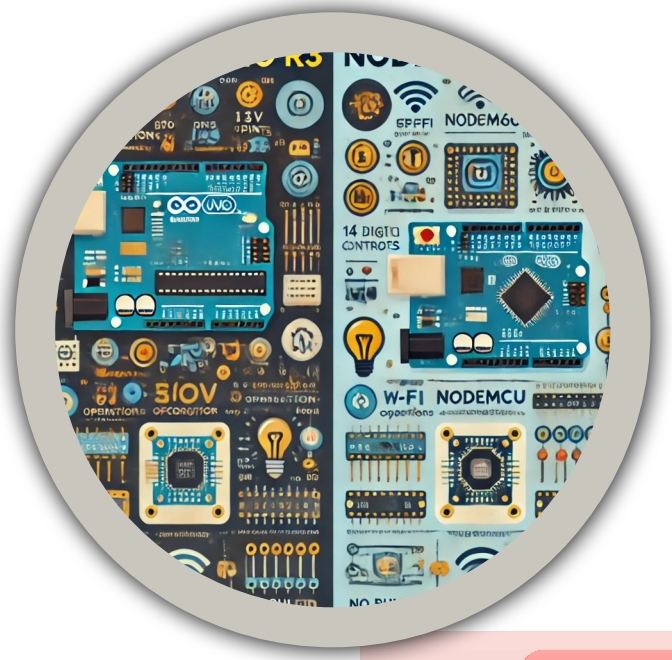
Documentação Técnica de Projeto IoT



Professor: Antonio Fernando Traina

O que é documentação técnica?

É o registro organizado das informações de um projeto, incluindo objetivos, arquitetura, funcionamento, testes e resultados.

Importância em projetos IoT

Garante entendimento do sistema, facilita replicação, manutenção, auditoria e integração com outros sistemas.

Referência para equipe

Serve como guia para desenvolvedores, técnicos e avaliadores, detalhando componentes e decisões técnicas.

Reprodutibilidade do projeto

Permite que outros repliquem o projeto com fidelidade, garantindo consistência e aprendizado contínuo.

Interoperabilidade entre sistemas

Facilita a integração entre diferentes dispositivos e plataformas por meio de padrões e documentação clara.

Validação e certificações

Necessária em ambientes regulados para garantir segurança, desempenho e conformidade técnica.

Longevidade do conhecimento

Preserva informações críticas do projeto mesmo após a saída de membros da equipe.

Estrutura básica da documentação

Inclui:

1) capa,
2) resumo,
3) objetivos,
4) Arquitetura e esquemas,
5) código,
6) Testes,
7) resultados e
8) plano de manutenção.

Capa e folha de rosto: função

Identificam o projeto, autores, instituição e contexto acadêmico ou técnico de forma padronizada.

Capa: informações essenciais

Título do projeto, nomes, curso, instituição, disciplina, professor orientador e data.

Folha de rosto: descrição formal

Inclui descrição do objetivo acadêmico do projeto e vínculo com a disciplina ou curso.

Resumo executivo: propósito

Apresenta visão geral do projeto, contexto, tecnologias utilizadas e impacto esperado.

Resumo: o que deve conter

Problema abordado, solução proposta, principais tecnologias e resultados esperados.

Objetivo do projeto: definição clara

Descreve o problema e a finalidade do sistema proposto, com foco técnico e aplicabilidade. Não descreve o que foi feito ou implementado, mas o que será resolvido, qual a solução esperada.

Assim, o objetivo do projeto

- Descreve o problema e a finalidade do sistema proposto, com foco técnico e aplicabilidade."
- O objetivo de um projeto técnico deve:
  - Delimitar claramente o problema a ser resolvido (por exemplo, falhas no monitoramento manual de umidade do solo).
  - Indicar a finalidade da solução proposta, ou seja, o que se pretende alcançar com o projeto (como criar um sistema de monitoramento automatizado).
  - Enfatizar o foco técnico (quais aspectos tecnológicos são aplicáveis) e a relevância prática (aplicabilidade no mundo real, como na agricultura, indústria, cidades inteligentes).

Assim, o objetivo do projeto...

- Não deve conter verbos no passado ou na voz ativa sobre a execução (ex: "foi montado", "foi programado").
- Deve indicar a meta a ser alcançada: o que o sistema pretende resolver, qual lacuna ou necessidade ele busca suprir.
- Deve ser formulado com verbos no infinitivo, expressando intenção e direcionamento (ex: "Desenvolver um sistema capaz de...", "Criar uma solução para...").

Objetivo: alinhamento com a solução

Deve estar coerente com a arquitetura e os resultados planejados do projeto.

Arquitetura do sistema: visão geral

Apresenta como o projeto foi estruturado com hardware e software integrados.

Hardware: sensores, atuadores

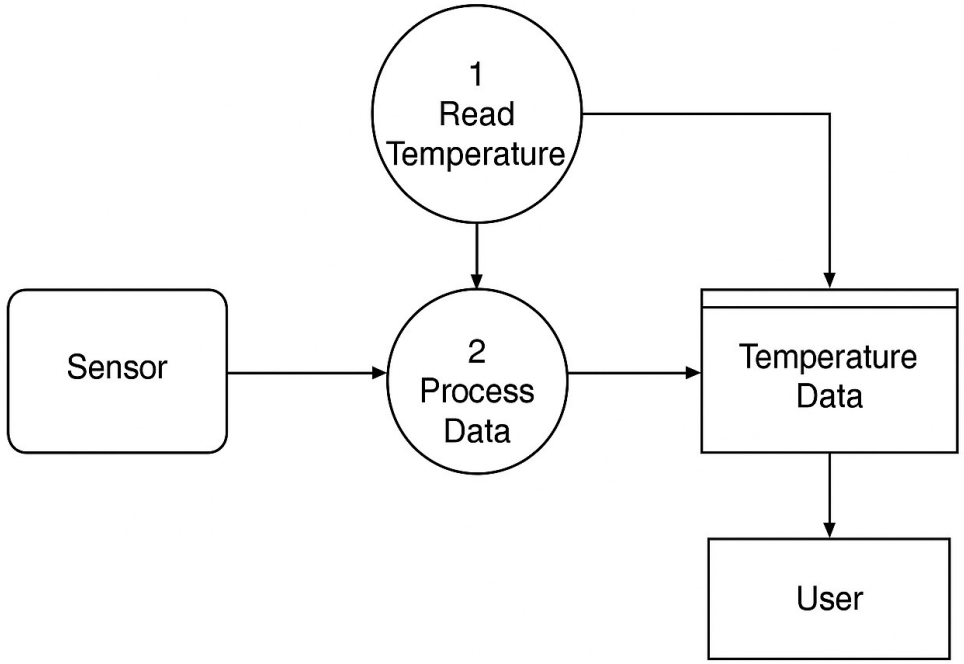
Detalha todos os componentes físicos usados e suas interações no sistema.

Software: protocolos, bibliotecas

Inclui as linguagens, bibliotecas, plataformas e protocolos usados na solução.

Diagramas de blocos

Representam funcionalmente os principais módulos e como se comunicam.



Ferramentas para Desenvolver DFDs (1/2)

- Draw.io (diagrams.net): Ferramenta gratuita, online e offline. Ideal para DFDs, fluxogramas e diagramas técnicos. Integra com Google Drive e GitHub.
- Lucidchart: Ferramenta online colaborativa com suporte a DFD, fluxos lógicos, BPMN e UML. Versão gratuita com limitações.
- yEd Graph Editor: Software desktop gratuito. Cria diagramas técnicos com layout automático e exportação profissional.
- Microsoft Visio: Ferramenta corporativa robusta para diagramas técnicos e fluxos de processos. Parte do pacote Microsoft 365.
- Pencil Project: Aplicação open-source para wireframes, fluxogramas e lógicas simples. Leve e gratuito.

Ferramentas para Desenvolver DFDs (2/2)

- Creately: Editor online colaborativo com templates para DFD e fluxogramas. Suporta trabalho em equipe em tempo real.
- Tinkercad Circuits + Blockly: Simulador online da Autodesk para projetos Arduino com blocos visuais de lógica (ideal para IoT educacional).
- Figma (com plugins): Editor moderno com plugins para diagramas de fluxo. Suporte à colaboração e estética refinada.
- Mermaid.js: Ferramenta baseada em código para desenhar fluxogramas e DFDs diretamente com texto Markdown.

Esquemas elétricos: precisão

Mostram conexões físicas, pinagens, alimentação e proteções do circuito.

Diagramas visuais: clareza

Devem ser legíveis, bem organizados e incluir legendas explicativas.

Fluxo de dados: DFD e lógica

Descreve como os dados fluem desde a aquisição até o destino final.

Fluxogramas e pseudocódigo

Ajudam a entender a lógica do software com representação visual ou textual.

Código-fonte comentado: importância

Permite a compreensão da lógica do programa e facilita manutenção e revisão.

Boas práticas de comentários

Comente blocos importantes, explique decisões e nomeie variáveis de forma clara.

Montagem do sistema: passo a passo

Explica como montar fisicamente o sistema, com imagens e precauções técnicas.

Testes: unitário, integração e final

Validam o funcionamento de cada parte do sistema e o comportamento geral.

Critérios e registros de testes

Definem parâmetros de sucesso e documentam os resultados obtidos.

Resultados esperados: definição clara

Listam os comportamentos funcionais esperados e metas de desempenho.

Métricas e objetivos mensuráveis

Devem ser quantificáveis, como frequência de leitura, tempo de resposta, etc.

Plano de manutenção: previsibilidade

Prevê trocas de componentes, atualizações de software e inspeções regulares.

Evolução futura do sistema

Sugere melhorias e adaptações, como novos sensores ou integração com apps.

Definição de responsabilidades

Define quem manterá o projeto e será responsável por atualizações.

Boas práticas: clareza e organização

Estrutura lógica, linguagem objetiva, uso de diagramas e padrões visuais.

Completude e padronização

Evita omissões técnicas, utiliza templates e segue normas da área.

Documentação ativa e atualizável

Permite modificações contínuas e acompanha a evolução do projeto.

Exemplos práticos de projetos

Estação meteorológica, irrigação automática, monitoramento de água, etc.

Ferramentas: Word, Markdown, LaTeX

- Word: acessível e institucional
- Markdown: leve e ideal para GitHub
- LaTeX: alta qualidade técnica
- OverLeaf
- GitHub README

GitHub README e integração

README apresenta resumo, instalação e uso; permite integração com código e controle de versão.

Overleaf: plataforma online colaborativa

Editor LaTeX na nuvem com suporte a colaboração, templates e visualização em tempo real.

Importância Estratégica

Base para IA e Machine Learning: Algoritmos só funcionam bem quando treinados com grandes volumes de dados relevantes e de qualidade.

Decisões Estratégicas: Dados permitem análises preditivas, identificação de padrões e suporte a decisões mais precisas e embasadas no agronegócio, indústria, saúde, etc.

➔ “Sem dados, é apenas opinião.” – W. Edwards Deming

Considerações Finais

Qualidade dos Dados: Dados imprecisos, incompletos ou desatualizados comprometem toda a análise.

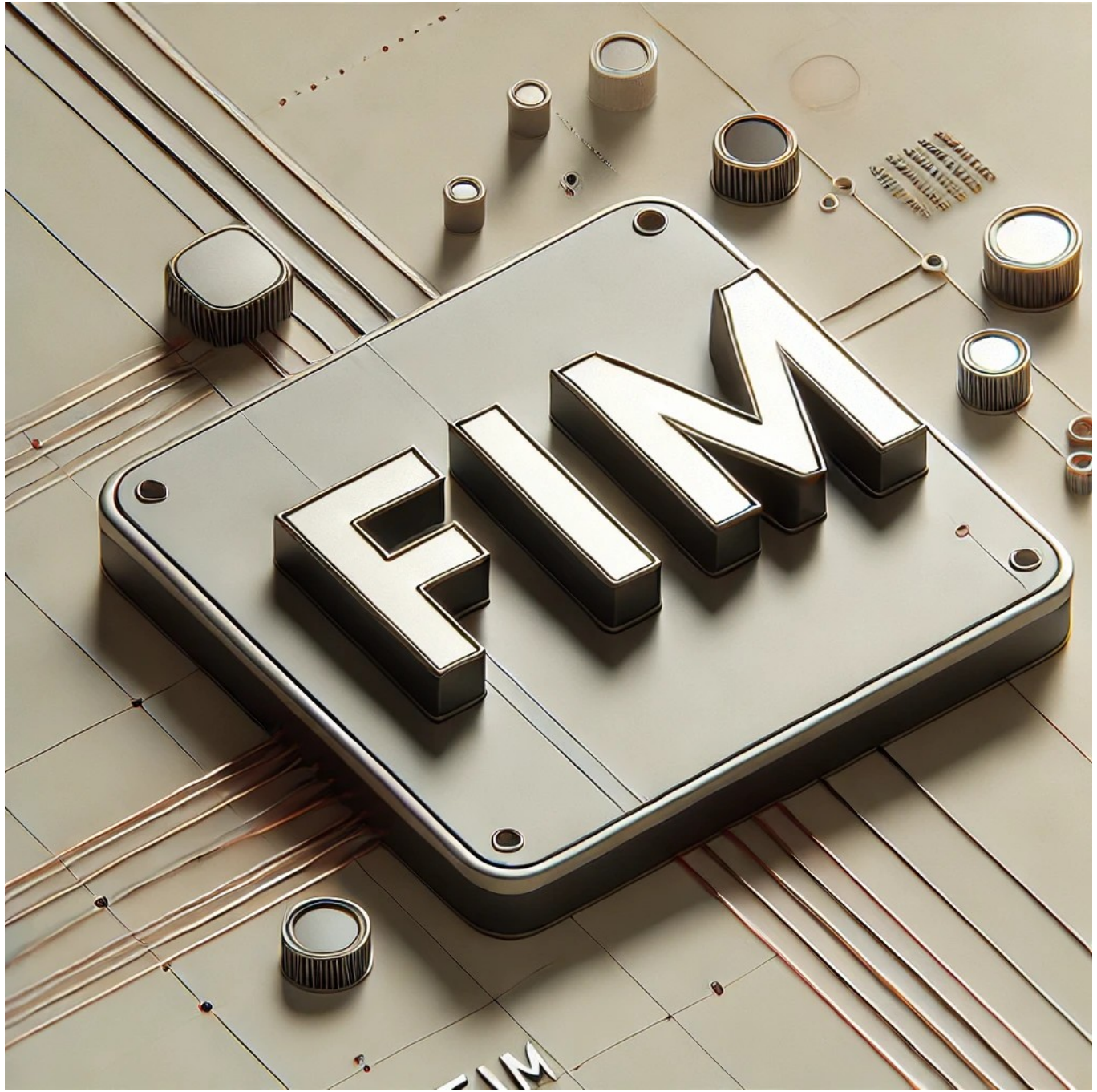
➔ Coleta e validação devem ser contínuas.

Segurança da Informação: Proteger os dados coletados contra vazamentos, acessos indevidos e usos indevidos é essencial.

➔ Encriptação, autenticação e governança são indispensáveis.

Valor Estratégico: Dados bem coletados e tratados se tornam ativos valiosos.

➔ Sustentam decisões, automatizam processos e geram inovação.


FIM