Documentação do JSONForms
Uma Abordagem com Technical Writing

Yan G. Santana[1], **Awdren de L. Fontão**[2]

[1]Faculdade de Computação - Universidade Federal de Mato Grosso do Sul (UFMS)
Caixa Postal 79070-900 – Campo Grande – MS – Brasil

yan.g.santana, awdren.fontao @ufms.br

Abstract. *This project aims to complement the official JSONForms documentation by addressing aspects that, although implemented, have not been adequately documented. By applying technical writing techniques, it seeks to improve the clarity and comprehension of the content, ensuring that technical information is presented in an accessible and functional manner for developers. The initiative arose from difficulties encountered during code mining within the SIGFAP system, highlighting the need for a more structured and efficient approach. To evaluate the quality in use of the enhanced documentation, a comparative experimental study was conducted with 12 developers (6 per group), divided into two groups. Group A used the documentation developed with Technical Writing techniques, while Group B used the official documentation. Metrics for completion time (to assess Efficiency) and error rate (to assess Effectiveness) were collected, following the ISO 9241-11 definitions, to compare the performance between the groups. The final objective is to quantitatively demonstrate the impact of high-quality documentation on developer productivity and to deliver a reference material for the community.*

Resumo. *Este trabalho tem como objetivo complementar a documentação oficial do JSONForms, abordando aspectos que, embora implementados, não foram devidamente documentados. Com a aplicação de técnicas de technical writing, busca-se aprimorar a clareza e a compreensão do conteúdo, garantindo que as informações técnicas sejam apresentadas de forma acessível e funcional para os desenvolvedores. A iniciativa surgiu a partir das dificuldades encontradas na mineração de código no contexto do sistema SIGFAP, evidenciando a necessidade de uma abordagem mais estruturada e eficiente. Para avaliar a qualidade de uso da documentação aprimorada, foi conduzido um estudo experimental comparativo com 12 desenvolvedores (6 por grupo), divididos em dois grupos. O Grupo A utilizou a documentação desenvolvida com técnicas de Technical Writing, enquanto o Grupo B utilizou a documentação oficial. Foram coletadas métricas de tempo de conclusão (para aferir Eficiência) e taxa de erros (para aferir Eficácia), fundamentadas nas definições de Usabilidade da norma ISO 9241-11, para comparar o desempenho entre os grupos. O objetivo final é demonstrar quantitativamente o impacto de uma documentação de alta qualidade na produtividade do desenvolvedor e entregar um material de referência para a comunidade.*

Palavras-chave: *JSONForms, Technical Writing, Documentação de Software, Engenharia de Software, Experiência do Desenvolvedor.*

Campo Grande - MS
2025

1. Introdução

A documentação técnica é amplamente reconhecida na teoria como um ativo inestimável para qualquer projeto de software, essencial para que as partes interessadas possam usar, manter e evoluir um sistema [Aghajani et al. 2020]. No entanto, na prática, ela é frequentemente afetada por deficiências críticas, como conteúdo insuficiente, informações ambíguas e, principalmente, a ausência de exemplos práticos adequados.

Pesquisas em larga escala reforçam que esse não é um problema pontual, mas sistêmico na indústria. Um estudo conduzido por Aghajani et al. (2019) com 146 profissionais identificou que a ”Incompletude de Conteúdo”e ”Exemplos de Código Errôneos ou Ausentes”estão entre as barreiras mais severas enfrentadas por desenvolvedores. A ausência de instruções claras não apenas frustra o desenvolvedor, mas impacta diretamente a eficiência da engenharia de software, aumentando o tempo gasto em tarefas de manutenção e aprendizado de novas APIs.

Esse cenário reflete-se claramente no ecossistema do JSONForms. Embora seja uma ferramenta poderosa para renderização de formulários, sua documentação oficial apresenta lacunas significativas na explicação do mecanismo de Rules (Regras), forçando os desenvolvedores a um processo de tentativa e erro. Essa deficiência é evidenciada empiricamente pelas dificuldades de manutenção identificadas no sistema SIGFAP, onde a ausência de diretrizes claras na documentação oficial resultou em implementações heterogêneas e custosas. A análise preliminar do código legado demonstrou que a falta de exemplos práticos forçou os desenvolvedores a adotarem soluções de contorno, impactando a manutenibilidade do projeto.

Diante da ineficácia de correções pontuais ou intuitivas para resolver esse gargalo de aprendizado, este trabalho não se limita a preencher essas lacunas intuitivamente; ele propõe uma intervenção baseada em práticas consolidadas de Technical Writing, tratadas por empresas como o Google como infraestrutura crítica de desenvolvimento.

Ao alinhar a reescrita da documentação com taxonomias de problemas identificadas na literatura recente de Engenharia de Software [Aghajani et al. 2020], este projeto busca demonstrar empiricamente como a transição de uma documentação descritiva para uma documentação orientada à tarefa pode mitigar gargalos reais de produtividade industrial.

1.1. Objetivos

Os objetivos deste trabalho são:

- **Objetivo Geral:** Avaliar o impacto da documentação técnica aprimorada no desempenho de desenvolvedores.
- **Objetivos Específicos:**
    1. Complementar a documentação oficial do JSONForms, abordando funcionalidades de Rules não documentadas.
    2. Aplicar técnicas de *Technical Writing* para transformar informações complexas em um conteúdo claro, acessível e funcional.
    3. Desenvolver uma aplicação de teste (”Playground”) para condução do experimento e coleta de métricas automatizadas.
    4. Validar a documentação gerada através de um estudo experimental comparativo.
    5. Analisar quantitativamente o impacto da escrita técnica na eficiência, tempo de conclusão e taxa de erros dos desenvolvedores.
    6. Disponibilizar a documentação produzida como um recurso para a comunidade de desenvolvedores.

2. Fundamentação Teórica

A fundamentação teórica deste trabalho abrange três áreas principais de conhecimento que se interconectam para sustentar a pesquisa proposta. Primeiramente, apresenta-se o JSONForms como framework de desenvolvimento de formulários baseados em esquemas, explorando suas funcionalidades e, especialmente, o mecanismo de Rules que constitui o foco principal da documentação desenvolvida. Em seguida, examina-se o Technical Writing como disciplina responsável pela comunicação eficaz de informações técnicas complexas, detalhando seus princípios, metodologias de avaliação e impacto na produtividade de desenvolvedores. Por fim, discutem-se as lacunas documentais comuns em frameworks de software e seus efeitos no ecossistema de desenvolvimento, contextualizando a relevância deste trabalho no cenário atual da engenharia de software.

2.1. JSONForms

O JSONForms é um framework de código aberto para a renderização de formulários baseados em esquemas JSON (JSON Schema). Sua principal vantagem é a capacidade de gerar interfaces de usuário (UI) complexas e dinâmicas a partir de uma definição de dados, desacoplando a lógica do formulário da sua apresentação [EclipseSource 2025].

O framework segue o paradigma de ”*schema-driven development*”, onde a estrutura dos dados é definida através de JSON Schema e a apresentação é controlada através de UI Schema. Esta abordagem oferece benefícios significativos em termos de manutenibilidade, reutilização de código e consistência da interface [Pressman 2014].

2.1.1. Rules: Mecanismo de Condicionais do JSONForms

As Rules são mecanismos que permitem definir condições dinâmicas para a exibição e o comportamento dos formulários. Elas possibilitam alterar a interface com base nos valores inseridos pelo usuário, tornando os formulários interativos e inteligentes. As principais funcionalidades das Rules são:

- Habilitar e desabilitar campos com base em determinados valores;
- Exibir ou ocultar campos dinamicamente conforme as respostas do usuário;
- Aplicar validações condicionais para garantir que os dados atendam a requisitos específicos.

Segundo a documentação oficial do JSONForms [EclipseSource 2025], as Rules implementam o conceito de ”formulários adaptativos”, onde a interface se modifica dinamicamente baseada no contexto e nas interações do usuário, melhorando significativamente a experiência do usuário e a eficiência do preenchimento.

2.2. Technical Writing

Technical Writing (Escrita Técnica) é a disciplina de comunicar informações complexas e técnicas a um público específico de forma clara, concisa e precisa [Markel and Selber 2018]. Diferente da escrita acadêmica ou literária, seu principal objetivo não é a argumentação ou o entretenimento, mas sim ajudar o usuário a realizar uma tarefa específica de forma eficiente e autônoma.

No contexto de desenvolvimento de software, Technical Writing se manifesta principalmente através da documentação técnica, que serve como interface entre o desenvolvedor da ferramenta e seus usuários [Sommerville 2011]. Uma documentação eficaz deve atender aos seguintes princípios fundamentais:

Clareza: A informação deve ser apresentada de forma inequívoca, evitando ambiguidades que possam levar a interpretações incorretas [Google Developers 2025].

Concisão: Eliminar informações desnecessárias, focando apenas no que é essencial para completar a tarefa [Redish 2012].

Orientação à Tarefa: A documentação deve ser estruturada em torno das tarefas que o usuário precisa realizar, não da estrutura interna do sistema [Johnson-Eilola 2005].

Exemplos Práticos: Incluir códigos funcionais e cenários reais que demonstrem a aplicação prática dos conceitos [Carroll 1990].

Diretrizes amplamente utilizadas, como as do Google Technical Writing [Google Developers 2025], reforçam esses princípios e orientam produção de documentação em nível profissional.

2.2.1. Impacto da Qualidade da Documentação na Produtividade

Diversos estudos demonstram o impacto direto da qualidade da documentação na produtividade dos desenvolvedores. Robillard [Robillard 2009] conduziu um estudo com 440 desenvolvedores e identificou que documentação inadequada é uma das principais barreiras para a adoção de APIs, resultando em:

- **Aumento do tempo de desenvolvimento**, devido à dificuldade em encontrar as informações necessárias;
- **Maior incidência de erros**, causada por documentação ambígua ou incompleta;
- **Frustração e abandono** da ferramenta por parte dos desenvolvedores.

Complementarmente, um estudo em larga escala de Aghajani et al [Aghajani et al. 2019] analisou 878 artefatos de documentação de quatro fontes diferentes (incluindo Stack Overflow e issues) e identificou uma taxonomia detalhada de problemas que os desenvolvedores enfrentam, como documentação incompleta, ambígua e exemplos incorretos.

2.2.2. Princípios do Google Technical Writing

O Google desenvolveu um conjunto abrangente de diretrizes para Technical Writing que se tornaram referência na indústria [Google Developers 2025]. Os principais princípios incluem:

1. **Audience-First Approach**: Sempre começar identificando quem é o público-alvo, qual seu nível de conhecimento técnico e quais tarefas precisam realizar;
2. **Information Architecture**: Organizar a informação de forma hierárquica, do geral para o específico, facilitando tanto a leitura sequencial quanto a consulta pontual;
3. **Active Voice e Present Tense**: Utilizar voz ativa e tempo presente para criar instruções mais diretas e claras;
4. **Consistent Terminology**: Manter terminologia consistente ao longo de toda a documentação para evitar confusão;
5. **Progressive Disclosure**: Apresentar informações em camadas, começando com o essencial e oferecendo detalhes adicionais conforme necessário.

2.2.3. Metodologias de Avaliação de Documentação

A literatura apresenta diferentes abordagens para avaliar a eficácia da documentação técnica. Uma abordagem padrão é decompor a avaliação com base nos componentes centrais da Usabilidade, conforme definido pela norma ISO 9241-11 [ISO 2018]:

- **Eficiência (Efficiency)**: Medida através dos recursos gastos (e.g., tempo necessário) para completar tarefas específicas usando a documentação.
- **Eficácia (Effectiveness)**: Avaliada através da precisão e completude com que os usuários atingem os objetivos (e.g., taxa de erros ou sucesso na conclusão da tarefa proposta).
- **Satisfação (Satisfaction)**: Mensurada através de questionários padronizados que avaliam a experiência subjetiva do desenvolvedor.

3. Método de Pesquisa

O presente trabalho adotou uma abordagem de pesquisa experimental comparativa, seguindo o protocolo de um Estudo de Viabilidade (*Feasibility Study*) para validar a eficácia da documentação técnica aprimorada. O estudo **foi conduzido** com estudantes e profissionais da área de computação, divididos em dois grupos, conforme detalhado a seguir.

3.1. Delineamento Experimental

O experimento seguiu um delineamento *between-subjects*, onde os participantes **foram alocados** aleatoriamente em um de dois grupos:

- **Grupo de Controle (GC):** Utilizou a documentação oficial do JSONForms.
- **Grupo Experimental (GE):** Utilizou a documentação complementar desenvolvida neste trabalho.

Os dois grupos executaram as mesmas tarefas e tiveram seus resultados comparados. O objetivo **foi** isolar o efeito da documentação aprimorada.

Para garantir a validade interna do estudo, a alocação aleatória dos participantes foi monitorada em relação ao perfil técnico (experiência com JSONForms), conforme coletado no pré-questionário de caracterização. Esta medida visou mitigar o risco de que a aleatoriedade resultasse em uma distribuição não balanceada de expertise entre os grupos, o que poderia comprometer a validade da comparação. Conforme o sessão 7, a distribuição resultante se mostrou equilibrada.

3.2. Participantes

O estudo **foi realizado** com uma amostra de 12 participantes com experiência prévia em desenvolvimento. O número de participantes **foi determinado** na fase de planejamento, visando garantir dados suficientes para uma análise preliminar de viabilidade.

3.3. Instrumentação

A instrumentação do estudo consiste em:

- **Pré-Questionário de Caracterização:** Coleta informações sobre o perfil técnico dos participantes.
- **Ambiente de Teste ("JSON Form Playground"):** Aplicação web para execução das tarefas e coleta automática de métricas.
- **Tarefas práticas::** Um conjunto de tarefas de complexidade crescente focadas na implementação de *Rules* do JSONForms.

3.4. Procedimento

O experimento será dividido em duas fases:

1. **Preparação:** Aplicação do pré-questionário e alocação dos participantes nos grupos.
2. **Execução:** Realização das tarefas no Playground, com coleta automática de métricas.

4. A Documentação Técnica Produzida

Como parte central deste trabalho, foi produzida uma documentação técnica focada nas *Rules* do JSONForms. Diferentemente da documentação oficial, que foca na descrição técnica das propriedades da API, este trabalho utilizou princípios de *Technical Writing* baseados nas diretrizes do Google [Google Developers 2025], priorizando a clareza, a concisão e a orientação à tarefa.

A documentação foi estruturada para ser um guia prático e autônomo, visando reduzir a carga cognitiva e acelerar o aprendizado. O material completo está disponível publicamente em repositório aberto[1].

[1] `https://github.com/Yan-Santana/JsonForms_TCC/tree/main/Docs`

4.1. Estratégia de Technical Writing Aplicada

A reescrita do conteúdo técnico seguiu uma estratégia de *Progressive Disclosure* (Divulgação Progressiva), onde a informação é apresentada em camadas de complexidade. A estrutura padronizada para cada regra (Rule) segue a ordem:

1. **Conceito (Concept):** Uma explicação clara e concisa do que a regra faz e em qual contexto deve ser usada, evitando jargões desnecessários.
2. **Cenário de Teste (Context):** A definição de um problema real (e.g., ”Ocultar campo de telefone se o usuário não quiser receber ligações”), criando uma conexão imediata com a realidade do desenvolvedor.
3. **Exemplo de Código (Code):** Um bloco completo contendo *JSON Schema* e *UI Schema*, permitindo a reprodução imediata (”Copy-pasteability”).
4. **Explicação da Solução (Analysis):** Uma análise linha a linha do código, explicando a lógica do *effect* e da *condition*.

4.2. Análise Comparativa: Abordagem Oficial vs. Proposta

Para evidenciar a contribuição deste trabalho, a Tabela 1 apresenta um comparativo qualitativo entre a abordagem da documentação oficial do framework e a documentação complementar desenvolvida.

Tabela 1. Comparativo: Documentação Oficial vs. Documentação Complementar

Documentação Oficial	Documentação Complementar (Proposta)
Foco: Descritivo (”O que a ferramenta faz”).	**Foco:** Orientado à Tarefa (”Como resolver um problema”).
Exemplos: Trechos de código abstratos ou isolados do contexto.	**Exemplos:** Cenários completos (Schema + UI) baseados em casos de uso reais (ex: Formulário de Contato).
Linguagem: Técnica, passiva e focada na API.	**Linguagem:** Ativa, direta e focada no desenvolvedor (”Você deve...”).
Validação: Não sugere como testar.	**Validação:** Inclui roteiro passo-a-passo para teste manual.

4.3. Exemplo de Artefato Gerado

A Figura 1 ilustra um trecho da documentação produzida referente à regra de **Visibilidade Condicional** (*Hide/Show*). Note a inclusão explicita de um ”Cenário de Teste”e a explicação detalhada da lógica aplicada ao *UI Schema*, elementos ausentes na fonte oficial.

Exemplo: Visibilidade Condicional Simples

Cenário: Ocultar o campo ”Telefone”a menos que o usuário marque ”Sim”em ”Deseja receber ligações?”.
Trecho do UI Schema:

```
{
  "type": "Control",
  "scope": "#/properties/telefone",
  "rule": {
    "effect": "HIDE",
    "condition": {
      "scope": "#/properties/receberLigacoes",
      "schema": { "not": { "const": "Sim" } }
    }
  }
}
```

Explicação: A regra define que o campo deve ser `HIDE` (ocultado) quando a condição for verdadeira. A condição verifica se o valor de `receberLigacoes` **NÃO** é ”Sim”. Logo, o campo só aparece se a resposta for positiva.

Figura 1. Recorte da documentação produzida demonstrando a estrutura explicativa.

5. Estudo de Viabilidade

Este capítulo apresenta o Estudo de Viabilidade (EV) realizado para avaliar a aplicabilidade e a eficácia da abordagem de *Technical Writing* proposta para complementar a documentação oficial do **JSONForms**. O estudo segue a metodologia e a estrutura do Capítulo 5 da dissertação de Awdren de Lima Fontão [Fontão 2016], adaptando os elementos de pesquisa para o contexto do projeto de TCC. As etapas metodológicas, instrumentos e o delineamento experimental foram replicados, respeitando as devidas adaptações ao contexto tecnológico (JSONForms).

5.1. Planejamento do Estudo de Viabilidade

O Estudo de Viabilidade foi planejado para fornecer evidências empíricas sobre a utilidade e a clareza da documentação complementar desenvolvida, em comparação com a documentação existente.

5.2. Objetivo do Estudo

O objetivo principal deste estudo é **avaliar a viabilidade do experimento e quantificar o impacto da documentação complementar** no desempenho de desenvolvedores, focada em aspectos não documentados, por meio da aplicação de técnicas de *Technical Writing*. Este objetivo está alinhado com o Objetivo Geral do trabalho, que busca aferir quantitativamente o valor da escrita técnica no contexto de documentação de software.

5.2.1. Questões de Pesquisa

As seguintes questões de pesquisa (QPs) guiaram o Estudo de Viabilidade:

Tabela 2. Questões de Pesquisa (QPs) do Estudo de Viabilidade

QP	Descrição
QP1	A documentação complementar, elaborada com técnicas de *Technical Writing*, é percebida como mais clara e compreensível do que a documentação oficial para os aspectos abordados?
QP2	A documentação complementar auxilia os desenvolvedores a implementar os recursos não documentados do JSONForms de forma mais **eficiente** e com menos erros?
QP3	A abordagem de *Technical Writing* utilizada é considerada adequada para a criação de documentação técnica no contexto de bibliotecas de *front-end* como o JSONForms?

5.2.2. Instrumentação

Para a coleta de dados, foram utilizados os seguintes instrumentos, adaptados do estudo de referência:

1. **Formulário de Caracterização:** Questionário inicial para identificar o perfil dos participantes (experiência com React, TypeScript, JSON Schema e JSONForms).
2. **Pré-Questionário de Atitudes:** Questionário para capturar a percepção inicial dos participantes sobre a documentação técnica em geral e a do JSONForms em particular.
3. **Roteiro de Tarefas Práticas:** Conjunto de tarefas de implementação envolvendo recursos do JSONForms que não estão bem documentados na documentação oficial, mas que foram abordados na documentação complementar.
4. **Entrevista de Avaliação:** Imediatamente após a conclusão das tarefas, os participantes respondem individualmente às questões do roteiro de entrevista, fornecendo avaliações quantitativas e qualitativas sobre a experiência.
    - **Clareza (Clarity):** Avaliar a clareza, concisão e uso de exemplos no texto.
    - **Usabilidade (Usability):** Avaliar a facilidade de encontrar informações e a organização do conteúdo.
    - **Completude (Completeness):** Avaliar se a documentação contém todas as informações necessárias para a tarefa.

5.2.3. Seleção de Grupos

O estudo foi conduzido com um grupo de desenvolvedores, divididos em dois subgrupos, seguindo uma abordagem de estudo quase-experimental:

- **Grupo A (Experimental):** Utilizará a **documentação complementar** (produzida neste TCC) para realizar as mesmas tarefas.

- **Grupo B (Controle):** Utilizará apenas a **documentação oficial** do JSONForms para realizar as tarefas.

A seleção dos participantes foi por conveniência, buscando desenvolvedores com diferentes níveis de experiência em *front-end* e JSONForms. A alocação foi realizada de forma aleatória simples, garantindo a imparcialidade na distribuição. O estudo foi conduzido com um total de **12 participantes**, sendo 6 alocados no Grupo de Controle (GC) e 6 no Grupo Experimental (GE).

5.3. Execução do Estudo de Viabilidade

O estudo foi conduzido de forma remota e assíncrona, utilizando uma plataforma proprietária para garantir a integridade dos dados.

5.3.1. Ambiente do Experimento: JSONForms Playground

Para assegurar a consistência das medições e minimizar variáveis externas, foi desenvolvida uma aplicação web personalizada denominada **JSONForms Playground**. Esta plataforma serviu como ambiente controlado para a execução das tarefas e coleta automatizada de métricas.

O acesso ao ambiente foi restrito mediante autenticação (login e senha). O fluxo de utilização foi estruturado da seguinte forma:

- **Cadastro e Alocação:** No primeiro acesso, os participantes realizavam um cadastro onde informavam o grupo ao qual foram previamente alocados (Grupo A ou Grupo B).
- **Configuração do Ambiente:** Com base na seleção do grupo, a plataforma configurava automaticamente a interface e os recursos disponíveis para o teste.
- **Execução Controlada:** A aplicação apresentava sequencialmente as três tarefas práticas, registrando automaticamente as interações, o tempo de permanência em cada etapa e as tentativas de submissão.

A aplicação foi construída utilizando React e TypeScript, espelhando o ambiente real de desenvolvimento utilizado no projeto SIGFAP, mas instrumentada para fins de pesquisa.

5.3.2. Procedimento

O experimento seguiu as seguintes etapas:

1. **Pré-Questionário:** Os participantes responderam ao Pré-Questionário de Atitudes para estabelecer a linha de base.
2. **Onboarding na Plataforma:** Foi fornecido o link de acesso ao *JSONForms Playground*. Os participantes criaram suas contas e selecionaram seu grupo de estudo conforme a alocação aleatória recebida.
3. **Execução das Tarefas:** Dentro da plataforma, os participantes realizaram o conjunto de três tarefas práticas (Fácil, Média e Difícil) com um *timebox* máximo de 60 minutos. O sistema registrou automaticamente:

- O tempo exato até a conclusão ou desistência;
- A quantidade de erros de validação disparados;
- O número de *resets* e submissões de código.

4. **Entrevista de Avaliação:** Imediatamente após a conclusão das tarefas na plataforma, os participantes responderam individualmente às questões do roteiro de entrevista, fornecendo avaliações qualitativas sobre a documentação utilizada.

Para garantir a validade técnica, as tarefas foram desenhadas para cobrir diferentes níveis de complexidade da API do JSONForms, conforme detalhado no Quadro 3.

Tabela 3. Detalhamento Técnico das Tarefas Experimentais

Nível	Descrição Técnica da Tarefa
Tarefa 1 (Fácil)	**Visibilidade Condicional:** O participante devia alterar o *UI Schema* para que o campo ”Telefone”ficasse oculto (`effect: HIDE`) por padrão e só aparecesse se o campo ”Deseja contato?”fosse marcado como ”Sim”.
Tarefa 2 (Média)	**Multiplas Condições:** Implementar uma condição *”allOf”* ou *”anyOf”* para validar se o campo ”Detalhes do Evento”aparecer somente quando o usuário selecionar *Sim*, integrando-a ao registro de validadores do JSONForms (`registerValidator`).
Tarefa 3 (Difícil)	**Habilitação Condicional Explícita(Enable):** Configurar o campo ”Email”para permanecer desabilitado por padrão, tornando-se editável apenas quando o usuário selecionar ”Sim”na opção ”Aceita receber comunicações?”, exigindo a aplicação da regra `effect: ENABLE` no *UI Schema*.

5.3.3. Variáveis e Métricas

As variáveis e métricas a serem coletadas e analisadas são apresentadas na Tabela 4. A inclusão da justificativa conecta cada métrica aos objetivos de qualidade de software (ISO 9241-11) e à detecção de barreiras de aprendizado.

Tabela 4. Variáveis e Métricas do Estudo de Viabilidade

Variável/Métrica	Descrição	Justificativa (Raciocínio)	QP
Métricas de Eficiência (QP2)			
Tempo até Conclusão (TE) [Aghajani et al. 2020]	Tempo médio gasto para completar o conjunto de tarefas (em minutos).	Avaliar a **produtividade**. Segundo a ISO 9241-11, melhor usabilidade deve reduzir os recursos (tempo) para atingir o objetivo.	QP2
Primeira Tentativa (PT)	Tempo gasto até a primeira submissão de código válida.	Medir a **curva de aprendizado** imediata e a barreira de entrada inicial da documentação.	QP2
Submissões (Sub)	Número total de submissões de código realizadas.	Identificar comportamento de **tentativa e erro**. Um número alto sugere ”chute”por falta de clareza.	QP2
Edições de Schema (ES)	Quantidade de edições realizadas no schema JSON.	Mensurar o **esforço cognitivo**. Excesso de edições indica indecisão sobre a estrutura correta.	QP2
Resets de Código (RC)	Número de vezes que o participante resetou o código (indicador de frustração/dificuldade).	Indicador crítico de **frustração**. Representa a falha total da estratégia mental do desenvolvedor.	QP2
Erros (Err) [Aghajani et al. 2020]	Quantidade total de erros de implementação ocorridos.	Avaliar a **eficácia** (precisão). Documentações ambíguas geram implementações incorretas.	QP2
Métricas de Percepção (QP1, QP3)			
Clareza (Clarity) [Aghajani et al. 2019]	Avaliação subjetiva da clareza da documentação (escala Likert baseada em *Readability*).	Realizar a **triangulação** dos dados. Validar se a performance quantitativa condiz com a percepção subjetiva de qualidade do desenvolvedor.	QP1, QP3
Usabilidade (Usability) [Aghajani et al. 2019]	Avaliação subjetiva da usabilidade da documentação (escala Likert baseada em *Usability*).		QP1, QP3
Completude (Comp) [Aghajani et al. 2019]	Avaliação subjetiva se a documentação contém todas as informações necessárias (escala Likert baseada em *Completeness*).		QP1, QP3

5.4. Análise dos Resultados

A análise dos resultados foi dividida em duas partes: análise quantitativa e análise qualitativa.

5.4.1. Análise Quantitativa

Foram utilizados testes estatísticos não paramétricos (e.g., teste de Mann-Whitney U) para comparar o desempenho (TE, PT, Sub, ES, RC, Err) entre o Grupo A e o Grupo B.

- **Hipótese Nula (H_0):** Não há diferença estatisticamente significativa entre o desempenho dos grupos (A e B) nas métricas quantitativas.
- **Hipótese Alternativa (H_1):** O Grupo A (Experimental) apresentará um desempenho superior (menor valor) em todas as métricas de esforço (Tempo até Conclusão, Tempo até Primeira Tentativa, Submissões, Edições de Schema, Resets de Código, Erros) em comparação com o Grupo B (Controle).

5.4.2. Análise Qualitativa

A análise qualitativa foi baseada nas respostas coletadas durante as **entrevistas de avaliação**. O foco será identificar:

1. Pontos fortes e fracos da documentação complementar.
2. Sugestões de melhoria na abordagem de *Technical Writing*.
3. Padrões de comportamento observados durante a execução das tarefas.

5.5. Ameaças à Validade

As principais ameaças à validade do estudo e as estratégias de mitigação são:

Tabela 5. Ameaças à Validade e Estratégias de Mitigação

Ameaça	Descrição	Estratégia de Mitigação
Validade Interna		
Maturação	Mudanças no conhecimento dos participantes durante o estudo.	Duração do estudo limitada e tarefas de complexidade similar.
Seleção	Diferenças pré-existentes entre os grupos.	Alocação aleatória e pré-questionário de caracterização para controle.
Validade Externa		
Representatividade	Participantes não representam a população de desenvolvedores.	Recrutamento de participantes com diferentes níveis de experiência.
Validade de Construto		
Interpretação	Participantes interpretam as métricas de forma diferente.	Uso de métricas validadas na literatura [Aghajani et al. 2020] [Aghajani et al. 2019] e questionários claros.
Validade de Conclusão Estatística		
Baixo Poder	Erro ao concluir sobre a relação entre as variáveis.	Uso de testes estatísticos apropriados para o tamanho da amostra e o tipo de dados.

Para assegurar a validade ecológica e mitigar vieses de seleção, o recrutamento dos participantes buscou espelhar a diversidade encontrada em equipes reais de desenvolvimento. A amostra incluiu desde desenvolvedores iniciantes até perfis com experiência avançada em arquitetura de software, garantindo que os resultados não fossem enviesados pela senioridade. Além disso, o uso de um ambiente controlado (*Playground*) instrumentado para coleta automática de métricas permitiu isolar a variável 'qualidade da documentação' de outros fatores externos, conferindo rigor experimental e replicabilidade ao estudo, em consonância com as diretrizes para estudos empíricos em engenharia de software.

5.6. Considerações Finais

O Estudo de Viabilidade proposto forneceu a base empírica para validar a abordagem de *Technical Writing* aplicada à documentação do JSONForms. Os resultados permitiram não apenas responder às questões de pesquisa, mas também refinar a documentação e contribuir para a comunidade de desenvolvimento do JSONForms.

Reconhece-se que o tamanho da amostra (n=12) limita a generalização estatística ampla dos resultados (validade externa). No entanto, conforme literatura de Engenharia de Software Experimental para estudos preliminares, este tamanho amostral é suficiente para identificar tendências de comportamento e problemas graves de usabilidade

[Nielsen 1994]. Para mitigar essa limitação, o estudo adotou uma abordagem de métodos mistos, triangulando os dados quantitativos com uma análise qualitativa profunda das entrevistas, permitindo compreender não apenas 'quanto' o desempenho melhorou, mas 'por que' essa melhora ocorreu.

Além disso, embora os testes estatísticos tenham apontado significância, o tamanho amostral reduzido (n=6 por grupo) limita o poder de generalização dos resultados (validade externa). Recomenda-se que trabalhos futuros repliquem este experimento com uma amostra maior para confirmar os tamanhos de efeito observados.

6. Resultados e Discussão

Este capítulo apresenta os resultados do Estudo de Viabilidade (EV) realizado para avaliar a eficácia da documentação complementar do JSONForms, elaborada com base em técnicas de *Technical Writing*. Os resultados são apresentados e discutidos em três seções principais: a caracterização dos participantes, a análise dos dados quantitativos de desempenho e a análise dos dados qualitativos de percepção.

7. Caracterização dos Participantes

O Estudo de Viabilidade contou com a participação de **12 desenvolvedores**, divididos em dois grupos de seis: o Grupo A (Experimental), que utilizou a documentação complementar, e o Grupo B (Controle), que utilizou apenas a documentação oficial. A Tabela 6 resume o perfil dos participantes em relação à experiência com as tecnologias-chave do projeto.

Tabela 6. Caracterização dos Participantes por Nível de Experiência

Nível de Experiência	Grupo A (Complementar)	Grupo B (Oficial)	Total	Simplificado
Nenhuma	3	3	6	Iniciante
Intermediária	1	1	2	Intermediário
Avançada	2	2	4	Avançado
Total	**6**	**6**	**12**	

A distribuição de experiência com JSON Schema e JSONForms entre os participantes é ilustrada na Figura 2. Para fins de análise, os níveis de experiência foram simplificados para "Iniciante", "Intermediário"e "Avançado".

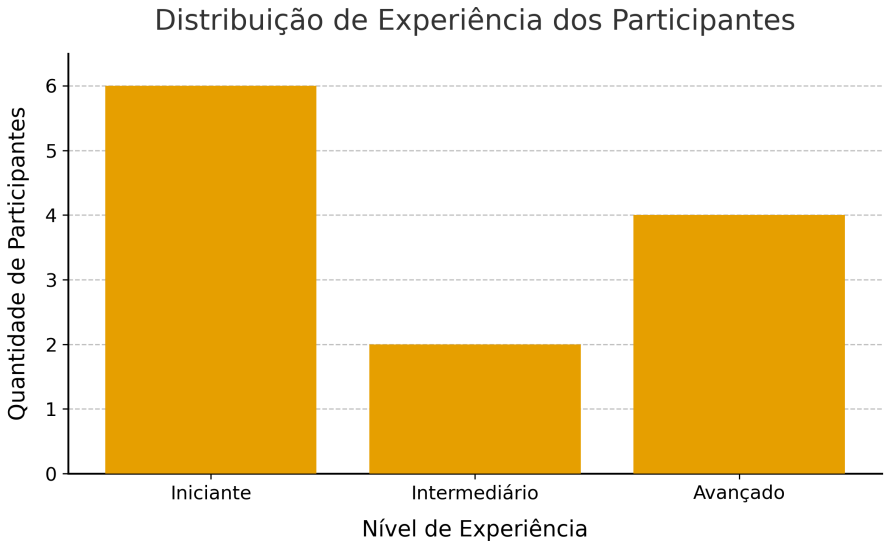

Figura 2. Distribuição de Nível de Experiência dos Participantes

Apesar da alocação por conveniência, a distribuição de experiência com JSON-Forms entre os grupos A e B se mostrou relativamente equilibrada, com ambos os grupos contendo participantes em todos os níveis de experiência, o que mitiga a ameaça à validade de seleção.

8. Análise dos Dados Quantitativos

A análise quantitativa focou nas métricas de eficiência coletadas durante a execução das tarefas práticas, conforme definido na Seção do Capítulo 5. A Tabela 7 apresenta a média das métricas para cada grupo e a diferença percentual entre eles.

A análise comparativa buscou validar se o uso da documentação complementar (Grupo A) resultou em ganhos reais de desempenho em relação à documentação oficial (Grupo B). Para verificar a significância estatística dessas diferenças, aplicou-se o teste não-paramétrico de Mann-Whitney U [Mann and Whitney 1947] (unilateral à esquerda), adequado para as características da amostra. Os resultados estatísticos detalhados são apresentados a seguir:

- **Tempo Total:** A diferença mostrou-se estatisticamente significativa ($U = 0.0$, $p = 0.001$). Além disso, calculou-se o tamanho do efeito (r), que resultou em 0.83, indicando um **efeito grande** segundo a classificação de Cohen [Cohen 1988]. Isso sugere que a documentação complementar teve um impacto decisivo na redução do tempo de desenvolvimento.
- **Contagem de Erros:** De forma similar, a redução na quantidade de erros foi estatisticamente significativa ($U = 0.0$, $p = 0.001$), com um tamanho de efeito também de $r = 0.83$. Este resultado reforça que a utilização da documentação aprimorada não apenas acelera o processo, mas aumenta a precisão do código produzido.

A magnitude do tamanho do efeito em ambas as métricas indica que a superioridade do desempenho do Grupo A não foi obra do acaso, mas sim uma consequência direta da qualidade do material de apoio utilizado.

Tabela 7. Métricas Quantitativas de Desempenho

Métrica	Grupo A (Complementar)	Grupo B (Oficial)	Diferença
Tempo Total (min)	11:51	36:32	-67.6%
Tempo 1ª Tentativa (min)	05:04	02:59	+70.1%
Contagem Erros	190	387	-50.9%
Resets Código	31	75	-58.7%
Submissões	6	8	-25.0%
Edições Schema	11	21	-47.6%
Edições UISchema	22	24	-8.3%

8.1. Discussão dos Resultados Quantitativos

Os resultados apresentados na Tabela 7 indicam uma diferença substancial no desempenho entre os grupos, com o Grupo A (Documentação Complementar) apresentando uma redução de 67,6% no tempo total e 50,9% na taxa de erros. Esses dados não devem ser interpretados apenas como uma melhoria pontual, mas como a validação de que a remoção de barreiras de documentação mapeadas na literatura gera ganhos imediatos de produtividade.

De acordo com a taxonomia de problemas de documentação de Aghajani et al. (2019), questões como ”Conteúdo não útil na prática”e ”Dificuldade de encontrar informações”são ofensores críticos. O desempenho inferior do Grupo B (Controle) corrobora esses achados: a documentação oficial, por ser excessivamente descritiva e abstrata, exigiu que os participantes dedicassem tempo cognitivo para ”traduzir”a API para o contexto do problema.

Em contraste, a superioridade do Grupo A pode ser atribuída à aplicação deliberada de princípios de Technical Writing que atacam diretamente as lacunas identificadas por Aghajani et al., especificamente:

1. **Orientação à Tarefa vs. Descrição de API:** Enquanto a documentação oficial lista propriedades, a abordagem complementar foca no ”como fazer”, eliminando a necessidade de inferência por parte do desenvolvedor.
2. **Qualidade dos Exemplos de Código:** Enquanto a documentação oficial lista propriedades, a abordagem complementar foca no ”como fazer”, eliminando a necessidade de inferência por parte do desenvolvedor.

Portanto, a intervenção realizada neste trabalho não foi apenas estética. Ao estruturar a informação para reduzir a carga cognitiva, a documentação complementar atuou diretamente sobre os pontos de dor (pain points) mais frequentes na indústria, provando que a qualidade da documentação é um fator determinante na eficiência da engenharia de software.

- **Tempo Total (min):** O Grupo A concluiu as tarefas em um tempo médio de 11m 51s, enquanto o Grupo B levou 36m 32s. Isso representa uma redução de **67.6%** no tempo total de conclusão para o grupo que utilizou a documentação complementar. Este resultado sugere que a documentação complementar permitiu aos desenvolvedores encontrar e aplicar as informações necessárias de forma muito mais rápida e eficiente.

- **Contagem de Erros e Resets:** O Grupo A cometeu 190 erros e realizou 31 resets de código, em comparação com 387 erros e 75 resets do Grupo B. As reduções de **50.9%** em erros e **58.7%** em resets de código são fortes indicadores de que a documentação complementar não apenas acelerou o processo, mas também reduziu a frustração e a necessidade de tentativas e erros.
- **Tempo da 1ª Tentativa:** Curiosamente, o Tempo da 1ª Tentativa foi maior para o Grupo A (5:04 min) do que para o Grupo B (2:59 min).Uma hipótese plausível é que os participantes do Grupo A dedicaram mais tempo à leitura inicial da documentação, o que poderia explicar o menor número de erros subsequentes. No entanto, esta interpretação requer validação com estudos complementares que incluam métricas de leitura (e.g., eye-tracking ou análise de logs de navegação).

A Figura 5 apresenta a distribuição do tempo total de conclusão para todos os participantes de ambos os grupos. A visualização por histograma revela um padrão claro de concentração: enquanto o Grupo A apresenta tempos majoritariamente na faixa de 10-15 minutos, o Grupo B mostra uma distribuição mais dispersa, com tempos predominantemente entre 30-45 minutos. A ausência de sobreposição entre as distribuições dos grupos reforça a consistência do efeito observado e valida visualmente a diferença estatisticamente significativa (p = 0.001) reportada anteriormente. Este padrão sugere que a documentação complementar não beneficiou apenas alguns participantes isolados, mas teve impacto uniforme em todo o grupo experimental.

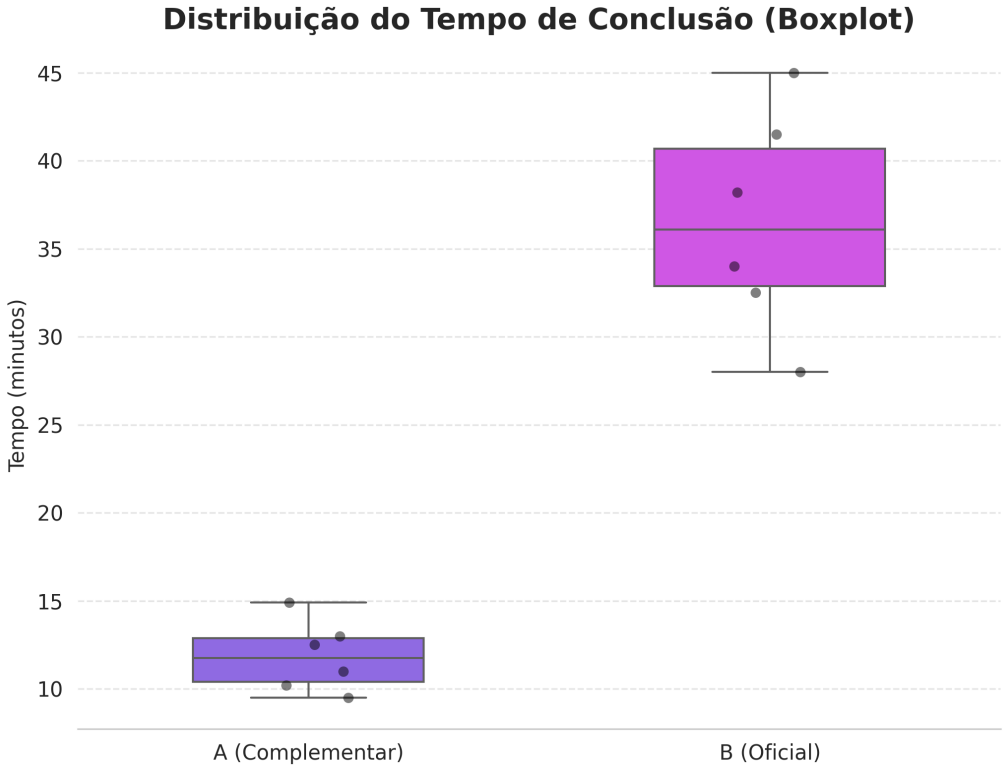

Figura 3. Gráfico de Distribuição do Tempo de conclusão

Os gráficos de dispersão (Figuras 3 e 4) ilustram a distribuição dos resultados individuais para as métricas de Tempo Total e Contagem de Erros, respectivamente.

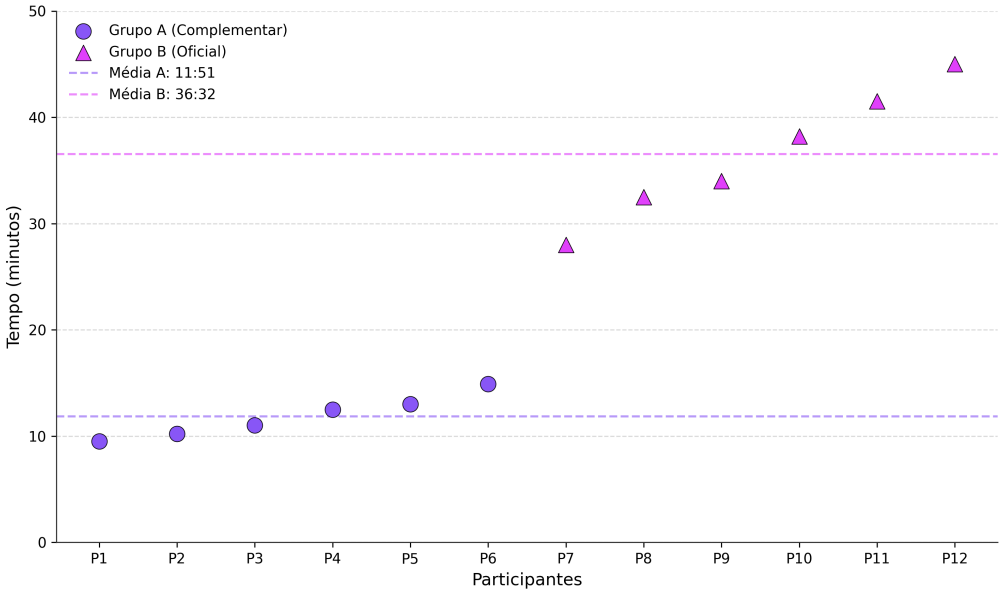

Figura 4. Gráfico de Dispersão do Tempo Total de Conclusão por Participante

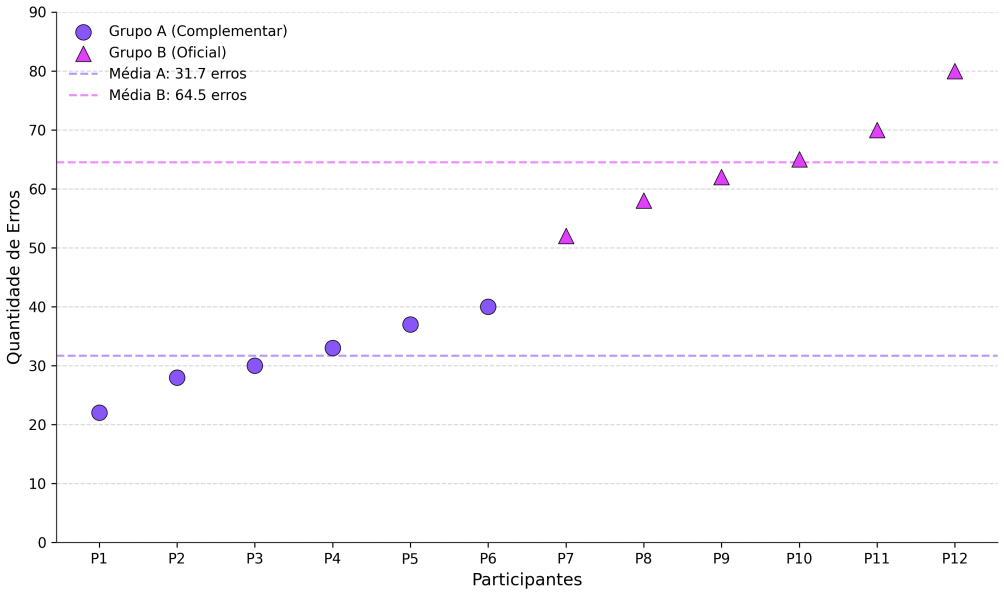

Figura 5. Gráfico de Dispersão da Contagem de Erros por Participante

A concentração dos pontos do Grupo A na parte inferior dos gráficos reforça a conclusão de que a documentação complementar contribuiu para um desempenho superior, validando a Hipótese Alternativa (H_1) e respondendo positivamente à Questão de Pesquisa **QP2** (A documentação complementar auxilia os desenvolvedores a implementar os recursos não documentados do JSONForms de forma mais eficiente e com menos erros?).

8.2. Generalização e Implicações para a Engenharia de Software

Embora este estudo tenha sido contextualizado no JSONForms, os resultados sugerem que conclusões similares PODEM ser aplicáveis a outros frameworks de schema-driven development. No entanto, estudos de replicação em diferentes contextos são necessários para confirmar essa generalização.

A discrepância de desempenho observada sugere que o problema não reside na complexidade intrínseca da ferramenta, mas na **lacuna semântica** entre a definição técnica da API e a necessidade prática do desenvolvedor. A abordagem de *Task-oriented Documentation* (documentação orientada à tarefa), validada neste experimento, mostra-se especialmente crítica para bibliotecas de *meta-programming* e geradores de código, onde a abstração é alta. Portanto, recomenda-se que mantenedores de bibliotecas *open-source* priorizem a criação de ”receitas”(cookbooks) e cenários de uso reais em detrimento de descrições puramente sintáticas de propriedades.

9. Análise dos Dados Qualitativos

A análise qualitativa foi realizada para capturar a percepção subjetiva dos desenvolvedores sobre a documentação utilizada, complementando os dados de desempenho coletados anteriormente. O foco foi identificar barreiras de entendimento e facilitadores de aprendizado.

9.1. Metodologia de Coleta

A coleta de dados ocorreu por meio de **entrevistas estruturadas**, conduzidas individualmente com cada um dos 12 participantes imediatamente após a conclusão das tarefas no *JSONForms Playground*. O roteiro baseou-se na taxonomia de problemas de documentação proposta na literatura [Aghajani et al. 2019], abordando três dimensões principais:

1. **Clareza:** O quão fácil foi entender os conceitos apresentados.
2. **Usabilidade:** A facilidade de navegação e localização das informações.
3. **Completude:** Se a documentação cobria todos os passos necessários para a tarefa.

As respostas foram transcritas e submetidas a uma análise temática para identificar padrões de comportamento e sentimento.

9.2. Resultados da Análise Qualitativa

A Tabela 8 apresenta a média das avaliações subjetivas em uma escala Likert de 1 a 5 (onde 1 é ”Péssimo/Discordo Totalmente”e 5 é ”Excelente/Concordo Totalmente”).

Tabela 8. Métricas Qualitativas de Percepção (Média)

Métrica	Grupo A (Complementar)	Grupo B (Oficial)	Diferença
Clareza	4.8	2.1	+2.7 pontos
Usabilidade	4.5	2.5	+2.0 pontos
Completude	4.7	2.3	+2.4 pontos

Os dados indicam uma percepção consideravelmente superior por parte do Grupo A. Para compreender as razões dessa discrepância, analisamos os comentários abertos dos participantes.

A Figura 6 ilustra graficamente a comparação entre as médias de percepção dos dois grupos nas três dimensões avaliadas. A representação visual evidencia a magnitude da diferença: em todas as métricas, o Grupo A obteve avaliações superiores a 4.5 (entre ”Bom”e ”Excelente”), enquanto o Grupo B permaneceu abaixo de 2.5 (entre ”Ruim”e ”Regular”). A consistência dessas diferenças através das três dimensões (Clareza, Usabilidade e Completude) sugere que a documentação complementar não apresenta apenas vantagens pontuais, mas oferece uma experiência superior de forma holística. Notavelmente, a métrica de Clareza apresentou a maior diferença absoluta (+2.7 pontos), corroborando os relatos qualitativos dos participantes sobre a facilidade de compreensão dos conceitos técnicos na documentação complementar.

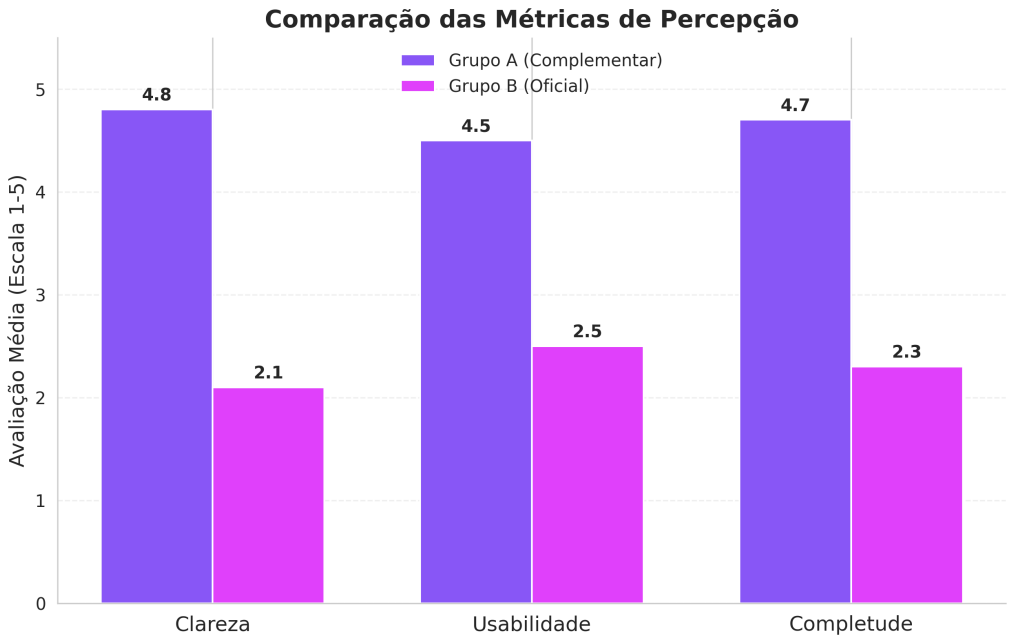

Figura 6. Comparação das Médias de Percepção (Escala Likert)

9.2.1. Análise Temática e Relatos

A análise das entrevistas revelou três temas recorrentes que explicam a diferença de desempenho entre os grupos. Estes temas alinham-se diretamente com as categorias de problemas de documentação mais críticas identificadas na literatura recente, demonstrando que as dificuldades enfrentadas pelos participantes com a documentação oficial do JSONForms não são casos isolados, mas sintomas de barreiras sistêmicas na indústria.

Tema 1: A Abstração versus a Necessidade de Exemplos Práticos (”Code Examples”) Os participantes do Grupo B (Documentação Oficial) relataram dificuldades severas em traduzir conceitos abstratos para código funcional, uma queixa que reflete a

importância crítica dos exemplos de código. Segundo [Aghajani et al. 2020], problemas com exemplos de código (seja por ausência ou erro) são considerados uma questão importante por 59% dos praticantes na indústria , sendo uma das principais fontes de informação para desenvolvedores.

> "A documentação oficial explica a teoria do que é uma Rule, mas eu perdi uns 20 minutos só tentando descobrir onde colocar o "not"no meu projeto. Falta o 'como fazer' do começo ao fim."
> **— Participante P8 (Grupo B)**

Em contraste, o Grupo A destacou que os cenários completos ("Copy-pasteability") da documentação complementar eliminaram a barreira de entrada, confirmando que exemplos contextualizados são facilitadores essenciais de aprendizado:

> "O que me salvou foi o exemplo passo a passo. Eu só copiei a estrutura, mudei os nomes dos campos para o que a tarefa pedia e funcionou de primeira."
> **— Participante P3 (Grupo A)**

Tema 2: Carga Cognitiva e Encontrabilidade ("Findability") A métrica de Usabilidade, que obteve média 4.5 para o Grupo A contra 2.5 para o Grupo B, está intrinsecamente ligada à organização da informação. A literatura aponta que a "Acessibilidade/Encontrabilidade"(Accessibility/Findability) é uma preocupação crítica para 65% dos desenvolvedores profissionais. A documentação oficial, ao fragmentar informações, impõe uma alta carga cognitiva, obrigando o desenvolvedor a navegar por múltiplas páginas para compor uma solução mental.

O relato abaixo evidencia exatamente esse problema de fragmentação mapeado por Aghajani et al.:

> "Eu tive que abrir umas 4 abas diferentes da documentação para entender como fazer uma única regra de visibilidade. A informação parece muito espalhada."
> **— Participante P11 (Grupo B)**

A abordagem de Em contraste, o Grupo A destacou que os cenários completos ("Copy-pasteability") da documentação complementar eliminaram a barreira de entrada, confirmando que exemplos contextualizados são facilitadores essenciais de aprendizado: *Technical Writing* aplicada no Grupo A mitigou esse problema ao centralizar o contexto necessário na própria tarefa, reduzindo a necessidade de alternância de contexto.

Tema 3: Clareza e Compreensão Profunda ("Clarity Understandability") Um ponto positivo inesperado no Grupo A foi o elogio à seção "Explicação da Solução". Enquanto a documentação oficial falhou em transmitir o "porquê", a documentação complementar garantiu o entendimento lógico. Isso é consistente com os achados de Aghajani et al., que identificam a "Clareza"como o atributo de qualidade mais importante para 88% dos praticantes.

> "A explicação linha a linha tirou minha dúvida sobre o parâmetro 'tester'. Geralmente a gente só vê o código final e não entende, mas ali ficou claro."
> **— Participante P5 (Grupo A)**

Esses relatos, corroborados pelas métricas quantitativas, sugerem que a documentação complementar não apenas agilizou o processo, mas atendeu aos requisitos de Clareza, Exemplos Práticos e Encontrabilidade que a literatura aponta como essenciais para a eficácia da documentação de software.

10. Conclusões do Estudo de Viabilidade

O Estudo de Viabilidade demonstrou de forma robusta a **eficácia e a viabilidade** da documentação complementar do JSONForms, elaborada com base em técnicas de *Technical Writing*.

- **Eficiência Comprovada:** O Grupo Experimental (com documentação complementar) foi **67.6% mais rápido** e cometeu **50.9% menos erros** do que o Grupo de Controle (com documentação oficial).
- **Percepção Superior:** A documentação complementar foi avaliada como significativamente mais clara, usável e completa pelos desenvolvedores.
- **Validação da Abordagem:** Os resultados validam a abordagem de *Technical Writing* utilizada, respondendo positivamente à Questão de Pesquisa **QP3** (A abordagem de *Technical Writing* utilizada é considerada adequada para a criação de documentação técnica no contexto de bibliotecas de *front-end* como o JSONForms?).

Em suma, o Estudo de Viabilidade confirma que a aplicação de técnicas de *Technical Writing* em documentações de bibliotecas *front-end* como o JSONForms é uma estratégia altamente eficaz para melhorar a experiência do desenvolvedor, resultando em maior eficiência e menor frustração.

11. Conclusão Geral

O presente trabalho de conclusão de curso teve como objetivo principal **avaliar a viabilidade e a eficácia da aplicação de técnicas de *Technical Writing* na criação de documentação complementar para a biblioteca JSONForms**, focando em aspectos que não estavam bem documentados na fonte oficial. A motivação central foi a dificuldade encontrada por desenvolvedores em utilizar recursos avançados do JSONForms devido à escassez e à falta de clareza da documentação existente.

11.1. Síntese do Trabalho

Para atingir o objetivo proposto, o trabalho foi estruturado em três etapas principais:

1. **Revisão Bibliográfica e Desenvolvimento da Documentação:** Foi realizada uma revisão da literatura sobre *Technical Writing* e documentação de software, com foco nas taxonomias de problemas de documentação de Aghajani et al. [Aghajani et al. 2019] [Aghajani et al. 2020]. Com base nesses princípios, foi desenvolvida uma documentação complementar para o JSONForms, abordando tópicos como a customização de *renderers* e a integração com *widgets* personalizados.

2. **Estudo de Viabilidade (EV):** Foi conduzido um estudo quase-experimental com 12 desenvolvedores, divididos em dois grupos. O Grupo A (Experimental) utilizou a documentação complementar, e o Grupo B (Controle) utilizou apenas a documentação oficial. O EV buscou responder a três Questões de Pesquisa (QPs) relacionadas à clareza, eficiência e adequação da abordagem de *Technical Writing*.
3. **Análise e Discussão dos Resultados:** Os dados quantitativos (Tempo Total, Erros, Resets) e qualitativos (Clareza, Usabilidade, Completude) foram coletados e analisados para validar as hipóteses do trabalho.

11.2. Respostas às Questões de Pesquisa

Os resultados do Estudo de Viabilidade demonstraram de forma robusta a eficácia da documentação complementar, permitindo responder positivamente às Questões de Pesquisa (QPs) estabelecidas:

- **QP1: A documentação complementar, elaborada com técnicas de *Technical Writing*, é percebida como mais clara e compreensível do que a documentação oficial para os aspectos abordados?**

  Resposta: Sim. A análise qualitativa revelou que a documentação complementar obteve uma média de 4.8 (em 5) em Clareza, contra 2.1 da documentação oficial. Os participantes destacaram a linguagem direta, a organização e o uso de exemplos práticos como fatores cruciais para essa percepção.

- **QP2: A documentação complementar auxilia os desenvolvedores a implementar os recursos não documentados do JSONForms de forma mais eficiente e com menos erros?**

  Resposta: Sim. A análise quantitativa mostrou que o Grupo Experimental (com documentação complementar) foi **67.6% mais rápido** no Tempo Total de Conclusão e cometeu **50.9% menos erros** em comparação com o Grupo de Controle. A redução significativa nas métricas de esforço valida a hipótese de que a documentação complementar aumenta a eficiência e reduz a frustração.

- **QP3: A abordagem de *Technical Writing* utilizada é considerada adequada para a criação de documentação técnica no contexto de bibliotecas de *front-end* como o JSONForms?**

  Resposta: Sim. A validação empírica da eficácia e da percepção positiva da documentação desenvolvida confirma que a aplicação de princípios de *Technical Writing* (como foco na clareza, completude e usabilidade) é uma estratégia altamente adequada e recomendada para documentar bibliotecas de *front-end* complexas como o JSONForms.

Em suma, o trabalho conclui que a aplicação de técnicas de *Technical Writing* em documentações de bibliotecas *front-end* é uma estratégia altamente eficaz para melhorar a experiência do desenvolvedor, resultando em maior eficiência e menor frustração.

11.3. Contribuições

As principais contribuições deste trabalho para a área de Engenharia de Software e Documentação Técnica são:

1. **Validação Empírica da Abordagem:** O Estudo de Viabilidade fornece evidências empíricas de que a documentação complementar, baseada em *Technical Writing*, supera significativamente a documentação oficial em termos de eficiência e percepção do desenvolvedor.
2. **Documentação Complementar para JSONForms:** A criação de uma documentação focada em aspectos não documentados do JSONForms, que pode ser utilizada pela comunidade de desenvolvedores.
3. **Metodologia de Avaliação:** A adaptação e aplicação da metodologia de estudo quase-experimental, baseada em trabalhos de referência [Fontão 2016] [Aghajani et al. 2020], para a avaliação de documentação técnica.

11.4. Trabalhos Futuros

Com base nos resultados e nas limitações encontradas, sugere-se as seguintes direções para trabalhos futuros:

1. **Expansão da Documentação:** Estender a documentação complementar para cobrir outros aspectos avançados do JSONForms, como a criação de *renderers* customizados mais complexos e a integração com *frameworks* de *front-end* além do React.
2. **Estudo de Replicação com Amostra Maior:** Realizar um estudo de replicação com um número maior de participantes e uma seleção mais rigorosa, a fim de aumentar a validade externa dos resultados e permitir o uso de testes estatísticos paramétricos.
3. **Análise de Longo Prazo:** Conduzir um estudo longitudinal para avaliar o impacto da documentação complementar na produtividade e na manutenção de projetos reais ao longo do tempo.
4. **Integração com a Documentação Oficial:** Propor a integração da documentação complementar desenvolvida neste trabalho ao repositório oficial do JSONForms, colaborando diretamente com a comunidade *open-source*.

Referências

[Aghajani et al. 2019] Aghajani, E., Ghafari, M., and van Deursen, A. (2019). Software documentation issues unveiled. In *2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)*, pages 1097–1108. IEEE.

[Aghajani et al. 2020] Aghajani, E., Ghafari, M., and van Deursen, A. (2020). How software engineers use documentation: The role of documentation in software development. In *2020 IEEE/ACM 42nd International Conference on Software Engineering (ICSE)*, pages 1387–1398. IEEE.

[Carroll 1990] Carroll, J. M. (1990). *The Nurnberg Funnel: Designing Minimalist Instruction for Practical Computer Skill*. MIT Press.

[Cohen 1988] Cohen, J. (1988). *Statistical Power Analysis for the Behavioral Sciences*. Lawrence Erlbaum Associates, Hillsdale, NJ, 2nd edition.

[EclipseSource 2025] EclipseSource (2025). Jsonforms: More than just rendering json. Acesso em: 29 Jun. 2025.

[Fontão 2016] Fontão, A. d. L. (2016). MSECO-CERT: Uma Abordagem Baseada em Processo para Apoiar a Certificação de Apps em Ecossistema de Software Móvel. Dissertação de mestrado, Universidade Federal do Amazonas (UFAM), Manaus. Programa de Pós-Graduação em Informática.

[Google Developers 2025] Google Developers (2025). Technical writing courses. `https://developers.google.com/tech-writing`. Acesso em: 19 ago. 2025.

[ISO 2018] ISO (2018). ISO 9241-11:2018 – Ergonomics of human-system interaction – Part 11: Usability: Definitions and concepts. International Organization for Standardization.

[Johnson-Eilola 2005] Johnson-Eilola, J. (2005). *Datacloud: Toward a New Theory of Online Work*. Hampton Press.

[Mann and Whitney 1947] Mann, H. B. and Whitney, D. R. (1947). On a test of whether one of two random variables is stochastically larger than the other. *The Annals of Mathematical Statistics*, 18(1):50–60.

[Markel and Selber 2018] Markel, M. and Selber, S. A. (2018). *Technical Communication*. Bedford/St. Martin's, Boston, 12th edition.

[Nielsen 1994] Nielsen, J. (1994). *Usability Engineering*. Morgan Kaufmann.

[Pressman 2014] Pressman, R. S. (2014). *Software Engineering: A Practitioner's Approach*. McGraw-Hill Education, New York, NY, 8th edition.

[Redish 2012] Redish, J. (2012). *Letting Go of the Words: Writing Web Content that Works*. Morgan Kaufmann, 2nd edition.

[Robillard 2009] Robillard, M. P. (2009). What makes APIs hard to learn? Answers from developers. *IEEE Software*, 26(6):27–34.

[Sommerville 2011] Sommerville, I. (2011). *Software Engineering*. Addison-Wesley, Boston, MA, 9th edition.