

Desempenho: "Tempo de resposta < 2s para 10k usuários concorrentes."
Disponibilidade: "99,99% uptime em ambiente de produção."
Segurança: "Autenticação via OAuth2.0 e criptografia AES-256."

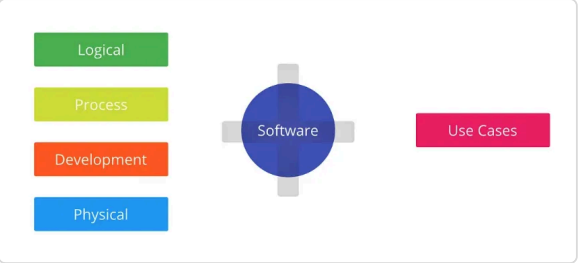
♡ 0 ◯ 0

ALINE LIMA MARTINS COELHO 4/13/25 2:44PM

4+1 Views (Kruchten)

O modelo possui quatro visões: lógica, desenvolvimento, processo e física. Além disso, casos de uso ou cenários selecionados são utilizados como a visão "mais um" para mostrar o design. Por isso, 4+1 visões.

- **Visão Lógica:** se preocupa com a funcionalidade do sistema no que se refere aos usuários finais.
- **Visão de Processo:** concentra-se no comportamento do sistema em tempo de execução e lida com seus elementos dinâmicos. Ela explica os processos do sistema e como eles se comunicam.
- **Visão de Desenvolvimento:** descreve um sistema do ponto de vista de um programador e se preocupa com a administração do software.
- **Visão Física:** a camada física se preocupa com a topologia dos componentes de software, bem como com as conexões físicas entre esses componentes.
- **Cenários:** Um pequeno número de casos de uso, ou cenários, que se tornam a quinta visão, são usados para ilustrar a descrição da arquitetura.



♡ 0 ◯ 0

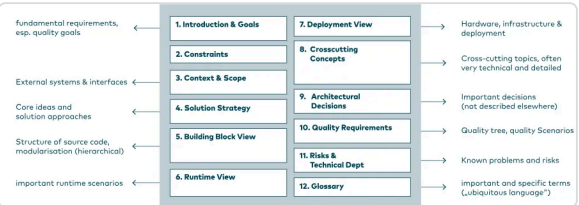
ALINE LIMA MARTINS COELHO 4/13/25 2:52PM

Arc42

O arc42 é um modelo para documentação de arquitetura que responde às duas perguntas a seguir de forma pragmática, mas pode ser adaptado às suas necessidades específicas:

- O *que* devemos documentar/comunicar sobre nossa arquitetura?
- *Como* devemos documentar/comunicar isso?

O Arc42 é de código aberto e gratuito, independente de ferramentas e tecnologias, portanto, você pode usar o arc42 para sistemas arbitrários.Trata-se de um modelo de documentação bastante minimalista em comparação com outros modelos. Não importa o tamanho do projeto, ele mantém a



O arc42 propõe uma dúzia de seções para sua documentação. Elas são organizadas de cima para baixo para facilitar a compreensão.

documentação relativamente enxuta e fácil de manter.

♡ 0 ◯ 0

ALINE LIMA MARTINS COELHO 4/13/25 3:22PM

Norma ISO/IEC/IEEE 42010:2022

A norma ISO/IEC/IEEE 42010:2022 é um padrão internacional que define a terminologia, conceitos e requisitos para descrições de arquitetura de sistemas e sistemas com uso intensivo de software. Baseia-se na ideia de que um sistema pode ter múltiplas arquiteturas, dependendo do ponto de vista e das preocupações de diferentes partes interessadas. Por exemplo, um sistema pode ter uma arquitetura funcional, uma arquitetura de segurança, uma arquitetura de desempenho e assim por diante. Cada arquitetura é descrita por um conjunto de modelos, diagramas e documentos que abordam as preocupações e pontos de vista relevantes.
Os principais conceitos
trazidos pela norma são:

- **Stakeholders**: pessoas ou organizações com interesse nas decisões arquiteturais.
- **Preocupações (concerns)**: questões ou necessidades relevantes dos stakeholders (ex: segurança, desempenho, escalabilidade).
- **Visão (views)**: representação da arquitetura sob uma determinada perspectiva (por exemplo: lógica, processos, implantação).
- **Modelos (models)**: artefatos concretos (como diagramas ou textos) que compõem uma visão.

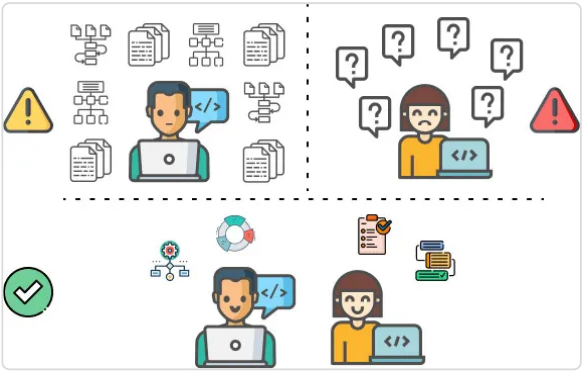
♡ 0 ◯ 0

ALINE LIMA MARTINS COELHO 4/13/25 3:30PM

Padronização na
Documentação de Arquitetura

Uma boa documentação arquitetural geralmente segue a seguinte estrutura:

1. Introdução e contexto
2. Visão geral da arquitetura
3. Stakeholders e preocupações
4. Pontos de vista e modelos
5. Decisões arquiteturais
6. Padrões e tecnologias
7. Riscos e trade-offs
8. Glossário e referências



Documentos de arquitetura de software promovem a reutilização de componentes e a padronização das práticas de desenvolvimento. Seguir um padrão na documentação da

arquitetura é essencial para garantir a coerência e a qualidade do projeto; facilitar a comunicação entre as equipes e com os stakeholders; facilitar a manutenção e a evolução do sistema; e ajudar na tomada de decisões e no gerenciamento de riscos.