Universidade Federal do Amazonas
Instituto de Computação
Programa de Pós-Graduação em Informática

Gretchen Torres de Macedo

Uma Abordagem Assistida de Inspeção Funcional de Protótipos de Interface de Software

Manaus
Setembro de 2025

Gretchen Torres de Macedo

Uma Abordagem Assistida de Inspeção Funcional de Protótipos de Interface de Software

Tese apresentada ao Programa de Pós-Graduação em Informática do Instituto de Computação da Universidade Federal do Amazonas como requisito para obtenção do grau de Doutor em Informática.
Área de concentração: Engenharia de Software

Orientador: Prof. Dr. Bruno Freitas Gadelha
Coorientador: Prof. Dr. Awdren de Lima Fontão

Manaus
2025

Ficha Catalográfica

Elaborada automaticamente de acordo com os dados fornecidos pelo(a) autor(a).

M141a Macedo, Gretchen Torres de
Uma Abordagem Assistida de Inspeção Funcional de Protótipos de Interface de Software / Gretchen Torres de Macedo. - 2025.
252 f. : il., color. ; 31 cm.

Orientador(a): Bruno Freitas Gadelha.
Coorientador(a): Awdren de Lima Fontão.
Tese (doutorado) - Universidade Federal do Amazonas, Programa de Pós-Graduação em Informática, Manaus, 2025.

1. Inspeção Funcional. 2. Prototipação de Interface de Usuário. 3. Inspeção de Interface de Usuário. 4. Heurísticas. I. Gadelha, Bruno Freitas. II. Fontão, Awdren de Lima. III. Universidade Federal do Amazonas. Programa de Pós-Graduação em Informática. IV. Título

---

Ministério da Educação
Universidade Federal do Amazonas
Coordenação do Programa de Pós-Graduação em Informática

FOLHA DE APROVAÇÃO

"UMA ABORDAGEM ASSISTIDA DE INSPEÇÃO FUNCIONAL DE PROTÓTIPOS DE INTERFACE DE SOFTWARE"

GRETCHEN TORRES DE MACEDO

Tese de Doutorado defendida e aprovada pela banca examinadora constituída pelos professores:

Prof. Dr. Bruno Freitas Gadelha - **Presidente**

Documento assinado digitalmente
gov.br EDNA DIAS CANEDO
Data: 23/09/2025 08:05:14-0300
Verifique em https://validar.iti.gov.br

Profa. Dra. Edna Dias Canedo - **Membro Externo**

Documento assinado digitalmente
gov.br Gleison dos Santos Souza
Data: 02/10/2025 17:29:18-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Gleison dos Santos Souza - **Membro Externo**

Documento assinado digitalmente
gov.br IGOR FABIO STEINMACHER
Data: 22/09/2025 14:58:30-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Igor Steinmacher - **Membro Externo**

Documento assinado digitalmente
gov.br RODRIGO PEREIRA DOS SANTOS
Data: 04/10/2025 20:46:35-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Rodrigo Pereira dos Santos - **Membro Externo**

Manaus, 10 de setembro de 2025.

sei! assinatura eletrônica

Documento assinado eletronicamente por **Bruno Freitas Gadelha**, **Professor do Magistério Superior**, em 17/09/2025, às 14:22, conforme horário oficial de Manaus, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015 .

A autenticidade deste documento pode ser conferida no site https://sei.ufam.edu.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **2760546** e o código CRC **CC238F39**.

Avenida General Rodrigo Octávio, 6200 - Bairro Coroado I Campus Universitário Senador Arthur Virgílio Filho, Setor Norte - Telefone: (92) 3305-1181 / Ramal 1193
CEP 69080-900, Manaus/AM, coordenadorppgi@icomp.ufam.edu.br

Para Claudemir, Getúlio e Isadora, que são o meu mundo inteiro.

Busque a liberdade e torne-se cativo de seus desejos.
Busque a disciplina e encontre sua liberdade.

Frank Herbert

AGRADECIMENTOS

Aproveito este espaço para registrar os meus mais sinceros agradecimentos às pessoas que, de uma forma ou de outra, me ajudaram, me apoiaram ou me incentivaram durante esta jornada de transformação pessoal que eu estou vivenciando. Em especial, gostaria de agradecer:

Ao meu esposo, Claudemir Ivan, por todo incentivo desde o primeiro dia em que eu cogitei fazer doutorado, por sempre me ouvir durante os momentos de questionamentos e pela paciência e amor infinitos.

Aos meus filhos, Getúlio e Isadora, maiores fontes de amor e inspiração da minha vida, que me fazem sempre saber diferenciar as coisas importantes das sem importância.

Ao meu orientador e amigo de longa data, Prof. Dr. Bruno Gadelha, pelo privilégio de poder tê-lo ao meu lado nessa fase tão transformadora da minha vida.

Ao meu coorientador, Prof. Dr. Awdren Fontão, por todas as suas contribuições para a realização deste trabalho.

À Professora Dra. Tayana Conte, pela sua imensa generosidade em compartilhar seus conhecimentos e experiência e por ser sempre tão inspiradora.

Aos amigos Athos, Leomar e Thaís, pessoas que amo e admiro profundamente.

Ao meu pai, Wilmar, a pessoa que eu mais admiro pela inteligência e percepção aguçada, e às minhas irmãs, Ingrid e Sheron, mulheres inteligentes e guerreiras, com as quais tenho a honra de poder compartilhar minha vida.

A todos os amigos do grupo USES, sempre companheiros e dispostos a ajudar nos momentos difíceis.

Aos membros da banca examinadora, Professores Edna Canedo, Gleison Santos, Igor Steinmacher, Rodrigo Santos, Tayana Conte e Márcia Lima, que gentilmente aceitaram o convite para participar desta defesa de doutorado. Agradeço pela disponibilidade e generosidade em dedicar seu tempo à leitura e discussão da tese. Suas considerações foram fundamentais para o aprimoramento do trabalho e para o meu crescimento como pesquisadora.

Aos participantes voluntários dos estudos que integram esta pesquisa, pela colaboração, pelo tempo e pela dedicação.

À Universidade Federal do Amazonas, ao Instituto de Computação e ao programa de pós-graduação em informática da UFAM (PPGI/UFAM).

Ao Ministério Público do Estado do Amazonas, pelo apoio concedido através do programa de aperfeiçoamento funcional de servidores.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001. Este trabalho foi parcialmente financiado pela Fundação de Amparo à Pesquisa do Estado do Amazonas – FAPEAM – por meio do projeto POSGRAD.

Resumo

A prototipação de interface de usuário (IU) é amplamente utilizada no desenvolvimento de software para explorar soluções, demonstrar e validar funcionalidades, servindo também como guia para o design, a implementação e os testes. Para apoiar a identificação precoce de defeitos funcionais e evitar retrabalho, é essencial incluir protótipos em inspeções de qualidade que verifiquem se contêm elementos necessários à representação de funcionalidades. No entanto, as técnicas existentes de inspeção de IU concentram-se, em sua maioria, na avaliação de usabilidade ou de problemas de design, sendo aplicadas a protótipos interativos ou interfaces já implementadas. A carência de abordagens de inspeção funcional de protótipos de IU motivou a questão de pesquisa desta tese: **(QP)** - *Como inspecionar protótipos de interface de usuário em relação à sua funcionalidade?* Para respondê-la, utilizou-se a *Design Science Research* (DSR) no desenvolvimento de uma abordagem de inspeção funcional de protótipos de IU que produziu dois instrumentos de inspeção: (i) um catálogo de heurísticas de inspeção funcional (CHIF), que descreve funcionalidades comumente representadas em protótipos de IU, elaborado a partir da análise de protótipos de interface e de capturas de telas de aplicações; (ii) um *checklist* de propósito geral (UIProtoCheck), voltado para para inspeção de protótipos de funcionalidades não caracterizadas pelas heurísticas. Para apoiar a realização de inspeções funcionais, foi desenvolvida a ferramenta GUIPFI (*Guided User Interface Prototype Functional Inspection*), que integra o CHIF e o UIProtoCheck com recursos de recomendação de instrumentos de inspeção, anotação de defeitos e geração de relatórios. Os artefatos elaborados foram avaliados através de um *focus group* com especialistas, um experimento controlado e estudos de inspeção funcional de protótipos de IU na academia e com profissionais da indústria de software. As avaliações mostraram evidências empíricas da eficácia e utilidade da abordagem na identificação de defeitos funcionais em protótipos de IU, contribuindo para a teoria e a prática da Engenharia de Software.

Palavras-chave: Inspeção Funcional, Prototipação de Interface de Usuário, Inspeção de Interface de Usuário, Heurísticas.

Abstract

User interface (UI) prototyping is widely used in software development to explore solutions, demonstrate, and validate functionality, and also serves as a guide for design, implementation, and testing. To support the early identification of functional defects and avoid rework, it is essential to include prototypes in quality inspections that verify they contain all the necessary elements to represent their intended functionality. However, existing UI inspection techniques primarily focus on assessing usability or design issues and are typically applied to interactive prototypes or already implemented interfaces. The lack of functional inspection approaches for UI prototypes motivated the research question of this thesis: **(QP)** - *How to inspect user interface prototypes regarding their functionality?* To answer it, *Design Science Research* (DSR) was used in the development of a functional inspection approach for UI prototypes that produced two inspection tools: (i) a catalog of functional inspection heuristics (CHIF), which describes functionalities commonly represented in UI prototypes, developed from the analysis of interface prototypes and application screenshots; (ii) a general-purpose *checklist* UIProtoCheck, aimed at inspecting prototypes of functionalities not characterized by heuristics. To support functional inspections, the GUIPFI (Guided User Interface Prototype Functional Inspection) tool was developed. It integrates CHIF and UIProtoCheck, featuring inspection tool recommendations, defect annotation, and report generation. The developed artifacts were evaluated through a focus group with experts, a controlled experiment, and functional inspection studies of UI prototypes in academia and with software industry professionals. The evaluations demonstrated empirical evidence of the approach's effectiveness and usefulness in identifying functional defects in UI prototypes, contributing to the theory and practice of Software Engineering.

Keywords: Functional Inspection, User Interface Prototyping, User Interface Inspection, Heuristics.

Lista de Figuras

1.1 Ciclos de pesquisa baseados na DSR 6

1.2 Visual Abstract da abordagem de inspeção funcional de protótipos de interface. 8

3.1 Papéis dos participantes nos times de desenvolvimento. 33

3.2 Tempo de experiência dos participantes. 34

3.3 Finalidade das empresas dos participantes. 34

3.4 QP1 - Finalidade de uso da prototipação - protótipos de baixa fidelidade. 35

3.5 QP1 - Finalidade de uso da prototipação - protótipos de alta fidelidade. 37

3.6 QP2 - Prototipação no processo de desenvolvimento adotado pelas empresas. 39

3.7 QP3 - Códigos relacionados às atividades de qualidade de software com uso de prototipação. 43

4.1 Abordagem de ensino remoto de Engenharia de Requisitos com desenvolvimento de soft skills. 56

4.2 Método para desenvolvimento de heurísticas de usabilidade e UX (Fonte: Quiñones et al. (2018)). 57
4.3 CHIF - página inicial 67
4.4 CHIF - heurística de Carrinho de Compras 68
4.5 Exemplos de protótipos de Funcionalidades Básicas – Lista (a) e Menu (b). 70
4.6 Exemplos de protótipos do tipo Funcional – Registro de Conta (a) e Autenticação (b). 70

5.1 Processo para anotação de *datasets* com atividades de garantia da qualidade. Fonte: Klie et al. (2024). 82
5.2 Processo de anotação adaptado para contemplar atividades de expansão do catálogo de heurísticas. 83
5.3 Funcionalidades do CHIF e EnRICO. 86
5.4 Exemplos de instâncias do tópico *Dialer*. (a) Calculadora. (b) Botões de elevador. (c) Criação de PIN. 88
5.5 Análise de instância. (a) Uma captura de tela de *Lista* reclassificada como *Configurações*. (b) Uma captura de tela de *Configurações* reclassificada como *Visualização*. 90
5.6 Exemplos de protótipos de interface classificados como Configurações. 91
5.7 Protótipo de IU usado no estudo - Página de Busca. 101
5.8 Natureza dos defeitos encontrados por instrumento de inspeção – CHIF-E e UIProtoCheck. 111

6.1 GUIPFI - diagrama de casos de uso. 125
6.2 GUIPFI - Aplicação de suporte à inspeção de protótipos de interface. 126

6.3 GUIPFI - Arquitetura da ferramenta GUIPFI. 127
6.4 GUIPFI - Dados informados para o Recomendador. 128
6.5 Funcionalidade de Recomendação 128
6.6 GUIPFI - Recomendador de instrumentos de inspeção. 131
6.7 Funcionamento do Assistente de Inspeção. 131
6.8 GUIPFI - Heurísticas *Lista* e *consulta* selecionadas como instrumentos de inspeção. 132
6.9 GUIPFI - Relatório de inspeção inicial. 132
6.10 Exemplos de protótipos de interface e *screenshots* utilizados na avaliação das sugestões de heurísticas do LLM. 133
6.11 Exemplo de protótipo que representa mais de uma funcionalidade (Menu, Lista e Busca). 134
6.12 Exemplo de protótipo (*Carrinho de Compra*) que recebeu sugestões de heurísticas de funcionalidades não presentes na tela, mas que possivelmente fariam parte da aplicação. 135
6.13 Ciclo 3 - Ocupação dos participantes. 142
6.14 Ciclo 3 - Tempo de experiência na indústria de software. 142
6.15 Ciclo 3 - Área de atuação na indústria de software. 143
6.16 Ciclo 3 - Análise estatística das inspeções funcionais de protótipos de interface. 144
6.17 Utilidade percebida. 145
6.18 Utilidade percebida. 145
6.19 Facilidade de uso percebida. 147
6.20 Dificuldades encontradas e pontos fracos da ferramenta. 147
6.21 Intenção de uso da GUIPFI. 149

6.22 Intenção de uso da ferramenta GUIPFI. 149
6.23 Avaliação da qualidade dos resultados - Recomendador. 151
6.24 Q4 - Qualidade das recomendações de instrumentos de inspeção obtidas com o uso da GUIPFI. 151

7.1 GUIPFI - Relatório de inspeção gerado pela ferramenta. 159
7.2 Faixa etária dos participantes da indústria. 164
7.3 Tempo de experiência dos participantes da indústria. 164
7.4 Ciclo 3 - Análise estatística das inspeções realizadas por participantes da indústria. 165
7.5 Utilidade percebida - participantes da indústria. 166
7.6 Facilidade de uso percebida - participantes da indústria. 168
7.7 Intenção de uso da GUIPFI - profissionais da indústria. 170
7.8 Avaliação da qualidade dos resultados do componente Recomendador - profissionais da indústria. 172

D.1 Protótipo de IU usado no estudo - Página de Busca. 231
D.2 Protótipo de IU usado no estudo - Página de Produto. 232
D.3 Protótipo de IU usado no estudo - Página de Carrinho de Compra. 232

Lista de Tabelas

3.1 Perguntas do questionário relativas ao uso de prototipação. 31

3.2 Lista de empresas com mais de um respondente. 33

4.1 Categorias de protótipos obtidas durante o estágio Experimental. 61

4.2 Informações selecionadas de diferentes fontes durante o passo 3, estágio Descritivo. 63

4.3 Correspondência entre categorias do processo de rotulagem e trabalhos relacionados. 65

4.4 Adaptações no modelo de especificação de heurísticas de usabilidade de Quiñones et al. (2018). 66

4.5 Heurística HEU1-06 – Lista. 71

4.6 Heurística HEU2-08 – Registro de Conta. 72

4.7 Questionário de avaliação das heurísticas por especialistas. 74

5.1 Resumo das funcionalidades do CHIF-E. 87

5.2 Heurística HEU2-05 – Configurações. 92

5.3 Funcionalidades Específicas do CHIF-E e seus contextos de aplicação 93

5.4 UIProtoCheck - *checklist* para inspeção de protótipos de interface de usuário 96

5.5 Lista de defeitos inseridos intencionalmente nos protótipos do experimento controlado. 102
5.6 Experiência dos participantes e times. CT - Time *Checklist*; HT - Time CHIF-E; PC - Participante do bloco *Checklist*; PH - Participante do bloco CHIF-E. 103
5.7 Equipes de revisores e desenvolvedores e temas de projeto. 107
5.8 Eficácia, eficiência e falso-positivos - CHIF-E. 108
5.9 Eficácia, eficiência, e falso-positivos - UIProtoCheck. 108
5.10 Teste de normalidade Shapiro-Wilk. 109
5.11 Teste estatístico não-paramétrico Mann-Whitney. 109
5.12 Falso-positivos por tipo de instrumento de inspeção. 110
5.13 Número de defeitos encontrados por natureza e instrumento de inspeção. 111
5.14 Quadro comparativo dos principais aspectos - CHIF-E e UIProtoCheck. 112
5.15 Temas resultantes da análise temática. 114
5.16 Defeitos relatados por time e categoria. 115

6.1 Designação de equipes de inspetores e desenvolvedores 138
6.2 Caracterização do perfil dos participantes 139
6.3 Questionário de avaliação da ferramenta GUIPFI - TAM 140
6.4 Questionário de avaliação da ferramenta GUIPFI - Perguntas complementares (abertas) 141

7.1 Exemplos de ajustes de termos e expressões na especificação das heurísticas - Lista 157

7.2 Exemplo de ajustes de termos e expressões na especificação das heurísticas - Câmera 157
7.3 Ocupação, tempo de experiência e faixa etária dos participantes. . 163
7.4 Caracterização das empresas onde os participantes trabalham. . . 163

A.1 Heurística HEU1-01 – Alteração. 197
A.2 Heurística HEU1-02 – Busca. 198
A.3 Heurística HEU1-03 – Cadastro. 199
A.4 Heurística HEU1-04 – Visualização. 200
A.5 Heurística HEU1-05 – Galeria. 201
A.6 Heurística HEU1-06 – Lista. 202
A.7 Heurística HEU1-07 – Menu. 203
A.8 Heurística HEU1-08 – Mestre-Detalhe. 204
A.9 Heurística HEU1-09 – Upload de Arquivos. 205
A.10 Heurística HEU2-01 – Autenticação. 206
A.11 Heurística HEU2-02 – Câmera. 207
A.12 Heurística HEU2-03 – Carrinho de Compra. 208
A.13 Heurística HEU2-04 – Chat. 209
A.14 Heurística HEU2-05 – Configurações. 210
A.15 Heurística HEU2-06 – Editor de Imagens. 211
A.16 Heurística HEU2-07 – Finalizar Compra. 212
A.17 Heurística HEU2-08 – Home. 213
A.18 Heurística HEU2-09 – Mapas. 214
A.19 Heurística HEU2-10 – Media Player. 215
A.20 Heurística HEU2-11 – Notícias. 216

A.21 Heurística HEU2-12 – Página de Produto. 217
A.22 Heurística HEU2-13 – Perfil de Usuário. 218
A.23 Heurística HEU2-05 – Quiz. 219
A.24 Heurística HEU2-15 – Registro de Conta. 220
A.25 Heurística HEU2-16 – Tutorial. 221
A.26 Heurística HEU2-17 – Videoconferência. 222

E.1 Designação de equipes de inspetores e desenvolvedores 234

Sumário

1 Introdução **1**

1.1 Objetivos 5

1.2 Metodologia 5

1.3 Organização do Trabalho 10

2 Referencial Teórico **12**

2.1 Prototipação de Interfaces de Usuário 12

2.2 Técnicas de Inspeção de Interfaces de Usuário 15

2.3 Abordagens Automatizadas para Detecção de Defeitos em Interfaces de Software 17

2.4 Identificação de Significado em Interfaces de Usuário 20

2.5 Trabalhos Relacionados 23

2.6 Originalidade e Relevância da Pesquisa 27

3 Ambiente: Prototipação no Desenvolvimento de Software **28**

3.1 Metodologia 29

3.2 Resultados 32

3.2.1 QP1: Finalidade da Prototipação 35

3.2.2 QP2: Prototipação no Processo de Desenvolvimento 38
3.2.3 QP3: Prototipação e Qualidade de Software 42
3.3 Discussão 45
3.3.1 Lições 47
3.3.2 Limitações 49
3.4 Considerações Finais 49

4 Catálogo Heurísticas de Inspeção Funcional (CHIF) 51

4.1 Introdução 51
4.2 Obtenção de Conjunto de Protótipos 54
4.3 Método de Especificação de Heurísticas de Inspeção Funcional . . 57
4.3.1 Estágio Exploratório 58
4.3.2 Estágio Experimental 59
4.3.3 Estágio Descritivo 62
4.3.4 Estágio Correlacional 64
4.3.5 Estágio de Seleção 64
4.3.6 Estágio de Especificação 65
4.3.7 Estágio de Validação 68
4.3.8 Estágio de Refinamento 69
4.3.9 Heurísticas de Inspeção Funcional 69
4.4 Validação das Heurísticas com Especialistas 71
4.4.1 Preparação e Execução 73
4.4.2 Resultados 73
4.5 Considerações Finais 77
4.5.1 Limitações 78

5 Expansão do Catálogo de Heurísticas de Inspeção Funcional (CHIF-E) 79

5.1 Introdução 80

5.2 Adição de novas funcionalidades ao CHIF 81

5.2.1 Planejamento 84

5.2.2 Anotação Iterativa em Lote 89

5.2.3 Adjudicação - Especificação de novas heurísticas 91

5.3 UIProtoCheck: Um *checklist* genérico para inspeção funcional de protótipos de interface 94

5.4 Avaliação 97

5.4.1 Experimento Controlado: CHIF-E X UIProtoCheck 97

5.4.1.1 Planejamento 98

5.4.1.2 Execução 101

5.4.2 Estudo Experimental: Inspeção de Projetos de Interface Utilizando o CHIF-E 104

5.4.2.1 Planejamento 105

5.4.2.2 Execução 106

5.5 Resultados 106

5.5.1 QP1: Suporte do CHIF-E e do UIProtoCheck à inspeção funcional de protótipos de Interface de Usuário 108

5.5.2 QP2: Tipos de defeitos encontrados durante a inspeção funcional de protótipos de interface usando o CHIF-E . . . 112

5.6 Ameaças à Validade 117

5.7 Considerações Finais 119

6 GUIPFI – Uma Ferramenta para Inspeção Funcional de Protótipos de Interface 123

6.1 Design da Ferramenta GUIPFI 124

6.1.1 Recomendação 127

6.1.2 Assistente de Inspeção 130

6.1.3 Avaliação do desempenho do LLM na sugestão de heurísticas relevantes 133

6.2 Avaliação da GUIPFI na Academia 137

6.2.1 Planejamento 137

6.2.2 Execução 139

6.2.3 Resultados 141

6.3 Considerações Finais 153

7 GUIPFI: Estudo de Campo na Indústria de Software 155

7.1 Planejamento 155

7.1.1 Melhorias nas heurísticas do CHIF-E 156

7.1.2 Melhorias na GUIPFI 157

7.1.3 Ajustes no prompt do Recomendador 159

7.2 Execução 162

7.3 Resultados 164

7.4 Ameaças à Validade 172

7.5 Considerações Finais 174

8 Conclusão 176

8.1 Contribuições da Pesquisa 180

8.1.1 Artigos Publicados 181

8.2 Trabalhos Futuros 181

Referências Bibliográficas **183**

A Catálogo de Heurísticas de Inspeção Funcional - CHIF-E **196**

A.1 Funcionalidades Básicas 196

A.2 Funcionalidades Específicas 196

B Termo de Consentimento Livre e Esclarecido – TCLE **223**

C Artefatos do Primeiro Ciclo de Design **228**

C.1 Cenário da Avaliação das Heurísticas por Especialistas 228

C.2 Roteiro de Atividades do Grupo Focal 229

D Artefatos do Segundo Ciclo de Design **231**

D.1 Protótipos de interface inspecionados no experimento controlado . 231

D.2 Instruções para condução da inspeção 232

E Artefatos do Terceiro Ciclo de Design **234**

E.1 Instruções para Realização da Atividade de Inspeção 234

E.2 Material da Aula - Inspeção Funcional de Protótipos de Interface de Usuário 236

F Artefatos do Estudo de Validação na Indústria **244**

F.1 Carta Convite 244

F.2 Instruções para realização do estudo 245

Capítulo 1

Introdução

Protótipos de interface do usuário são representações gráficas, interativas ou estáticas da interface de usuário (IU) de um software (Bjarnason et al., 2021). Eles são um recurso amplamente utilizado por engenheiros de software para representar as funcionalidades de um software durante o processo de desenvolvimento de software (Bjarnason et al., 2021).

No estágio de elicitação de requisitos, protótipos de interface de usuário são utilizados para facilitar a comunicação com clientes, explorar soluções alternativas e validar ideias (Dow et al., 2011; Lauff et al., 2020), pois ilustram como os usuários irão interagir com o software por meio da interface. Em projetos que utilizam a metodologia de desenvolvimento ágil, a prototipação tem se mostrado um instrumento crítico de elicitação de requisitos que pode promover a melhoria da comunicação em times de desenvolvimento, principalmente quando associada a boas práticas de gerenciamento (Käpyaho and Kauppinen, 2015; Garcia et al., 2017).

No entanto, a prototipação é uma tarefa desafiadora (Lee et al., 2020). Durante a atividade de design de protótipos de interface, é necessário analisar quais funcionalidades serão representadas e quais elementos de design caracterizam tais funcionalidades. É preciso, ao mesmo tempo, seguir regras de design, dominar tecnologias de desenvolvimento e ser criativo e inovador para atrair e manter o engajamento do usuário (Chen et al., 2020a). Como resultado, as atividades que

envolvem o desenvolvimento de interfaces se tornam extremamente complexas e costumam ser realizadas de maneira pouco sistemática (Belikova et al., 2018).

Além disso, durante o desenvolvimento de um software, os protótipos de interface constituem um recurso para representar e coletar requisitos que podem servir de referência para as fases posteriores do processo de desenvolvimento, incluindo o design de interface, o desenvolvimento incremental, a implementação, a validação e testes de aceitação (Käpyaho and Kauppinen, 2015; Bjarnason et al., 2023). Por este motivo, possíveis erros de representação de requisitos em protótipos que não forem detectados logo no início de um projeto podem resultar em custos maiores de correção conforme o desenvolvimento do software avança.

O cenário apresentado destaca a necessidade de desenvolvimento de formas de auxiliar a elaboração de protótipos de interface de usuário, bem como viabilizar a sua verificação. Atividades voltadas à garantia da qualidade podem ser empregadas para avaliar a qualidade dos protótipos de interface. Dentre as técnicas de garantia da qualidade mais utilizadas para verificação de interfaces de usuário, destaca-se a inspeção (Wilson, 2013), que é uma técnica que consiste na análise de artefatos do projeto de um software para identificar defeitos. A inspeção funcional de artefatos de design de software propõe-se a verificar o quanto um determinado design está em conformidade com uma especificação ou com requisitos funcionais (Salleh et al., 2017).

As técnicas existentes de inspeção de interface de usuário são projetadas principalmente para avaliação e identificação de defeitos de usabilidade (Hollingsed and Novick, 2007; Islam et al., 2020). Dentre elas, pode-se citar a avaliação heurística (Nielsen, 1994a), a inspeção baseada em perspectivas (Zhang et al., 1999), inspeções formais de usabilidade (Kahn and Prail, 1994) e a avaliação de comunicabilidade de sinais de interface (De Souza et al., 2006; Islam et al., 2020).

Outras abordagens procuram identificar defeitos de design em interfaces de software através da análise automática de interfaces gráficas de usuário (Graphical User Interfaces - GUI) (Moran et al., 2018; Packevičius et al., 2018; Yang et al., 2021). Por exemplo, o trabalho de Moran et al. (2018) utiliza técnicas de visão computacional para verificar se a implementação de uma interface está de acordo com seu design. Yang et al. (2021) apresentam uma abordagem automática para detecção de *design smells*, que são características encontradas em interfaces de

software que violam regras de design pré-estabelecidas. Packevičius et al. (2018) apresentam um método de análise de código-fonte com base em conjuntos de regras associados a capturas de tela para identificar defeitos de apresentação de texto e problemas semânticos. Também há trabalhos que buscam auxiliar o desenvolvimento de interfaces no momento da sua concepção, através da busca de interfaces gráficas (Huang et al., 2019) ou sugerindo GUIs semelhantes, com base na disposição dos componentes e destacando áreas onde a atenção do usuário possivelmente se concentra em um protótipo (Lee et al., 2020). Neste tipo de busca, é possível pesquisar, a partir de um design de entrada, uma variedade de exemplos de interfaces de usuário semelhantes (Bernal-Cárdenas et al., 2019; Chen et al., 2020b).

As abordagens apresentadas de auxílio à prototipação e verificação de design de interface de usuário baseiam-se em imagens e não consideram o significado dos elementos e funcionalidades representados no protótipo. Isto reduz os tipos de verificações que podem ser realizadas durante a avaliação da interface. No caso da verificação da disposição dos componentes no protótipo, não é verificado se os elementos realmente representam a funcionalidade correta. Já a busca de designs com base em imagens procura apenas GUIs visualmente semelhantes. Isto limita a recuperação de exemplos inovadores de interfaces que representem a mesma funcionalidade, mas cuja aparência difere da interface utilizada como parâmetro de busca. Além disso, estas abordagens são destinadas a protótipos interativos, de elaboração mais complexa e demorada, ou à própria interface do software, que só pode ser examinada após o software ter sido implementado (Wilson, 2013).

Protótipos criados durante a fase de elicitação de requisitos, por outro lado, costumam focar nos aspectos mais críticos da funcionalidade projetada, frequentemente desconsiderando convenções de design ou considerações de usabilidade. Assim, a inspeção de protótipos de interface preliminares deve considerar essencialmente sua funcionalidade subjacente. Por funcionalidade entende-se um conjunto de funções e respectivas propriedades que satisfazem um ou mais requisitos de sistema (Salleh et al., 2017). Desta forma, destaca-se a necessidade de técnicas ou abordagens de inspeção de protótipos de interface que os analisem de acordo com seu significado, possibilitando antecipar a detecção de defeitos relacionados à funcionalidade representada.

O problema tratado nesta tese refere-se à identificação tardia de defeitos de funcionalidade em protótipos de interface de usuário (IU), etapa que frequentemente ocorre apenas após a implementação do software, resultando em retrabalho e aumento de custos de correção. Embora os protótipos de IU sejam amplamente utilizados para comunicar e validar requisitos durante o desenvolvimento, inexistem abordagens de inspeção com foco explícito na verificação funcional desses artefatos, capazes de avaliar se as funcionalidades esperadas estão corretamente representadas e completas. As técnicas de inspeção existentes concentram-se predominantemente em aspectos de usabilidade, acessibilidade e comunicabilidade, não contemplando a dimensão funcional necessária para garantir a consistência entre o que é representado no protótipo e o comportamento esperado do sistema. Neste contexto, esta pesquisa visa contribuir com a identificação de defeitos funcionais em protótipos de interface e, consequentemente, com a melhoria da qualidade do software desenvolvido. Assim, esta pesquisa buscou responder à seguinte questão de pesquisa:

> **QP: "Como inspecionar protótipos de interface de usuário em relação à sua funcionalidade?"**

Para abordar esta questão, primeiramente, verificou-se que a inspeção de protótipos de interface em termos de sua funcionalidade deve iniciar-se pela identificação da funcionalidade representada, para, em seguida, verificar se a interface projetada se alinha com os requisitos funcionais. Tais requisitos podem ser explícitos, derivados de especificações e modelos, e intrínsecos (conhecimento tácito), associados a essa funcionalidade. Por exemplo, em um aplicativo de comércio de veículos, existem requisitos específicos de domínio, como, por exemplo, *"A página de produto deve apresentar a quilometragem do veículo"*, e requisitos típicos de comércio eletrônico, como a apresentação de imagens do veículo na página do produto.

Diante disso, a abordagem de inspeção funcional de protótipos de interface desta pesquisa considerou tanto as características inerentes às funcionalidades representadas quanto especificações e modelos específicos das aplicações. Ela foi viabilizada através da caracterização de funcionalidades amplamente conhecidas,

comuns a diferentes aplicativos, como autenticação de usuário, pesquisa ou formulários de cadastro. Como o conhecimento sobre as características peculiares destas funcionalidades recorrentes é tácito, ou seja, desenvolvedores de software as conhecem, mas elas não são formalmente especificadas, buscou-se neste trabalho descrever explicitamente cada funcionalidade comumente conhecida para realizar inspeções segundo o aspecto funcional. Além disso, também foi elaborado um *checklist* de propósito geral para inspeção de protótipos de IU que não representam as funcionalidades diretamente especificadas. Através da ênfase na caracterização de funcionalidades, buscou-se contribuir para a detecção precoce de defeitos e inconsistências na representação de requisitos funcionais em protótipos da interface de usuário.

1.1 Objetivos

O objetivo principal desta pesquisa é dar suporte a times de desenvolvimento de software na atividade de inspeção funcional de protótipos de interface de usuário através de uma abordagem assistida de inspeção funcional. Com esta abordagem, busca-se oferecer mecanismos que possibilitem antecipar a identificação de defeitos relacionados à funcionalidade. Os objetivos específicos desta pesquisa são:

- Caracterizar funcionalidades comumente representadas em protótipos de interface de usuário em diferentes domínios de aplicação;
- Elaborar instrumentos para inspeção de protótipos de interface com foco na sua funcionalidade;
- Apoiar a identificação de defeitos funcionais através de ferramenta automatizada baseadas nos instrumentos de inspeção desenvolvidos nesta pesquisa.

1.2 Metodologia

A abordagem metodológica utilizada nesta pesquisa é a *Design Science Research* - DSR (Hevner and Chatterjee, 2010; Dresch et al., 2015). O DSR é um método de pesquisa que operacionaliza a pesquisa baseada na *Design Science* (Hevner

et al., 2004), que por sua vez é um paradigma epistemológico para a realização de pesquisas dedicadas ao estudo da construção de novos artefatos, sejam para resolver problemas ou para melhorar os sistemas existentes. A ênfase na produção de artefatos é a razão pela qual a DSR foi escolhida para este projeto, uma vez que o mesmo visa contribuir com um artefato para a inspeção funcional de protótipos da interface de usuário de maneira automatizada. As etapas deste projeto de pesquisa seguem a visão de DSR descrita por Hevner (2007). Em seu modelo, a DSR compreende três ciclos de atividades intimamente relacionados: Rigor, Relevância e Design. A Figura 1.1 ilustra como os ciclos se relacionam entre si e com os escopos do Ambiente, da Pesquisa e da Base de Conhecimento, com atividades adaptadas ao contexto específico deste projeto de pesquisa.

O ciclo de Relevância compreende a realização de atividades entre os escopos do Ambiente e da Pesquisa visando identificar um problema prático, definir o domínio da pesquisa e avaliar o artefato produzido pela pesquisa no ambiente real. O ciclo do Rigor é realizado visando identificar na literatura as bases teóricas e metodológicas a serem utilizadas na pesquisa, bem como contribuir com uma abordagem de inspeção funcional de interfaces para a base de conhecimento existente. Já o ciclo do Design compreende as atividades de desenvolvimento e aprimoramento iterativo do artefato da pesquisa, visando resolver o problema previamente identificado no ciclo de Relevância.

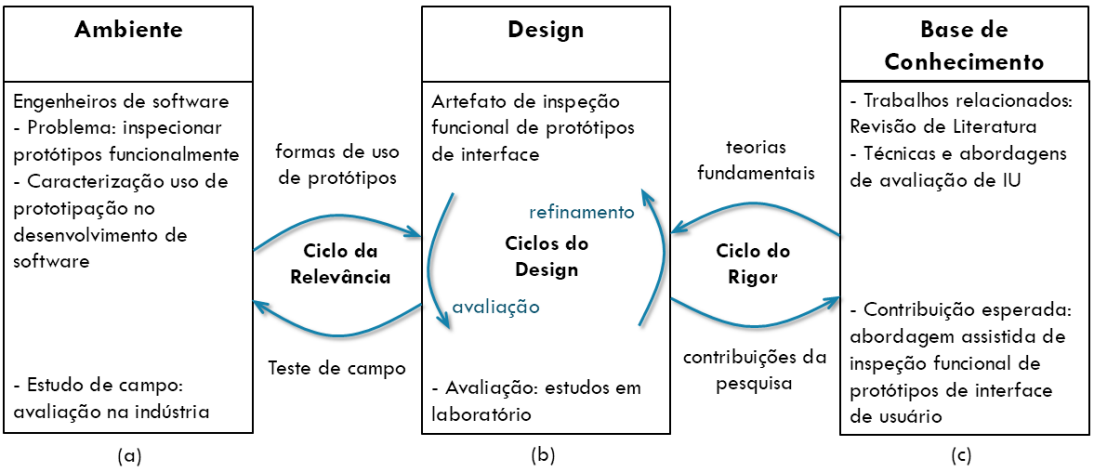


Figura 1.1: Ciclos de pesquisa baseados na DSR

A abordagem de inspeção funcional de protótipos de interface desenvolvida nesta pesquisa visa solucionar o problema de identificação tardia de defeitos associados à funcionalidade em protótipos de interface de usuário. Ela se baseia em um conjunto de instrumentos de inspeção funcional desenvolvidos no decorrer do projeto: (i) um conjunto de heurísticas que descrevem funcionalidades comumente encontradas em aplicações de software e (ii) um *checklist* de propósito geral como alternativa de inspeção para protótipos de funcionalidades não contempladas no conjunto de heurísticas. A abordagem foi implementada na ferramenta GUIPFI (*Guided User Interface Prototype Functional Inspection*), que oferece mecanismos de seleção de instrumentos de inspeção adequados ao protótipo inspecionado, além de suporte ao registro de defeitos e geração de relatórios de inspeção. A Figura 1.2 apresenta o *Visual Abstract* da solução elaborada nesta pesquisa, elaborado a partir do modelo definido por Storey et al. (2017). O *Visual Abstract* apresenta o escopo, o problema e a solução adotados nesta pesquisa na forma de uma Regra Tecnológica.

Nesta pesquisa, a primeira atividade está situada no **Ciclo de Rigor** e consistiu na realização de uma revisão de literatura (Capítulo 2) visando encontrar trabalhos existentes que tratam de técnicas de inspeção de protótipos de interface de usuário, assim como técnicas de identificação de significado em protótipos de interface. Nesta fase, também foi avaliada a aplicabilidade dessas técnicas ao contexto desta pesquisa. Nesta revisão, foi possível verificar que as abordagens existentes para inspeção de protótipos de interface focam-se nos aspectos de regras de design e de usabilidade, não havendo ainda abordagens que suportem a inspeção de protótipos de IU com foco na sua funcionalidade.

A segunda atividade do método de trabalho, localizada no escopo do **Ciclo Relevância**, foi a realização de uma pesquisa de opinião direcionada a profissionais de diferentes funções em times de desenvolvimento de software (Capítulo 3). A pesquisa visou caracterizar com que finalidade os profissionais da indústria de software utilizam protótipos de UI e como estes estão relacionados às etapas do processo de desenvolvimento e às atividades de garantia da qualidade. Com este estudo, foram identificadas diversas etapas do processo de desenvolvimento de software que utilizam a prototipação, desde a ideação, a implementação e os testes. Os resultados revelaram a importância de incluir protótipos de interface

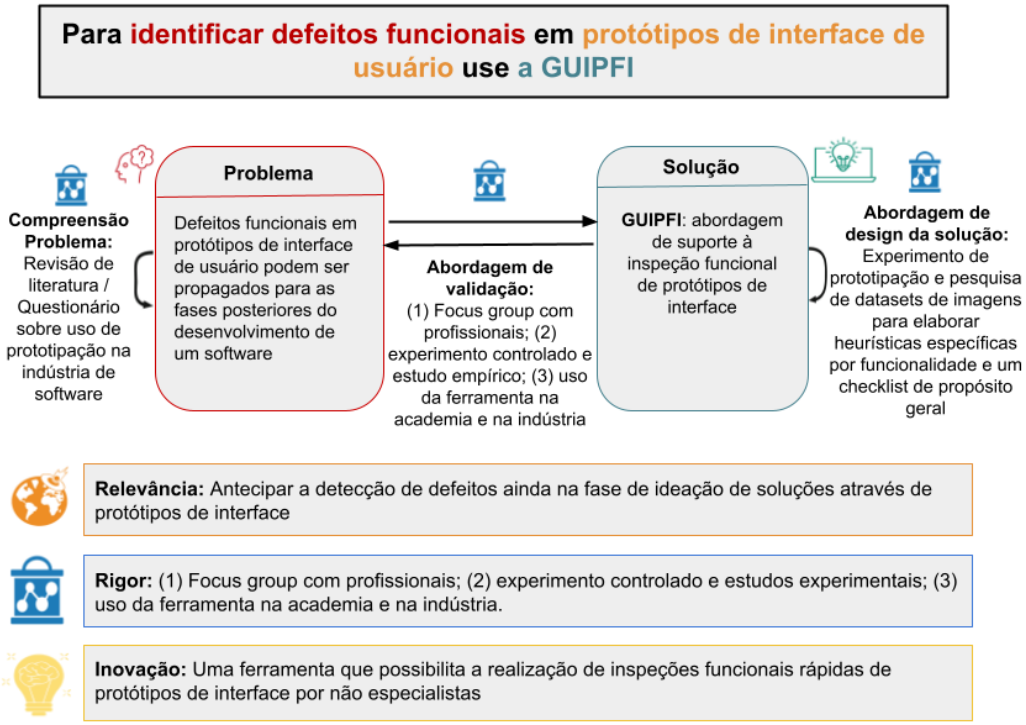


Figura 1.2: Visual Abstract da abordagem de inspeção funcional de protótipos de interface.

em atividades de garantia da qualidade, de maneira a antecipar a identificação de defeitos em protótipos de IU no processo de desenvolvimento de software, reduzindo assim o seu impacto nas fases de desenvolvimento subsequentes. Os resultados desta pesquisa foram publicados em De Macedo et al. (2022).

O **Ciclo de Design** foi realizado em três iterações, com o objetivo de desenvolver uma abordagem de apoio à inspeção funcional de protótipos de interface de usuário. Na primeira iteração do ciclo de Design (Capítulo 4), foi elaborado um conjunto de heurísticas de funcionalidades comumente encontradas em sistemas de informação (De Macedo et al., 2022), com o intuito de utilizá-las como instrumentos de apoio à inspeção funcional. O conjunto de heurísticas foi elaborado a partir da análise de protótipos de interface do ponto de vista de sua funcionalidade, obtidos em um estudo que utilizou Pesquisa-Ação para avaliar uma metodologia colaborativa para o ensino de Engenharia de Requisitos com o desenvolvimento de *soft skills* adaptada ao contexto remoto (De Macedo et al.,

2024). As heurísticas foram avaliadas através de um grupo focal com especialistas, que indicaram pontos importantes de melhoria, sugestões de unificação de algumas heurísticas e a adição de novas funcionalidades ao catálogo.

A segunda iteração do Ciclo de Design (Capítulo 5) consistiu na inclusão de novas funcionalidades no conjunto de heurísticas de funcionalidades recorrentes. Além disso, foi desenvolvido o UIProtoCheck (De Macedo et al., 2023), um instrumento de inspeção de propósito geral para ser utilizado na inspeção de protótipos de funcionalidades não descritas no conjunto de heurísticas. Estes instrumentos de inspeção foram avaliados em um estudo em laboratório, onde os participantes voluntários elaboraram projetos de aplicações de software com o uso de prototipação de interface e utilizaram os instrumentos de inspeção para verificar se os protótipos atendem aos requisitos estabelecidos para a funcionalidade representada. Os resultados deste estudo indicaram a necessidade de desenvolvimento de um software de apoio às atividades de inspeção funcional.

Na terceira iteração do Ciclo de Design (Capítulo 6), foi desenvolvido um artefato de software, o GUIPFI (Guided User Interface Prototype Functional Inspection), o qual, dado um protótipo de interface, recomenda o instrumento de inspeção funcional mais adequado para inspecioná-la. Além disso, o artefato oferece suporte às atividades de inspeção, possibilitando a consulta e seleção de instrumentos, o registro de defeitos encontrados e a geração de relatórios de inspeção. Os resultados indicaram pontos de melhoria nas heurísticas, na recomendação de instrumentos e nas funcionalidades da ferramenta.

A finalização da pesquisa ocorreu novamente no escopo do Ciclo de Relevância. Ela consistiu em um estudo de campo na indústria de software (Capítulo 7), onde o artefato refinado foi utilizado por profissionais de times de desenvolvimento de diferentes perfis e empresas. Neste estudo, os profissionais utilizaram o software elaborado para inspecionar protótipos de IU de projetos de software reais.

Em atenção aos princípios éticos que regem esta pesquisa, esta pesquisa foi submetida ao Comitê de Ética em Pesquisa - CEP da Universidade Federal do Amazonas - UFAM, sob o número CAAE 40937120.6.0000.5020, em 26/07/2021, tendo sido aprovada no Parecer Consubstanciado N. 4.923.486, de 22 de agosto de 2021. Devido a alterações significativas no escopo da pesquisa, um novo projeto de pesquisa foi submetido ao referido comitê, sob o número CAAE: 82963224.1.0000.5020,

em 18/07/2024, tendo sido aprovado no Parecer Consubstanciado N. 7.072.722, de 12 de setembro de 2024.

1.3 Organização do Trabalho

Os demais capítulos deste trabalho de pesquisa estão organizados da seguinte maneira:

- **Capítulo 2 – Referencial Teórico e Trabalhos Relacionados:** Apresenta o referencial teórico dos conceitos abordados nesta pesquisa, prototipação de interfaces de usuário e inspeção de software, bem como trabalhos relacionados às técnicas de inspeção existentes.

- **Capítulo 3 – Prototipação no Desenvolvimento de Software:** Apresenta os resultados da pesquisa realizada com profissionais que trabalham com desenvolvimento de software, visando caracterizar o uso de prototipação na indústria de software.

- **Capítulo 4 – Catálogo de Heurísticas de Inspeção Funcional (CHIF):** Neste capítulo é apresentado o processo de elaboração das heurísticas de inspeção funcional de interfaces de usuário utilizadas pela abordagem de inspeção funcional apresentada nesta pesquisa.

- **Capítulo 5 – Expansão do Catálogo de Heurísticas de Inspeção Funcional (CHIF-E):** Este capítulo apresenta a metodologia utilizada para ampliar a abrangência do CHIF com a inclusão de heurísticas para novas funcionalidades, resultando no CHIF expandido (CHIF-E), e para a elaboração do *checklist* de uso geral para inspeção funcional de protótipos de interface (UIProtoCheck).

- **Capítulo 6 – GUIPFI – Uma Ferramenta para Inspeção Funcional de Protótipos de Interface:** Apresenta o processo de desenvolvimento da ferramenta GUIPFI e a avaliação da ferramenta no contexto da academia.

- **Capítulo 7 – GUIPFI: Estudo de Campo na Indústria de Software:** Apresenta o estudo de avaliação da abordagem implementada na ferramenta GUIPFI na indústria de software.

- **Capítulo 8 – Conclusão e Trabalhos Futuros:** Apresenta as principais contribuições deste trabalho, suas limitações e oportunidades de pesquisas futuras.

Capítulo 2

Referencial Teórico

Este capítulo apresenta o referencial teórico e a revisão de literatura adotados no desenvolvimento deste projeto de pesquisa, que abordam conceitos relacionados à prototipação de interfaces de usuário. O capítulo aborda ainda a revisão de literatura de diversos temas relacionados à inspeção funcional de protótipos de interface: métodos de inspeção de interfaces de usuário, abordagens automatizadas de detecção de defeitos e identificação de significados em interfaces de software. Em seguida, na seção Trabalhos Relacionados, são apresentadas as pesquisas que guardam maior proximidade com o tema abordado nesta tese: a inspeção funcional de protótipos de interface de usuário.

2.1 Prototipação de Interfaces de Usuário

Em atividades de engenharia de software, protótipos de interface de usuário são modelos preliminares de funcionalidades de um software. Eles são utilizados com diferentes finalidades, tais como demonstrar ideias, explorar soluções e elicitar requisitos (Ogedebe and Jacob, 2012; Suranto, 2015; Karhapää et al., 2021).

No processo de desenvolvimento, a prototipação tem sido utilizada na elicitação de requisitos ou na descoberta do produto de software. No entanto, alguns trabalhos relatam seu uso também como instrumento na definição de iterações, para auxiliar estimativas, para servir de referência para o desenvolvimento ou para apoiar atividades de teste de usabilidade (Käpyaho and Kauppinen, 2015;

Garcia et al., 2017; Behutiye et al., 2020). Além disso, estudos mostram que a prototipação pode ser utilizada em dinâmicas de design que estimulem a criação e o compartilhamento de múltiplas soluções, de forma a favorecer a aprendizagem, a comunicação em equipe e a produção de resultados de maior qualidade (Dow et al., 2011).

Bjarnason et al. (2021) apresentam um mapeamento sistemático da literatura sobre prototipação, que resultou em *framework* para classificação de protótipos (*PAM - Prototyping Aspects Model*), o qual os classifica segundo quatro aspectos: o propósito para sua elaboração, o escopo, a forma de uso e a estratégia de exploração de ideias durante a elaboração do protótipo. No aspecto do propósito, oito tipos diferentes são apresentados: exploração, comunicação, desenvolvimento incremental, melhoria da qualidade, viabilidade de mercado e técnica e usabilidade.

Em relação a aspectos de qualidade no desenvolvimento de software com o uso de prototipação, há relatos do uso de prototipação para gerenciar requisitos de qualidade. O trabalho de Behutiye et al. (2020), sobre tipos de recursos utilizados para gerenciar requisitos de qualidade em desenvolvimento ágil, mostrou que a prototipação em ciclos iterativos é utilizada na identificação de requisitos de arquitetura, usabilidade e desempenho. Karhapää et al. (2021) relatam o uso de *mockups* como uma das estratégias utilizadas pelas empresas de desenvolvimento de software para elicitar requisitos de qualidade em situações em que o time de desenvolvimento tem acesso direto a clientes ou usuários.

O trabalho de Käpyaho and Kauppinen (2015) apresenta um estudo de caso para investigar como a prototipação pode ser utilizada para resolver problemas na engenharia de requisitos em processos de desenvolvimento ágil. Seus resultados mostram que os protótipos facilitaram a comunicação com clientes, porém, não ajudaram na visão do projeto como um todo e nem melhoraram a atenção aos requisitos de qualidade e não funcionais. Além disso, eles apresentam uma lista de medidas para amenizar problemas normalmente relacionados ao uso de protótipos, tais como fazer revisão da visão geral do projeto periodicamente, promover o envolvimento do cliente e manter melhor controle de requisitos de qualidade e não-funcionais.

Bellomo et al. (2013) analisaram os fatores de uso da prototipação com foco em atributos de qualidade que e identificaram diversos aspectos da prática de

prototipação que permitem uma resposta eficiente a mudanças, tais como a integração entre o processo ágil e a arquitetura, com colaboração entre arquiteto e *product owner* durante o planejamento, a demonstração de protótipos e a análise do *feedback* sobre o protótipo. A partir desses resultados, são apresentados *guidelines* de prototipação com foco na qualidade do software.

Apesar de ser utilizada para fins diversos, apoiando atividades que vão desde a concepção de ideias até a gestão de requisitos de qualidade (De Macedo et al., 2022), processos existentes não oferecem procedimentos explícitos para a utilização da prototipação no desenvolvimento de software, sendo seu uso predominantemente empírico (Belikova et al., 2018; Bjarnason et al., 2023). Alguns trabalhos, entretanto, apresentam ferramentas que visam apoiar a ideação de interfaces de usuário. Por exemplo, o Guigle (Bernal-Cárdenas et al., 2019) implementa uma abordagem de busca e recomendação de interfaces, utilizando textos de telas, componentes de IU, nomes de aplicativos e cores para recuperar telas semelhantes da loja Google Play. O Swire (Huang et al., 2019) propõe a recuperação de GUIs a partir de esboços, baseando-se em um banco de dados construído com exemplos do dataset Rico (Deka et al., 2017). Chen et al. (2020b) também investigam a busca e recuperação de GUIs por meio de *wireframes*. Já o Rewire (Swearngin et al., 2018) apoia a reutilização de exemplos ao converter capturas de tela em representações vetoriais editáveis, possibilitando que designers manipulem componentes extraídos de maneira a agilizar a reconstrução e edição de interfaces a partir de exemplos. De forma similar, o GUI2WiRe (Kolthoff et al., 2020) possibilita a extração e composição automática de elementos em *wireframes* ao recuperar interfaces gráficas a partir de requisitos expressos em linguagem natural, utilizando um repositório de larga escala de GUIs mineradas de aplicativos.

Além de abordagens e ferramentas assistivas, diversas técnicas de detecção de defeitos em interfaces de usuário foram desenvolvidas nas últimas décadas. A próxima seção apresenta a revisão de literatura de técnicas de inspeção voltadas para interfaces de usuário.

2.2 Técnicas de Inspeção de Interfaces de Usuário

Técnicas de inspeção são métodos de engenharia de software para melhorar a qualidade de requisitos, documentação e código, usados principalmente na avaliação de problemas de usabilidade em softwares (Wilson, 2013). As técnicas de inspeção mais popularmente utilizadas para interfaces de usuário são as de inspeção de usabilidade. Dentre elas, podemos destacar a avaliação heurística (Wilson, 2013), onde a interface é avaliada de acordo com um conjunto de princípios de *design* (heurísticas), oriundos do senso comum. As Heurísticas de Nielsen (Nielsen, 1994a) constituem um dos métodos de inspeção de usabilidade de interfaces mais amplamente utilizados. Ele é formado por dez heurísticas que descrevem as características gerais desejáveis de uma interface de usuário e podem indicar potenciais fontes de problemas relacionados à usabilidade.

Com base nessa abordagem, outros trabalhos foram propostos contendo heurísticas voltadas para domínios de aplicação mais específicos. Dentre eles, podemos citar o MIT2 (Valentim and Conte, 2014), uma técnica de inspeção para *mockups* de interfaces de aplicações. Silva et al. (2015) propõem um conjunto de trinta e três heurísticas de usabilidade para avaliar a usabilidade de aplicativos móveis voltados para usuários idosos. As heurísticas avaliam fatores relevantes de usabilidade para o público-alvo em aplicações de saúde e bem-estar: percepção, cognição, habilidade, navegação, conteúdo e *design* visual. O HE4DWUX (Yeratziotis and Zaphiris, 2018) é um conjunto de heurísticas para avaliar a experiência do usuário e a acessibilidade de aplicações web para usuários surdos. O trabalho de Baños Díaz and Zapata Del Río (2018) propôs um conjunto de heurísticas para projetar interfaces seguras e satisfatórias para sites de *internet banking*. Uma revisão sistemática da literatura sobre heurísticas de usabilidade para aplicações móveis é apresentada no trabalho de Da Costa et al. (2019), que propõe ainda um conjunto de heurísticas de usabilidade para esse contexto. As heurísticas propostas consideram o usuário, a tarefa realizada por ele e o contexto de uso, assim como a carga cognitiva, como fatores importantes da usabilidade. Já Bashir and Farooq (2019) propuseram um conjunto de 14 heurísticas de usabilidade de aplicativos móveis obtidas a partir do mapeamento entre problemas encontrados em

um estudo com 800 usuários e conjuntos de heurísticas disponíveis na literatura.

Além da avaliação heurística, outros métodos de inspeção de interface são voltados para avaliar a forma como os usuários interagem com as aplicações, considerando os diversos perfis ou papéis que eles podem assumir ao usar o software. O Walkthrough Cognitivo (Wharton, 1994) é um método de avaliação de capacidade de aprendizagem no qual especialistas em UI percorrem a interface e exploram todos os recursos oferecidos, avaliando, a cada tarefa executada, a facilidade com que o usuário final entenderia o seu propósito e a sua forma de uso. Uma variação desse método é o *Heuristic Walkthrough* (Sears, 1997), que combina a Avaliação Heurística e o Walkthrough Cognitivo usando uma lista de tarefas que especificam o que deve ser feito, mas não como deve ser feito, de maneira a estimular a exploração da IU. A Inspeção da Interface do Usuário Baseada em Perspectiva (Zhang et al., 1999; Shull et al., 2000) é outra abordagem que seleciona diferentes aspectos de uso do software e define procedimentos de revisão personalizados para eles, podendo detectar problemas de performance e eficiência, além de usabilidade. Por fim, inspeções formais de usabilidade (Hollingsed and Novick, 2007) revisam artefatos com um grupo de especialistas em fatores humanos que executam cenários de tarefas na interface como se fossem usuários de diferentes perfis. Os defeitos encontrados são avaliados pelo grupo e encaminhados aos desenvolvedores para correção.

Finalmente, alguns métodos buscam avaliar diferentes aspectos da comunicação entre sistema e usuário. A avaliação de comunicabilidade é um tipo específico de inspeção de interfaces de usuário que busca verificar como a interface comunica ao usuário o que pode ser feito, o que está acontecendo e o que aconteceu após uma ação. O Método de Inspeção Semiótica (*Semiotic Inspection Method* - SIM)(De Souza et al., 2006) avalia como a interface transmite a lógica do sistema por meio de sinais estáticos e dinâmicos presentes na interface, como links, ícones e botões, e como os usuários entendem esses sinais. Seu objetivo é identificar diferentes formas de uso do software e possíveis interpretações de seu comportamento. Para isso, é utilizada a documentação da aplicação associada à análise exploratória e aprofundada em partes bem definidas do software. Derivado do Método de Inspeção Semiótica, o *framework* SIDE - *Semiotic Interface Sign Design and Evaluation* (Islam et al., 2020) foi desenvolvido para projetar e avaliar a comuni-

cabilidade de sinais de interface web e de aplicativos móveis. Este *framework* é orientado à identificação de problemas de usabilidade em elementos de interface relacionados à sua intuitividade, que não são captados por técnicas de inspeção de usabilidade convencionais. Já o *Collaborative Critique* (Lucas and Babaian, 2015) é um método de inspeção de usabilidade que avalia o software como um parceiro colaborativo do usuário na realização de tarefas, ao invés de avaliar se o usuário consegue usar a interface de forma eficiente. Ele é conduzido por especialistas como um *walkthrough*, que realizam um conjunto definido de tarefas e avaliam os aspectos de encontrabilidade de opções, esforço e confusão, apoio contextual, *feedback* de progresso e suporte a erros.

As técnicas de inspeção apresentadas se mostraram eficazes na detecção de problemas de usabilidade, interação e violação de regras de design. No entanto, elas não cobrem a avaliação de aspectos relacionados à funcionalidade representada nas interfaces e são mais adequadas para protótipos interativos ou interfaces implementadas. Além disso, elas requerem um tempo considerável de preparação e necessitam de avaliadores especialistas.

2.3 Abordagens Automatizadas para Detecção de Defeitos em Interfaces de Software

Embora amplamente utilizados, os métodos manuais de inspeção de interfaces apresentam limitações relacionadas ao custo, ao tempo necessário e à dependência de avaliadores especializados. Visando reduzir essas limitações, diversas abordagens automatizadas têm sido propostas, com o objetivo de acelerar o processo de detecção de defeitos, aumentar a cobertura da avaliação e reduzir a subjetividade característica das análises humanas. Essas soluções incluem modelos de falhas para viabilizar a automação da identificação de defeitos (Lelli et al., 2015; Marenkov et al., 2016), *frameworks* que combinam múltiplas análises e fornecem *feedback* em tempo de design (Charfi et al., 2015; Cheng, 2016; Lee et al., 2020) e ferramentas que realizam a análise automática em busca de defeitos (Packevičius et al., 2018; Moran et al., 2018; Yang et al., 2021).

Dentre os modelos propostos visando fornecer bases para a elaboração de

ferramentas de detecção de defeitos, podemos destacar o trabalho de Lelli et al. (2015), que apresenta um modelo de falhas de GUI para apoiar atividades de teste de software e o desenvolvimento de ferramentas de teste. O modelo descreve falhas em duas categorias: falhas de interface de usuário, como problemas estruturais, estéticos e de apresentação de dados GUI, e falhas de interação, como a execução de ações incorretas, ausência de ações esperadas e problemas de *feedback* nas interações. Seu objetivo é o de apoiar o desenvolvimento e a avaliação de ferramentas de teste automatizado. Já o modelo proposto por Marenkov et al. (2016) define conceitos e uma estrutura para definição de diretrizes de usabilidade, de maneira a possibilitar a avaliação automática e imediata de atributos da interface de usuário, como estrutura de navegação, densidade de conteúdo e legibilidade de textos, durante a implementação. A avaliação do modelo e da estrutura proposta mostrou que são adequados para avaliar a maioria das interfaces de usuário de acordo com as diretrizes definidas ainda durante a fase de desenvolvimento.

Em relação aos *frameworks* de detecção de defeitos em interfaces, Charfi et al. (2015) desenvolveram o RITA (useR Interface evaluaTion frAmework), um *framework* modular para avaliação de interfaces de usuário que integra como técnicas de análise a inspeção automática de conformidade com diretrizes ergonômicas configuráveis, a análise de dados de interação capturados em tempo real e a avaliação subjetiva por meio de questionários aplicados aos usuários. Desta forma, ele consegue detectar automaticamente defeitos relacionados à violação de diretrizes de design, problemas de fluxo de interação, desempenho de tarefas e percepção dos usuários quanto à qualidade da interface. No entanto, os autores apontam limitações como a necessidade de instrumentação do código-fonte da aplicação para capturar os dados de interação.

O MAUi - Mobile App Usability Inspection (Cheng, 2016), por sua vez, é um framework para inspeção de usabilidade de aplicativos móveis em fases iniciais de desenvolvimento no contexto de *lean startups*. Ele incorporou regras de Avaliação Heurística (Nielsen, 1994b) em uma abordagem que coordena a inspeção com atividades de preparação, execução baseada em casos de uso e avaliação por meio de questionários. Os resultados do estudo de avaliação do *framework* demonstraram alta confiabilidade na detecção de defeitos de usabilidade associados à navegabilidade, clareza de rótulos e adequação de fluxos de interação. Já

o GUIComp (Lee et al., 2020) é uma ferramenta de auxílio à prototipagem de GUI que oferece *feedback* automatizado e em tempo real sobre defeitos visuais e estruturais durante o processo de design de interfaces gráficas. A ferramenta pode recomendar interfaces de usuário semelhantes, avaliar aspectos visuais, como alinhamento de componentes, tamanho de elementos e consistência de layout, e mostrar pontos no protótipo que chamam mais a atenção do usuário.

Outros trabalhos desenvolveram abordagens automatizadas de detecção de defeitos em GUIs. A TLDD - *Text and Layout Defect Detector* (Packevičius et al., 2018) é uma abordagem de análise automatizada para detecção de defeitos que impactam a legibilidade, a compreensão das funcionalidades e a consistência visual da interface. Para isso, a abordagem se baseia em uma lista de regras de análise estática e combina a execução dinâmica da aplicação com a análise automatizada de capturas de tela para identificar problemas como textos ambíguos ou inconsistentes, mal posicionados ou sobrepostos.

Outra ferramenta de análise automatizada de interface de usuário é o GVT - *GUI Violation Tester* (Moran et al., 2018), que compara *mockups* de design com implementações reais de interfaces de aplicativos móveis, com o objetivo de garantir a fidelidade entre a especificação visual e o produto final. A técnica emprega visão computacional para detectar discrepâncias visuais entre o design planejado e a interface implementada, identificando defeitos como desalinhamentos, uso incorreto de cores e diferenças de tamanho entre componentes. O trabalho de Yang et al. (2021) apresenta um detector automatizado de *design smells* em interfaces gráficas de usuário, com o objetivo de melhorar a clareza visual e a organização dos elementos de interface. *Design smells* são características encontradas em designs de interface que violam boas regras de design (Yang et al., 2021). A ferramenta analisa UIs complexas para identificar violações em sete dimensões e quatro aspectos, como agrupamentos confusos de componentes, densidade excessiva de informações e uso inadequado de hierarquias visuais.

Os modelos, *frameworks* e ferramentas apresentados nesta seção demonstram o potencial de abordagens automatizadas em apoiar o processo de inspeção, auxiliando a detecção de diferentes tipos de problemas, como defeitos visuais, estruturais ou de usabilidade. No entanto, há uma carência de abordagens que identifiquem problemas de natureza funcional, como a ausência ou inadequação

de funcionalidades esperadas pelos usuários, o que pode acarretar a propagação de defeitos relacionados à funcionalidade para as demais fases do processo de desenvolvimento. A próxima seção apresenta a revisão de literatura de trabalhos que buscam identificar aspectos relacionados ao significado em interfaces de usuário.

2.4 Identificação de Significado em Interfaces de Usuário

A identificação de significado em interfaces de usuário tem sido tema de trabalhos recentes tanto na área de Engenharia de Software quanto na de Interação Humano-Computador, visando principalmente o desenvolvimento de aplicações de aprendizado de máquina para entendimento visual de interfaces, acessibilidade, automação e análise de design. As abordagens apresentadas nesta revisão de literatura concentram-se na anotação de significado em diferentes níveis de granularidade, desde a identificação de características semânticas em elementos individuais da interface, como ícones, componentes ou botões, passando por anotações estruturais e identificação de regiões funcionais de interfaces, como menus e rodapés, até a caracterização da interface como um todo.

Em relação à identificação e anotação automática de significado em componentes individuais, como ícones, links, botões e campos de texto, diversos trabalhos foram desenvolvidos visando o aprimoramento de recursos de acessibilidade, de geração de descrições automáticas de interfaces e de suporte a ferramentas de design (Liu et al., 2018; Zang et al., 2021; Chen et al., 2022; Sunkara et al., 2022).

O trabalho de Liu et al. (2018) descreve um método de aprendizagem de anotação semântica de elementos presentes em imagens de interfaces de aplicativos móveis, especificamente botões textuais e ícones, além de alguns elementos estruturais, como menus e rodapés. Essa abordagem utiliza um dicionário lexical de componentes, ícones e botões obtidos após a análise de 720 telas e 73.000 elementos de interface. As anotações podem ser usadas para realizar operações baseadas em semântica, como busca de GUIs com o mesmo significado ou busca de sequências de telas relacionadas a uma mesma funcionalidade em *datasets* de mineração de interação.

Zang et al. (2021) apresentam um método multimodal de anotação de significado de ícones de aplicativos móveis que combina imagens, contexto textual ao redor do ícone e alguns atributos textuais presentes na hierarquia de componentes da UI para inferir sua função. Chen et al. (2022) ampliaram a identificação de ícones de aplicações móveis através da elaboração de um modelo multimodal de classificação em aprendizagem de máquina. O modelo utiliza um dataset anotado contendo 327.879 ícones, associados à informação contextual (textos por perto) e símbolos modificadores nos ícones, visando prover uma classificação de ícones mais completa, que abranja também a classificação de ícones menos frequentemente encontrados em aplicações.

Sunkara et al. (2022) também exploraram a anotação semântica de interfaces móveis. Sua pesquisa produziu um dataset elaborado manualmente contendo mais de 500.000 anotações de ícones, que associam componentes de interface a rótulos textuais que descrevem sua funcionalidade. A pesquisa visou possibilitar o desenvolvimento de aplicações com interpretações mais completas do significado de interfaces, indo além da identificação de tipos de componentes.

Além da identificação de significado em elementos isolados, outros trabalhos têm se dedicado a analisar o significado de áreas ou regiões das interfaces, buscando reconhecer agrupamentos de elementos que, juntos, representam uma unidade funcional ou conceitual. O trabalho de Akpinar and Yeşilada (2017) apresenta uma abordagem de identificação automática de elementos visuais e regiões em páginas web e de seu significado semântico em nível estrutural, tais como cabeçalho, barra lateral ou mecanismo de busca. Para identificar a funcionalidade, foi criado um *dataset* anotado por usuários que classificaram áreas de páginas web reais, pré-divididas automaticamente em blocos visuais, de acordo com sua percepção do papel funcional de cada região, como navegação, conteúdo ou publicidade.

Já o trabalho de Wu et al. (2021) propôs um método de engenharia reversa de modelos de UI a partir de capturas de tela, com o objetivo de reconstruir uma representação hierárquica da interface. A partir dessa representação, é possível segmentar a tela em regiões com papéis distintos, rotulando grupos de componentes que representam aspectos estruturais, como barra de ferramentas, tabela, navegação inferior.

De maneira similar, o WebUI (Wu et al., 2023) é um *dataset* com 400.000

páginas web e metadados anotado com informações semânticas estruturais e de elementos individuais extraídas automaticamente da árvore de acessibilidade dos navegadores. Esse dataset possibilita o desenvolvimento de modelos de aprendizado de máquina destinados ao entendimento visual de interfaces de usuário, como a análise de padrões de regiões de tela e de layout, a classificação de telas com base na aparência visual, a geração de listas de rótulos mais prováveis para uma interface ou a análise de similaridade entre diferentes interfaces.

Por fim, algumas iniciativas têm buscado atribuir significado à interface como um todo, categorizando telas inteiras segundo tópicos ou funcionalidades globais. O *dataset* EnRICO (Leiva et al., 2020) é um conjunto rotulado formado por 1.460 capturas de tela de aplicativos e suas respectivas informações estruturais. As telas são uma amostra do *dataset* RICO (Deka et al., 2017) e foram categorizadas em 20 tópicos de design de UI, que fornecem uma compreensão de alto nível da intenção de cada tela e podem ser utilizadas em aplicações de apoio ao design de interfaces de usuário.

Já a abordagem de Chen et al. (2020a) usa a identificação semântica para descobrir *tags* ausentes que possam ser usadas para descrever imagens em repositórios de compartilhamento de imagens de design. A solução apresentada descreve uma técnica baseada em aprendizado profundo que combina imagens e metadados do design de interface de aplicativos para sugerir *tags* que complementem a descrição de significado de UIs em cinco diferentes aspectos: plataforma, cor, funcionalidade da aplicação, funcionalidade da tela e layout da tela. Além disso, oferece um método de visualização para destacar quais partes das imagens deram origem à *tag* sugerida.

Em outro trabalho mais recente, Wang et al. (2021) propuseram o *Screen2Words*, uma abordagem de resumo automático de interfaces móveis que utiliza aprendizado multimodal (imagem da tela, a estrutura hierárquica da interface e o texto visível na tela) para gerar descrições textuais de alto nível sobre o propósito de uma tela de aplicativo. Esse trabalho contribui também com um *dataset* anotado de telas de aplicações móveis associadas a descrições feitas por anotadores humanos. Esse trabalho pode ser utilizado no desenvolvimento de aplicações como acessibilidade, recuperação de interfaces e geração automática de documentação.

Os trabalhos revisados nesta seção apresentam diversas abordagens de carac-

terização de significado em interfaces de usuário, abrangendo níveis elementares (componentes individuais), intermediários (regiões funcionais) e globais (UI como um todo). No entanto, observa-se que a maioria dessas abordagens está voltada para a classificação de elementos ou telas com base em características visuais ou estruturais, com foco em aplicações como acessibilidade, recuperação de design ou entendimento visual. Poucos trabalhos buscam conectar essa identificação semântica com a verificação da presença das funções esperadas a partir de requisitos ou cenários de uso. Neste sentido, a abordagem proposta neste projeto de pesquisa visa contribuir para a caracterização semântica de interfaces de usuário ao apresentar uma abordagem baseada em heurísticas para identificação de defeitos funcionais em protótipos de interface de usuário. As heurísticas utilizadas na abordagem descrevem diferentes funcionalidades através de suas características específicas e podem ser utilizadas em inspeções funcionais rápidas de protótipos, com ênfase na detecção de defeitos relacionados à ausência ou representação inadequada das funcionalidades especificadas.

2.5 Trabalhos Relacionados

Estudos recentes têm apresentado métodos e ferramentas visando antecipar a identificação de defeitos em protótipos de interface de usuário, abordando diferentes aspectos de garantia da qualidade, tais como a verificação de consistência entre requisitos e protótipos baseada em BDD (Rocha Silva et al., 2019) ou LLMs (Kolthoff et al., 2024), a análise automatizada de interfaces visando identificar problemas de agrupamento semântico de elementos de interação (Duan et al., 2023) e violações de heurísticas de usabilidade (Duan et al., 2024). Esta seção apresenta os quatro trabalhos mais relacionados à abordagem assistida de identificação de defeitos funcionais em protótipos de interface proposta neste projeto de pesquisa.

Rocha Silva et al. (2019) propuseram uma abordagem automatizada para verificar a consistência entre requisitos funcionais descritos em formato *Behavior-Driven Development* (BDD) e protótipos de interface de usuário. A solução utiliza uma ontologia (Silva et al., 2017) para mapear semanticamente os comportamentos descritos nos requisitos com os elementos presentes nos protótipos, permitindo

a validação automática, a nível de interação, de cenários de uso desde as primeiras etapas do desenvolvimento. Para isso, os requisitos funcionais precisam estar formalizados na DSL (Domain-Specific Language) Gherkin[1], com cenários no formato *Given-When-Then*, especificando, respectivamente, condições de ocorrência, eventos e resultados esperados. A ferramenta analisa cada cenário BDD através do XML do protótipo, verificando se os campos existem no protótipo, se os nomes são consistentes com os requisitos e se os tipos de campo utilizados para entrada de dados são compatíveis com o comportamento esperado.

Embora essa abordagem automatize a verificação de cenários de histórias de usuário e elementos de protótipos, ela depende fortemente da existência de especificações formais completas e de protótipos estruturados em formatos como XML, que possam ser lidos por ferramentas de automatização, o que não é o cenário comumente encontrado na indústria de desenvolvimento de software. Além disso, os tipos de verificação realizadas são limitados a rótulos, campos e tipos de elementos presentes na interface. Por outro lado, esta tese apresenta a ferramenta GUIPFI para a realização de inspeções funcionais assistidas em protótipos de interface de diferentes tipos, incluindo os de baixa fidelidade ou não estruturados. Esta ferramenta utiliza um catálogo de heurísticas de design elaborado a partir do conhecimento tácito de funcionalidades recorrentes em sistemas de informação, o que possibilita a detecção de defeitos funcionais mesmo na ausência de requisitos formalizados, podendo ser utilizada em cenários da indústria de software com diferentes graus de maturidade em engenharia de requisitos. Aspectos de avaliação de elementos de exibição, entrada de dados e interação são cobertos pelo *checklist* de inspeção funcional (UIProtoCheck).

Duan et al. (2023) propuseram um conjunto de cinco diretrizes de agrupamento semântico, com foco na detecção automatizada de problemas de organização lógica de elementos em interfaces de usuário. Sua abordagem utiliza protótipos com hierarquia de layout estruturada e elementos com descrições textuais (p. ex.: XML de Android, RICO dataset) a partir dos quais são elaboradas métricas baseadas em *embeddings* semânticos (Sentence-BERT) e análise de *clusters* para identificar violações como agrupamentos incoerentes, redundância e falta de rótulos explicativos.

[1] https://cucumber.io/docs/gherkin/reference/

A solução apresentada por Duan et al. (2023) oferece contribuições importantes na verificação automatizada de aspectos estruturais da interface, com foco na coerência semântica entre elementos de UI. Nesse sentido, a abordagem proposta neste trabalho complementa tais esforços ao propor uma abordagem voltada para a validação de aspectos funcionais de protótipos, possibilitando a identificação de defeitos relacionados à representação, ausência ou inadequação de funcionalidades esperadas, e ampliando o escopo de verificações precoces que podem ser realizadas visando à qualidade de interfaces de usuário.

Recentemente, outro trabalho de Duan et al. (2024) propôs uma abordagem para a avaliação automática de *mockups* de interface de usuário com uso de *Large Language Models* (LLMs), especificamente o modelo GPT-4[2]. A proposta consiste em um plugin para o Figma[3] que permite realizar avaliações automáticas com base em diferentes conjuntos de heurísticas selecionadas pelo usuário, como as heurísticas de usabilidade de Nielsen (Nielsen, 1994b), princípios de design visual e diretrizes de agrupamento semântico (Duan et al., 2023). A aplicação fornece ao LLM uma representação do *mockup* em formato JSON junto com o conjunto de heurísticas selecionadas para identificar violações de design. A ferramenta utiliza um processo iterativo no qual designers podem revisar o *mockup* com base na avaliação gerada pela ferramenta, rodar novamente a avaliação e continuar refinando a interface.

Os resultados mostraram um bom desempenho na detecção de problemas de alinhamento, espaçamento e tamanho de elementos, de representações textuais e de agrupamento semântico, mas mostraram limitações em relação à avaliação de usabilidade com heurísticas de Nielsen (Nielsen, 1994b), especialmente naquelas cujas avaliações dependem de interações ou fluxos, tais como *Ajuda e Documentação* ou *Prevenção de Erros*. A principal contribuição desta tese, por outro lado, é a de oferecer suporte à detecção precoce de defeitos funcionais em protótipos de interface, incluindo os relacionados à ausência de características essenciais ou inconsistências na representação de funcionalidades conhecidas, independentemente da existência de estrutura formal (como JSON) ou integração com ferramentas específicas de design.

[2]https://openai.com/index/gpt-4-research/
[3]https://www.figma.com

Kolthoff et al. (2024) apresentaram avanços significativos ao apresentar uma abordagem semiautomatizada baseada em LLMs para validação entre requisitos funcionais, descritos como história de usuário, e protótipos de interface de usuário. O método realiza a validação ao transformar os protótipos em uma representação textual abstrata e, a partir dessa abstração, utiliza LLMs para classificar cada história de usuário como implementada ou não. Além disso, o sistema sugere componentes de interface adequados para apoiar a implementação de requisitos ainda ausentes.

No entanto, a abordagem de Kolthoff et al. (2024) depende de requisitos formalizados como histórias de usuário, o que não costuma ser uma realidade comumente encontrada na indústria de desenvolvimento de software, e se limita a analisar protótipos representados com estrutura passível de análise textual, como os representados em DSLs de ferramentas de prototipação. Outra limitação dessa abordagem é o fato de que ela não cobre falhas funcionais que vão além da presença ou ausência de componentes diretamente mapeáveis. Em contraste, a proposta desta tese adota um enfoque baseado em conhecimento de domínio, utilizando heurísticas de design funcional (CHIF/CHIF-E) e um *checklist* de inspeção semântica (UIProtoCheck), o que permite a análise de protótipos de diferentes tipos e fidelidades, incluindo aqueles não estruturados ou desprovidos de especificações formais, como *wireframes* em formato de imagem ou papel. Enquanto a abordagem de Kolthoff et al. (2024) busca identificar a cobertura de requisitos explícitos, este projeto de pesquisa é voltado para a detecção de defeitos funcionais relacionados à representação, ausência ou inconsistência de funcionalidades esperadas, considerando tanto requisitos explícitos quanto conhecimento tácito de domínio.

A análise desses trabalhos evidencia o avanço recente nas soluções voltadas à identificação de problemas em protótipos de interface de usuário, com destaque para abordagens que exploram desde a validação automatizada de requisitos formais até a geração de *feedback* visual baseado em heurísticas e aprendizado de máquina. No entanto, tais abordagens carecem de suporte à avaliação da representação de funcionalidades em protótipos de interface. Além disso, observa-se que essas iniciativas apresentam limitações em termos de aplicabilidade, escopo funcional ou dependência de artefatos formais e estruturados. A proposta desta

tese busca oferecer uma alternativa a essas limitações através de uma abordagem de inspeção funcional que integra conhecimento de diferentes domínios de aplicação, heurísticas para funcionalidades básicas e específicas e suporte de software para guiar inspeções voltadas à identificação precoce de defeitos funcionais, independentemente do tipo de protótipo ou do nível de formalização dos requisitos.

2.6 Originalidade e Relevância da Pesquisa

Como visto nos trabalhos apresentados nas seções anteriores, há um conjunto consolidado de técnicas de avaliação de usabilidade, ferramentas voltadas à verificação de interfaces implementadas, além de diversas abordagens de caracterização de significado em interfaces de usuário. Entretanto, observa-se uma lacuna relevante na literatura de abordagens que ofereçam suporte à verificação sistemática de funcionalidades ainda na fase de prototipação.

Visando preencher a lacuna identificada, a presente pesquisa propõe uma abordagem de inspeção funcional voltada para protótipos de interface, com o objetivo de apoiar desenvolvedores de software na identificação precoce de falhas e complementar as técnicas tradicionais de avaliação, evitando o avanço de defeitos funcionais não identificados para fases mais custosas do processo de desenvolvimento. Desta forma, a presente pesquisa oferece contribuições teóricas e práticas para ampliar o alcance das técnicas de detecção antecipada de defeitos em interfaces de usuário, através de uma abordagem adaptável a diferentes contextos que possibilita o desenvolvimento de interfaces mais robustas desde suas etapas iniciais.

Capítulo 3

Ambiente: Prototipação no Desenvolvimento de Software

De acordo com a literatura, a prototipação de interface de usuário é um recurso amplamente utilizado nas diversas fases do desenvolvimento de um software, principalmente na engenharia de requisitos, para explorar ideias e facilitar a comunicação com o cliente (Käpyaho and Kauppinen, 2015). Além disso, protótipos são o artefato mais utilizado para melhorar a comunicação de times de desenvolvimento de software (Garcia et al., 2017).

Além das atividades relacionadas à elicitação de requisitos, os protótipos também são utilizados no desenvolvimento incremental, na melhoria da qualidade, além de análises de viabilidade de mercado, técnica e de usabilidade (Bjarnason et al., 2021). Bellomo et al. (2013) descrevem um processo de desenvolvimento onde protótipos e *sprints* são desenvolvidos paralelamente, de forma a atrelar requisitos e atributos de qualidade. Outros trabalhos também utilizam a prototipação em processos iterativos para elicitar requisitos de qualidade (Behutiye et al., 2020; Karhapää et al., 2021).

Diante dos diferentes usos da prototipação de IU relatados na literatura, foi realizada uma pesquisa de opinião com profissionais da indústria de software, através da aplicação de um questionário eletrônico, visando caracterizar como eles têm utilizado a prototipação e qual a relevância de seu uso para o processo de desenvolvimento de software.

Especificamente, o estudo realizado buscou entender: (i) como a prototipação tem sido utilizada; (ii) de que maneira ela está inserida no processo de desenvolvimento de software; e (iii) de que forma ela é utilizada nas atividades de garantia da qualidade de software. A análise destes três aspectos possibilitou avaliar a relevância do uso da prototipação no processo de desenvolvimento em aspectos relacionados à qualidade de software. O resultado desta pesquisa foi publicado em De Macedo et al. (2022).

3.1 Metodologia

O método de pesquisa adotado neste estudo é formado por duas etapas: Pesquisa de Opinião (Glasow, 2005) e Análise Qualitativa com procedimentos iniciais de codificação da *Grounded Theory* (Corbin and Strauss, 2014). A pesquisa de opinião é um método de pesquisa utilizado para coletar dados e descrever aspectos específicos de uma população dada. Esta pesquisa seguiu os procedimentos indicados por Linåker et al. (2015) e teve como objetivo verificar como a prototipação é utilizada; como se integra ao processo de desenvolvimento adotado e como esse uso se reflete na qualidade dos produtos de software desenvolvidos. Desta forma, foram elencadas as seguintes questões de pesquisa:

QP1: Com qual finalidade os protótipos de interface são elaborados?

QP2: Como a prototipação está inserida no processo de desenvolvimento de software?

QP3: Como a prototipação está relacionada com atividades de garantia da qualidade dos produtos de software desenvolvidos?

A partir destas questões de pesquisa, foram elaboradas as perguntas da pesquisa de opinião, começando com questões mais gerais sobre o perfil do participante, seguidas de perguntas mais específicas voltadas aos aspectos investigados para responder às questões de pesquisa definidas. O tempo de resposta do questionário foi estimado em 15 minutos.

A pesquisa de opinião foi realizada por meio de um questionário eletrônico[1],

[1]Material suplementar: https://doi.org/10.6084/m9.figshare.20407329

composto de três partes. A **primeira parte** compreende o Termo de Consentimento Livre e Esclarecido (TCLE), que descreve os objetivos da pesquisa, riscos, benefícios e procedimentos adotados, e cuja aceitação pelo participante é obrigatória para que ele proceda às demais perguntas do questionário. O Apêndice B apresenta o TCLE utilizado nesta pesquisa.

A **segunda parte** contém perguntas de caracterização do perfil dos participantes e da empresa para a qual trabalham. Estas perguntas envolvem: o tempo de experiência, o grau de escolaridade, a finalidade da empresa, a quantidade de desenvolvedores da empresa, a quantidade de pessoas no time de desenvolvimento e os papéis desempenhados no time. A **terceira parte** do questionário compreende perguntas tanto abertas quanto fechadas relacionadas ao uso de prototipação. As questões desta parte são apresentadas na Tabela 3.1.

Para avaliar o questionário, foi realizado um estudo piloto com um voluntário, profissional da indústria de desenvolvimento de software com mais de 30 anos de experiência. Durante o preenchimento do questionário, o participante relatou algumas dúvidas em relação às perguntas, que foram ajustadas para remover ambiguidades e torná-las mais claras.

O público-alvo desta pesquisa é formado por profissionais de times de desenvolvimento da indústria de software que utilizam prototipação em suas atividades de trabalho. Desta forma, o questionário foi divulgado no *LinkedIn*[2]. Além disso, o link para a pesquisa foi enviado via mensagem direta para profissionais da rede de contatos dos pesquisadores envolvidos neste trabalho que atendessem ao perfil da pesquisa. Também foi solicitado que estes profissionais encaminhassem o *link* do questionário para sua rede de contatos que atendessem ao mesmo perfil.

As respostas do questionário sobre o uso de prototipação foram analisadas com os seguintes procedimentos de codificação da *Grounded Theory* (Corbin and Strauss, 2014): codificação aberta (*open coding*) e codificação axial. Na codificação aberta, são definidos previamente os aspectos que guiam a análise, a partir dos quais os dados obtidos são rotulados com códigos criados segundo estes aspectos. No caso deste estudo, a análise inicialmente foi realizada pelo primeiro pesquisador, que a conduziu com base nas questões de pesquisa, de acordo com os seguintes aspectos: diferentes formas de uso de protótipos, o papel da prototipação no

[2]https://www.linkedin.com

Tabela 3.1: Perguntas do questionário relativas ao uso de prototipação.

ID	Pergunta	Questão de Pesquisa
Q1	No seu trabalho, sua equipe costuma elaborar protótipos das telas do software a ser desenvolvido?	–
Q1.1	Como os protótipos da interface são elaborados?	QP2
Q1.2	Conte-nos sobre o processo de elaboração desses protótipos. Como a demanda chega à equipe e quais as primeiras ações tomadas?	QP2
Q1.3	Durante a elaboração dos protótipos de tela, existe comunicação com o cliente? Conte-nos como essa comunicação ocorre e qual o objetivo.	QP1 e QP2
Q1.4	Quantas vezes o protótipo de telas elaborado é apresentado ao cliente?	QP2
Q1.5	O que acontece se o cliente discordar de algo que foi projetado no protótipo de telas?	QP2 e QP3
Q1.6	Os protótipos de telas que foram elaborados durante o projeto são utilizados em outras fases do desenvolvimento?	QP3
Q1.7	Se você respondeu "Sim"na pergunta anterior, conta pra gente em quais momentos ele é utilizado durante o processo de desenvolvimento.	QP1 e QP3
Q1.8	Seu time realiza alguma ação para verificar se o software implementado corresponde ao que foi especificado nos protótipos de telas e outros artefatos de desenvolvimento do produto? Se sim, como é feita essa validação?	QP3
Q1.9	Você utiliza os protótipos de telas para automatizar alguma etapa do desenvolvimento? De que forma?	QP1 e QP3
Q1.10	Na sua opinião, a prototipação de telas tem alguma utilidade no seu processo de desenvolvimento? Por quê?	QP1 e QP2
Q2	Que outra forma de comunicação você usa para conversar com seu cliente sobre a solução a ser desenvolvida?	QP2

processo de desenvolvimento e a prototipação relacionada a atividades de garantia da qualidade do produto de software. Os resultados foram revisados por outros dois pesquisadores. Um deles possui mais de 4 anos de experiência em estudos qualitativos e ensino de qualidade de software, além de ampla experiência na indústria. O outro pesquisador envolvido tem mais de 8 anos de experiência em estudos qualitativos e educação em qualidade de software. Como resultado, foram identificados conceitos-chave relativos ao uso de prototipação na indústria, tais como especificação de requisitos, análise de negócio, protótipos de baixa fidelidade, protótipos de alta fidelidade e seus diferentes usos.

Na codificação axial (Corbin and Strauss, 2014), foi analisado como os códigos obtidos durante a codificação aberta se relacionam entre si, formando uma rede de conceitos e relações. A partir dessas relações, foi possível obter uma visão de como a prototipação está inserida no processo de desenvolvimento de software, com qual finalidade ela é utilizada e de que forma ela está relacionada às atividades de garantia da qualidade. Os resultados desta análise são detalhados na seção seguinte. Uma imagem geral da rede de códigos resultante da análise qualitativa e uma lista de códigos e respectivas citações estão disponíveis no material suplementar[3].

3.2 Resultados

Após a divulgação do questionário, foram obtidas 56 respostas, das quais 39 (69,6%) alegam utilizar protótipos no desenvolvimento de software. Foram obtidas 42 respostas de 24 empresas diferentes, sendo que 14 participantes não identificaram a empresa onde trabalham. Das empresas identificadas, cinco empresas possuíam mais de um respondente, conforme apresentado na Tabela 3.2.

Em relação ao papel que os participantes exercem no time de desenvolvimento, foram obtidas respostas principalmente de analistas de sistemas, gerentes de projeto, programadores e desenvolvedores júnior, pleno e sênior. A Figura 3.1 mostra a proporção de cada papel.

Quanto ao tempo de experiência dos participantes, foi verificada uma maior

[3]Material suplementar: https://doi.org/10.6084/m9.figshare.20407329

Tabela 3.2: Lista de empresas com mais de um respondente.

Empresa	Quantidade
E1	3
E2	9
E4	3
E6	4
E10	3

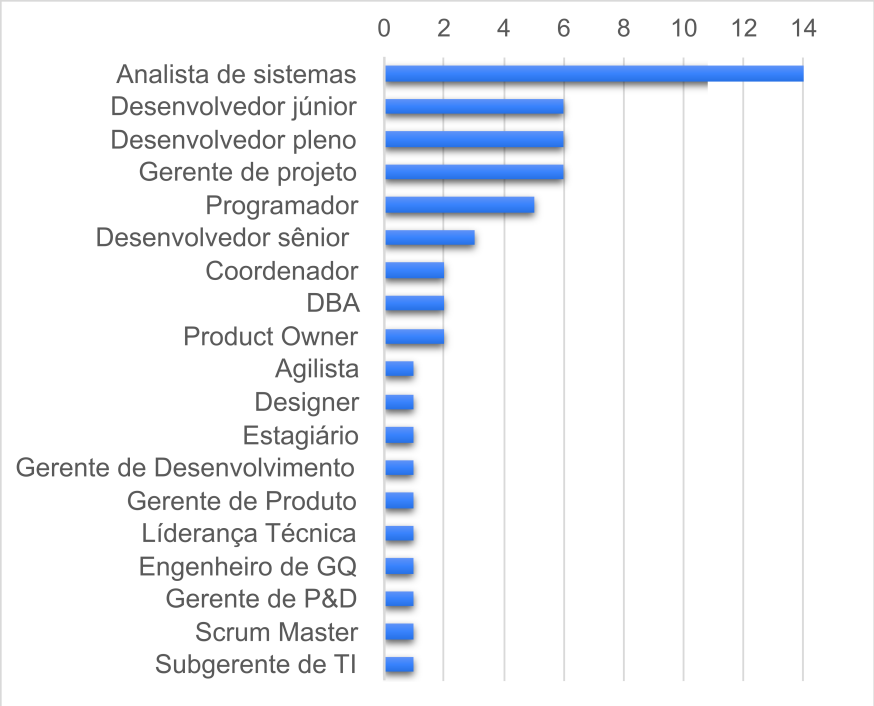


Figura 3.1: Papéis dos participantes nos times de desenvolvimento.

ocorrência de participantes com experiência de até 10 anos, sendo que essa frequência vai diminuindo conforme aumenta o tempo de experiência. A Figura 3.2 ilustra esta distribuição.

Em relação à finalidade das empresas dos participantes, 30 (53,6%) trabalham para empresas de desenvolvimento de software. Outras finalidades que se destacam são de funcionários de empresas cuja finalidade principal não é a de desenvolvimento de software (comércio de bens e serviços) e de empresas que terceirizam a prestação de serviços de desenvolvimento de software. Há também

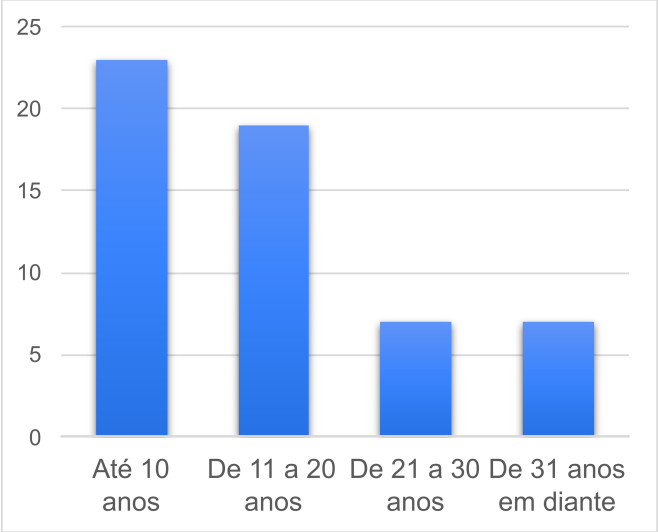


Figura 3.2: Tempo de experiência dos participantes.

participantes de empresas que desenvolvem e comercializam softwares de prateleira, universidades, institutos de pesquisa, empresas de realização de cursos, de administração de bancos de dados e do setor público, com um respondente cada, como ilustrado na Figura 3.3.

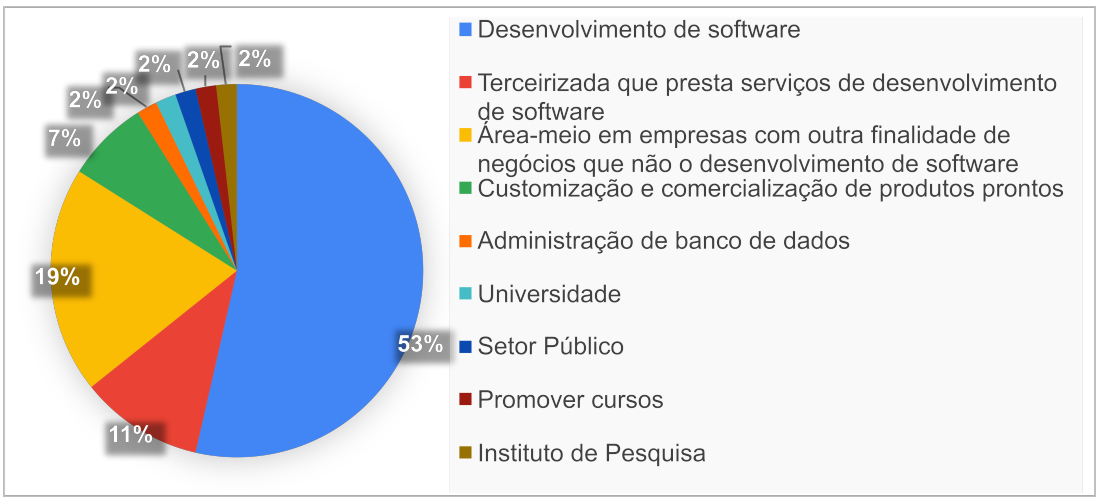


Figura 3.3: Finalidade das empresas dos participantes.

Nas próximas seções serão apresentados os resultados das demais questões apresentadas no questionário, tomando por base cada uma das questões de pes-

quisa definidas anteriormente.

3.2.1 QP1: Finalidade da Prototipação

Nesta seção, são apresentados os resultados relacionados às motivações dos desenvolvedores para utilizarem a prototipação, coletados das respostas às questões Q1.2, Q1.7 e Q1.9. Além disso, apresenta-se ainda a percepção destes desenvolvedores em relação aos benefícios obtidos com o uso da prototipação, coletada da questão Q1.10. A Figura 3.4 apresenta um trecho da rede de códigos criada neste trabalho durante a análise qualitativa. Nela, são apresentados os códigos relacionados à elaboração de protótipos de baixa fidelidade (Walker et al., 2002) que representam as finalidades com que os protótipos são elaborados.

As respostas fornecidas pelos participantes mostraram que os protótipos são bastante utilizados nas fases iniciais do desenvolvimento. As principais finalidades são as de elicitação de requisitos ou de histórias de usuário, exploração de ideias e soluções pela equipe de desenvolvimento, de forma a alinhar expectativas. Também são utilizados para analisar soluções existentes e tecnologias que serão necessárias para desenvolver o produto: *"Durante este processo da criação do protótipo de baixa fidelidade faz-se brainstorming com a equipe para obter ideias e o que será preciso estudar de novas tecnologias ou componentes para o projeto. (...) Depois é feita uma pesquisa benchmarking pelo Designer para entender melhor as soluções existentes."* (P1).

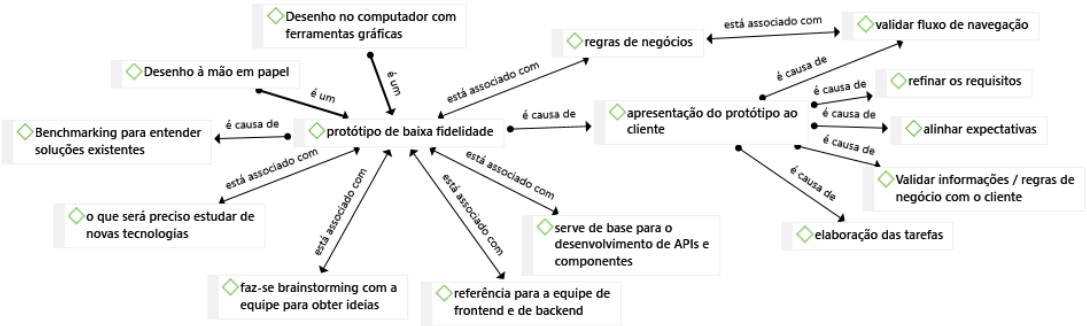


Figura 3.4: QP1 - Finalidade de uso da prototipação - protótipos de baixa fidelidade.

Após a elaboração dos protótipos iniciais, eles são apresentados aos clientes

para validação dos requisitos, das regras de negócio e dos fluxos de navegação: *"Com a prototipação, nos reunimos com o cliente para refinar os requisitos e o fluxo de navegação."* (P10); *"É nesta fase que as prototipações entram para validar as regras de negócio e então entrar no ciclo do dev team."* (P41). A partir desta primeira etapa, os protótipos são utilizados na elaboração de outros artefatos, como diagramas de casos de uso, elaboração do *backlog* de produto, planejamento de *sprints* e definição de tarefas para os desenvolvedores: *"É utilizado durante a criação do backlog e no planejamento da Sprint."* (P1).

Alguns participantes relataram que os protótipos são inicialmente criados sem muitos detalhes (baixa fidelidade) para depois das primeiras validações com o cliente serem complementados: *"Com base no protótipo de baixa fidelidade, o designer inicia a criação das telas de alta fidelidade."* (P1). Essas atualizações são feitas para cobrir outras funcionalidades do software ou refiná-lo em desenhos gráficos de alta fidelidade, protótipos navegáveis ou protótipos funcionais na linguagem de programação-alvo, conforme relata o participante P29: *"Os protótipos são complementados para cobrirem todos os cenários mapeados e serem utilizados nas etapas seguintes."*.

No caso de protótipos navegáveis, o protótipo é fornecido ao cliente para utilização, como citam os participantes P43 e P55: *"P43 - É enviado um link do protótipo, onde o cliente pode fazer anotações em cada tela. Ao final de cada sprint, é marcada uma reunião para avaliar tudo o que foi feito."*; *"P55 - (...) sempre informamos o cliente e deixamos que ele mesmo experimente antes de começarmos a fazer algo."*. Essa avaliação permite a realização de discussões mais ricas entre desenvolvedores e clientes sobre os requisitos da aplicação a ser desenvolvida: *"Os protótipos são o passo inicial para especificação visual e de experiência de uso do software. (...) É um passo importante para fomentar discussões ricas entre os envolvidos."* (P29).

Este tipo de protótipo é utilizado para que os clientes possam ter uma visão mais profunda do software e terem discussões mais ricas sobre o produto a ser desenvolvido. A Figura 3.5 ilustra os códigos relacionados às finalidades de uso de protótipos de alta fidelidade.

As versões de protótipos mais complexas são aproveitadas em alguns casos na etapa de implementação para criar estruturas de código reaproveitáveis ou

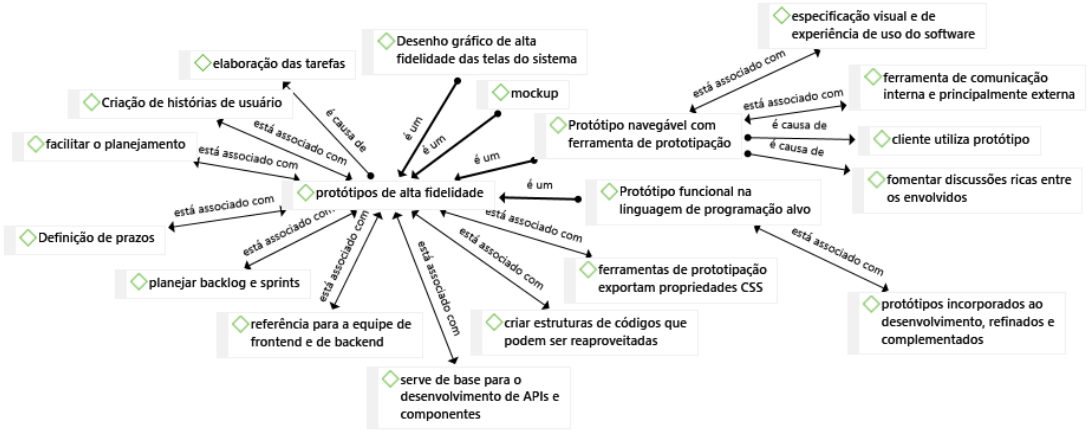


Figura 3.5: QP1 - Finalidade de uso da prototipação - protótipos de alta fidelidade.

servirem de base para a criação de componentes: *"Utilizo para criar estruturas de código que podem ser reaproveitadas."* (P54); *"Ele serve de base para o desenvolvimento de APIs e componentes."* (P8). Além disso, podem ser utilizados para exportar trechos de código de apresentação gerados pela ferramenta de prototipação adotada: *"Como desenvolvo também interfaces de software, eu considero fundamental ter um protótipo, uma vez que as ferramentas de prototipação exportam propriedades CSS."* (P18). No caso de protótipos funcionais, eles são incorporados à implementação e aprimorados até se tornarem o produto final: *"Os protótipos aprovados são incorporados ao desenvolvimento e não se perdem. Eles são refinados e complementados."* (P29).

Quanto à percepção dos participantes dos benefícios obtidos com a utilização de protótipos no desenvolvimento de software, foram apontados um menor custo de desenvolvimento, agilização na implementação, diminuição de retrabalho e previsão de erros antes da implementação.

> *"P32 - A partir dos protótipos podemos validar os requisitos junto ao cliente e realizar ajustes com menor custo, antes do desenvolvimento de cada sprint."*

> *"P55 - É uma forma de você prever erros futuros antes mesmo de ser programado, evitando correções e mudanças de usabilidade."*

Além disso, também foi indicado que o protótipo pode ser uma maneira de

engajar o cliente no processo de desenvolvimento e de acelerar as atividades de validação do produto. As citações dos participantes P3, P5 e P41 ilustram essas utilidades de uso dos protótipos:

> *"P3 - É fundamental para adquirir um entendimento claro do problema e do usuário, além de ser muito mais barato de desenvolver e descartar versões para provar hipóteses e mitigar riscos em comparação com software de produção."*

> *"P5 - Comprometimento do cliente durante o processo e agilizar a implementação."*

> *"P41 - Acredito que as etapas de validação são aceleradas com a prototipagem. Quando o time não a faz, é comum que revalidações sejam feitas após a entrega de produtos."*

Os resultados apresentados nesta seção mostram que a prototipação, além de ser utilizada durante a elicitação de requisitos, também é utilizada em outras etapas do processo de desenvolvimento, tais como a implementação, a validação, o treinamento e a implantação. Nas próximas seções, é detalhado como os protótipos desenvolvidos e suas aplicações estão inseridos nos processos de desenvolvimento adotados pelas empresas.

3.2.2 QP2: Prototipação no Processo de Desenvolvimento

Nesta seção, é analisada a forma como a prototipação é incorporada aos processos de desenvolvimento de software adotados em diferentes empresas. Para responder a esta questão de pesquisa, foram analisadas as respostas às questões Q1.1, Q1.2, Q1.3, Q1.4, Q1.5, Q1.10 e Q2.

A análise da utilização de protótipos nas empresas revelou diferentes processos de desenvolvimento. A Figura 3.6 ilustra os códigos relacionados ao processo de desenvolvimento com uso de prototipação.

Em geral, nos processos adotados pelas empresas analisadas, o processo costuma iniciar a partir de uma demanda do cliente. Esta demanda origina uma especificação de requisitos ou de histórias de usuários, a partir da qual os protótipos são elaborados.

Figura 3.6: QP2 - Prototipação no processo de desenvolvimento adotado pelas empresas.

> *"P36 - O cliente solicita uma demanda. Esta demanda gera uma especificação de requisitos e de projeto, que é aprovada pelo cliente. A prototipação das telas, caso seja feita, faz parte destas especificações."*

> *"P47 - A equipe participada da reunião de levantamento de requisitos com cliente, e em seguida elaboramos o protótipo."*

> *"P48 - Os protótipos são elaborados como a primeira etapa do nosso processo de desenvolvimento, esboçando as ideias do cliente com a prévia simplificada das funcionalidades requisitadas."*

Também foram identificados processos em que os protótipos são criados através de imersões com o cliente, em seções de *Product Discovery* (Brhel et al., 2015), nas quais ele participa ativamente, indicando as funcionalidades necessárias enquanto os protótipos são desenvolvidos. Desta forma, são exploradas diferentes soluções para as funcionalidades requisitadas pelo cliente. A cada iteração, os protótipos são apresentados e validados pelo cliente.

> *"P35 - É realizada imersão com o Cliente para entender o problema a ser resolvido e seu contexto. Com o entendimento, são elaborados os protótipos para concretizar o entendimento da solução para o problema."*

> *"P43 - O projeto é desenvolvido em parceria com o cliente, indicando quais funcionalidades ele precisa, sendo realizadas paralelamente."*

As sessões de *Product Discovery* também são utilizadas no desenvolvimento de software para o mercado, sem que a demanda se origine de um cliente específico. O processo inicia-se por uma análise de negócio, que envolve a definição de objetivos estratégicos e de cadeia de valor, análise de viabilidade e estimativas de custo. Uma vez que o produto é considerado viável por essa análise inicial, o designer ou time de UX realiza o processo de *design research* (Bayazit, 2004), onde protótipos são elaborados de forma exploratória com a participação de POs, designers e marketing.

> *"P29 - Durante o workshop de Discovery ocorrem checkpoints de design onde levamos esses protótipos como resultado do entendimento de uma interação, muito mais com o objetivo de dar suporte a discussões para validação do entendimento e refinamento."*

> *"P41 - Quando o produto se conecta com os objetivos estratégicos de negócio, há viabilidade técnica e foi refinado a níveis de regras de negócio e cadeia de valor, inicia-se um discovery mais aprofundado com o time de discovery & research, product owners, product marketings e product designers."*

> *"P20 - A demanda chega através dos stockholders, o PO faz as primeiras análises de negócio e em seguida o designer inicia o processo de design research, onde são desenvolvidos os protótipos de baixa, média e alta fidelidade."*

Alguns participantes revelaram usar processos de desenvolvimento mais formais. Nesses processos, uma vez que há uma demanda, que pode ser gerada pelo *stakeholder*, gerente de projeto, *product owner*, ou durante reuniões de planejamento, são elaborados o projeto de alto nível e uma proposta comercial, na qual são definidas as estórias de usuário a serem desenvolvidas. Uma vez que o negócio é aprovado, é elaborada a especificação de requisitos, a partir da qual são elaborados os protótipos.

> *"P10 - Após a demanda do cliente e a análise crítica de atendimento e estimativa de recurso, é iniciado levantamento de requisitos. Após essa fase é que utilizo a prototipação para refinar os requisitos e o fluxo de navegação."*

Além da análise dos processos adotados, verificou-se a consistência dos processos dentro das empresas. Para tanto, foram analisadas as respostas dos participantes pertencentes às empresas E1, E2, E4, E6 e E10, que obtiveram mais de um participante, conforme listado na Tabela 3.2.

Nesta análise, verificou-se que, na mesma empresa, desenvolvedores seguiam processos de desenvolvimento diferentes, principalmente em relação à forma como os protótipos são utilizados ao longo das etapas do desenvolvimento. Foram encontradas divergências em relação ao uso de protótipos em outras fases do desenvolvimento (Q1.6 - empresas E1, E2, E4, E6 e E10), à atualização do protótipo ao longo do projeto (Q1.5 - empresas E1, E2, E4 e E10), quanto à verificação da implementação versus o protótipo (Q1.8 - empresas E2, E4 e E10) e quanto ao uso, ou não, de protótipos de um modo geral (Q1 - empresas E2, E4 e E10).

Nas respostas obtidas para a questão Q1, foi verificado que 17 participantes (30,4%) alegaram não utilizar prototipação em suas atividades de trabalho. Isso pode demonstrar que ainda existe uma proporção grande de empresas em que não se tem a percepção da importância do uso de protótipos em seus processos de desenvolvimento. Em alguns casos, o desenvolvedor trabalha apenas com implementação de software de apoio a outros desenvolvedores, como bibliotecas ou APIs.

> *"P4 - Trabalho em uma equipe que evolui os produtos da engenharia, como implementando CI/CD, melhorando libs internas etc."*

Por outro lado, a análise das respostas destes desenvolvedores mostrou casos em que isto pode ter ocorrido devido a uma visão parcial do processo de desenvolvimento, em que o desenvolvedor tem conhecimento apenas das atividades desempenhadas pelo seu próprio papel, como pode ser observado em algumas respostas à pergunta Q2.

> *"P13 - Eu não converso com os clientes; outro funcionário hierarquicamente acima de mim que se comunica com os clientes."*

> *"P49 - Dentro da cadeia de desenvolvimento de um produto, o time de desenvolvimento costuma receber uma documentação de design já pronta com a descrição das funcionalidades e aspectos visuais do que deve ser implementado."*

> *"P56 - Quem desenvolve os wireframes é o time de design."*

Nesta seção, verificou-se que a prototipação é utilizada em processos com diferentes níveis de complexidade e relacionada a diferentes tipos de artefatos. Também foi verificado que participantes da mesma empresa relataram o uso da prototipação de diferentes maneiras, como em diferentes times da mesma empresa que enxergam e utilizam a prototipação de formas diversas. Na próxima seção, são apresentadas as relações entre a prototipação e as atividades associadas à qualidade de software.

3.2.3 QP3: Prototipação e Qualidade de Software

Nesta seção são apresentados os resultados da pesquisa sobre atividades relacionadas à qualidade de software durante o uso de prototipação. Os resultados foram obtidos das respostas às questões Q1.5, Q1.6, Q1.7, Q1.8 e Q1.9.

Na fase de validação de software, a prototipação é utilizada em diversas atividades. A Figura 3.7 ilustra os códigos relacionados às atividades de qualidade de software com o uso de prototipação. Em relação à fase de análise e validação da qualidade, foi verificado que os protótipos são usados para revisar a implementação de acordo com os requisitos solicitados.

> *"P32 - Durante a validação do requisito solicitado versus desenvolvido."*

> *"53 – Utiliza na fase de testes para verificar se a implementação corresponde aos protótipos."*

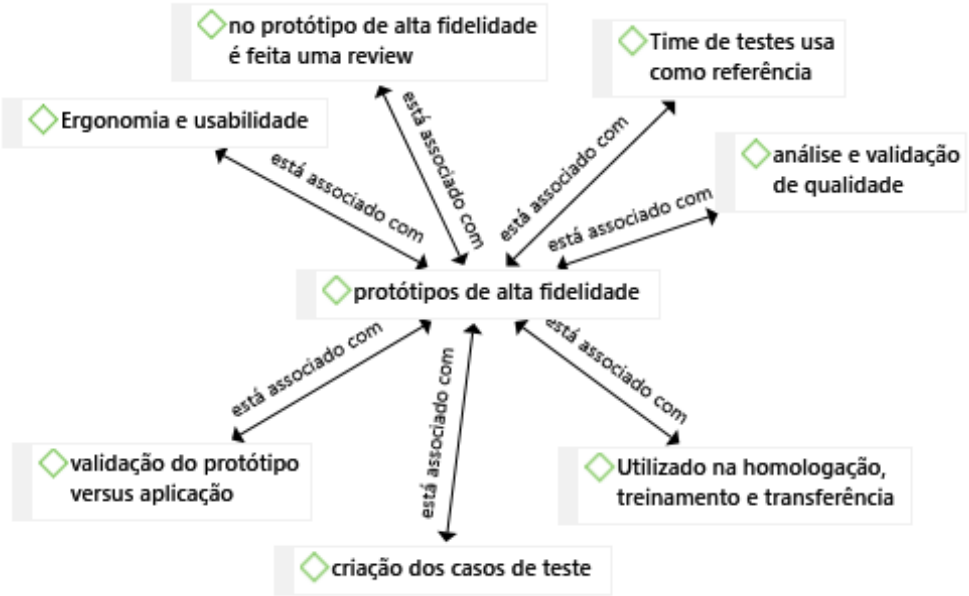


Figura 3.7: QP3 - Códigos relacionados às atividades de qualidade de software com uso de prototipação.

Nas respostas às questões Q1.8 e Q1.9, também foi identificado que os protótipos são usados como referência pelos times de testes na criação de casos de teste e para guiar testes exploratórios.

> *"P24 - Bastante utilizado para criação dos casos de teste. (...) utilizado pelo QA para validação do layout desenvolvido."*

> *"P8 - Um grupo de usuários é escolhido para testar o software. Nós ainda aplicamos testes de ponta a ponta."*

> *"P41 - O time de Quality Assurance projeta um ciclo de qualidade de ponta a ponta. É comum os protótipos serem usados para testes no ciclo, validação de componentes, testes automatizados de interface etc."*

Além das diferentes atividades relatadas, foram encontrados alguns problemas na utilização de protótipos durante as atividades de qualidade de software. Primeiramente, identificou-se que alguns participantes não verificam se o software

desenvolvido corresponde aos protótipos (Q1.8). No caso de protótipos de baixa fidelidade, há situações em que não é necessário que a implementação seja igual ao protótipo, principalmente quando este é criado de maneira exploratória, sem muitos detalhes e com a intenção de identificar requisitos e demonstrar funcionalidades: *"Sim, durante os testes de validação. Mas não há obrigatoriedade de seguir o que foi definido."* (P50). No entanto, foram identificados casos em que o protótipo desenvolvido é de alta fidelidade, mas ainda assim esta verificação não é realizada, como nos dos participantes P26 e P54.

Também foram identificados diversos participantes que alegaram não utilizar protótipos em outras fases do desenvolvimento, ou que usam durante a implementação, mas não em atividades de verificação, validação e testes: *"A partir dos protótipos, definimos quais elementos serão componentes genéricos, quais já devem existir e afins."* (P11).

Outro problema encontrado foi que alguns participantes responderam que os protótipos não são atualizados caso o cliente não concorde com algo que foi projetado nos protótipos de tela (Q1.5), sendo as modificações implementadas diretamente no código. Desta forma, identifica-se o descarte de um artefato que poderia ser útil em outras fases do desenvolvimento, como implementação ou testes.

> *"41 - Times sem Product Designers geralmente não prototipam ou, se prototipam, fazem as alterações na documentação e isso é refletido direto no produto, sem alterações de mock-up."*

Além disso, verificou-se que alguns dos participantes que alegaram não manter os protótipos atualizados afirmaram utilizá-los nas fases de implementação, testes, homologação, treinamento e implantação (Q1.7). Assim, foram identificadas situações em que o uso de um artefato desatualizado pode invalidar as atividades relacionadas à qualidade de software.

> *"28 – Utiliza na implementação como instrumento de consulta do desenvolvedor."*

> *"53 – Utiliza na fase de testes para verificar se a implementação corresponde aos protótipos."*

As implicações envolvidas na adoção de prototipação no desenvolvimento que impactam a qualidade de software são discutidas na próxima seção.

3.3 Discussão

A análise dos resultados revelou que diferentes modelos de processo de desenvolvimento de software comportam o uso de prototipação de interface, mostrando que esta é uma ferramenta versátil, de fácil utilização e de grande capacidade de adaptação. Verificou-se que os protótipos não foram usados somente na elicitação de requisitos, para explorar opções e definir um entendimento comum da aplicação com o usuário. Além disso, serviram para identificar novas tecnologias a serem estudadas, especificar histórias de usuário, identificar regras de negócio e definir fluxos das informações através das telas. Também foram utilizados em tarefas relacionadas ao planejamento do projeto, como na criação do *backlog*, na definição de *sprints* e na elaboração de tarefas. Finalmente, foram utilizados também como base para desenvolvimento de código *front-end* e *back-end*, na realização de revisões, como apoio a testes, treinamento e implantação.

Estes resultados alinham-se aos relatados por Garcia et al. (2017), que constataram que os protótipos são o principal artefato utilizado durante ciclos iterativos de desenvolvimento devido ao fato de serem utilizados com diversos propósitos, desde servirem de referência para o desenvolvimento de código *front-end* e *back-end* até atuarem como instrumento nos testes de usabilidade. Além disso, Garcia et al. relatam que os protótipos são relevantes nas reuniões de planejamento, onde são usados para definir o que integrará uma iteração e auxiliam a estimativa de software ao oferecer uma visão do que vai ser desenvolvido. Os propósitos apresentados nesta pesquisa também se alinham aos propósitos de uso de prototipação definidos no *framework* PAM (Bjarnason et al., 2021).

Em relação às atividades relacionadas à garantia da qualidade, foram identificados problemas na forma como os protótipos foram utilizados em relação ao processo de desenvolvimento. Entre estes problemas, destacam-se:

1. Falta de padronização do uso de prototipação entre profissionais de uma mesma empresa. Verificou-se que, dentro da mesma empresa, foram encontradas diferentes maneiras de uso e até mesmo a existência de equipes que

não fazem uso de protótipos de interface. A princípio, a coexistência de equipes que adotam ou não a prototipação em uma mesma empresa não é necessariamente um problema. Porém, a falta de padronização de uso em relação a manter os protótipos atualizados, usar em outras fases do processo ou fazer a verificação da implementação em relação ao protótipo pode ocasionar problemas, mesmo que existam acordos de uso bem estabelecidos dentro de um time em relação à prototipação de interface. Um exemplo disto seria na ocasião de troca de desenvolvedores entre times. Neste cenário, um desenvolvedor que tem o hábito de implementar testes baseados em protótipos poderia ter problemas ao ser remanejado para uma equipe que não possui o hábito de manter os protótipos atualizados.

2. Uso de protótipos desatualizados em diversas atividades do processo de desenvolvimento de software. Foi relatado o uso de artefatos desatualizados como referência para a criação de *backlogs*, durante a implementação, na criação de casos de teste, validação da implementação com o protótipo, testes funcionais e de integração, homologação, treinamento e implantação. O uso de protótipos desatualizados compromete a qualidade do produto gerado e leva a um maior retrabalho, uma vez que inconsistências e requisitos defasados podem afetar o produto entregue ao cliente, acarretando prejuízos para a qualidade do software.

3. Ausência de verificação e inspeção dos protótipos de interface. Alguns participantes relataram não verificar o software desenvolvido em relação aos protótipos. Isto pode acarretar problemas no caso de protótipos de alta fidelidade, que especificam exatamente como a interface do produto deve ser. Além disso, não foram obtidos relatos sobre a realização de atividades de inspeção dos protótipos, de maneira a verificar se eles foram corretamente elaborados em relação às regras de design e às funcionalidades por eles representadas.

4. Descarte de protótipos que poderiam ser úteis. Participantes alegaram não utilizar os protótipos em fases posteriores do desenvolvimento, ou que os utilizam durante a implementação, mas não em atividades de verificação, validação e testes. Alguns participantes relataram também que não atualizam os protótipos caso o cliente não concorde com algo que foi projetado, sendo as modificações implementadas diretamente no código. Desta forma, identifica-se o descarte de um artefato que poderia ser útil em outras fases do desenvolvimento,

como implementação ou testes.

Os problemas identificados indicam que a adoção da prototipação de interface, por si só, não garante a qualidade do software desenvolvido. Trabalhos relacionados indicam situações nas quais o uso da prototipação deve ser aplicado segundo alguns critérios para que se atinja os objetivos de qualidade de software. O trabalho de Bellomo et al. (2013) identificou diferenças nos resultados obtidos por dois times de desenvolvimento e, a partir disto, defendeu a adoção de um processo com foco em atributos de qualidade, oferecendo um conjunto de diretrizes para o uso de prototipação. Já Käpyaho and Kauppinen (2015) sugerem a adoção de práticas complementares, tais como a elaboração de testes de aceitação com base nos protótipos ou a checagem periódica de protótipos em relação aos requisitos de qualidade, para que a prototipação possa contribuir plenamente com a melhoria da qualidade.

Assim como os trabalhos citados, este estudo contribui com lições para a prática de desenvolvimento de software com o uso de prototipação, obtidas a partir da observação dos resultados. Estas lições são apresentadas na seção seguinte.

3.3.1 Lições

Os problemas identificados na análise possibilitaram a observação de implicações para a prática de desenvolvimento de software e para o ensino de qualidade de software com o uso de prototipação. Estas lições podem ajudar a mitigar os problemas identificados.

É importante que os acordos de uso de ferramentas de apoio ao processo de desenvolvimento, como a prototipação, estejam claramente definidos para todos os membros dos times. Isto possibilita um alinhamento da visão dos membros do time em relação ao uso de prototipação e ajuda a evitar problemas com o uso de protótipos desatualizados.

Lição 1: Acordos de uso de prototipação devem estar claramente definidos para todos os membros do time de desenvolvimento.

Além disso, manter os protótipos atualizados pode ajudar a evitar problemas de uso de artefatos inconsistentes durante o desenvolvimento, testes, treinamento

e implantação. Isto ajudaria a garantir a realização das atividades de teste de maneira consistente. Também pode ser adotada a estratégia de colocar os protótipos desenvolvidos em repositórios de versionamento, mantendo assim um histórico das modificações realizadas.

Lição 2: Manter os protótipos atualizados pode ajudar a evitar problemas de uso de artefatos inconsistentes.

Finalmente, é importante também que os protótipos sejam incluídos nas atividades de inspeção, de forma a verificar se eles foram elaborados adequadamente, tanto no aspecto de regras de design e usabilidade quanto no aspecto funcional, quando representam funcionalidades recorrentes. A identificação prévia de erros nos protótipos elaborados poderia evitar a realização de ciclos adicionais de prototipação e apresentação ao cliente.

Lição 3: A verificação do software pode se beneficiar da inclusão de protótipos nas atividades de inspeção.

Além das implicações citadas para a prática de desenvolvimento de software, o uso de prototipação também tem implicações para o ensino de qualidade de software. As lições apresentadas podem ser incorporadas em atividades de aprendizagem prática, onde a prototipação é utilizada desde a elicitação de requisitos até as fases de verificação e validação. Desta forma, seria possível motivar a necessidade de se manter os protótipos atualizados em todas as etapas de desenvolvimento. Esta atividade também pode ser associada ao método *Share Multiple* de design de artefatos (Dow et al., 2011), onde grupos de alunos colaborariam na criação e no compartilhamento de designs alternativos, explorando diferentes ideias e se engajando em interações produtivas que favoreçam o aprendizado e resultem em protótipos de melhor qualidade.

Lição 4: O ensino de prototipação com atividades práticas em todas as fases do processo de desenvolvimento pode ajudar a refletir as necessidades da prática de engenharia de software.

3.3.2 Limitações

Este estudo apresenta algumas limitações, que são detalhadas a seguir. A primeira delas é relativa a problemas de compreensão das perguntas do questionário pelos participantes. Para amenizar este problema, foi conduzido um teste piloto com um participante voluntário, em que as perguntas foram revisadas de forma a melhorar a compreensibilidade. No entanto, verificamos que muitos participantes entenderam a pergunta Q2, sobre outras formas de se comunicar com o cliente além da prototipação, como o meio de comunicação adotado (e-mail, mensagem, telefone). Desta forma, não foi possível ter informações sobre quais artefatos ou procedimentos são utilizados na elicitação de requisitos quando os protótipos não são adotados.

Outra limitação encontrada foi relacionada ao fornecimento de informações sensíveis, como o nome da empresa na qual os participantes trabalhavam, mesmo com o compromisso assumido pelos pesquisadores, no TCLE, de anonimizar os dados pessoais fornecidos. Desta forma, ainda obtivemos 14 respondentes que não informaram o nome da empresa onde trabalham.

Finalmente, devido ao tamanho da amostra obtido neste estudo, não é possível afirmar que os resultados se aplicam de maneira geral ao desenvolvimento de software utilizando prototipação. No entanto, os problemas encontrados, bem como as lições que visam mitigar estes problemas, permanecem válidos em contextos de desenvolvimento que utilizam prototipação de interface do usuário.

3.4 Considerações Finais

Os resultados do estudo com profissionais da indústria de software mostraram a importância do uso de prototipação como ferramenta de elicitação de requisitos e a necessidade de estarem inseridos no processo de desenvolvimento de maneira clara e bem definida. Além disso, a pesquisa mostrou que os protótipos são um artefato versátil e de fácil elaboração que pode ser utilizado de diversas maneiras em diversas fases do processo de desenvolvimento, desde a elicitação de requisitos até a implementação e validação.

Além disso, foram identificados problemas envolvidos no uso de protótipos em

processos de desenvolvimento com baixo nível de maturidade. Verificou-se que, embora seja um artefato muito popular, apenas a adoção da prototipação não garante que o produto de software desenvolvido obtenha resultados melhores em relação à qualidade.

Desta forma, evidencia-se a demanda por técnicas que apoiem a verificação da qualidade de protótipos de IU. Neste sentido, esta pesquisa apresenta uma abordagem de inspeção de protótipos de interface de usuário em termos de suas funcionalidades, de forma a oferecer uma maneira de verificá-los desde a sua concepção. O próximo capítulo apresenta a primeira iteração do ciclo de design da abordagem desenvolvida nesta pesquisa.

Capítulo 4

Catálogo Heurísticas de Inspeção Funcional (CHIF)

Este capítulo apresenta a primeira iteração do ciclo de pesquisa desenvolvido nesta tese, conduzida segundo os princípios da *Design Science Research* (DSR). O método de pesquisa adotado nesta iteração combinou abordagens experimentais e analíticas, com o objetivo de definir e estruturar o Catálogo de Heurísticas de Inspeção Funcional (CHIF), concebido como um artefato inicial para apoiar a elaboração e verificação de protótipos de interface segundo seu aspecto funcional. As próximas seções apresentam as atividades realizadas para obtenção de protótipos para análise, identificação de diferentes funcionalidades e suas características e validação das heurísticas especificadas.

4.1 Introdução

O estudo com profissionais da indústria de software, relatado no Capítulo 3, indicou que protótipos de interface podem ser utilizados em diversas fases do processo de desenvolvimento. Isso mostrou a importância da identificação antecipada de defeitos em interfaces de usuário.

Por outro lado, a revisão de literatura (Capítulo 2) identificou uma carência de abordagens de identificação de defeitos em protótipos de interface com foco na funcionalidade. As abordagens de inspeção existentes se concentram em sua

maioria na identificação de defeitos de usabilidade e em violações de regras de design. Outros trabalhos analisam automaticamente GUIs encontrar problemas de design, como o trabalho de Moran et al. (2018) que utiliza técnicas de visão computacional para verificar se uma implementação de GUI está de acordo com seu design, ou o GUIComp (Lee et al., 2020), uma ferramenta para auxiliar o design de GUIs sugerindo outras semelhantes, avaliando o arranjo dos componentes e mostrando áreas de atenção no protótipo. No entanto, estas abordagens não consideram o significado dos elementos e a funcionalidade representados no protótipo, o que limita os tipos de verificações que poderiam ser realizadas para avaliação da GUI.

A caracterização de significado de interfaces, por outro lado, possibilitaria a realização de verificações detalhadas e, portanto, poderia ser usada para fornecer *feedbacks* mais precisos durante o design da interface ou para indicar erros de design. Por exemplo, o trabalho de Liu et al. (2018) utiliza um *dataset* de capturas de tela e seus metadados para gerar anotações semânticas para interfaces de aplicativos móveis. Essas anotações semânticas podem ser usadas para buscas de telas de aplicativos com interações semelhantes, por exemplo. Já o trabalho de Chen et al. (2020a) utiliza uma combinação de imagens e informações textuais para prever categorias de tela e sugerir rótulos para descrever designs de interface.

Seguindo a linha de pesquisa desses trabalhos, o design e a verificação de protótipos poderiam se beneficiar de uma descrição explícita de funcionalidades comuns a diferentes aplicativos, como autenticação de usuário, pesquisa ou formulários de registro, fornecendo um meio para identificar a funcionalidade representada e indicando possíveis inconsistências entre o protótipo e a especificação da funcionalidade. Desta forma, a primeira iteração do ciclo de design deste trabalho buscou responder à seguinte questão de pesquisa:

> **QP – "Como caracterizar funcionalidades representadas em protótipos de interface de usuário?"**

Para responder a essa questão de pesquisa, investigou-se a aplicabilidade de heurísticas, que são regras derivadas da experiência usadas na implementação de processos de decisão para encontrar soluções para problemas específicos (Calle-Escobar et al., 2016). Diferentemente dos métodos determinísticos, que consideram

todas as variáveis do problema em busca de uma solução ótima — o que geralmente implica maior custo computacional e tempo de processamento elevado —, as heurísticas simplificam o processo de busca, ignorando parte das informações para encontrar soluções suficientemente boas para resolver um problema (Gigerenzer, 2008). Dessa forma, heurísticas são utilizadas para obter resultados satisfatórios com menor esforço computacional (Shah and Oppenheimer, 2008).

Heurísticas de design são aquelas usadas para apoiar o design de produtos. Trabalhos como o de Figueiredo et al. (2012) apresentam heurísticas de projeto para avaliar projetos sensíveis, ou seja, aspectos de um produto de software que podem afetar a manutenção de módulos de software. Já o trabalho de Yilmaz et al. (2016) apresenta uma compilação de 77 heurísticas de design comumente utilizadas no design de produtos.

Neste estudo, o problema de caracterização de funcionalidades representadas em protótipos de interface foi abordado através da definição de heurísticas de design. A descrição das características mais relevantes e determinísticas de funcionalidades de software através de heurísticas pode ser utilizada para auxiliar designers durante o desenvolvimento de protótipos ou para verificar se um determinado protótipo atende aos requisitos associados à funcionalidade representada.

Na solução desenvolvida, foram identificadas funcionalidades comumente representadas em protótipos de interface, que tiveram suas principais características descritas através de heurísticas segundo o modelo de Quiñones et al. (2018). Para identificar tais funcionalidades, inicialmente foi conduzido um estudo sobre o ensino de engenharia de requisitos com o desenvolvimento de *soft skills*, no qual os participantes atuaram em equipes simulando projetos reais de desenvolvimento nos papéis de cliente e desenvolvedor. Como resultado, foram produzidos 31 projetos e 856 protótipos de interface, que constituíram o corpus de análise utilizado nesta iteração.

Em seguida, a análise dos protótipos foi conduzida de forma sistemática com base em procedimentos da *Grounded Theory* (Corbin and Strauss, 2014), utilizando codificação aberta para a identificação iterativa de categorias representativas das funcionalidades presentes nas interfaces. Esse processo resultou em 23 categorias de protótipos, abrangendo funcionalidades recorrentes em diferentes tipos de sistemas. Com base nas categorias resultantes, foi elaborado o Catálogo

de Heurísticas de Inspeção Funcional (CHIF), estruturado como um conjunto de heurísticas destinadas a apoiar a elaboração e verificação de protótipos de interface.

O CHIF foi avaliado através de um grupo focal com especialistas. Os resultados mostraram que o CHIF e o conjunto de protótipos rotulados podem auxiliar na prototipagem de IU ou integrar soluções automatizadas de análise de protótipos de IU. Os resultados deste ciclo foram publicados em De Macedo et al. (2022). As próximas seções detalham a metodologia adotada e as etapas do processo de elaboração e avaliação do CHIF.

4.2 Obtenção de Conjunto de Protótipos

Os protótipos de interface de usuário utilizados como base para a elaboração do Catálogo de Heurísticas de Inspeção Funcional (CHIF) foram obtidos a partir de um estudo experimental conduzido para avaliar uma abordagem de aprendizagem colaborativa (Jeong and Hmelo-Silver, 2016) para o ensino remoto de Engenharia de Requisitos com o desenvolvimento de *soft skills* (Matteson et al., 2016). Esta abordagem adaptou a técnica de aprendizagem colaborativa de Zowghi and Paryani (2003) para o contexto de ensino remoto e o desenvolvimento de habilidades relacionadas à comunicação, empatia e resolução de conflitos, consideradas essenciais para engenheiros de requisitos. [1]

O estudo foi realizado em duas iterações de Pesquisa-Ação – Action Research (Davison et al., 2004) no contexto de disciplinas de graduação em Engenharia de Software da Universidade Federal do Amazonas (UFAM). O método adotado seguiu o processo canônico de Pesquisa-Ação proposto por Davison et al. (2004), composto pelas fases de diagnóstico, planejamento, intervenção, avaliação e reflexão. Cada ciclo foi conduzido em uma disciplina distinta, com participantes em diferentes estágios de formação acadêmica, permitindo analisar o impacto da abordagem em contextos de aprendizado com níveis variados de experiência prévia.

[1] Pesquisa aprovada pelo Comitê de Ética em Pesquisa - CEP da Universidade Federal do Amazonas - UFAM, sob o número CAAE 40937120.6.0000.5020, em 26/07/2021, Parecer Consubstanciado N. 4.923.486, de 22 de agosto de 2021.

No primeiro ciclo, a intervenção ocorreu durante a disciplina Engenharia de Requisitos e Análise de Sistemas, ofertada a estudantes mais avançados do curso. Nesta etapa, 35 participantes desenvolveram projetos de software completos, aplicando uma abordagem de aprendizagem colaborativa baseada em projetos – *Project-Based Learning* (Guo et al., 2020), que integrou as técnicas *Role-Play* (Svensson and Regnell, 2017) e *Send-a-Problem* (Barkley et al., 2014). Os estudantes foram divididos em equipes que atuaram alternadamente como clientes e desenvolvedores de sistemas, criando especificações de requisitos e protótipos de interface de usuário para representar as funcionalidades planejadas. A atividade foi mediada por tecnologias educacionais e ferramentas de comunicação, como videoconferências e mensagens instantâneas, possibilitando a realização remota das tarefas durante o período de ensino à distância imposto pela pandemia de COVID-19. A realização das atividades de projeto foi acompanhada pelos instrutores através de reuniões remotas. Ao término das atividades, as equipes apresentaram seus projetos às suas respectivas equipes clientes. A Figura 4.1 ilustra as etapas da abordagem através de um diagrama de sequência. Os dados para avaliação dos aspectos de aprendizagem do estudo foram coletados através de questionários, respondidos pelos alunos de maneira voluntária através da assinatura de um Termo de Consentimento (TCLE - Apêndice B). No termo, os alunos concordaram em ceder seus protótipos para este projeto de pesquisa.

Os resultados do primeiro ciclo mostraram que a dinâmica favoreceu a prática de habilidades interpessoais e o desenvolvimento de artefatos de qualidade. Entretanto, identificou-se a necessidade de maior acompanhamento das equipes, pois fatores como nível de experiência profissional e vínculos prévios de amizade entre estudantes influenciaram a execução das tarefas e a complexidade dos projetos propostos.

Para refinar o método e mitigar essas variáveis, foi conduzido um segundo ciclo de Pesquisa-Ação na disciplina Fundamentos de Engenharia de Software, ofertada a alunos ingressantes do curso de Engenharia de Software. Essa segunda iteração manteve a mesma estrutura metodológica, mas incluiu reuniões semanais de acompanhamento e uma apresentação final dos projetos em videoconferência, permitindo aos alunos vivenciar de forma mais realista os papéis de cliente e desenvolvedor. Os resultados mostraram que, mesmo com participantes menos

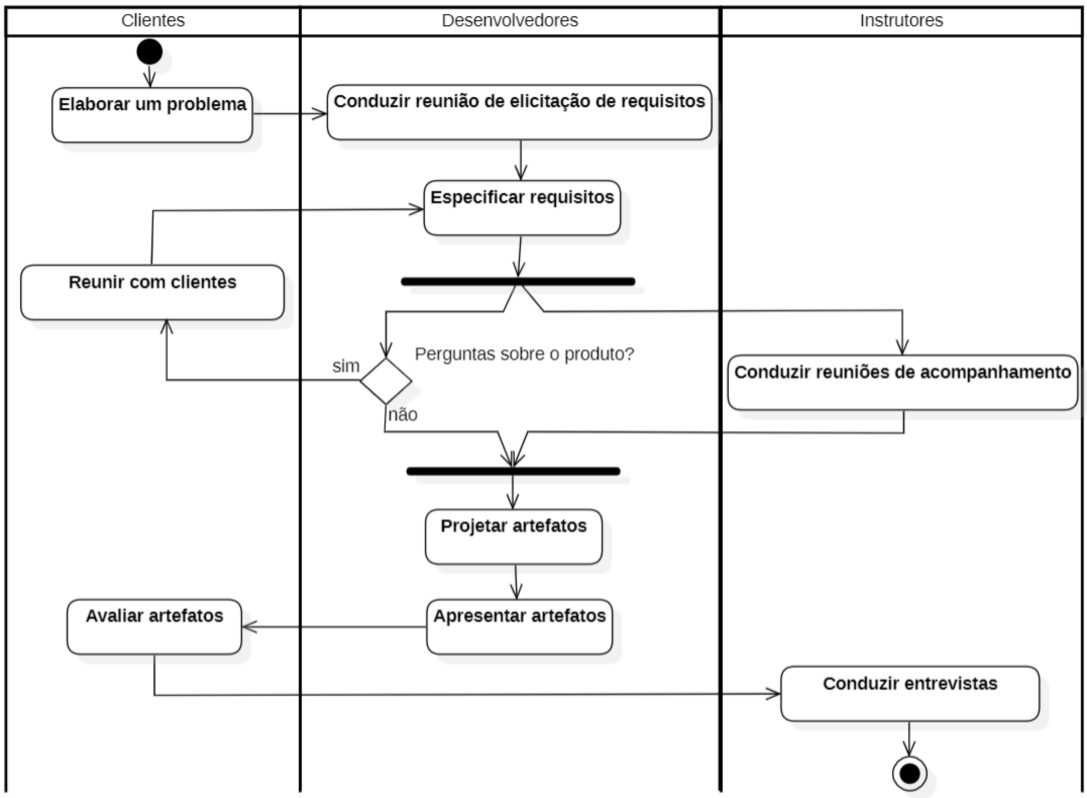


Figura 4.1: Abordagem de ensino remoto de Engenharia de Requisitos com desenvolvimento de soft skills.

experientes, a abordagem possibilitou o desenvolvimento de competências como raciocínio analítico, comunicação e resolução de conflitos, além de estimular a compreensão prática dos conceitos de Engenharia de Requisitos.

Os resultados deste estudo foram publicados em De Macedo et al. (2024). No total, as duas iterações resultaram em 31 projetos de software e 856 protótipos de interface de usuário. Os projetos abordaram sistemas de diferentes domínios, como comércio eletrônico, gestão acadêmica, controle financeiro, saúde e serviços sob demanda, refletindo a diversidade de contextos em que funcionalidades típicas de sistemas interativos são representadas. Esses protótipos foram analisados qualitativamente para identificar padrões de funcionalidades recorrentes em interfaces de usuário. As categorias resultantes dessa análise fundamentaram a estruturação do Catálogo de Heurísticas de Inspeção Funcional (CHIF), descrito nas seções

seguintes deste capítulo.

4.3 Método de Especificação de Heurísticas de Inspeção Funcional

A partir do conjunto de protótipos de interface obtidos nos experimentos, foram identificadas as características predominantes de cada categoria de protótipo, as quais compõem a especificação da heurística. Para obter esse conjunto de categorias de protótipos e suas características predominantes, o método proposto por Quiñones et al. (2018), originalmente concebido para a especificação de heurísticas de usabilidade, foi adaptado para o contexto de especificação de heurísticas de inspeção funcional. Esse método é composto por 8 estágios, que podem ser realizados de forma sequencial ou iterativa, em ciclos envolvendo um ou mais passos. A Figura 4.2 mostra como os estágios do método foram realizados neste trabalho, em que os passos foram realizados de maneira sequencial, com ciclos iterativos entre os passos 4 e 2 e entre os passos 8 e 6. Nos passos 1 e 2, estágios Exploratório e Experimental, foram coletados subsídios para o passo 3 - estágio Descritivo.

Figura 4.2: Método para desenvolvimento de heurísticas de usabilidade e UX (Fonte: Quiñones et al. (2018)).

No passo 4 - estágio Correlacional, foi realizada a correspondência entre as categorias de protótipos obtidas no estágio Experimental e as obtidas em trabalhos relacionados. Nesta correlação, foi realizada uma padronização de nomenclatura que resultou na modificação do nome de algumas categorias obtidas no passo 2 - Experimental. Já no passo 8 - Refinamento, algumas das heurísticas especificadas no passo 6 foram modificadas através da união de heurísticas semelhantes ou

da adição de novas regras. A próxima seção descreve com maiores detalhes os estágios da metodologia e como eles foram adaptados ao contexto deste trabalho.

4.3.1 Estágio Exploratório

O passo 1, o Estágio Exploratório, inicia-se através da definição de um domínio de aplicação específico para desenvolver heurísticas de usabilidade e UX. No caso específico deste trabalho, visou-se auxiliar as atividades de design de IU; então, foram desenvolvidas heurísticas de design para protótipos de IU de diferentes domínios e plataformas. Desta forma, o domínio de pesquisa consiste na análise de protótipos da interface de usuário no contexto do design de interfaces de usuário.

Neste passo, também devem ser coletados três tipos de dados: informações sobre o domínio, atributos de usabilidade/UX a serem avaliados pelo conjunto de novas heurísticas e princípios, heurísticas, diretrizes e padrões já existentes. Primeiramente, conforme apresentado no Capítulo 2, foi realizada uma revisão de literatura, em que foram identificados trabalhos relacionados à assistência ao design de interfaces de usuário, à identificação semântica em protótipos de interface e à busca de GUIs a partir de um modelo de entrada. Nesses trabalhos, o design de IU e a classificação de protótipos são abordados em diferentes níveis semânticos: no nível de componentes, pela identificação de componentes de UI e elementos interativos, tais como botões e ícones (Liu et al., 2018); no nível estrutural, onde o *layout* de seus componentes caracteriza a semântica dos designs de IU (Leiva et al., 2020); e no nível funcional, descrevendo funcionalidades mais complexas (Chen et al., 2020a). No contexto deste trabalho, foram coletados os seguintes tipos de informações dos trabalhos relacionados:

- Heurística de identificação de componentes, conceitos de botões de texto e classes de ícones, de Liu et al. (2018);
  - Heurísticas de identificação de componentes baseadas em código Android - Determina o tipo de componente de um elemento examinando sua classe Android e as classes das quais ele herda, através das propriedades *class* e *ancestors*;

  - Heurísticas de identificação de conceitos de botões - Qualquer elemento cuja classe ou propriedade *ancestors* contenha o texto "*button*";
  - Heurística para identificação de ícones - Um componente de imagem clicável e visível ao usuário, cuja área seja inferior a 5% da área total da tela e cuja proporção seja superior a 0,75 (dimensão menor dividida pela dimensão maior).

- Tópicos de design de interfaces de usuário, de Leiva et al. (2020) - bare, discador, câmera, chat, editor, formulário, galeria, lista, login, mapas, media player, menu, modal, notícias, outro, perfil, pesquisa, configurações, termos, tutorial;

- Categorias semânticas de interface de usuário de Chen et al. (2020a) - plataforma, cor, funcionalidade do aplicativo, funcionalidade da tela, layout da tela;

- Padrões de design de interface do usuário, do site UI Patterns (Toxboe, 2022).

4.3.2 Estágio Experimental

O passo 2, Estágio Experimental, utilizou os protótipos obtidos resultantes do estudo sobre ensino remoto de Engenharia de Requisitos com desenvolvimento de *soft skills*, descrito na Seção 4.2. Este estudo resultou em 31 projetos e 856 protótipos de UI para plataformas móveis, web e *desktop*.

O conjunto de protótipos obtido foi analisado pelo primeiro pesquisador para identificar diferentes categorias de protótipos. Antes de iniciar a análise dos protótipos, foram removidas telas duplicadas e elementos explicativos da interface de usuário. Em seguida, a rotulagem dos protótipos de UI foi realizada adotando-se o procedimento de *Grounded Theory* (Corbin and Strauss, 2014) de codificação aberta, onde os rótulos dos protótipos de UI foram criados e modificados iterativamente durante a análise. Alguns rótulos de protótipos criados inicialmente foram substituídos por outros mais representativos ao longo deste processo.

Em seguida, a rotulagem inicial foi revisada por dois outros pesquisadores com experiência em engenharia de requisitos. Neste processo, pontos de discordância

foram discutidos até que se chegasse a um consenso, resultando na unificação de rótulos distintos atribuídos a protótipos com o mesmo significado semântico, na separação de rótulos atribuídos a protótipos com significados semânticos diferentes, sendo atribuídos rótulos mais significativos, e no agrupamento de rótulos muito semelhantes e com poucos protótipos rotulados.

O processo de rotulagem resultou em 23 categorias de protótipos, em que 47 protótipos de UI receberam mais de um rótulo. A Tabela 4.1 apresenta uma lista das categorias de protótipos obtidas após o processo de rotulagem. É importante ressaltar que são listadas apenas as quantidades de protótipos aos quais foi atribuído um único rótulo.

Tabela 4.1: Categorias de protótipos obtidas durante o estágio Experimental.

Nº	**Rótulo**	**Qtde.**	**Descrição**
1	Alteração	34	Um formulário que apresenta os valores dos atributos de uma instância de entidade para que possam ser modificados
2	Autenticação	40	Formulário de informação de credenciais para acessar os recursos da aplicação
3	Busca	50	Mecanismo de busca
4	Cadastro	80	Formulário para registrar uma instância de uma entidade no domínio do aplicativo
5	Calendário	8	Consulta com base em datas apresentadas na forma de um calendário
6	Carrinho de Compras	12	Lista de produtos ou serviços e suas quantidades, a serem adquiridos em um aplicativo de *e-commerce*
7	Chat	24	Funcionalidade para troca de mensagens instantâneas entre usuários da aplicação
8	Filtro de Conteúdo	7	Um conjunto de campos que filtram as informações exibidas
9	Finalizar Compra	17	Informações sobre opções de pagamento, envio e cobrança em aplicativos de comércio eletrônico
10	Grade	20	Um conjunto de instâncias de uma entidade do domínio do aplicativo exibida em um formato de grade
11	Home	11	Página inicial da aplicação apresentando as informações mais relevantes em uma ampla variedade de formatos.
12	Lista	123	Conjunto de instâncias de uma entidade de domínio
13	Mestre-Detalhe	28	Exibição de atributos de uma instância de uma entidade de domínio seguida por uma lista de instâncias de outra entidade relacionada à primeira
14	Menu	82	Opções de recursos da aplicação
15	Modal	21	Uma janela pop-up solicitando uma ação do usuário.
16	Notificação	48	Informações de status do sistema
17	Página do Produto	20	Informações detalhadas sobre um produto
18	Perfil do usuário	32	Informações da conta do usuário
19	Registro de Conta	57	Criação de conta de usuário para acessar os recursos de uma aplicação
20	Upload de Arquivo	6	Um formulário de upload de arquivo
21	Termos de Uso	10	Termos de uso da aplicação
22	Videoconferência	20	Funcionalidade de videoconferência
23	Visualização	59	Exibe os valores de atributo de um objeto de uma entidade no domínio da aplicação

4.3.3 Estágio Descritivo

No passo 3, Estágio Descritivo, os tópicos coletados nos passos 1 e 2 são selecionados e priorizados. Desta forma, foi realizada uma análise dos trabalhos relacionados para seleção de heurísticas e categorias de IU. Primeiramente, de Liu et al. (2018), foram selecionados todos os botões de conceito e definições de sinônimos para serem utilizados nas definições de heurísticas deste trabalho. Já as heurísticas deste trabalho selecionadas previamente não foram utilizadas porque foram concebidas para a detecção de componentes e ícones da interface do usuário, enquanto o objetivo deste trabalho é o de identificar a semântica para protótipos da interface do usuário como um todo.

Em seguida, do site UI Patterns (Toxboe, 2022) e dos trabalhos de Leiva et al. (2020) e Chen et al. (2020a), foram selecionadas as categorias correspondentes às encontradas durante o processo de análise e rotulagem dos protótipos analisados. De Chen et al. (2020a) também foram selecionadas as categorias de aplicativos para classificar os projetos aos quais os protótipos deste trabalho pertencem. As categorias de interface *Screen Functionality* e *Screen Layout* foram adotadas por este trabalho para agrupar as heurísticas desenvolvidas, sendo chamadas, respectivamente, de Funcionalidades Específicas e Funcionalidades Básicas. A Tabela 4.2 apresenta um resumo das informações selecionadas de cada trabalho relacionado.

Tabela 4.2: Informações selecionadas de diferentes fontes durante o passo 3, estágio Descritivo.

Fonte	**Informações coletadas**	**Informações selecionadas**
Liu et al. (2018)	28 botões de conceito UX (não, login, voltar, ir, ok, todos, próximo, adicionar, criar, mais, tentar novamente, pular, definir, excluir, Facebook, visualizar, reservar, continuar, listar, salvar, pesquisar, terminar, concordar, mostrar, compartilhar, comprar, atualizar, editar)	Todos os botões de conceito e sinônimos foram selecionados.
Liu et al. (2018)	Heurísticas para identificação de botões de texto, extração de ícones e identificação de componentes	Nenhuma das heurísticas foi selecionada.
Leiva et al. (2020)	Tópicos do conjunto de dados Enrico (bare, discador, câmera, chat, editor, formulário, galeria, lista, login, mapas, media player, menu, modal, notícias, outro, perfil, pesquisa, configurações, termos, tutorial	Chat, formulário, lista, login, menu, modal, perfil, pesquisa, termos.
Chen et al. (2020a)	Categorias semânticas de interface do usuário (plataforma, cor, funcionalidade do aplicativo, funcionalidade da tela, layout da tela)	Todas as categorias de aplicações; categorias de funcionalidade da tela: login, inscrição, checkout, pesquisa, perfil; categorias de layout: formulário, lista, grade.
UI Design Patterns (Toxboe, 2022)	Padrões de design da interface do usuário	Padrões modal, notificações, filtro de conteúdo, página de produto, carrinho de compras, chat, cadastro de conta.

4.3.4 Estágio Correlacional

No passo 4, Estágio Correlacional, as características e heurísticas existentes, coletadas nos passos anteriores, devem ser combinadas. No caso desta pesquisa, as heurísticas pré-existentes obtidas no Estágio 1 não foram selecionadas durante o Estágio 3. Desta forma, as categorias de protótipos resultantes do processo de rotulagem foram mapeadas com as listas de categorização de trabalhos relacionados. A Tabela 4.3 apresenta este mapeamento. Além disso, as categorias obtidas no Estágio 2 - Experimental foram divididas em dois grupos: Funcionalidades Básicas e Funcionalidades Específicas. As categorias de Funcionalidades Básicas referem-se a recursos genéricos de software, como formulários de cadastro e listas de objetos. As categorias de Funcionalidades Específicas estão relacionadas a recursos mais complexos, como, por exemplo, *Autenticação* e *Chat*. A identificação das categorias de Funcionalidades Básicas começa com o prefixo *HEU1* e as categorias de Funcionalidades com o prefixo *HEU2*.

4.3.5 Estágio de Seleção

No passo 5, Estágio de Seleção, deve ser determinado quais heurísticas serão mantidas, adaptadas, criadas ou eliminadas. Desta forma, as categorias de protótipos deste trabalho foram reanalisadas, de forma a identificar quais delas correspondem a requisitos funcionais de projeto de software. Essa análise verificou que quatro das categorias não representam funcionalidades do software: Calendário, Termos de Uso, Notificação e Modal. Calendário é um componente de entrada de dados diferenciado para seleção de datas, que pode ser utilizado em diversas situações. Os Termos de Uso representam um contrato de uso da aplicação e não possuem requisitos de design específicos. Por fim, as categorias Notificação e Modal representam situações em que textos informativos são mostrados ao usuário e podem ou não exigir algum tipo de interação. Essas categorias podem ser usadas em uma ampla variedade de situações e também não representam funcionalidades específicas de software. Portanto, estas quatro categorias não foram selecionadas para o estágio de Especificação.

Tabela 4.3: Correspondência entre categorias do processo de rotulagem e trabalhos relacionados.

ID	Categoria	Enrico (Leiva et al., 2020)	Chen et al. (2020a)	UI Design Patterns (Toxboe, 2022)
Categorias de Funcionalidades Básicas				
HEU1-01	Alteração	Formulário	Formulário	
HEU1-02	Busca	Busca	Busca	
HEU1-03	Cadastro	Formulário	Formulário	
HEU1-04	Filtro de Conteúdo			Filtro de Tabela
HEU1-05	Grade		Grade	
HEU1-06	Lista	Lista	Lista	
HEU1-07	Menu	Menu		
HEU1-08	Mestre-Detalhe			
HEU1-09	Upload de Arquivo			
HEU1-10	Visualização			
Categorias Funcionalidades Específicas				
HEU2-01	Autenticação	Login	Login	
HEU2-02	Carrinho de Compras			Carrinho de Compras
HEU2-03	Chat	Chat		Chat
HEU2-04	Finalizar Compra		Checkout	
HEU2-05	Home			
HEU2-06	Página de Produto	Perfil		Página de Produto
HEU2-07	Perfil de Usuário	Perfil	Perfil	
HEU2-08	Registro de Conta		Inscrição	Registro de Conta
HEU2-09	Videoconferência			

4.3.6 Estágio de Especificação

O passo 6 corresponde ao Estágio de Especificação, onde as heurísticas são formalmente especificadas. Neste trabalho, foram especificadas heurísticas para as 19 categorias restantes do processo de rotulagem. O modelo padrão de definição de heurísticas proposto por Quiñones et al. (2018) foi adaptado para o contexto

deste trabalho, de forma a possibilitar a descrição de categorias de protótipos de interface de acordo com a funcionalidade por eles representada.

Foi adicionado o campo *Sinônimos* para facilitar o entendimento da heurística. Como não estão sendo especificadas heurísticas de usabilidade, os campos *Prioridade*, *Recurso da Aplicação*, *Benefícios*, *Problema* e *Atributos de Usabilidade/UX* foram removidos. A Tabela 4.4 apresenta as adaptações feitas no modelo de heurísticas original.

Tabela 4.4: Adaptações no modelo de especificação de heurísticas de usabilidade de Quiñones et al. (2018).

Modelo original **Heurísticas de Usabilidade**	**Modelo adaptado** **Heurísticas de Design**
ID	ID
Prioridade	
Nome	Nome
	Sinônimos
Definição	Definição
Explicação	Explicação
Recurso da aplicação	
Exemplo	Exemplo
Benefícios	
Problemas	
Checklist	Checklist
Atributo de Usabilidade/UX	
	Informações relacionadas
Heurísticas relacionadas	Heurísticas relacionadas

As heurísticas de design possuem atributos gerais que as descrevem, como *Nome, Sinônimos, Definição, Explicação* e *Exemplos*. Além disso, elas são descritas em termos de características que um protótipo deve apresentar para caracterizar uma funcionalidade. No campo *checklist*, são listados os critérios a que o protótipo da funcionalidade representada pela heurística deve atender. Essas características podem estar relacionadas às entidades do domínio da aplicação ou a componentes que devem estar presentes nos protótipos e às propriedades destes componentes. Por exemplo, o rótulo do botão de conexão em um protótipo do tipo Autenticação deve descrever claramente esta ação, como os termos *(login, conectar, entrar)*. Para que estas informações possam integrar a especificação das heurísticas, foi incluído

no modelo um campo chamado *Informações Relacionadas*, no qual esses conjuntos de dados são elencados.

As heurísticas especificadas neste ciclo de design estão listadas no Apêndice A e estão disponibilizadas no *website* do catálogo de heurísticas[2]. Nele, as heurísticas estão agrupadas de acordo com seu nível de abstração: Funcionalidades Básicas ou Específicas. Na página de cada uma das heurísticas é apresentada a sua especificação acompanhada de exemplos da base de protótipos desenvolvida pelos voluntários. As Figuras 4.3 e 4.4 apresentam telas do site do CHIF.

Figura 4.3: CHIF - página inicial

[2]Link para o CHIF: https://sites.google.com/view/design-heuristics

Figura 4.4: CHIF - heurística de Carrinho de Compras

4.3.7 Estágio de Validação

No passo 7, Estágio de Validação, foi realizada a validação das heurísticas de design por meio de um estudo de avaliação por especialistas (Quiñones et al., 2018), com dois profissionais da indústria de software e dois especialistas em interação humano-computador. Neste estudo, foi proposto um cenário de desenvolvimento de uma aplicação em que os participantes projetaram protótipos de UI para os requisitos de software.Os resultados foram avaliados através das dimensões de Utilidade, Clareza, Facilidade de Uso e Necessidade de uma Lista de Verificação Adicional para as heurísticas. A seção 4.4 apresenta os resultados deste estudo em detalhes.

4.3.8 Estágio de Refinamento

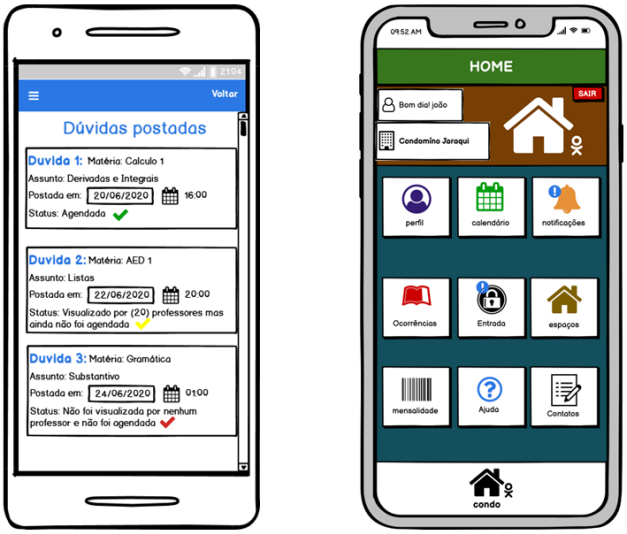
No passo 8, Estágio de Refinamento, foram realizados refinamentos no conjunto de heurísticas utilizando os resultados do estágio de Validação. As heurísticas *Filtro de Conteúdo* e *Busca* foram unificadas. Também foram mescladas as heurísticas *Grade* e *Lista*, porque ambas representam grupos de objetos organizados de maneiras diferentes. Além destas modificações, foi adicionado o formato Carrossel como forma de apresentação de objetos nesta heurística. A análise de especialistas também sugeriu a inclusão de heurísticas para a funcionalidade de *Avaliação* e para o projeto de protótipos com requisitos de acessibilidade. Os resultados deste estágio são detalhados na próxima seção.

4.3.9 Heurísticas de Inspeção Funcional

O catálogo de heurísticas de design, obtido após a realização dos passos do processo de especificação e cujas funcionalidades estão listadas na Tabela 4.3, pode ser consultado no Apêndice A (versão final da tese). Para ilustrar as heurísticas obtidas nesta iteração do ciclo de Design, esta seção apresenta uma heurística de cada nível, Funcionalidades Básicas e Funcionalidades Específicas. As modificações realizadas em decorrência do estágio de Validação das Heurísticas (Seção 4.4) já estão refletidas nas heurísticas ilustradas a seguir.

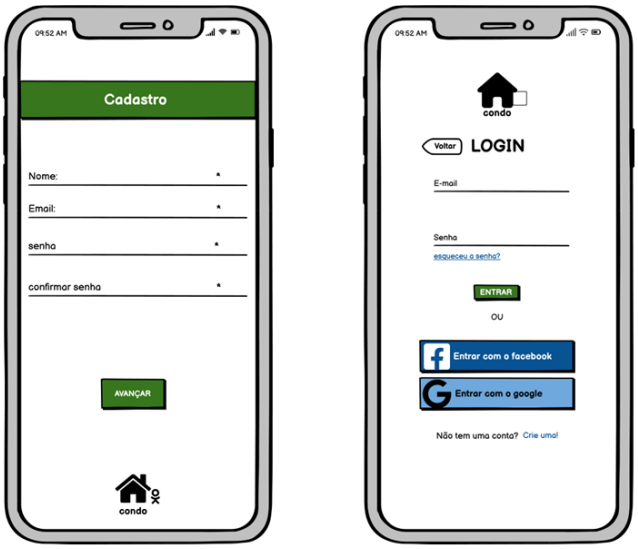
As heurísticas de Funcionalidades Básicas compreendem protótipos de UI que representam funcionalidades genéricas de uma aplicação de software, como *Cadastro* e *Busca*. A Figura 4.5 mostra exemplos de protótipos de interface para as funcionalidades *Lista* (4.5(a)) e *Menu* (4.5(b)). Como exemplo ilustrativo de uma heurística de *Funcionalidades Básicas*, a Tabela 4.5 apresenta a heurística da funcionalidade *Lista* obtida ao final da primeira iteração do ciclo de design.

As heurísticas de *Funcionalidades Específicas* compreendem protótipos de UI que representam funcionalidades de alto nível que são frequentes em diversas aplicações. A Figura 4.6 mostra exemplos de protótipos das funcionalidades específicas *Registro de Conta* (a) e *Autenticação* (b). A Tabela 4.6 apresenta a heurística da funcionalidade Registro de Conta para ilustrar este tipo de funcionalidade.



(a) (b)

Figura 4.5: Exemplos de protótipos de Funcionalidades Básicas – Lista (a) e Menu (b).



(a) (b)

Figura 4.6: Exemplos de protótipos do tipo Funcional – Registro de Conta (a) e Autenticação (b).

Tabela 4.5: Heurística HEU1-06 – Lista.

ID: HEU1-06

Nome: **Lista**
Sinônimos: Lista vertical, Lista horizontal
Definição: Exibição de um conjunto de objetos relacionados a uma entidade do domínio da aplicação.
Explicação: Uma lista apresenta um conjunto de objetos de uma entidade no domínio da aplicação.
Exemplo: Um usuário visualiza uma lista de filmes em um aplicativo de serviço de *streaming*.
Checklist:

1. O protótipo apresenta uma lista de objetos de uma entidade do domínio da aplicação.
2. Cada item da lista apresenta valores de atributos do objeto.
3. Opcionalmente, o conjunto de objetos pode ser apresentado verticalmente, horizontalmente, como grade ou como um carrossel.
4. Se a lista está organizada em forma de tabela, os títulos das colunas correspondem a nomes de atributos dos objetos.

Informações relacionadas:

1. Entidades do domínio da aplicação e seus respectivos atributos.

Heurísticas relacionadas: Busca, Mestre-Detalhe, Videoconferência

4.4 Validação das Heurísticas com Especialistas

Esta seção apresenta os resultados do estudo realizado no Estágio de Validação da metodologia de especificação do conjunto de heurísticas de design, que analisou a percepção de especialistas em design de interface. Durante o estudo, os especialistas foram instruídos a criar protótipos de interface utilizando o catálogo de heurísticas de inspeção funcional (CHIF). Após esta atividade, os especialistas avaliaram as heurísticas por meio da aplicação de um questionário e da realização de um grupo focal. As seções a seguir apresentam os detalhes da preparação do experimento e da análise dos resultados.

Tabela 4.6: Heurística HEU2-08 – Registro de Conta.

ID: HEU2-08

Nome: **Registro de Conta**
Sinônimos: Sign in, sign up, registro de usuário
Definição: Criação de uma conta de usuário para acesso a funcionalidades de um software.
Explicação: Ilustra a criação de uma conta de acesso feita por um usuário de um software para acessar recursos personalizados ou restritos.
Exemplo: Um usuário se cadastra em um site de um restaurante para pedir uma pizza.
Checklist:

1. O protótipo apresenta um formulário de entrada de dados com um ou mais campos cujos valores não estão preenchidos.
2. Os campos do formulário representam os atributos da entidade da aplicação relativa a usuários.
3. O formulário tem um campo para identificação do usuário, um campo para senha e um campo para redigitar a senha.
4. O rótulo do campo de identificação de usuário é usado para descrever campos de identificação do usuário. Exemplos: id, login, e-mail, email, identificação, cpf, nome, usuário, matrícula, nome de usuário, telefone.
5. O rótulo do campo de senha é usado para descrever campos de senha. Exemplos: senha, digite a senha.
6. O rótulo do campo de redigitar a senha é usado para descrever campos de redigitação de senha. Exemplos: redigite a sua senha, confirmar senha, confirme sua senha.
7. O formulário tem um botão de registro e seu rótulo descreve botões de registro de conta (exemplos: cadastrar, confirmar, confirmar cadastro, cadastrar-se, finalizar, continuar, criar conta, finalizar cadastro, concluir, registrar) ou seu ícone representa botões de registro de conta. Exemplos:

Informações relacionadas:

1. Entidades do domínio da aplicação e suas respectivas propriedades.

Heurísticas relacionadas: Cadastro

4.4.1 Preparação e Execução

Para validar o catálogo de heurísticas, foi realizado um estudo de validação por meio de julgamento de especialistas – *Validation Through Expert Judgment* (Quiñones et al., 2018). Por ser o primeiro experimento de avaliação das heurísticas, o estudo teve como objetivo avaliar a percepção geral dos especialistas sobre as dimensões propostas por Quiñones et al. (2018), que consistem em Utilidade, Clareza, Facilidade de Uso e Necessidade de uma Lista de Verificação Adicional. Participaram do estudo dois especialistas em interação humano-computador e dois profissionais da indústria de software, selecionados por conveniência através de convite direto para pessoas da rede de contatos dos pesquisadores que tivessem cinco anos ou mais de experiência na sua área de atuação.

O estudo consistiu na realização de atividades de prototipação de interface para um *website* fictício de comércio eletrônico. Para isso, foi realizada uma reunião remota com os especialistas, na qual foram apresentadas as informações e instruções para a realização do estudo. Foi apresentada a eles uma descrição geral do *website* e uma lista de requisitos a que o site deveria atender (Apêndice B), solicitando-se que projetassem protótipos de interface do usuário para três requisitos escolhidos livremente dentre os apresentados na especificação da aplicação. Foi recomendado aos especialistas que utilizassem o conjunto de heurísticas durante a elaboração dos protótipos, disponibilizado no *website* do CHIF. Cada especialista trabalhou individualmente conforme sua disponibilidade e respondeu a um questionário de avaliação das heurísticas após a elaboração dos protótipos. A Tabela 4.7 apresenta as perguntas deste questionário. Três dias após a atividade de prototipação e o preenchimento do questionário, foi realizado um grupo focal com os especialistas, no qual eles relataram os pontos positivos, dificuldades e dúvidas no uso das heurísticas durante a atividade de prototipação.

4.4.2 Resultados

Os resultados do estudo experimental foram avaliados através de análise qualitativa das respostas dos especialistas ao questionário e das considerações obtidas no grupo focal. As respostas foram analisadas de acordo com os aspectos de Utilidade, Clareza, Facilidade de Uso e Necessidade de uma Lista de Verificação

Tabela 4.7: Questionário de avaliação das heurísticas por especialistas.

Nº	**Pergunta**
1	Alguma das heurísticas foi útil no desenvolvimento do protótipo? Por favor, marque abaixo as que foram utilizadas.
2	Você fez alguma modificação nos seus protótipos após consultar as heurísticas? Se sim, por favor detalhe essas modificações.
3	Você gostaria de sugerir modificações nas heurísticas existentes? Quais modificações?
4	Você gostaria de sugerir a adição de novas heurísticas? Quais?
5	Você gostaria de sugerir a remoção de heurísticas que considera desnecessárias? Quais?

Adicional, apresentados por Quiñones et al. (2018) para validação de heurísticas.

Em relação à dimensão Utilidade, todos os participantes afirmaram ter utilizado heurísticas para desenvolver ou aprimorar seus protótipos. Especificamente, os avaliadores mencionaram o uso das heurísticas *Lista, Busca, Visualização, Grade, Cadastro, Página de Produto, Chat, Carrinho de Compras* e *Filtro de Conteúdo*, todas relacionadas a funcionalidades contidas na especificação utilizada no estudo. Um dos desenvolvedores (P1) relatou que durante o projeto de um protótipo de *Cadastro* (HEU1-03), que ele frequentemente implementa, incluiu elementos que não estavam no *checklist* da heurística e eram, na verdade, desnecessários. Da mesma forma, ele não desenhou vários componentes ao projetar um protótipo da funcionalidade *Carrinho de Compras*, que ele não costuma projetar. Ele observou os itens ausentes somente após usar a heurística do Carrinho de Compras (HEU2-02). O outro desenvolvedor (P2) mencionou que as heurísticas podem ser usadas sozinhas ou combinadas para desenhar protótipos mais complexos.

Em relação à dimensão da Clareza, os especialistas relataram alguns problemas. Os dois especialistas em IHC e um dos desenvolvedores indicaram que estavam confusos sobre as diferenças entre as heurísticas Busca (HEU1-02) e Filtro de Conteúdo (HEU1-04) e entre as heurísticas Lista (HEU1-06) e Grade (HEU1-05). Em relação às heurísticas de Busca e Filtro de Conteúdo, os especialistas relataram que as especificações são complementares e podem ser unificadas. Em relação

às heurísticas Lista e Grade, eles relataram que são apenas formas diferentes de apresentar um conjunto de instâncias, sugerindo ainda a inclusão de uma forma de exibição em carrossel. Um dos especialistas em IHC não compreendeu o significado da heurística *Mestre-Detalhe* (HEU1-08), que é usada para representar uma relação de dependência entre um objeto principal e outros subordinados. Por outro lado, ambos os desenvolvedores a relataram como uma heurística de grande relevância nos projetos que desenvolvem para ilustrar relações do tipo um para muitos. Além disso, um dos especialistas em IHC apontou dificuldade em compreender a relação estabelecida entre as heurísticas *Visualização* (HEU1-10) e *Perfil de Usuário* (HEU2-07), observando que essa correspondência não estava suficientemente clara. O especialista sugeriu que a associação mais coerente seria entre *Visualização* e *Cadastro* (HEU1-03), o que indica a necessidade de tornar o item "Heurísticas Relacionadas" mais explícito e consistente, de modo que as conexões entre heurísticas reflitam de forma mais evidente a lógica funcional subjacente ao catálogo.

Quanto à dimensão Facilidade de Uso, os avaliadores também identificaram alguns problemas. Por exemplo, um dos especialistas em IHC teve dificuldade em memorizar o nome das heurísticas e o seu significado. Como consequência, este especialista usou a heurística *Visualização* (HEU1-10) no design do protótipo da página de produto porque não encontrou a heurística mais adequada (HEU2-06), indicando uma necessidade de disponibilização de mecanismos de seleção ou recomendação de heurísticas. Apesar disso, os avaliadores também relataram facilidade na aplicação das heurísticas ao protótipo modelado e elogiaram a navegação entre as heurísticas por meio de links na seção Heurísticas Relacionadas.

Em relação à dimensão *Necessidade de uma Lista de Verificação Adicional*, os avaliadores sugeriram a adição de duas novas heurísticas de design: para o design de interfaces de avaliação e de interfaces acessíveis. A sugestão dessas novas categorias mostrou que as heurísticas de design precisam abranger funcionalidades de diferentes domínios de aplicação. Já a sugestão de heurísticas para design de interfaces acessíveis está relacionada a regras de design de um modo geral, e não a funcionalidades específicas, que são o escopo deste trabalho.

Os problemas e sugestões indicados durante a avaliação por especialistas foram debatidos entre os pesquisadores e resultaram nas seguintes melhorias no catálogo

de heurísticas:

- Adição de um item no *checklist* da heurística *Registro de Conta* para indicar o formato de senha aceito (HEU2-08);

- Unificação das heurísticas de Filtro de Conteúdo (HEU1-04) e Busca (HEU1-02);

- Unificação das heurísticas Lista (HEU1-06) e Grade (HEU1-05) e adição do formato de exibição Carrossel como opção para exibir uma lista de objetos;

- Adição de um relacionamento entre as heurísticas de Perfil de Usuário (HEU2-07) e Cadastro (HEU1-03).

As heurísticas apresentadas neste trabalho especificam características que descrevem funcionalidades comumente encontradas em sistemas de informação. Elas foram desenvolvidas com o objetivo de auxiliar a elaboração e validação de protótipos de interface. No entanto, alguns avaliadores confundiram seu propósito. Em algumas situações, eles assumiram que as heurísticas os ajudariam a projetar protótipos em relação a aspectos de usabilidade. Por exemplo, um dos especialistas em IHC (P3) relatou que um exemplo de Filtro de Conteúdo (HEU1-04) usou as cores vermelho e verde brilhantes e relatou que isso era uma recomendação de design equivocada. O outro especialista em IHC (P4) relatou que decidiu projetar um protótipo de Cadastro (HEU1-03) em várias páginas após analisar exemplos desta heurística, o que exigiria mais cliques do usuário.

Ao analisar os dados obtidos no grupo focal, foi verificado que a disponibilidade de exemplos de projeto a partir da base de protótipos obtida no passo 2 induziu os participantes a entender que os protótipos deveriam ter uma aparência semelhante aos exemplos. Outro problema que pode ter influenciado esta interpretação dos avaliadores foi que as heurísticas *Grade* (HEU1-05) e *Lista* (HEU1-06) continham, em sua especificação, indicações de como os componentes do protótipo deveriam estar dispostos, em desacordo com as demais heurísticas e a finalidade do CHIF.

4.5 Considerações Finais

Neste ciclo de design, foi desenvolvido um conjunto de heurísticas de apoio à inspeção funcional de protótipos de interface. Essas heurísticas, agrupadas em dois níveis de abstração, podem ser utilizadas para auxiliar em tarefas de design de interface do usuário, como a busca por interfaces com funcionalidade semelhante ou a verificação da adequação do protótipo a características das funcionalidades representadas através das heurísticas.

Em comparação com métodos de análise de protótipos de IU baseados em imagens, a abordagem baseada em heurísticas tem a vantagem de se basear nas características semânticas do design da IU e, portanto, não é limitada ao arranjo de elementos na interface nem sofre interferência de menus ou notificações sobrepostos. Apesar disso, abordagens baseadas em imagens, metadados de imagens e dados de interação do usuário (Zecheng et al., 2020) podem ser combinadas com a abordagem baseada em heurísticas durante o processo de classificação, conforme indicado por Li et al. (2021), ou complementadas após a classificação, onde as heurísticas funcionariam como *checklists* para verificar a adequação do design da IU.

Em relação aos aspectos semanticamente analisados, o trabalho de Liu et al. (2018) classifica componentes estruturais, botões de texto e ícones, localizados em um nível semântico anterior aos apresentados neste artigo, que classifica as interfaces como um todo e não como elementos isolados. O conjunto de dados EnRICO (Leiva et al., 2020), por sua vez, filtrou e classificou um subconjunto do conjunto de dados Rico, encontrando 20 categorias, embora não considere níveis semânticos. Seu trabalho lista diversas aplicações para o conjunto de dados rotulado, algumas das quais se aplicam ao conjunto de dados que apresentamos neste trabalho: descrição da IU com base no rótulo identificado, classificação da IU, listagem dos rótulos mais prováveis para uma IU e busca otimizada de GUIs baseadas em classificação.

Assim como esta pesquisa, o trabalho de Chen et al. (2020a) também identificou categorias de *tags* em diferentes níveis semânticos: plataforma, cor, funcionalidade do aplicativo, funcionalidade da tela e layout da tela. Em seu trabalho, cada IU pode receber até uma *tag* de cada uma das categorias identificadas. Na

abordagem apresentada neste capítulo, mais de uma heurística pode ser utilizada para avaliar um protótipo de interface.

4.5.1 Limitações

Como limitação deste estudo, foi identificada a quantidade reduzida de instâncias representando algumas das funcionalidades identificadas, o que pode ter limitado a identificação de características relevantes nos exemplos durante a definição das respectivas heurísticas. Para mitigar essa limitação, o catálogo de heurísticas pode ser ampliado com novas instâncias e funcionalidades, a partir da incorporação de *datasets* rotulados, como o EnRICO (Leiva et al., 2020).

Além disso, a avaliação do catálogo contou com um número restrito de especialistas em Interação Humano-Computador (IHC), o que limita a generalização dos resultados obtidos. Essa restrição decorre do fato de se tratar de uma avaliação preliminar da primeira versão do catálogo, cujo propósito principal foi verificar sua consistência interna e clareza conceitual antes de sua ampliação e aplicação em contextos empíricos mais amplos. Adicionalmente, reconhece-se a possibilidade de viés decorrente da familiaridade dos avaliadores com processos de inspeção. Especialistas com maior experiência prévia em técnicas de inspeção heurística podem ter interpretado as heurísticas sob a ótica de abordagens tradicionais voltadas à usabilidade, o que pode ter influenciado a percepção de sua aplicabilidade funcional.

Outro fator limitante refere-se à ausência de um contexto real de uso, uma vez que a validação foi conduzida em ambiente controlado e não inserida em projetos reais de desenvolvimento. A aplicação das heurísticas em situações autênticas de inspeção de protótipos poderia revelar desafios adicionais, como restrições de tempo, pressões de projeto e variações nas práticas de modelagem de interface. Para contornar essas limitações, avaliações de futuras versões do catálogo poderão ser realizadas por um grupo maior e mais diversificado de especialistas ou profissionais da indústria de software. O próximo capítulo apresenta a segunda iteração do ciclo de Design, que foi realizada com esta finalidade.

Capítulo 5

Expansão do Catálogo de Heurísticas de Inspeção Funcional (CHIF-E)

No Capítulo 4, o CHIF (Catálogo de Heurísticas de Inspeção Funcional) foi desenvolvido como um instrumento de suporte à prototipação de interfaces de usuário, permitindo a identificação de omissões potenciais nas funcionalidades descritas no catálogo de heurísticas. Tais funcionalidades estão agrupadas em dois níveis semânticos: (i) as de *Funcionalidades Básicas*, como *Cadastro*, *Alteração* ou *Busca*, e (ii) as de *Funcionalidades Específicas*, como *Notícias* ou *Configurações*. A avaliação preliminar do CHIF mostrou que ele pode ser utilizado para auxiliar a elaboração de protótipos de interface. No entanto, para torná-lo um instrumento mais abrangente de maneira que pudesse dar suporte à inspeção funcional de protótipos de IU, foi necessário ampliar o escopo de funcionalidades nele descritas, incluindo novas funcionalidades de diferentes domínios de aplicação. Este capítulo apresenta a segunda iteração do ciclo de pesquisa conduzido nesta tese, fundamentada na abordagem de Design Science Research (DSR). O objetivo desta etapa foi ampliar e avaliar o Catálogo de Heurísticas de Inspeção Funcional (CHIF-E), de modo a verificar seu potencial de apoio à identificação de defeitos funcionais em protótipos de interface de usuário em diferentes contextos de uso.

O método de pesquisa adotado neste capítulo envolveu a condução de um

estudo experimental estruturado em duas fases. Na primeira fase, foi realizado um experimento controlado com o objetivo de comparar o desempenho dos dois instrumentos de inspeção desenvolvidos nesta iteração na detecção de defeitos funcionais em protótipos de um sistema de comércio eletrônico. Na segunda fase, realizou-se um estudo experimental exploratório, voltado à análise qualitativa dos tipos de defeitos identificados pelos participantes durante o uso do CHIF expandido, buscando compreender o potencial do catálogo para apoiar a inspeção funcional de protótipos em diferentes contextos de uso.

5.1 Introdução

A inclusão de novas funcionalidades no CHIF envolveu a adaptação do processo proposto por Klie et al. (2024) para anotação de *datasets* com atividades de garantia da qualidade. As adaptações incluídas no processo proposto envolveram atividades como a seleção de *datasets* rotulados em tópicos para identificação de novas funcionalidades, a análise de novas funcionalidades potenciais, a revisão e anotação de instâncias, a especificação de novas heurísticas, além de procedimentos para incorporá-las ao catálogo atual.

O processo adaptado foi utilizado para incorporar novas funcionalidades ao CHIF a partir do EnRICO (Leiva et al., 2020), um *dataset* de imagens capturadas de aplicativos móveis rotulado em tópicos, contendo 1.460 capturas de tela categorizadas em 20 tópicos. Este processo resultou no CHIF-E, um catálogo de heurísticas de inspeção funcional expandido, com nove novas funcionalidades especificadas através de heurísticas, e na reclassificação de 781 das 1460 capturas de tela do EnRICO.

Além disso, visando definir uma *baseline* para avaliar o desempenho do CHIF-E em inspeções funcionais de protótipos de interface, foi elaborado o UIProto-Check, um *checklist* genérico para inspeção funcional de protótipos de interface. O UIProtoCheck não é voltado para funcionalidades específicas; em vez disso, ele avalia a interface de usuário em três diferentes perspectivas, cada uma de acordo com os critérios de Corretude, Completude e Coesão, independentemente da funcionalidade representada.

Para investigar como os instrumentos desenvolvidos neste ciclo, CHIF-E e

UIProtoCheck, podem ser empregados na detecção de defeitos funcionais em protótipos de interface de usuário durante inspeções de artefatos de software, foram formuladas as seguintes questões de pesquisa:

> **QP1:** Como o CHIF-E e o UIProtoCheck oferecem suporte à inspeção funcional de protótipos de interface de usuário?
> **QP2:** Quais tipos de defeitos podem ser encontrados durante inspeções funcionais de protótipos de interface de usuário usando o CHIF-E?

Para responder a estas questões, foi realizado um estudo experimental de duas fases. A primeira fase consistiu em um experimento controlado no qual os participantes, com base em um cenário de aplicação, inspecionaram três protótipos de IU contendo defeitos inseridos intencionalmente, visando avaliar a eficácia dos instrumentos de inspeção. Metade dos participantes usou o CHIF-E, enquanto a outra metade utilizou o UIProtoCheck como instrumento de apoio à inspeção. Na segunda fase, participantes organizaram-se em equipes e desenvolveram projetos de aplicações com o uso de prototipação de interfaces. Em seguida, cada equipe inspecionou os protótipos de interface do projeto de outra equipe utilizando o CHIF-E. O processo de incorporação de novas funcionalidades ao CHIF, a elaboração do UIProtoCheck e os resultados da avaliação do desempenho destes instrumentos em inspeções funcionais de protótipos de interface são apresentados nas próximas seções.

5.2 Adição de novas funcionalidades ao CHIF

A expansão do CHIF com a inclusão de novas funcionalidades utilizou o processo apresentado por Klie et al. (2024) para anotação de *datasets* com avaliação da qualidade. Este processo foi adotado porque o método de incorporação de novas heurísticas ao catálogo envolve a análise e rotulagem de imagens de interfaces de usuário, mesmo que sejam usados *datasets* previamente rotulados. Além disso, o processo proposto envolve um ciclo iterativo de avaliação da qualidade da rotulagem e de ajustes no esquema de anotação.

O processo de anotação descrito por Klie et al. (2024), ilustrado pela Figura 5.1, é estruturado em ciclos iterativos que visam assegurar qualidade e consistência

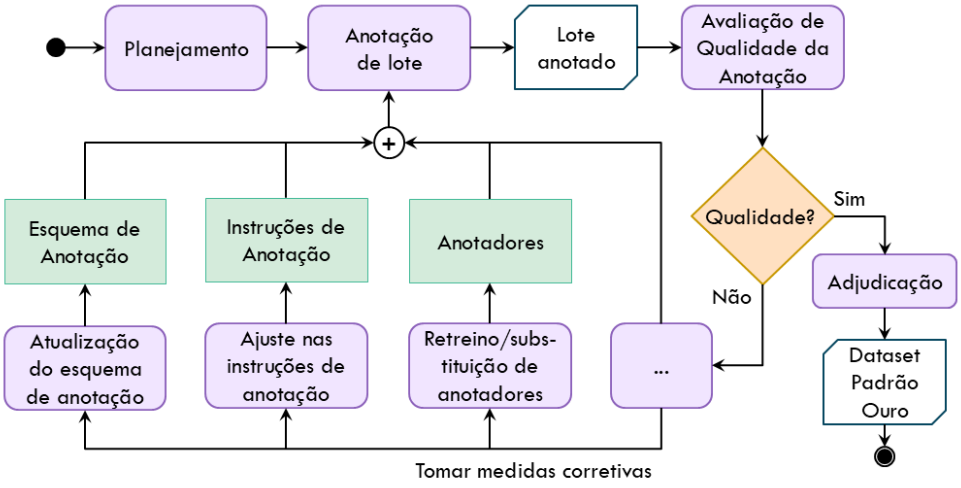


Figura 5.1: Processo para anotação de *datasets* com atividades de garantia da qualidade. Fonte: Klie et al. (2024).

nas anotações. Ele se inicia com a fase de *Planejamento*, que envolve a tomada de decisões preliminares sobre o procedimento de anotação, tais como a definição de objetivos, anotadores, dados, cronograma e esquema de anotação. Em seguida, na fase *Anotação de lote*, apenas um lote (batch) do conjunto de dados é anotado, o que permite detectar problemas iniciais e realizar ajustes antes de avançar. Após a anotação, ocorre a fase de *Avaliação de Qualidade da Anotação*, onde a qualidade é estimada por métodos como inspeção manual, uso de instâncias de controle ou cálculo de concordância entre anotadores; se o nível de qualidade for insuficiente, aplicam-se medidas corretivas como atualização de diretrizes, treinamento de anotadores ou reanotação. Finalmente, quando a qualidade do lote é considerada adequada, passa-se à fase de *Adjudicação*, que consiste em consolidar múltiplas anotações em uma única versão, geralmente por métodos como votação majoritária, desempate manual ou técnicas probabilísticas. Esse ciclo se repete até que um número suficiente de lotes alcance a qualidade esperada, obtendo-se um *dataset Gold Standard*.

Para adaptar o processo de Klie et al. (2024) para o contexto deste trabalho, foram incluídas atividades específicas de ampliação do catálogo de heurísticas nas etapas de planejamento e anotação do processo, destacadas em negrito na Figura 5.2. À fase de planejamento, foram acrescentadas as atividades de (i) seleção de

datasets que estivessem previamente rotulados em tópicos, representando candidatos a funcionalidades, e de (ii) categorização dos tópicos do *dataset* selecionado em funcionalidades existentes ou novas funcionalidades potenciais, cujo resultado compôs o conjunto de *tags* utilizado durante a rotulagem. Na fase iterativa com ciclos de rotulagem, foram realizadas as atividades de revisão e reanotação das instâncias tanto do *dataset* selecionado quanto do original (CHIF), considerando o novo conjunto de tags utilizado no processo de anotação.

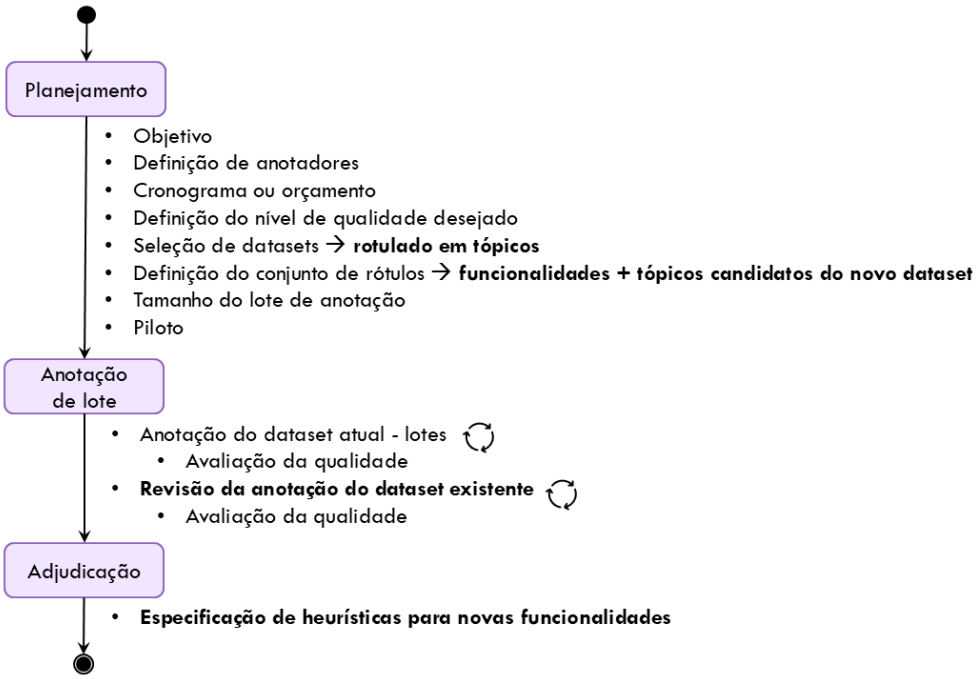


Figura 5.2: Processo de anotação adaptado para contemplar atividades de expansão do catálogo de heurísticas.

Finalmente, após a anotação do novo *dataset*, procedeu-se à especificação de heurísticas para as novas funcionalidades identificadas, seguindo a mesma metodologia utilizada durante a criação do CHIF (Quiñones et al., 2018), sendo realizadas as fases de Especificação, Validação e Refinamento. As etapas do processo de anotação para expansão do catálogo de heurísticas são detalhadas nas seções a seguir.

5.2.1 Planejamento

No contexto desta iteração do Ciclo de Design, a anotação foi realizada com o objetivo de descobrir novas funcionalidades e incorporá-las ao CHIF, visando aumentar sua abrangência e, com isso, dar suporte à inspeção funcional de protótipos de interface. Para alcançar esse objetivo, a seleção de dados da fase de planejamento precisou se limitar a *datasets* de protótipos de interface de usuário ou de capturas de tela de aplicativos que estivessem anotados em tópicos que indicassem a funcionalidade representada nas imagens. Desta forma, foi realizada uma revisão de literatura visando encontrar contribuições de *datasets* de interfaces de usuários que pudessem ser utilizados neste estudo. Na ocasião em que esta revisão foi realizada, foram encontrados diversos *datasets* relacionados à interface de usuário.

O RICO (Deka et al., 2017) e o RaWi (Kolthoff et al., 2023) são *datasets* não rotulados de GUIs extraídas de aplicativos disponíveis em *app stores*, o que os torna inadequados para identificação de novas funcionalidades. O *dataset* ReDraw (Moran et al., 2020) inclui capturas de tela de aplicativos Android, metadados e componentes de interface, mas também não possui anotações relativas à funcionalidade de cada tela. Os estudos de Sermuga Pandian et al. (2021a,b) fornecem *datasets* de interfaces de usuário com elementos de IU rotulados, mas sem anotações referentes à funcionalidade representada na interface como um todo. De maneira similar, o LoFiSketch (Sermuga Pandian et al., 2022) é uma coleção de esboços de componentes de interface desenhados à mão com anotações sobre o tipo de componente esboçado.

Diferentemente dos *datasets* listados anteriormente, o EnRICO (Leiva et al., 2020) e o Screen2Words-dataset (Wang et al., 2021) são *datasets* de imagens capturadas de interfaces de aplicativos móveis com anotações sobre o significado do conteúdo nelas representado. Ambos são formados por amostras do *dataset* RICO (Deka et al., 2017). O Screen2Words-dataset possui 112.085 anotações sobre 22.417 telas únicas realizadas por anotadores humanos. As anotações são descrições textuais de alto nível que indicam o propósito e a função principal da interface de forma descritiva. Por oferecer múltiplas descrições textuais para a mesma tela, seria necessária uma etapa adicional de tratamento das anotações

do Screen2Words-dataset de maneira a unificar e condensar as anotações de cada tela em um tópico único e representativo de sua funcionalidade. Já o EnRICO (Leiva et al., 2020) é um dataset rotulado em 20 tópicos, com 1.460 capturas de tela de aplicativos acompanhadas de suas respectivas hierarquias de visualização e metadados. Por este motivo, este foi o *dataset* escolhido neste estudo para incorporar novas funcionalidades ao catálogo de heurísticas de inspeção funcional.

A fase de planejamento do processo de anotação adotado compreende também a definição do conjunto de *tags* a ser usado na rotulagem. Esse conjunto é revisado a cada ciclo da fase de anotação. No contexto deste trabalho, o conjunto de *tags* original, com as funcionalidades do CHIF, foi complementado com as funcionalidades candidatas obtidas do *dataset* selecionado EnRICO (Leiva et al., 2020). Para selecionar o conjunto de *tags* candidatas, cada tópico do EnRICO foi avaliado a partir da sua descrição, de maneira a verificar se correspondia a uma funcionalidade distinta. Por exemplo, o tópico *Chat* corresponde a uma funcionalidade identificável, visto que representa o recurso de bate-papo. Já no caso do tópico *Dialer*, cuja descrição era *Entrada de número*, não foi possível associá-lo a nenhuma funcionalidade.

Os tópicos foram avaliados por dois especialistas em Engenharia de Requisitos, o primeiro deles com mais de 27 anos de experiência na indústria de desenvolvimento de software e o segundo com mais de 15 anos de experiência na academia e de 12 anos na indústria de software. Os tópicos identificados pelos especialistas como equivalentes a funcionalidades comumente encontradas em sistemas de informação foram pré-selecionados para compor o conjunto de *tags*, como foi o caso do tópico *Maps (Mapa)*. Em seguida, os tópicos pré-selecionados foram avaliados visando verificar se correspondiam a alguma funcionalidade já especificada na primeira versão do CHIF. Nos casos em que houve uma correspondência, foi definido o nome mais adequado para representar a funcionalidade no conjunto de *tags*.

O resultado da definição do conjunto inicial de *tags*, contemplando funcionalidades obtidas a partir dos tópicos do dataset EnRICO, é ilustrado na Figura 5.3. Foram identificadas oito novas funcionalidades/tags, representadas pelos tópicos Câmera, Editor, Galeria, Mapas, *Media Player*, Notícias, Configurações e Tutorial. Os tópicos Bate-papo, Formulário (Cadastrar e Atualizar), Lista, Login (Auten-

ticação), Menu, Perfil e Busca correspondem a funcionalidades já existentes no CHIF. Já os tópicos do EnRICO não identificados inicialmente como novas funcionalidades são *Bare, Dialer, Other, Modal* e *Term*. O tópico *Bare* é caracterizado por telas com extensas áreas em branco, sem funcionalidade específica. O tópico *Dialer* não representa uma funcionalidade distinta; em vez disso, ele é uma forma de entrada de dados com teclado numérico. O tópico *Other* inclui telas que não se enquadram em nenhum outro tópico. Os tópicos *Modal* e *Term* já haviam sido desconsiderados no Estágio de Seleção da metodologia adotada para especificação de heurísticas na primeira iteração do Ciclo de Design (Capítulo 4). A Tabela 5.1 apresenta uma descrição das funcionalidades que formaram o conjunto de funcionalidades usado nas anotações.

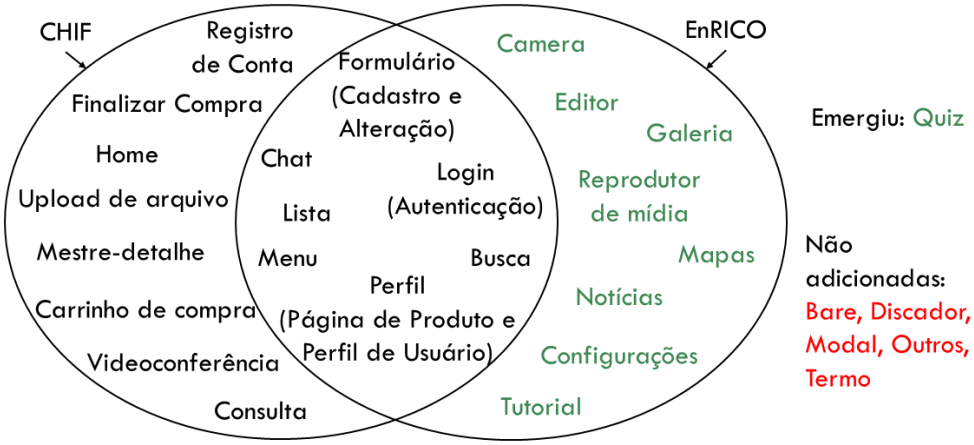


Figura 5.3: Funcionalidades do CHIF e EnRICO.

As imagens do *dataset* foram anotadas/revisadas em lotes de acordo com os tópicos do EnRICO. Os tópicos maiores foram divididos em lotes menores de 50 imagens. Para a anotação, foi utilizada uma planilha contendo uma lista das imagens do EnRICO e o tópico com o qual foram rotuladas.

Antes de iniciar o processo de anotação, foi conduzido um piloto com as imagens rotuladas com o tópico *Dialer*, que possuía uma quantidade reduzida de instâncias (6). O resultado obtido com este estudo revelou que o tópico agrupou capturas de tela por sua aparência (teclado numérico) em vez de sua funcionalidade (Calculadora - 1, Autenticação - 3 e Menu - 2). Este resultado, ilustrado na Figura 5.4, indicou que os tópicos não necessariamente representam funcio-

Tabela 5.1: Resumo das funcionalidades do CHIF-E.

Tópicos do EnRICO mapeados para funcionalidades existentes no CHIF	
Chat	Troca de mensagens instantâneas entre usuários
Formulário (Cadastro e Alteração)	Funcionalidade de preenchimento de formulário
Lista	Conjunto de instâncias de uma entidade de domínio
Login (Autenticação)	Informação de credenciais de acesso a recursos da aplicação
Menu	Opções de recursos da aplicação
Perfil (Página de Produto e Perfil de Usuário)	Informações sobre um perfil de usuário ou produto
Busca	Funcionalidade de mecanismo de busca
Funcionalidades do CHIF não presentes no EnRICO	
Registro de Conta	Criação de uma conta de usuário para acessar os recursos de um aplicativo
Finalizar Compra	Informações sobre opções de pagamento, envio e cobrança em aplicativos de comércio eletrônico
Upload de Arquivo	Um formulário de upload de arquivo
Home	Página inicial com principais informações e recursos
Mestre-detalhe	Objeto de uma entidade seguido por uma lista de objetos de outra entidade relacionados ao primeiro
Carrinho de Compra	Lista de produtos ou serviços e suas quantidades para compra em aplicações de e-commerce
Videoconferência	Funcionalidade de videoconferência
Visualização	Exibição de informações de um objeto de uma entidade da aplicação
Tópicos do EnRICO adicionados ao CHIF-E	
Câmera	Funcionalidade de câmera
Editor	Edição de texto/imagem
Galeria	Layout no estilo grade com imagens
Mapas	Exibição geográfica
Reprodutor de mídia	Reprodutor de música ou vídeo
Notícias	Lista de itens com imagem, título, texto
Configurações	Controles para alterar configurações da aplicação
Tutorial	Tela de *onboarding*
Tópicos do EnRICO não adicionados ao CHIF-E	
Bare	Grande área não utilizada
Discador	Entrada numérica
Modal	Uma janela pop-up
Outros	Todo o resto (classe de rejeição)
Termo	Termos e condições de serviço
Funcionalidade emergida durante análise de instância	
Quiz	Um questionário com respostas corretas predefinidas

nalidades. Desta forma, o procedimento de anotação foi elaborado de maneira a garantir a análise de cada imagem de acordo com sua funcionalidade.

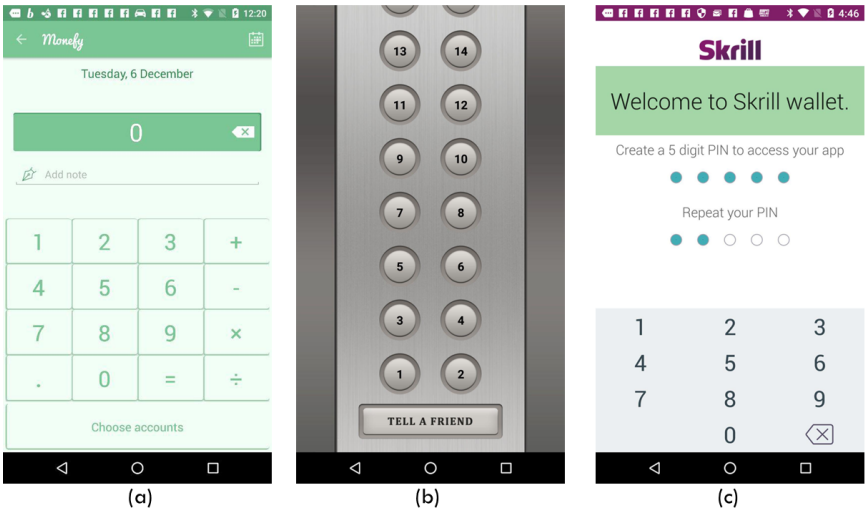


Figura 5.4: Exemplos de instâncias do tópico *Dialer*. (a) Calculadora. (b) Botões de elevador. (c) Criação de PIN.

No procedimento de anotação, cada imagem foi analisada por um especialista para verificar se a funcionalidade representada estava de acordo com o tópico com que foi anotada. Ao lado do tópico originalmente anotado, foi anotado o rótulo atribuído pelo especialista. As funcionalidades representadas que não estavam contempladas no conjunto de *tags* foram adicionadas ao conjunto com uma breve descrição da nova funcionalidade.

Nos casos em que vários rótulos poderiam ser atribuídos a uma imagem, foi selecionado aquele que melhor descrevia a funcionalidade representada de acordo com a descrição do tópico, priorizando funcionalidades específicas às básicas. Quando não estava claro qual rótulo era mais adequado, foi atribuído mais de um rótulo à imagem.

Como medida de avaliação da qualidade, a cada lote anotado, um segundo especialista revisou a rotulagem. Pontos de discordância foram debatidos, com eventuais ajustes nas anotações e no conjunto de *tags*.

5.2.2 Anotação Iterativa em Lote

As imagens de cada tópico do *dataset* EnRICO foram revisadas e anotadas conforme definido no procedimento de anotação. Com base no nome e na descrição do tópico, foi verificado se elas se enquadravam adequadamente no tópico atribuído, se deveriam ser anotadas com outro tópico do conjunto de *tags*, ou se deveriam ser anotadas com uma nova funcionalidade identificada durante a análise. Ao final desta fase, 781 de 1.460 do EnRICO telas foram reclassificadas, o que corresponde a 53% do *dataset*.

A atividade de anotar as imagens utilizando o conjunto de *tags* revelou-se uma tarefa subjetiva. Frequentemente, uma tela podia ser anotada com mais de uma funcionalidade, especialmente considerando que as funcionalidades do CHIF são organizadas em dois níveis de abstração: Funcionalidades Básicas e Funcionalidades Específicas. Por exemplo, na tela da Figura 5.5(a) é possível verificar que ela foi inicialmente rotulada como *Lista*, mas após a reanálise, foi reclassificada como *Configurações*. Esta tela mostra uma lista de configurações, o que significa que ela pode ser anotada com ambos os rótulos. Outro fator que pode ter contribuído para o alto número de telas reanotadas é que os anotadores do EnRICO poderiam estar inclinados a anotar uma tela com base em sua forma em vez de sua finalidade. Um exemplo é a captura de tela ilustrada na Figura 5.5(b), que foi inicialmente anotada como *Configurações*, mas foi reanotada como *Visualização* durante a revisão das anotações.

Estabelecidos estes critérios de anotação, esta etapa iniciou-se pela análise de instâncias dos oito tópicos do EnRICO identificados como novas funcionalidades: Câmera, Editor, Galeria, Mapas, Media Player, Notícias, Configurações e Tutorial. Esses tópicos corresponderam a 523 telas do EnRICO, das quais 261 foram reanotadas pelos dois especialistas durante a análise em tópicos que melhor descreviam a funcionalidade que representavam. Posteriormente, foram examinados os sete tópicos do conjunto de dados do EnRICO correspondentes às funcionalidades do CHIF *Bate-papo, Formulário, Lista, Login, Menu, Perfil e Busca*. Esses tópicos foram mapeados diretamente para as funcionalidades correspondentes no catálogo de heurísticas, exceto pelos tópicos *Formulário*, que foi mapeado para as funcionalidades *Cadastrar* e *Atualizar*, e *Perfil*, mapeado para as funcionalidades

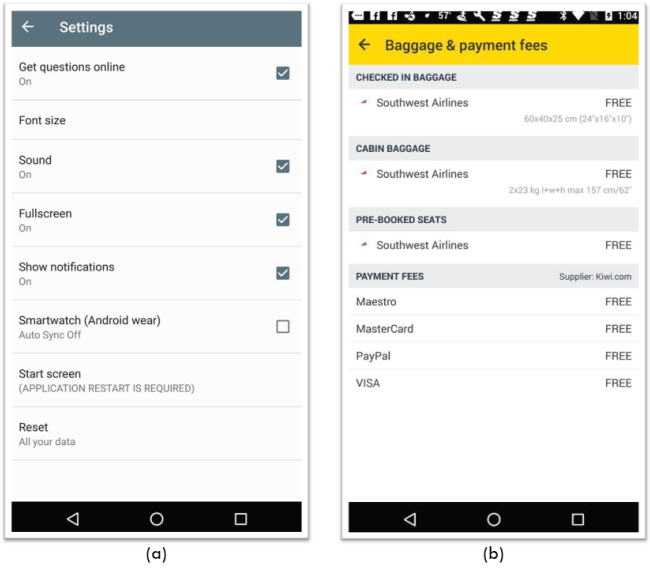


Figura 5.5: Análise de instância. (a) Uma captura de tela de *Lista* reclassificada como *Configurações*. (b) Uma captura de tela de *Configurações* reclassificada como *Visualização*.

Página do produto e *Perfil do usuário*. Esses tópicos corresponderam a 697 telas no conjunto de dados, com 325 passando por reclassificações.

Também foram avaliadas as instâncias dos tópicos não incorporados anteriormente, *Bare, Other, Modal* e *Terms*. A avaliação das instâncias do tópico *Bare* levou à reclassificação de 59 das 76 telas originais para os seguintes tópicos: Lista (2), Mestre-Detalhe (1), Menu (6), Notificação (39), Cadastro (1), Tutorial (9) e Visualização (1). Em relação ao tópico *Other*, uma reavaliação das 52 telas resultou na reclassificação de 34 telas em dez tópicos diferentes, incluindo Cadastro, Lista e Notícias. Também foram revisadas as telas dentro dos tópicos *Modal* e *Terms*, reclassificando 16 das 67 telas do tópico *Modal* e 30 das 39 telas do tópico *Terms*. Por fim, uma nova categoria de funcionalidade, *Quiz*, emergiu durante o processo de análise e revisão. Esta categoria representa interfaces que fazem perguntas e oferecem uma lista de opções para o usuário, em que apenas uma resposta é a correta.

Finalmente, a anotação dos protótipos de interface originalmente utilizados na definição do CHIF foi revisada, considerando as novas funcionalidades identificadas no *dataset* EnRICO. Esta revisão teve como objetivo avaliar se essas

instâncias estariam melhor rotuladas com alguma das funcionalidades recentemente introduzidas. Ao longo deste processo, 23 protótipos de interface passaram por reclassificação nas categorias Câmera (1), Galeria (17), Mapas (1), Configurações (3) e Tutorial (1).

5.2.3 Adjudicação - Especificação de novas heurísticas

Após a fase de anotação das imagens do *dataset* EnRICO (Leiva et al., 2020) e do *dataset* original do CHIF, procedeu-se à especificação de heurísticas para as funcionalidades adicionadas. As instâncias enquadradas nas novas categorias foram examinadas pelo primeiro especialista para identificar características específicas que pudessem compor a especificação das heurísticas.O modelo adotado para especificação de heurísticas de design foi o proposto por De Macedo et al. (2022), o qual foi adaptado da metodologia apresentada por Quiñones et al. (2018). Consequentemente, as novas heurísticas são compatíveis com a versão inicial do CHIF. As heurísticas derivadas dessa análise foram revisadas pelo segundo especialista. A heurística da funcionalidade Configurações é apresentada na Tabela 5.2, e a Figura 5.6 mostra exemplos dessa funcionalidade.

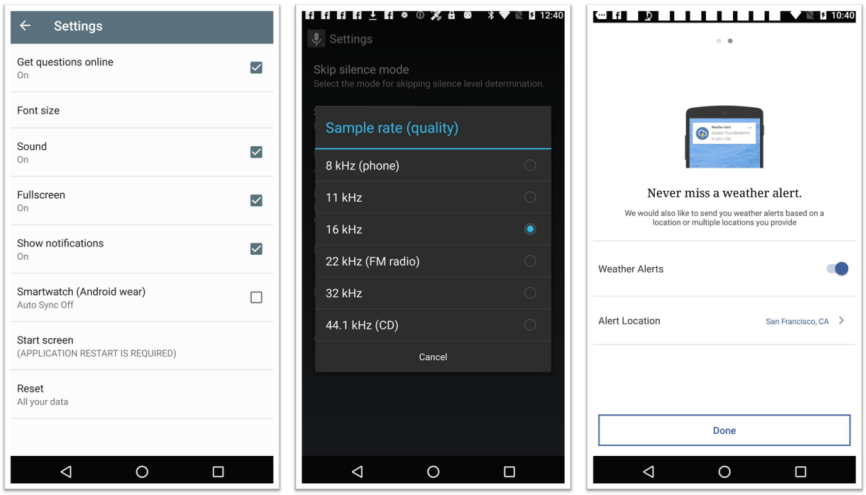


Figura 5.6: Exemplos de protótipos de interface classificados como Configurações.

As demais heurísticas incluídas nesta iteração do Ciclo de Design estão disponíveis no Apêndice A, que apresenta a versão final do Catálogo de Heurísticas de

Tabela 5.2: Heurística HEU2-05 – Configurações.

ID: HEU2-05

Nome: **Configurações**
Sinônimos: Preferências
Definição: Representa as configurações de preferência de uso de uma aplicação.
Explicação: Um protótipo do tipo Configurações representa as diferentes maneiras de configurar uma aplicação.
Exemplo: Um usuário escolhe que tipo de notificações receber de um aplicativo de previsão do tempo.
Checklist:

1. O título do protótipo é um termo comumente usado para descrever telas de configuração. Exemplos: Configurações, Notificações, Preferências.
2. O protótipo apresenta um ou mais itens correspondentes às opções personalizáveis da aplicação.
3. Opcionalmente, as opções de configuração podem ser agrupadas em seções ou subtelas.

Informações relacionadas: –
Heurísticas relacionadas: Alteração

Inspeção Funcional Expandido (CHIF-E).

Em relação às características das funcionalidades descritas no CHIF-E, é importante salientar que, enquanto as *Funcionalidades Básicas* são transversais a diversos domínios, as *Funcionalidades Específicas* tendem a se aplicar a contextos e tipos de aplicação bem definidos. Por exemplo, a funcionalidade *Câmera* é bastante utilizada em aplicativos móveis, enquanto é muito menos utilizada em aplicações *desktop* ou *web*. Como o *dataset* utilizado para a geração do CHIF-E tem como fonte aplicações móveis, as funcionalidades adicionadas tenderam a ser voltadas para este tipo de aplicação. Iterações futuras podem se concentrar na análise de protótipos de domínios de aplicações específicos, de maneira a identificar e contemplar suas funcionalidades típicas. A Tabela 5.3 apresenta o contexto em que as *Funcionalidades Específicas* do CHIF-E são esperadas.

Tabela 5.3: Funcionalidades Específicas do CHIF-E e seus contextos de aplicação

Funcionalidade	**Descrição**	**Contexto de Uso**
Autenticação(Login)	Acesso mediante credenciais	Qualquer sistema com área restrita ou personalizada
Câmera	Captura de imagem via dispositivo	Apps mobile com upload de imagens, verificação de identidade, QR code
Carrinho de Compra	Lista de produtos a serem adquiridos	E-commerce, apps de supermercado ou delivery
Chat	Troca de mensagens entre usuários	Apps colaborativos, batepapos, atendimento ao cliente
Configurações	Preferências do usuário	Aplicativos personalizáveis.
Editor de Imagens	Interface para modificação de imagens	Redes sociais, editores de documentos, apps de design gráfico, editores de postagens.
Finalizar Compra	Etapa de finalização de compra	E-commerce, apps de delivery, reservas
Home	Tela inicial com principais recursos	Qualquer app ou sistema estruturado em seções
Mapas	Exibe a localização geográfica	Apps de entrega, mobilidade, turismo, imóveis, eventos
Media Player	Reprodução de áudio ou vídeo	Apps de streaming, educação, treinamento
Notícias	Lista de atualizações ou notícias	Portais, intranets, apps informativos
Página de Produto	Informações detalhadas de um produto	E-commerce, catálogos, *marketplaces*
Perfil de Usuário	Informações do usuário de uma aplicação	Redes sociais, e-commerce
Quiz	Perguntas avaliativas	E-learning, jogos educacionais, avaliação em apps corporativos
Registro de Conta	Cadastro de novo usuário	Apps com autenticação, redes sociais, sistemas de serviço
Tutorial / Onboarding	Apresentação inicial do sistema	Apps com curva de aprendizado inicial ou fluxo de ativação
Videoconferência	Comunicação por vídeo e áudio em tempo real	Ferramentas de reunião, EAD, saúde remota

5.3 UIProtoCheck: Um *checklist* genérico para inspeção funcional de protótipos de interface

A avaliação do desempenho do CHIF-E como instrumento de apoio à identificação de defeitos funcionais em protótipos de interface necessitou de um instrumento de inspeção que estabelecesse uma linha de base para comparação. Tal instrumento, ao contrário do CHIF-E, não deveria se ater a funcionalidades específicas. Em vez disso, precisava ser genérico o suficiente para ser aplicado a qualquer funcionalidade representada, de maneira a auxiliar a identificação de defeitos relacionados à interpretação e representação visual de recursos de acordo com a especificação de requisitos.

A Leitura Baseada em *Checklists* (Checklist-Based Reading - CBR) é uma técnica de leitura para revisão de artefatos de software (Thelin et al., 2004) que orienta o processo de inspeção com base em recomendações ou perguntas. A CBR é considerada a técnica básica para inspecionar artefatos de engenharia de software que utiliza *checklists* que informam aos revisores quais perspectivas e características devem ser consideradas durante a inspeção (De Mello et al., 2016). Por este motivo, foi desenvolvido um *checklist* de propósito geral como *baseline* para avaliação do CHIF-E. O UIProtoCheck foi desenvolvido com o objetivo de auxiliar a realização de inspeções rápidas em protótipos de IU segundo seu aspecto funcional. Ele é destinado a aplicativos com uso intensivo de dados, como sistemas de informação e comércio eletrônico. Por se tratar de um instrumento de inspeção genérico, inspeções de protótipos de interface usando este *checklist* necessitam de um documento de referência que especifique o comportamento esperado dos recursos representados nos protótipos de IU. Este documento pode ser uma descrição textual do comportamento do sistema, uma especificação de requisitos ou um documento descrevendo casos de uso.

Para criar o UIProtoCheck, foram adotados os princípios tradicionais definidos por Laitenberger et al. (2000) apud Chernak (1996) para desenvolvimento de *checklists*. Os princípios são um esquema formado por dois componentes: *Onde Olhar* e *Como Detectar*. Primeiro, para definir o componente *Onde Olhar*, foram analisadas as diferentes perspectivas de uma IU que devem ser avaliadas durante a inspeção em busca de defeitos. Foram identificadas três perspectivas: (i) as

informações exibidas no protótipo, como rótulos e imagens; (ii) campos de entrada de dados, geralmente relacionados a atributos de domínio; e (iii) elementos de interação, como ícones, links e botões, representando ações que um usuário pode executar na interface.

Após identificar essas três perspectivas, foi definido como cada uma deve ser analisada (componente Como Detectar). Dessa forma, foram elaboradas questões dentro de cada perspectiva para avaliar o protótipo de acordo com os critérios de Correção, Completude e Consistência, usados para avaliar a qualidade dos artefatos de engenharia de software (Laitenberger et al., 2000). No contexto do método proposto, esses critérios foram interpretados da seguinte forma:

- **Correção** – se os elementos contidos no protótipo representam adequadamente a funcionalidade descrita no documento de referência;

- **Completude** – se o protótipo contém todos os elementos da funcionalidade representada conforme especificado no documento de referência;

- **Consistência** – se o protótipo possui elementos adicionais e contradições entre seus elementos ou entre o protótipo de IU e o documento de referência.

Além das perguntas relacionadas às três perspectivas, foram incluídas perguntas gerais que visam avaliar se o protótipo representa de forma clara e inequívoca um recurso descrito no documento de referência. As perguntas resultantes são apresentadas na Tabela 5.4. A primeira parte do *checklist* refere-se ao protótipo como um todo, enquanto as demais seções são específicas para as três perspectivas identificadas: informações exibidas, elementos de entrada de dados e itens de interação.

A próxima seção detalha o estudo realizado para avaliar como o CHIF-E e o UIProtoCheck podem ser utilizados como instrumentos de identificação de defeitos funcionais em protótipos de interface do usuário durante inspeções de artefatos de software.

Tabela 5.4: UIProtoCheck - *checklist* para inspeção de protótipos de interface de usuário

Onde procurar	#	Como detectar
Protótipo de IU	1	A funcionalidade representada está claramente identificada?
	2	A funcionalidade representada no protótipo corresponde a uma funcionalidade especificada no documento de referência?
	3	O protótipo apresenta informações ambíguas entre seus elementos ou entre seus componentes e o documento de referência?
	4	O protótipo usa nomes, termos e descrições de acordo com a funcionalidade correspondente no documento de referência?
Informações exibidas	5	Há consistência entre as informações exibidas e definidas no documento de referência?
	6	Há informações faltando no protótipo que devem ser exibidas?
	7	Há alguma informação adicional exibida no protótipo?
Elementos de entrada de dados	8	Os campos de entrada de dados do protótipo são consistentes com o documento de referência (se aplicável)?
	9	Algum campo de entrada deve ser incluído no protótipo?
	10	Há campos de entrada adicionais no protótipo além daqueles definidos no documento de referência?
Elementos de interação (links, ícones e botões)	11	Os elementos de interação oferecidos pelo protótipo de interface são consistentes com a funcionalidade especificada no documento de referência?
	12	De acordo com o documento de referência, há algum elemento de interação na interface que precisa ser adicionado?
	13	De acordo com o documento de referência, o protótipo oferece algum elemento de interação além daqueles correspondentes à sua funcionalidade?

5.4 Avaliação

Para avaliar como o CHIF-E oferece suporte a inspeções funcionais de protótipos de interface de usuário, a metodologia de pesquisa adotada consiste em um estudo com duas fases (Easterbrook et al., 2008). A primeira fase compreendeu um experimento controlado para avaliar o desempenho do CHIF-E (funcionalidades catalogadas) na identificação de defeitos em protótipos de interface, comparando seu desempenho ao do UIProtoCheck. Na segunda fase, foi realizado um estudo experimental para analisar os tipos de defeitos encontrados pelos participantes usando o CHIF-E. Nesta fase, times de participantes utilizaram o CHIF-E para inspecionar protótipos de interface de oito projetos de software. As seções a seguir detalham o planejamento e a execução das duas fases do estudo.

5.4.1 Experimento Controlado: CHIF-E X UIProtoCheck

O objetivo da primeira fase do estudo foi avaliar se os instrumentos de inspeção desenvolvidos nesta iteração quanto ao suporte à inspeção funcional de protótipos de interface do usuário. Para orientar o estudo, a seguinte questão de pesquisa foi formulada:

> **QP1:** Como o CHIF-E e o UIProtoCheck oferecem suporte à inspeção funcional de protótipos de interface do usuário?

Para responder a essa questão, foi planejado um experimento controlado com o intuito de comparar o desempenho do CHIF-E com o do UIProtoCheck na detecção de defeitos funcionais em protótipos de interface. Desta forma, seria possível determinar se os participantes poderiam identificar tanto defeitos relacionados a requisitos específicos de aplicação quanto problemas associados à funcionalidade representada em cada protótipo ao usar os dois instrumentos de inspeção. Enquanto o UIProtoCheck é um *checklist* genérico para a inspeção funcional de protótipos de interface, o CHIF-E visa oferecer suporte à identificação de defeitos funcionais ao descrever os requisitos de um conjunto de funcionalidades comumente utilizadas em sistemas de informação.

5.4.1.1 Planejamento

Este estudo seguiu as etapas do modelo proposto por Wohlin et al. (2012) no planejamento do experimento controlado. O escopo do estudo experimental foi projetado de acordo com o modelo GQM (Goal/Question/Metric):

> **Analisar** instrumentos de inspeção funcional de protótipos de interface de usuário, CHIF-E e UIProtoCheck, com o **propósito** de avaliar **com relação à** eficácia e eficiência do **ponto de vista** do pesquisador no **contexto** de alunos de graduação em engenharia de software inspecionando protótipos de interface em termos de suas funcionalidades.

O desenho do estudo seguiu o modelo de um fator e dois tratamentos. O fator avaliado foi a identificação de defeitos em protótipos de interface de usuário. Os dois tratamentos avaliados foram os instrumentos de inspeção de protótipos de interface: CHIF-E, o catálogo de heurísticas expandido com as novas heurísticas descritas neste capítulo, e UIProtoCheck.

Os participantes foram selecionados por meio de amostragem não-probabilística por conveniência (Baltes and Ralph, 2022), formada por alunos do quinto semestre do curso de Engenharia de Software matriculados na disciplina de Engenharia de Requisitos e Análise de Sistemas, dos quais muitos já eram atuantes na indústria de software.

Os pesquisadores identificaram a vontade de participar e a experiência profissional em desenvolvimento de software como características dos participantes que poderiam influenciar os resultados do estudo (Rainer and Wohlin, 2022). Portanto, a participação no estudo experimental foi voluntária e formalizada através da assinatura de um Termo de Consentimento detalhado descrevendo as especificidades, riscos e benefícios do experimento.[1]

Além disso, foi realizado um levantamento sobre a experiência em desenvolvimento de software dos 30 participantes. Quinze alunos relataram ter mais de seis meses de experiência em desenvolvimento de software. Essas informações foram

[1]Este estudo está incluído no projeto número 40937120.6.0000.5020, aprovado pelo Comitê de Ética em Pesquisa da UFAM.

utilizadas para distribuir participantes experientes nos dois blocos do experimento controlado e para evitar equipes formadas apenas por membros inexperientes. Os dois blocos do estudo são os de participantes que utilizaram o CHIF-E e o de participantes que utilizaram o UIProtoCheck, com oito participantes experientes no primeiro bloco e sete no segundo. Cada bloco foi então dividido aleatoriamente em cinco times de três participantes, sendo assegurado que cada time tivesse pelo menos um participante experiente.

Os participantes de ambos os blocos receberam uma especificação textual de um site de comércio eletrônico de motocicletas e acessórios, junto com três protótipos de interface que representavam algumas das funcionalidades descritas na especificação: *Busca*, *Página de Produto* e *Carrinho de Compra*. Esses protótipos de interface são os objetos de estudo utilizados pelos participantes durante a inspeção. Cada bloco recebeu seu instrumento de inspeção correspondente — CHIF-E ou UIProtoCheck.

Outros materiais de pesquisa usados neste estudo incluem o Termo de Consentimento, instruções para conduzir a inspeção, uma planilha de inspeção individual e uma planilha para consolidação dos defeitos encontrados pelo time. Os materiais de pesquisa gerados durante este estudo estão disponíveis no material suplementar (https://doi.org/10.6084/m9.figshare.26907004).

A variável independente do estudo foi o instrumento de inspeção (CHIF-E ou UIProtoCheck). As variáveis dependentes analisadas no experimento são a *eficácia*, a *eficiência* e o número de *falso-positivos*. Falso-positivos são problemas relatados que não são realmente defeitos de funcionalidade, tais como os relacionados à interpretação da especificação ou problemas de usabilidade. A eficácia foi determinada pela razão dos defeitos identificados pelo instrumento de inspeção em relação ao número total de erros deliberadamente incorporados nos protótipos. A eficiência foi medida pelo número de defeitos encontrados por hora.

$$Eficácia = \frac{DefeitosEncontrados}{TotalDefeitos}$$

$$Eficiência = 60.\frac{DefeitosEncontrados}{TempoTotal}$$

Para analisar estatisticamente os resultados do estudo, foram formuladas as

seguintes hipóteses:

$\mathbf{H}_{01}$ (Hipótese nula 1): Não há diferença estatística significativa entre a eficácia do CHIF-E e do UIProtoCheck na detecção de defeitos funcionais em protótipos de interface.

$\mathbf{H}_{A1}$ (Hipótese alternativa 1): A eficácia do CHIF-E difere significativamente da do UIProtoCheck na detecção de defeitos funcionais em protótipos de interface do usuário.

$\mathbf{H}_{02}$ (Hipótese nula 2): Não há diferença estatística significativa entre a eficiência do CHIF-E e do UIProtoCheck na detecção de defeitos funcionais em protótipos de interface do usuário.

$\mathbf{H}_{A2}$ (Hipótese alternativa 2): A eficiência do CHIF-E difere significativamente da do UIProtoCheck na detecção de defeitos funcionais em protótipos de interface do usuário.

$\mathbf{H}_{03}$ (Hipótese nula 3): Não há diferença estatística significativa entre o número de falsos positivos encontrados usando CHIF-E e UIProtoCheck durante a inspeção funcional de protótipos de interface de usuário.

$\mathbf{H}_{A3}$ (Hipótese alternativa 3): O número de falsos positivos encontrados com CHIF-E difere significativamente daqueles encontrados usando UIProtoCheck na detecção de defeitos funcionais em protótipos de interface de usuário.

O procedimento de inspeção seguido durante o estudo foi o definido por Sauer et al. (2000), o qual consiste em três estágios: Preparação, Inspeção Individual e Consolidação do Grupo. No estágio de Preparação, os participantes recebem a descrição do problema, os artefatos a serem revisados e o instrumento de inspeção. Em seguida, os participantes são instruídos sobre as fases subsequentes. Na fase de inspeção individual, os participantes analisam cada um dos artefatos em busca de defeitos com base na especificação e anotam os defeitos encontrados em formulários de defeitos. Por fim, na fase de consolidação do grupo, os participantes se reúnem para consolidar os defeitos encontrados pelo grupo, removendo problemas duplicados e reavaliando cada defeito.

Cada protótipo de interface usado no estudo continha quatro defeitos inseridos intencionalmente, totalizando 12 defeitos, listados na Tabela 5.5. Esses defeitos foram inseridos visando avaliar o desempenho dos instrumentos de inspeção na identificação de defeitos funcionais. Seis deles estavam relacionados a inconsistên-

cias com o documento de referência *(descrição do problema)*, enquanto os outros seis estavam mais intimamente associados a características bem conhecidas das funcionalidades representadas nos protótipos *(características gerais)*, sendo mais propensos a serem identificados por meio das heurísticas correspondentes a cada funcionalidade.

A Figura 5.7 ilustra o protótipo *Busca*, um dos três protótipos utilizados no estudo. Os materiais do estudo foram preparados pelo autor da tese e revisados por outros dois pesquisadores, ambos com experiência em estudos qualitativos e educação em Engenharia de Software.

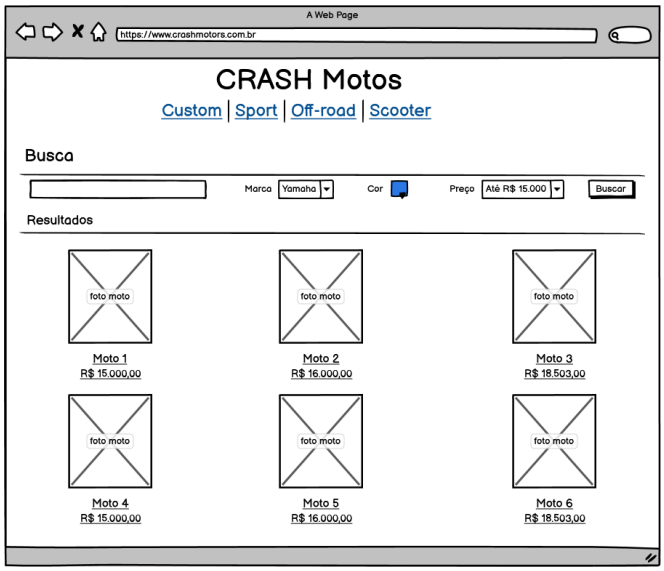


Figura 5.7: Protótipo de IU usado no estudo - Página de Busca.

5.4.1.2 Execução

No dia da primeira fase do estudo, no experimento controlado, compareceram ao estudo 26 participantes, sendo 20 do gênero masculino e 6 do feminino, com idades variando entre 20 e 27 anos. Desses participantes, 14 estavam previamente alocados no bloco CHIF-E (sete com experiência profissional) e 12 no bloco UIProtoCheck (cinco experientes). Os times foram reorganizados de maneira a manter pelo menos um participante experiente em cada time, conforme demonstrado na Tabela 5.6.

Os participantes foram divididos em duas salas separadas, de acordo com o

Tabela 5.5: Lista de defeitos inseridos intencionalmente nos protótipos do experimento controlado.

ID do defeito	Protótipo	Problema	Tipo de erro
D1	Busca	Campo "cor" não está presente na especificação.	descrição do problema
D2	Busca	Campos ausentes: modelo, cc, quilometragem.	descrição do problema
D3	Busca	O termo de pesquisa não tem um rótulo	características gerais
D4	Busca	Não há botão ou ícone de pesquisa.	características gerais
D5	Página de produto	Atributos de ano e cor não estão presentes na especificação.	descrição do problema
D6	Página de produto	A página não mostra o atributo CC.	descrição do problema
D7	Página de produto	Não há descrição do produto.	características gerais
D8	Página de produto	Não há opção para inserir o código postal e calcular o tempo de envio e entrega.	Descrição do problema
D9	Carrinho de compras	Um botão para remover itens do carrinho de compras está ausente.	características gerais
D10	Carrinho de compras	Um campo para alterar a quantidade de itens está ausente.	características gerais
D11	Carrinho de compras	Um botão para finalizar a compra está ausente.	características gerais
D12	Carrinho de compras	Uma lista com produtos relacionados está faltando.	descrição do problema

Tabela 5.6: Experiência dos participantes e times. CT - Time *Checklist*; HT - Time CHIF-E; PC - Participante do bloco *Checklist*; PH - Participante do bloco CHIF-E.

Participante	Experiência profissional	Equipe planejada	Equipe no estudo
PC1*	Não	CT1	–
PC2	2a	CT1	CT1
PC3	Não	CT1	CT1
PC4*	3a	CT2	–
PC5	Não	CT2	CT2
PC6	3a	CT2	CT2
PC7	Não	CT3	CT3
PC8*	2a	CT3	–
PC9	Não	CT3	CT3
PC10	2a	CT4	CT3
PC11	Não	CT4	CT4
PC12	7m	CT4	CT4
PC13	3m	CT5	CT5
PC14	3a	CT5	CT5
PC15	7m	CT5	CT5
PH1	2a	HT1	HT1
PH2	Não	HT1	HT1
PH3	3a	HT1	HT1
PH4	Não	HT2	HT2
PH5	4a	HT2	HT2
PH6	Não	HT2	HT2
PH7	1a	HT3	HT3
PH8	Não	HT3	HT3
PH9*	2a	HT3	–
PH10	Não	HT4	HT3
PH11	1,5a	HT4	HT4
PH12	Não	HT4	HT4
PH13	2a	HT5	HT5
PH14	Não	HT5	HT5
PH15	6m	HT5	HT5

* Participantes que não compareceram ao estudo.

instrumento de inspeção atribuído a eles (CHIF-E ou UIProtoCheck). Em cada sala, os participantes foram instruídos a seguir os estágios definidos do procedimento de inspeção: Preparação, Inspeção Individual e Consolidação do Grupo. Durante o estágio de preparação, os pesquisadores apresentaram a especificação do problema e os protótipos do estudo aos participantes. Em seguida, foi apresentado o respectivo instrumento de inspeção a cada bloco de participantes, CHIF-E ou UIProtocheck. Por fim, o estágio de Preparação foi finalizado com a explicação sobre o procedimento de inspeção adotado e com a apresentação dos formulários para registro dos defeitos durante o estudo.

Durante a Inspeção Individual, os participantes primeiramente realizaram inspeções individuais e registraram os defeitos encontrados em um formulário de defeitos individuais. Em seguida, durante a fase de Consolidação do Grupo, os participantes foram organizados em times de duas ou três pessoas e consolidaram a lista de defeitos encontrados, com a remoção de defeitos em duplicidade. Eles então avaliaram cada um dos problemas identificados para chegar a um consenso e os classificaram como Defeito (D) ou Falso-Positivo (FP). Ao final da execução da primeira fase do estudo, foram obtidos 10 relatórios de inspeção, sendo cinco de uso do CHIF-E e cinco do UIProtoCheck, que foram utilizados na avaliação dos resultados do experimento controlado (Seção 5.5.1).

5.4.2 Estudo Experimental: Inspeção de Projetos de Interface Utilizando o CHIF-E

Além de avaliar a aplicabilidade do CHIF-E para identificar defeitos associados à funcionalidade em protótipos de interface de usuário, foi necessário investigar mais detalhadamente a natureza dos defeitos que essa ferramenta tem potencial para identificar. Para tanto, foi definida a segunda questão de pesquisa:

> **QP2:** Quais tipos de defeitos podem ser encontrados durante inspeções funcionais de protótipos de interface de usuário usando o CHIF-E?

Para responder a essa questão de pesquisa, foi formulada a segunda fase do estudo, que consiste em usar o CHIF-E durante a inspeção funcional de protótipos de interface de dez projetos de software.

5.4.2.1 Planejamento

Os participantes da segunda fase do estudo foram, em sua maioria, os mesmos da primeira fase, compreendendo 34 alunos do quinto semestre do curso de graduação em Engenharia de Software na disciplina Engenharia de Requisitos e Análise de Sistemas. Ao longo do semestre, os participantes se auto-organizaram em dez equipes: sete equipes com quatro membros e três com três membros. Cada equipe projetou um aplicativo de software especificado por uma das outras equipes. Durante o projeto das aplicações, as equipes usaram técnicas de *design thinking* (Souza et al., 2020) para elicitar requisitos e elaborar os artefatos do projeto de software. Esses artefatos incluíram um documento de descrição do problema, a especificação de requisitos, diagramas UML (casos de uso, classe, sequência, estado, atividade, implantação) e protótipos da interface de usuário.

Utilizando os projetos elaborados pelos participantes, a segunda parte do estudo consistiu em uma atividade opcional proposta aos alunos. Nesta atividade, eles inspecionaram protótipos de interface do projeto de outra equipe usando CHIF-E e a descrição do problema. Os participantes que concordaram em participar do estudo e permitiram acesso aos dados gerados durante a atividade, incluindo os protótipos de interface elaborados, formalizaram sua decisão assinando um Formulário de Consentimento, o qual detalha todos os aspectos do estudo, incluindo riscos e benefícios.

As equipes foram instruídas pelo professor titular da disciplina sobre como utilizar o CHIF-E. Esta etapa foi necessária para nivelar o conhecimento dos participantes ao ensinar os participantes que ainda não haviam tido contato com o CHIF-E, por terem participado do experimento controlado no grupo que utilizou o UIProtoCheck como instrumento de inspeção.

Os participantes foram instruídos a seguir o mesmo procedimento de inspeção adotado durante o experimento controlado (Sauer et al., 2000), que consiste em três etapas: Preparação, Inspeção Individual e Consolidação do Grupo. Cada equipe foi instruída a preparar um relatório de inspeção utilizando um modelo fornecido previamente, incluindo defeitos encontrados individualmente por cada membro da equipe e a consolidação desses defeitos, classificados como *Defeito* ou *Falso-positivo*. Além disso, as equipes foram solicitadas a escrever um relatório

de experiência no qual 1) descrevessem as principais dificuldades encontradas, 2) sugerissem modificações nas heurísticas existentes e 3) recomendassem novas heurísticas para complementar as funcionalidades do catálogo.

5.4.2.2 Execução

Após o experimento controlado, os participantes que concordaram em participar da pesquisa foram designados para a segunda fase do estudo. Nessa fase, as equipes de inspeção seguiram a divisão em equipes já formadas ao longo do curso para executar tarefas relacionadas à Engenharia de Requisitos. Os participantes receberam orientações sobre como usar o CHIF-E para inspecionar protótipos de interface. Em seguida, cada equipe recebeu os protótipos de interface do projeto desenvolvido por outra equipe ao longo do curso, juntamente com a especificação de requisitos da respectiva aplicação. A segunda fase do estudo obteve oito relatórios de inspeção e respectivos relatos de experiência.

Nesta fase foram incluídos apenas dados produzidos por equipes que concordaram em fornecer seu relatório de inspeção para a pesquisa por meio do Termo de Consentimento Livre e Esclarecido. O professor da disciplina atribuiu às equipes de alunos a tarefa de revisar os protótipos elaborados por uma das outras equipes. A Tabela 5.7 lista as equipes de desenvolvedores e revisores que participaram do estudo.

5.5 Resultados

Nesta seção são apresentados os principais resultados do estudo sobre a avaliação de instrumentos de inspeção na detecção de defeitos funcionais em protótipos de interface de usuário. A seção 5.5.1 responde à QP1 apresentando como o CHIF-E suporta inspeções funcionais de interfaces de usuário, enquanto a seção 5.5.2 apresenta os tipos de defeitos encontrados durante inspeções funcionais usando CHIF-E (QP2).

Tabela 5.7: Equipes de revisores e desenvolvedores e temas de projeto.

Equipe de Revisão	Equipe de Desenvolvedores	Qtd. Participantes	Tema do projeto
RT3	1	4	Loja online sem estoque com gerenciamento de pedidos de clientes e fornecedores
RT4	2	4	Sistema de gerenciamento financeiro colaborativo de dívidas e metas
RT5	3	2	Uma plataforma para rastrear pedidos onde as empresas monitoram o envio de produtos e os clientes rastreiam seus pedidos
RT6	4	3	Site para anunciar e vender produtos para festas e aluguel de espaços.
RT7	5	3	Gestão de restaurante com controle de pedidos locais e para entrega
RT8	6	2	Aplicativo móvel que conecta pessoas que descartam e que utilizam materiais recicláveis
RT9	7	4	Aplicativo para facilitar a comunicação de pessoas com deficiência por meio da troca de mensagens e tradução entre vídeo (língua de sinais), voz e texto
RT2	10	3	Gestão de empresa comercial e suas filiais

5.5.1 QP1: Suporte do CHIF-E e do UIProtoCheck à inspeção funcional de protótipos de Interface de Usuário

Os resultados do experimento controlado foram avaliados por meio de análise estatística e qualitativa. Os defeitos identificados foram catalogados e organizados por equipe. Durante essa análise, foram anotados o participante, protótipo, funcionalidade, o problema identificado, a heurística aplicada e, caso existam, os códigos dos defeitos equivalentes na lista de defeitos pré-inseridos (Tabela 5.5). Após esta análise inicial, os defeitos duplicados foram removidos, sendo determinados o número final de defeitos, a quantidade de falso-positivos, a eficácia e a eficiência de cada equipe. Os resultados da análise estatística são exibidos nas Tabelas 5.8 e 5.9.

Tabela 5.8: Eficácia, eficiência e falso-positivos - CHIF-E.

Time	Defeitos	Falso-Positivos	Tempo(min)	Eficiência	Eficácia(%)
HT1	7	6	86	4.88	58
HT2	8	2	85	5.65	67
HT3	7	3	88	4.77	58
HT4	8	3	97	4.95	67
HT5	8	5	82	5.85	67

Em média, o grupo que utilizou CHIF-E obteve maior eficácia e eficiência (63% e 5,22 defeitos/hora) em comparação ao grupo UIProtoCheck (58% e 4,93 defeitos/hora). Para avaliar se as diferenças entre os resultados obtidos pelos dois instrumentos de inspeção nas variáveis dependentes foram estatisticamente

Tabela 5.9: Eficácia, eficiência, e falso-positivos - UIProtoCheck.

Time	Defeitos	Falso-Positivos	Tempo(min)	Eficiência	Eficácia(%)
CT1	8	5	82	5.85	67
CT2	5	10	88	3.41	42
CT3	7	6	88	4.77	58
CT4	7	9	96	4.38	58
CT5	8	7	77	6.23	67

significativas, primeiramente, foi realizado o teste de normalidade de Shapiro-Wilk para determinar a distribuição dos dados e escolher o teste estatístico apropriado.

O teste de normalidade de Shapiro-Wilk apresentou valor abaixo de 0,05 para a variável Eficácia do grupo CHIF-E, conforme indicado na Tabela 5.10, o que sugere que um teste estatístico não-paramétrico deve ser utilizado. Desta forma, foi realizado o teste estatístico de Mann-Whitney, com resultados apresentados na Tabela 5.11. Estes resultados não mostraram diferença significativa na eficácia e eficiência dos dois grupos (valor de p maior que 0,05), o que significa que as hipóteses nulas H_{01} e H_{02} não podem ser refutadas. Em relação aos falso-positivos, o grupo UIProtoCheck apresentou um número significativamente maior do que o grupo CHIF-E, confirmando a hipótese H_{A3}.

Tabela 5.10: Teste de normalidade Shapiro-Wilk.

	Eficiência		Eficácia		False-positives	
	CHIF-E	UIProto Check	CHIF-E	UIProto Check	CHIF-E	UIProto Check
Mediana	4.950	4.770	67.000	58.000	3.000	7.000
Média	5.220	4.928	63.400	58.400	3.800	7.400
Desvio padrão	0.493	1.137	4.930	10.213	1.643	2.074
Shapiro-Wilk	0.839	0.960	0.684	0.846	0.914	0.952
Valor de p	0.163	0.809	0.006	0.183	0.490	0.754

Tabela 5.11: Teste estatístico não-paramétrico Mann-Whitney.

	W	p
Eficiência	15.000	0.674
Eficácia	16.000	0.488
Falso-positivos	2.000	0.035

Falso-positivos referem-se a problemas relatados por participantes que não estavam entre aqueles intencionalmente incluídos nos protótipos do estudo. Os defeitos identificados como falso-positivos foram avaliados através de análise temática reflexiva (Braun and Clarke, 2006) e categorizados em seis temas, detalhados na Tabela 5.12. Esses falso-positivos podem resultar de problemas de compreensão da especificação ou dos protótipos, como os tipos FP1, FP2, FP5 e FP6, ou

Tabela 5.12: Falso-positivos por tipo de instrumento de inspeção.

ID	Descrição	CHIF-E	UIProtoCheck
FP1	Funcionalidade incorretamente identificada	3	10
FP2	Funcionalidade não representada nos protótipos do estudo	6	10
FP3	Problema de usabilidade	3	10
FP4	Comportamento presumido que não estava especificado	2	5
FP5	Texto ilustrativo interpretado como dado real	3	2
FP6	Não se aplica, de acordo com a heurística	2	0
Total		19	37

podem apontar diferentes tipos de defeitos, como os tipos FP3 e FP4, que são indicativos de problemas de usabilidade ou especificação. O grupo UIProtoCheck relatou mais frequentemente falso-positivos relacionados à identificação incorreta das funcionalidades descritas nos protótipos ou a defeitos de usabilidade. Além disso, foram relatados por ambos os grupos vários falso-positivos relacionados a funcionalidades descritas na especificação, mas que não estavam representadas nos protótipos do estudo.

Em relação à natureza dos defeitos relatados pelos dois grupos, os pesquisadores calcularam a frequência de cada defeito identificado pelas equipes usando os dois instrumentos de inspeção. A Tabela 5.13 exibe esses dados, categorizados pela natureza dos defeitos: os seis primeiros estão relacionados aos detalhes da especificação do problema usada no estudo, enquanto os seis últimos se relacionam às características gerais das funcionalidades especificadas no CHIF-E. A Tabela revela que o grupo que empregou o UIProtoCheck descobriu todos os defeitos associados à descrição do problema, mas falhou em reconhecer dois defeitos vinculados ao domínio do problema do estudo (comércio eletrônico). Por outro lado, o grupo que utilizou o CHIF-E identificou todos os defeitos relativos às funcionalidades gerais do domínio do problema, mas ignorou um defeito associado à especificação.

A Figura 5.8 apresenta a natureza dos defeitos identificados por cada instru-

Tabela 5.13: Número de defeitos encontrados por natureza e instrumento de inspeção.

Defeito	Natureza	CHIF-E	UIProtoCheck
D1	Especificação do problema	1	3
D2	Especificação do problema	9	10
D5	Especificação do problema	0	6
D6	Especificação do problema	1	7
D8	Especificação do problema	8	4
D12	Especificação do problema	7	8
D3	Funcionalidades gerais	3	3
D4	Funcionalidades gerais	8	1
D7	Funcionalidades gerais	2	1
D9	Funcionalidades gerais	10	0
D10	Funcionalidades gerais	4	0
D11	Funcionalidades gerais	12	7
	Total	65	50

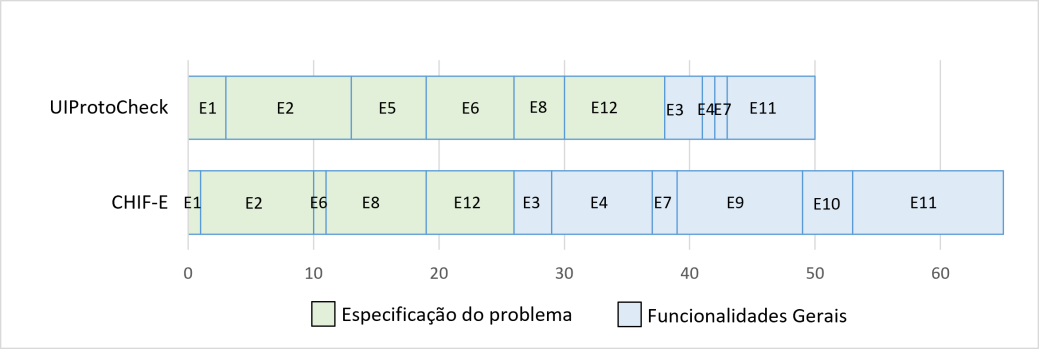


Figura 5.8: Natureza dos defeitos encontrados por instrumento de inspeção – CHIF-E e UIProtoCheck.

mento de inspeção. Ela mostra que os participantes que usaram o UIProtoCheck detectaram mais defeitos relacionados à especificação do problema. Em contraste, aqueles que usaram o CHIF-E identificaram mais defeitos vinculados às características gerais das funcionalidades, independentemente da especificação. Portanto, em relação a como o CHIF-E suporta a inspeção funcional de protótipos de interface do usuário (QP1), embora o CHIF-E seja mais adequado para encontrar defeitos relacionados às características gerais das funcionalidades, foi verificado

que os participantes também conseguiram encontrar quase todos os defeitos relacionados à especificação do problema. A Tabela 5.14 apresenta um comparativo dos aspectos principais dos dois instrumentos de inspeção funcional.

Tabela 5.14: Quadro comparativo dos principais aspectos - CHIF-E e UIProtoCheck.

Aspecto	CHIF-E	UIProtoCheck
Defeitos por funcionalidade	Mais eficaz	Menos eficaz
Defeitos por especificação	Menos eficaz	Mais eficaz
Falso-positivos	Menor incidência	Maior incidência
Adequação ao contexto genérico	Alta	Alta
Necessidade de documento de referência	Menor	Maior

5.5.2 QP2: Tipos de defeitos encontrados durante a inspeção funcional de protótipos de interface usando o CHIF-E

Os resultados da segunda fase foram analisados qualitativamente para investigar os diferentes tipos de defeitos que os participantes encontraram durante a inspeção dos protótipos. Os defeitos relatados pelas equipes foram analisados juntamente com suas respostas ao relatório de experiência. Os problemas foram categorizados através de análise temática (Braun and Clarke, 2006) para definir temas que representassem com precisão a natureza dos defeitos relatados.

Os defeitos relatados foram examinados para determinar se estavam conectados à funcionalidade descrita nas especificações ou aos aspectos de design e usabilidade do protótipo. Além disso, a análise avaliou a eficácia das heurísticas na detecção de defeitos, a adequação da heurística aplicada durante a inspeção ao protótipo e se os problemas descobertos poderiam aprimorar as heurísticas do CHIF-E. Os temas e seus significados foram revisados e atualizados até que representassem adequadamente os defeitos relatados.

Cinco temas que caracterizam os tipos de defeitos emergiram da análise temática: DF - defeito funcional associado ao CHIF-E, IE - inconsistência com a especificação de software, CP - comportamento presumido, US - usabilidade e FP -

falso-positivos. Os temas emergidos da análise temática resultaram no esquema de codificação apresentado na Tabela 5.15, sendo utilizados para codificar sistematicamente os defeitos relatados nas inspeções. Defeitos de especificação de software são aqueles diretamente associados às informações contidas na especificação do problema. Em contraste, defeitos de comportamento presumido são aqueles identificados pelos participantes como essenciais para as funcionalidades sob inspeção, mas que não estão listados como necessários nos *checklists* das heurísticas. Esta categoria de defeito geralmente indica melhorias que podem ser acrescentadas às heurísticas como requisitos opcionais. Por exemplo, o defeito identificado como 'opção de criação de conta' na funcionalidade *Autenticação* pode ser adicionado no seu respectivo *checklist* como item opcional.

A Tabela 5.16 exibe o número total de defeitos relatados, categorizados por tema. O tema de defeitos mais relatados foi *defeito funcional (DF)*, com 14 defeitos, seguido de perto por defeitos de *comportamento presumido (CP)*, com 13. Apenas três dos defeitos relatados estavam relacionados a recursos específicos da aplicação.

As equipes utilizaram 16 das 26 heurísticas para identificar defeitos nos protótipos de interface. As heurísticas mais utilizadas pelos times incluíram Pesquisa, Registro, Lista, Visualização, Galeria, Carrinho de Compras e Registro de Conta, cada uma duas vezes. As heurísticas não utilizadas foram Câmera, Editor de Imagens, Mapas, Media Player, Menu, Mestre-Detalhe, Notícias, Quiz, Tutorial e Upload de Arquivo. Em relação aos problemas de usabilidade relatados, os problemas mais identificados incluíram informações insuficientes sobre o conteúdo da tela, elementos truncados, detalhes ambíguos, padronização inconsistente de botões e ícones e excesso de informações na interface. Embora não sejam o foco da inspeção funcional, essas descobertas destacam as áreas críticas a serem aprimoradas para melhorar a experiência do usuário.

No relatório de experiência realizado após as atividades de inspeção, o principal desafio identificado pelos participantes foi o de selecionar a heurística mais adequada para avaliar um protótipo. A equipe 8, por exemplo, aplicou heurísticas que não eram adequadas para os protótipos que inspecionou. Já a equipe 6 observou que várias heurísticas poderiam ser aplicadas para inspecionar um único protótipo.

Tabela 5.15: Temas resultantes da análise temática.

Cod	Categoria	Definição	Critérios para Codificação	Exemplo típico
DF	Defeito Funcional	Falta ou erro relacionado a uma funcionalidade descrita no catálogo (CHIF-E)	Ausência de elemento essencial da heurística, comportamento esperado não representado.	Tela de carrinho sem botão de "remover item"; Filtro de busca sem campo de texto.
IE	Inconsistência com a especificação	Incompatibilidade com a especificação textual	Divergência entre o que está no protótipo e o que está descrito no documento de requisitos.	Campo "cor" presente no protótipo, mas ausente na especificação.
CP	Comportamento presumido	Ausência de elemento esperado, mesmo sem estar descrito como obrigatório	Requer conhecimento tácito do domínio; comportamento considerado "esperado" mesmo que não esteja no catálogo.	Tela de login sem opção de "recuperar senha".
US	Usabilidade	Problemas de uso, legibilidade, layout, ambiguidade	Dificuldade de interpretação, excesso de informações, interface truncada.	Botões não padronizados; layout confuso.
FP	Falso Positivo	Elemento marcado como defeito, mas que não se enquadra nos critérios acima	Interpretação errada; expectativa do avaliador que não se sustenta pelas heurísticas ou pela especificação.	Criticar um botão ausente quando ele não é esperado; confundir texto ilustrativo com real.

Tabela 5.16: Defeitos relatados por time e categoria.

Time	DF	IE	CP	US	FP	Total
RT3	0	2	3	1	4	10
RT4	2	1	0	0	0	3
RT5	0	0	0	0	0	0
RT6	1	0	1	3	1	6
RT7	3	0	1	2	0	6
RT8	1	0	2	3	1	7
RT9	7	0	4	0	0	11
RT2	0	0	2	0	3	5
Total	14	3	13	9	9	48

Equipe de Revisores 4 - *"Tivemos dificuldade em encontrar heurísticas adequadas devido às diferentes maneiras como as páginas podem ser construídas, mesmo seguindo as heurísticas."*

Equipe de Revisores 6 - *"Algumas telas podiam usar mais de uma heurística, pois misturavam funcionalidades. No entanto, para prosseguir, decidiu-se usar a heurística que melhor se ajustasse à funcionalidade encontrada na tela."*

Além disso, os participantes relataram que poucos erros eram identificáveis por meio das heurísticas. Eles também indicaram dificuldade em diferenciar os termos *funcionalidade* e *heurística* ao preencher a planilha.

Equipe de Revisores 3 - *"Os protótipos tiveram poucos erros identificáveis por heurísticas, então os membros da equipe apontaram praticamente os mesmos erros. (...) Algumas telas não se encaixavam muito bem nem com as heurísticas das funcionalidades principais nem com funcionalidades específicas."*

Equipe de revisores 9 - *"Tivemos dificuldade em entender o que é a heurística em si. Exemplo: "Galeria" está em "Funcionalidades Básicas", mas isso faz de "Galeria" uma heurística ou uma funcionalidade? Não está bem definido o que cada coisa é."*

Em relação ao aprimoramento do CHIF-E, as equipes propuseram as seguintes melhorias:

- Adicionar instruções ao site do CHIF-E indicando que várias heurísticas podem ser empregadas para avaliar um protótipo (RT 6).
- Apresentar uma seção no CHIF-E explicando as categorias heurísticas Básica e Específica (RT 5).
- Revisar o CHIF-E para incluir um item nos *checklists* das heurísticas para que os protótipos tenham títulos descritivos refletindo suas funcionalidades, quando aplicável (RT 3).
- Incluir informações de data e hora como um item obrigatório no *checklist* heurística *Bate-papo* (RT 6 e 9).
- Incluir 'Formulário' como um sinônimo nas heurísticas de *Cadastro* e *Alteração* (RT 5).
- Expandir o *checklist* da heurística *Autenticação* para incluir os itens opcionais *link para criação de conta* (RT 3), *autenticação por meio de outras plataformas*, como mídia social (RT 8) e *link para recuperação de senha* (RT 9).
- Melhorar os *checklists* das heurísticas *Perfil de Usuário* e *Home* com um item opcional indicando uma opção de logout para aplicativos que usam usuários autenticados (RT 8).

As equipes também propuseram novas heurísticas para as funcionalidades *Painel* (RT 3), *Avaliação* (RT 5), *Publicação de Conteúdo* (RT 8) e *Chamada de Áudio* (RT 9).

> **Equipe de Revisores 3** - *"Uma heurística de Dashboard ajudaria a criar protótipos que exibem dados e gráficos com informações relevantes da melhor maneira possível para o usuário."*

> **Equipe de Revisores 5** - *"Uma heurística que define métricas para inspeção da funcionalidade de Avaliação pode ser uma boa adição, já que o projeto inspecionado apresentou um recurso para avaliar perfis de empresas. O mesmo recurso ocorreu em nosso projeto, onde desenvolvemos um recurso para avaliar itens em um menu."*

A funcionalidade *Avaliação* poderia auxiliar na criação e inspeção de protótipos de interface que avaliam serviços, produtos ou até mesmo o próprio aplicativo. A especificação da funcionalidade *Publicação de Conteúdo* auxiliaria a criação de interfaces para publicação em plataformas de publicidade ou mídias sociais. Já uma heurística para a funcionalidade *Chamada de Áudio* descreveria recursos essenciais neste tipo de protótipo, tais como botões para iniciar ou atender chamadas, silenciar e ativar o microfone ou encerrar a chamada.

Finalmente, foram identificados defeitos em protótipos de funcionalidades elementares que não possuíam heurísticas no CHIF-E. O RT7 relatou um defeito em um protótipo do tipo *Notificação*. Esta funcionalidade, a princípio, não deveria exigir ações do usuário; no entanto, o protótipo incluía um botão de confirmação. Embora essa seja uma funcionalidade elementar, expansões futuras do CHIF-E podem incluir heurísticas para caracterizar e diferenciar estes tipos de funcionalidades, tais como *Notificação* ou *Modal*, onde o primeiro conteria apenas informações de status e um botão para fechar a mensagem, enquanto o último ofereceria opções variadas de ação, tais como *Confirmar* e *Cancelar*.

As melhorias sugeridas pelos participantes foram utilizadas para melhorar as heurísticas do CHIF-E. Para as novas funcionalidades sugeridas pelos participantes, novas heurísticas poderão ser criadas utilizando-se o processo apresentado na Seção 5.2 e um *dataset* de protótipos de interface com instâncias das novas funcionalidades.

5.6 Ameaças à Validade

O documento de referência utilizado na primeira fase do estudo foi identificado como uma ameaça à validade externa. Este documento consistiu em uma especificação simplificada de uma aplicação de comércio eletrônico. Além disso, os

protótipos do estudo incluíram apenas três das funcionalidades descritas. A seleção de uma descrição simplificada do problema teve como objetivo garantir a viabilidade do estudo, pois um conjunto mais extenso de protótipos poderia estender significativamente sua duração, afetando potencialmente a validade interna devido a fatores como fadiga ou indisposição dos participantes. Para mitigar esta ameaça, a segunda fase do estudo envolveu a inspeção de protótipos de projetos de software mais complexos (Tabela 5.7). Cada time examinou um projeto diferente nesta etapa e recebeu mais tempo para concluir as atividades de inspeção.

O nível de experiência dos participantes também foi identificado como uma ameaça à validade da conclusão, uma vez que participantes inexperientes podem produzir resultados que diferem significativamente daqueles de profissionais da indústria de software (Falessi et al., 2018). Consequentemente, equipes formadas apenas por membros experientes ou inexperientes poderiam apresentar resultados inconsistentes. Para abordar essa questão, foi realizado um levantamento preliminar da experiência dos participantes em desenvolvimento e inspeção de software. A partir dos resultados deste levantamento, as equipes foram formadas de maneira a garantir que cada uma incluísse pelo menos um membro experiente. Além disso, foi adotado neste estudo um método de inspeção em equipe, o qual envolve a revisão e consolidação de defeitos identificados individualmente. Essa abordagem tende a tornar mais abrangente a detecção de defeitos e a diminuir a incidência de falsos positivos, aproximando os resultados do estudo àqueles obtidos na prática real de engenharia de software.

Na análise temática, a classificação de falso-positivos no estágio inicial e a categorização de defeitos no estágio subsequente podem ameaçar a validade de construto, devido à sua dependência do julgamento subjetivo do avaliador principal. Para reduzir esse risco, dois outros avaliadores revisaram a codificação. Esse processo de revisão envolveu uma discussão completa e ajuste subsequente de falso-positivos e categorias de defeitos.

Em relação à generalização das descobertas, as evidências deste estudo são específicas para o campo da educação em Engenharia de Requisitos e Verificação e não podem ser extrapoladas para outras áreas. No entanto, este estudo fornece uma referência para a realização de novos estudos em diferentes contextos que visem avaliar a eficácia de ferramentas de inspeção de protótipos de interface,

particularmente aquelas com foco na funcionalidade.

5.7 Considerações Finais

Este ciclo de design consistiu em investigar como o conjunto de heurísticas de funcionalidades recorrentes, desenvolvido no primeiro ciclo de design (Capítulo 4), poderia dar suporte à inspeção de protótipos de interface de usuário com foco em sua funcionalidade. Para isso, o conjunto de heurísticas foi expandido com nove novas funcionalidades obtidas a partir do *dataset* rotulado EnRICO de capturas de tela de aplicações (Leiva et al., 2020), resultando no Catálogo de Heurística de Inspeção Funcional Expandido (CHIF-E). A inclusão de novas funcionalidades ao conjunto de heurísticas adaptou o processo de anotação de *datasets* proposto por Klie et al. (2024) e o método de especificação de heurísticas de usabilidade de Quiñones et al. (2018) para demonstrar como o CHIF-E pode ser incrementalmente ampliado com funcionalidades de diferentes domínios, tornando-o cada vez mais abrangente.

Em relação à inclusão de novas heurísticas no CHIF-E, iterações futuras podem replicar o processo apresentado na Seção 5.2 com protótipos de interface ou capturas de tela que representem novas funcionalidades, de maneira a fornecer subsídios para a elicitação das suas características durante o processo de expansão. Para fins de organização, conforme novas funcionalidades forem sendo adicionadas ao CHIF-E, o catálogo poderá ser segmentado em seções de acordo com o domínio de aplicação a que as heurísticas se aplicam.

O CHIF-E foi avaliado em um estudo de duas fases. A primeira fase consistiu em um experimento controlado que comparou o desempenho do CHIF-E com o UIProtoCheck, um *checklist* genérico de inspeção funcional, na detecção de defeitos funcionais em protótipos de interface. A segunda fase compreendeu um estudo experimental de inspeção funcional de protótipos de interface de oito projetos de software usando o CHIF-E. Os resultados do estudo de avaliação mostraram que tanto o CHIF-E quanto o UIProtoCheck foram efetivos na detecção de defeitos funcionais em protótipos de interface de usuário. A avaliação estatística do experimento controlado não mostrou diferenças significativas na eficácia e na eficiência. No entanto, a análise qualitativa da natureza dos defeitos encontrados por cada

instrumento de inspeção indicou que ambos são complementares. Enquanto a inspeção com CHIF-E tende a encontrar defeitos relacionados às características genéricas da funcionalidade representada, o UIProtoCheck considera as peculiaridades da aplicação definidas no documento de referência, tendendo a encontrar mais defeitos de inconsistência entre a interface do protótipo e a especificação da aplicação inspecionada.

Em relação ao UIProtoCheck, os resultados o mostraram como uma alternativa simples para realizar inspeções funcionais rápidas em versões iniciais de protótipos de interface, em comparação com abordagens de inspeção existentes que usam perguntas para orientar as inspeções de interfaces de usuário. Entre essas abordagens, podemos citar o *Cognitive Walkthrough* (Spencer, 2000), o *Heuristic Walkthrough* (Sears, 1997) e a Inspeção Semiótica (De Souza et al., 2006). Tais métodos tendem a ser mais demorados porque exigem que especialistas realizem revisões detalhadas. Além disso, são realizados diretamente no software, envolvendo a análise de recursos por meio da navegação.

O estudo de Islam et al. (2020), que avaliou ferramentas para detecção de problemas de usabilidade em interfaces de sites, também concluiu que cada ferramenta consegue identificar tipos de problemas diferentes. O estudo comparou o *framework* SIDE, baseado na Avaliação Semiótica (De Souza et al., 2006), com o método de Avaliação Heurística (Nielsen, 1994a). Seus resultados mostraram que ambas as abordagens são complementares quanto à detecção de problemas de usabilidade, sendo o SIDE mais adequado para descobrir defeitos relacionados à intuitividade de sinais de interface, como links, imagens, botões ou ícones.

Na primeira fase do estudo também foi verificada uma ocorrência significativamente menor de falso-positivos em inspeções usando CHIF-E em comparação com UIProtoCheck. Essa diferença pode ser atribuída à ênfase dada pelo CHIF-E aos requisitos de cada funcionalidade. Desta forma, os participantes orientaram-se durante a inspeção pelo *checklist* de cada heurística, o que os levou a ignorar outras classes de defeitos. Em relação aos tipos de falso-positivos relatados, destacam-se problemas de usabilidade e violações de regras de design.

Especificamente em relação a defeitos de usabilidade, estes tendem a ser confundidos com defeitos funcionais durante inspeções de interface de usuário. No entanto, vale notar que problemas de usabilidade geralmente são detectados

durante inspeções de artefatos mais completos, onde detalhes de interação já foram pensados, enquanto os instrumentos de inspeção funcional apresentados neste trabalho têm como objetivo inspecionar protótipos iniciais de IU quando os requisitos ainda estão sendo mapeados para possíveis soluções. A detecção de defeitos de usabilidade durante inspeções funcionais, no entanto, não indica um problema; ao contrário, ela ajuda a antecipar a identificação de defeitos potenciais antes que eles apareçam em protótipos de alta fidelidade ou durante a implementação da interface, ou mesmo a identificar problemas na especificação de requisitos.

Os demais tipos de falso-positivos analisados estão relacionados às interpretações dos participantes sobre o comportamento e o conteúdo de protótipos não especificados no documento de referência. Esses falso-positivos indicam o quanto o desempenho dos instrumentos de inspeção na detecção de defeitos em protótipos de interface de usuário depende da qualidade dos documentos de referência utilizados durante a inspeção. Os mesmos poderiam ser mitigados com informações mais detalhadas aos participantes, tais como uma lista de funcionalidades contidas na especificação do problema, ou ainda, poderia ser enfatizado aos participantes que apenas algumas das funcionalidades listadas na especificação seriam representadas nos protótipos.

Ambos os instrumentos de inspeção funcional (CHIF-E e UIProtoCheck) mostraram-se ferramentas eficazes para auxiliar inspeções rápidas com foco na detecção de defeitos relacionados à funcionalidade. É importante ressaltar que esse tipo de inspeção não demanda profissionais da área de qualidade ou etapas de preparação extensas. Essa adaptabilidade é especialmente crucial em contextos de *lean startups*, onde equipes de garantia da qualidade dedicadas são raras e o tempo de lançamento no mercado é curto (Melegati et al., 2020). Nesses cenários, entregas rápidas são essenciais, mas os produtos ainda devem atender a um padrão mínimo de qualidade para evitar a sua rejeição pelo mercado.

O *framework* MaUI (Cheng, 2016) também foi desenvolvido para o contexto de *lean startups*, especificamente para oferecer suporte a inspeções de usabilidade de aplicativos móveis com base na Avaliação Heurística (Nielsen, 1994b). Ele apresenta um conjunto de etapas para preparar a sistematização da aplicação das heurísticas de Nielsen. Os resultados do estudo de avaliação do *framework*

demonstraram alta confiabilidade na detecção de defeitos de usabilidade. No entanto, houve variações notáveis nos resultados da avaliação entre diferentes inspetores. Os autores também observaram que a estrutura do *framework* pode ser excessivamente complexa para que seja adotada por profissionais de *lean startups*.

Na segunda fase do nosso estudo, na qual os times usaram o CHIF-E para inspecionar protótipos de interface de oito projetos de software, os participantes identificaram algumas características que os protótipos deveriam apresentar, aproveitando seu conhecimento pessoal da funcionalidade representada. No entanto, esses requisitos não estavam presentes nem nas heurísticas nem nas especificações. Essas características foram aproveitadas para revisar e complementar a definição das heurísticas do CHIF-E.

Os participantes também relataram dificuldades para identificar a funcionalidade representada em cada protótipo e escolher a heurística mais apropriada para inspeção (Seção 5.5.2). Esse desafio surgiu porque alguns protótipos podem representar múltiplas funcionalidades. Consequentemente, há uma necessidade de auxiliar a inspeção de protótipos por meio da recomendação automatizada de heurísticas de catálogo, a exemplo da abordagem adotada pela ferramenta *UIS-Hunter* desenvolvida por Yang et al. (2021). A ferramenta *UIS-Hunter* detecta automaticamente violações de diretrizes de design. Em seu estudo, os autores pesquisaram 93 diretrizes *don't-design*, que mapeiam violações de design a partir dos valores das propriedades de 17 tipos de componentes da GUI API do sistema Android.

No contexto desta pesquisa, a automatização da detecção de defeitos funcionais deve levar em consideração cada uma das heurísticas das funcionalidades descritas no catálogo, além de informações específicas de entidades do domínio da aplicação inspecionada, das quais algumas heurísticas são dependentes. Considerando a dificuldade relatada na seleção de heurísticas e visando dar um passo no sentido de endereçar a necessidade de automação do processo de inspeção, o próximo capítulo apresenta o desenvolvimento de uma ferramenta de apoio à inspeção funcional, a qual contempla os recursos de recomendação de heurísticas, registro de defeitos e geração de relatórios de inspeção.

Capítulo 6

GUIPFI – Uma Ferramenta para Inspeção Funcional de Protótipos de Interface

Na segunda iteração do Ciclo de Design, os resultados mostraram que os instrumentos de inspeção desenvolvidos, UIProtoCheck e CHIF-E, dão suporte à inspeção funcional de protótipos de interface de maneira complementar. Enquanto as heurísticas do CHIF-E descrevem requisitos de funcionalidades específicas, o UIProtoCheck apoia a inspeção de aspectos gerais de um protótipo de interface, tais como informações exibidas, elementos de entrada de dados e de interação.

Além disso, os resultados indicaram que os participantes apresentaram dificuldades em selecionar as heurísticas mais adequadas para inspecionar os protótipos do estudo. Esse problema tende a intensificar-se à medida que o catálogo for expandido, uma vez que o aumento do número de heurísticas, decorrente da inclusão de novas funcionalidades, amplia o esforço cognitivo necessário para identificar quais delas são relevantes em cada contexto de inspeção. Com um conjunto mais extenso, a sobreposição entre heurísticas e a maior complexidade na categorização funcional podem tornar o processo de seleção menos intuitivo, exigindo mecanismos adicionais de apoio à escolha.

Este cenário indicou a necessidade de desenvolvimento de um artefato que auxilie a realização de inspeções funcionais, dando suporte à seleção e ao uso dos

instrumentos de inspeção funcional desenvolvidos durante esta pesquisa. Desta forma, foi formulada a seguinte questão de pesquisa:

> **QP:** Como apoiar a seleção e o uso de instrumentos de inspeção durante inspeções funcionais de protótipos de interface?

Neste capítulo, é apresentada a terceira iteração do Ciclo de Design, que consistiu no desenvolvimento de um artefato de software que integra o uso dos instrumentos de inspeção funcional, GUIPFI - *Guided User Interface Prototype Functional Inspection*. Este artefato oferece suporte à inspeção de protótipos de interface de usuário através de um recurso de recomendação de instrumentos de inspeção e de um ambiente para a realização de inspeções com o uso dos instrumentos selecionados. As próximas seções descrevem o desenvolvimento da GUIPFI e sua avaliação no ambiente da academia.

6.1 Design da Ferramenta GUIPFI

A GUIPFI é um artefato de software que foi desenvolvido com o objetivo de oferecer suporte à inspeção funcional de protótipos de interface utilizando os instrumentos previamente desenvolvidos nas iterações 1 e 2 do ciclo de Design da metodologia de pesquisa deste projeto: CHIF-E - Catálogo de Heurísticas de Inspeção Funcional Expandido e o UIProtoCheck - *checklist* genérico de inspeção funcional de protótipos de UI.

As principais funcionalidades da aplicação GUIPFI estão ilustradas no diagrama de casos de uso da Figura 6.1.

A GUIPFI visa oferecer suporte à inspeção funcional ao oferecer os seguintes recursos:

- Consulta ao CHIF-E e ao UIProtoCheck, através de links para os respectivos instrumentos;
- Campos para informar o nome da aplicação e do protótipo de interface inspecionado, que são utilizados no relatório de inspeção e para compor as informações passadas ao Recomendador;

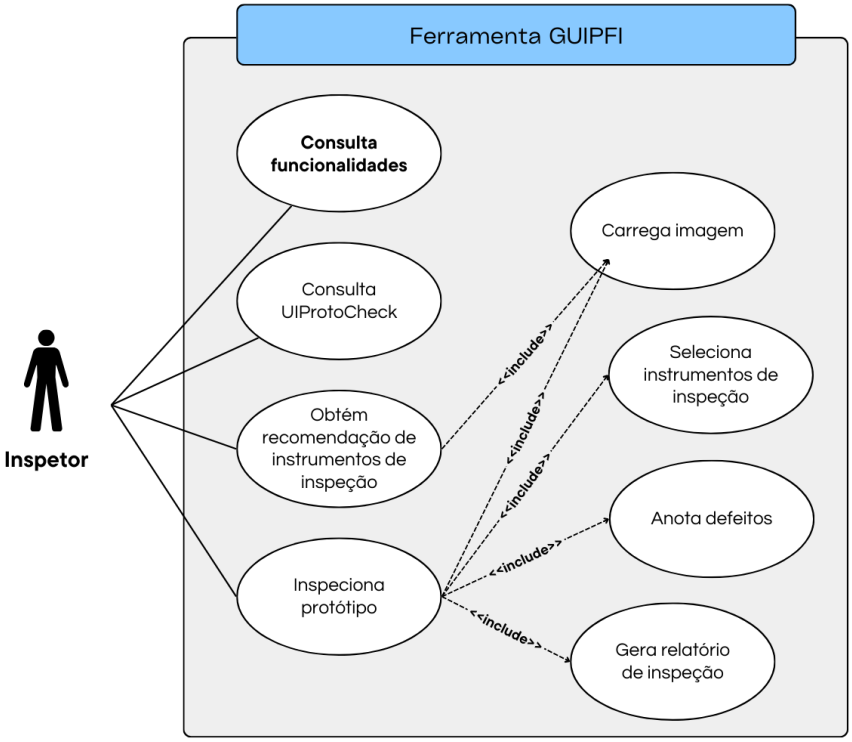


Figura 6.1: GUIPFI - diagrama de casos de uso.

- Opções para carregar e visualizar a imagem do protótipo a ser inspecionado;
- Funcionalidade de recomendação de instrumentos de inspeção relacionados ao protótipo de interface carregado;
- Seleção de instrumentos de inspeção, que exibe informações dos instrumentos selecionados e seus respectivos *checklists*;
- Formulário para preenchimento dos defeitos encontrados no protótipo a partir dos *checklists* instrumentos de inspeção previamente carregados;
- Recurso de geração de relatório de inspeção;
- Registro de um arquivo de *log* das requisições de recomendação de heurísticas, das respostas fornecidas pelo Recomendador e dos defeitos anotados pelo usuário durante a inspeção de um protótipo.

Estes requisitos foram implementados na ferramenta apresentada na Figura 6.2. Ela ilustra a inspeção de um protótipo de *Página de Produto*, o recurso de recomendação de instrumentos de inspeção e o uso da heurística *Página de Produto* para inspecionar o protótipo.

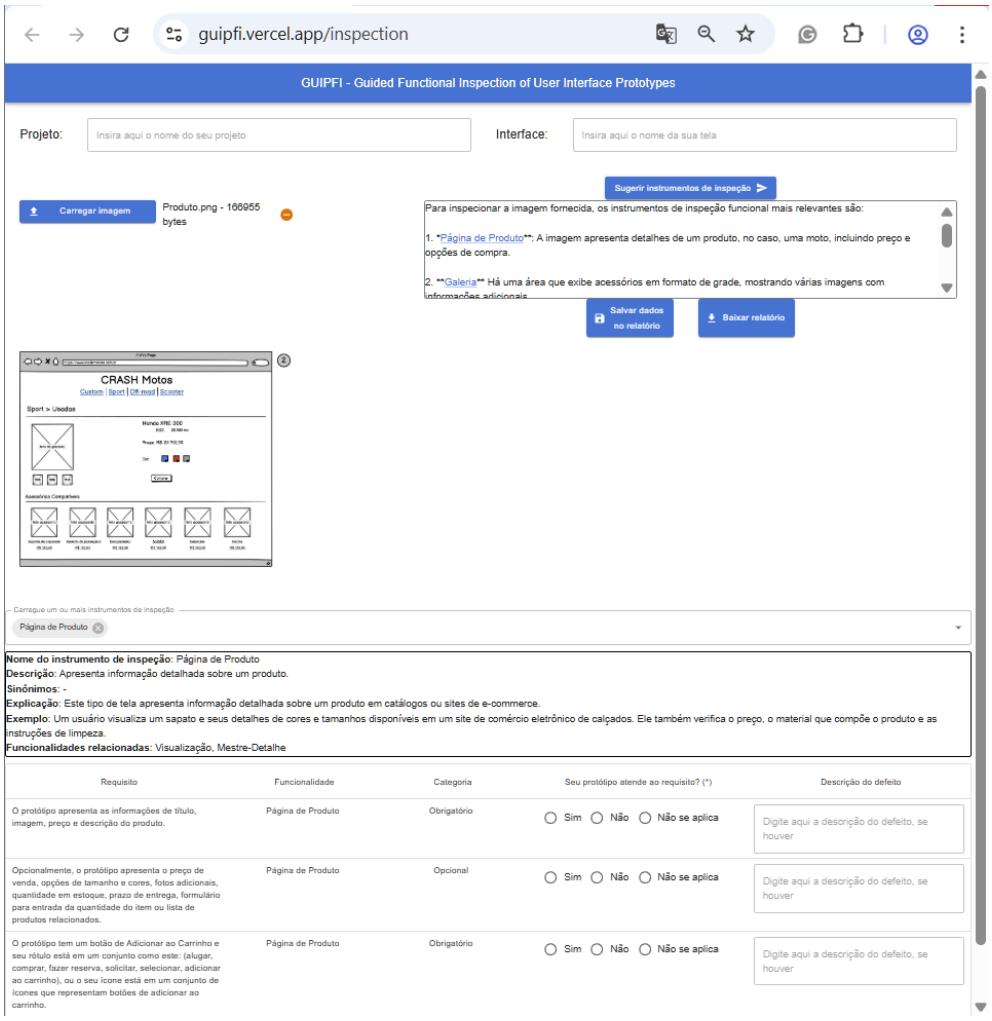


Figura 6.2: GUIPFI - Aplicação de suporte à inspeção de protótipos de interface.

A ferramenta GUIPFI foi desenvolvida com arquitetura Cliente-Servidor, na qual o componente cliente (GUIPFI-Client) é responsável pela interface da aplicação, onde são apresentados os instrumentos de inspeção e a funcionalidade de inspeção. Esta funcionalidade possibilita que o usuário informe o protótipo a ser inspecionado, obtenha recomendações de instrumentos, selecione instrumentos

para apoiar a inspeção e realize a inspeção com os *checklists* dos instrumentos selecionados. Já o componente GUIPFI-Server é responsável pelas interações com o banco de dados de instrumentos de inspeção e com o LLM usado na recomendação. A Figura 6.3 ilustra a arquitetura da ferramenta. As próximas seções detalham o comportamento das funcionalidades de Recomendação e do Assistente de Inspeção.

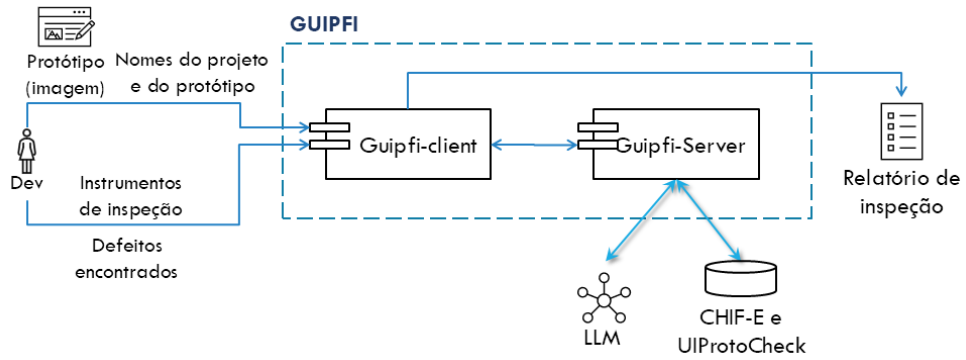


Figura 6.3: GUIPFI - Arquitetura da ferramenta GUIPFI.

6.1.1 Recomendação

O *Recomendador* é um recurso oferecido pela ferramenta GUIPFI que analisa a imagem a ser inspecionada e recomenda os instrumentos mais adequados para inspecioná-la. Para isso, é enviada uma requisição (*prompt*) a um LLM com as informações sobre as heurísticas de inspeção juntamente com o protótipo a ser inspecionado. O LLM adotado na implementação desta ferramenta foi o GPT-4 Vision [1], escolhido pela sua capacidade de lidar com imagens.

O usuário informa, na interface da ferramenta, o protótipo de IU a ser inspecionado em formato de imagem. Opcionalmente, também podem ser informados os nomes da aplicação e do protótipo. A Figura 6.4 ilustra como estes dados são informados na interface da aplicação.

O funcionamento do Recomendador é apresentado na Figura 6.5. Os dados informados compõem, junto com uma especificação resumida das heurísticas do

[1] https://platform.openai.com/docs/guides/vision

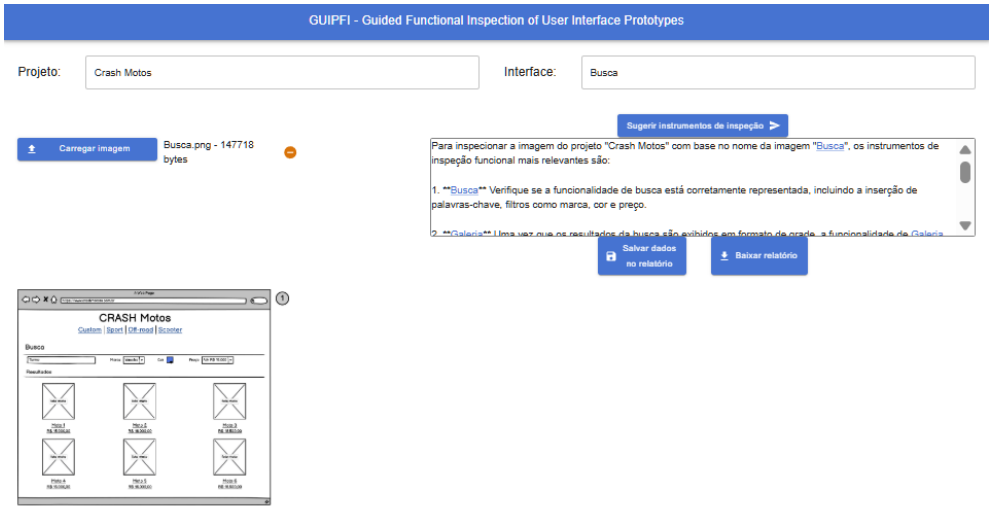


Figura 6.4: GUIPFI - Dados informados para o Recomendador.

CHIF-E e do UIProtoCheck, a requisição passada ao LLM. A resposta do LLM é tratada pela aplicação, que adiciona links para as heurísticas recomendadas, seguidos de uma informação sobre o CHIF-E e o UIProtoCheck e as repassa na forma de recomendação para o usuário da aplicação.

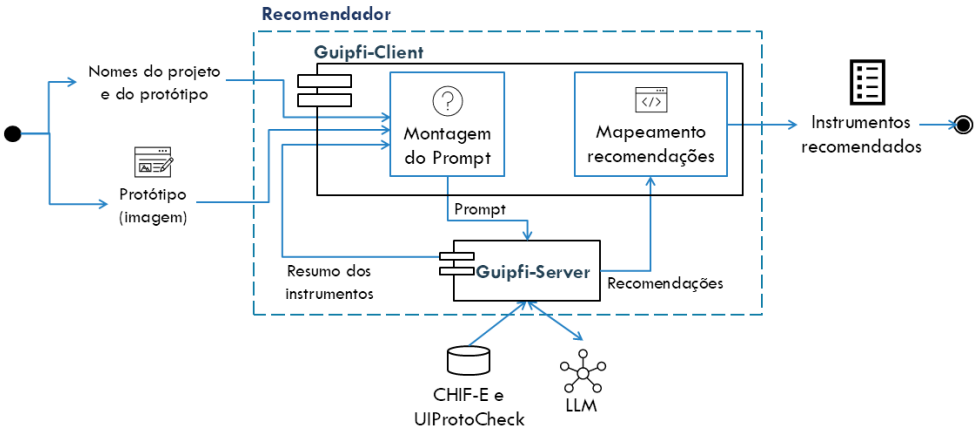


Figura 6.5: Funcionalidade de Recomendação

A requisição passada ao LLM, ou *prompt*, é formada por quatro partes: 1) Descrição geral das heurísticas e de que forma elas são utilizadas; 2) Descrição resumida de cada uma das heurísticas; 3) Imagem passada para análise, opcional-

mente acompanhada do nome do protótipo e do projeto; e 4) Pergunta solicitando sugestão de heurísticas para inspeção. A primeira parte está ilustrada a seguir.

> *Considere as seguintes informações: As heurísticas descritas a seguir descrevem funcionalidades comumente encontradas em sistemas de informação. Elas são usadas para inspecionar protótipos de interface de usuário de aplicações. Elas estão divididas em dois grupos: Funcionalidades Básicas e Funcionalidades Específicas.*
> *O grupo Funcionalidades Básicas descreve funcionalidades genéricas que existem em sistemas de informação e aplicações diversas. Já o grupo Funcionalidades Específicas descreve funcionalidades mais complexas existentes em domínios de aplicação específicos, tais como comércio eletrônico ou redes sociais.*
> *As Funcionalidades Básicas são: Alteração, Busca, Cadastro, Galeria, Lista, Menu, Mestre-Detalhe, Upload de Arquivo e Visualização.*
> *As Funcionalidades Específicas são: Autenticação, Câmera, Carrinho de Compra, Chat, Configurações, Editor de Imagens, Finalizar Compra, Home, Mapas, Media Player, Notícias, Página de Produto, Perfil de Usuário, Quiz, Registro de Conta, Tutorial, Videoconferência.*

A segunda parte, que contém a descrição resumida de cada uma das funcionalidades, concatena as informações de nome, sinônimos, definição, explicação e exemplos, que fazem parte da definição das heurísticas (De Macedo et al., 2022). O exemplo a seguir ilustra o resumo da heurística Busca:

> *A funcionalidade Busca representa um mecanismo de busca. Use este tipo de tela quando quiser ilustrar como os usuários inserem palavras-chave para localizar informações em uma aplicação de software. Uma situação de uso desta funcionalidade ocorre quando um usuário procura um livro pelo título em uma livraria online. Esta funcionalidade também pode ser chamada de Pesquisa.*

A terceira parte do *prompt* contém as informações específicas do protótipo inspecionado. Nesta etapa, caso tenham sido informados, são adicionados os nomes do projeto e da interface, além da imagem do protótipo.

> *Considere a seguinte tela de uma aplicação: <Imagem inspecionada>*;
> *Se o nome do projeto for informado, ele é incluído no prompt:
> *O nome do projeto é: <nome do projeto>*
> *Se o nome do protótipo for informado, ele é incluído no prompt:
> *O nome do protótipo é: <nome do protótipo>*

Finalmente, é enviada a pergunta para que sejam obtidas recomendações de instrumentos de inspeção:

> *Com base nessas informações e nas heurísticas informadas e na tela de aplicação que a imagem apresenta, quais as heurísticas mais relevantes para inspecionar esta tela?*

A Figura 6.6 ilustra o funcionamento do Recomendador de instrumentos de inspeção na ferramenta GUIPFI.

6.1.2 Assistente de Inspeção

O *Assistente de Inspeção* é responsável por oferecer suporte às atividades de inspeção do protótipo de interface. Ele oferece recursos de adição e remoção de instrumentos de inspeção, carregando ou removendo, respectivamente, os itens de cada instrumento selecionado (heurística ou UIProtoCheck). Seu funcionamento está ilustrado na Figura 6.7.

Os itens dos *checklists* dos instrumentos selecionados para inspecionar o protótipo são exibidos em forma de lista, com um campo para anotação de defeitos em cada linha. A Figura 6.8 ilustra essa funcionalidade.

O Assistente de Inspeção também é responsável pela geração do relatório de inspeção, que contém as informações do protótipo, sua imagem, as recomendações de heurísticas e os defeitos registrados pelo usuário da aplicação durante a

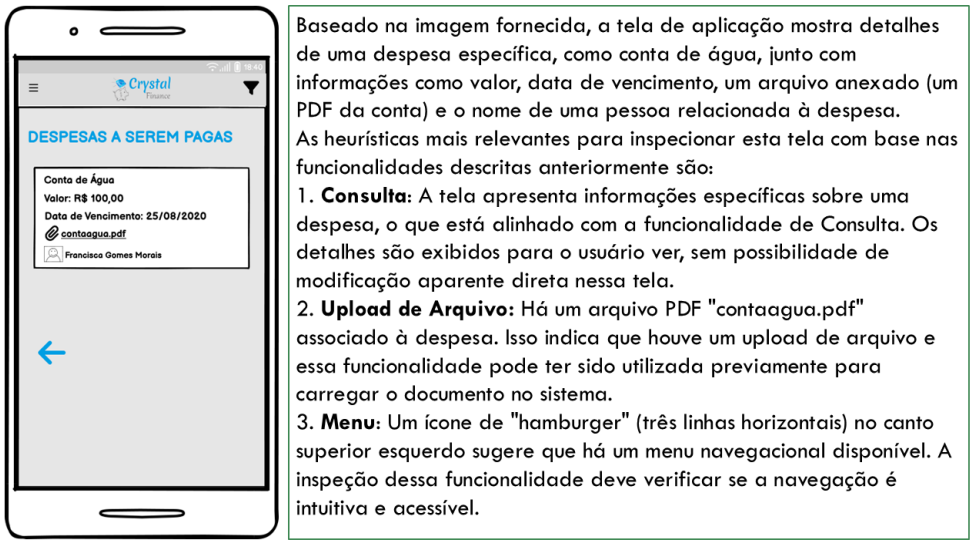


Figura 6.6: GUIPFI - Recomendador de instrumentos de inspeção.

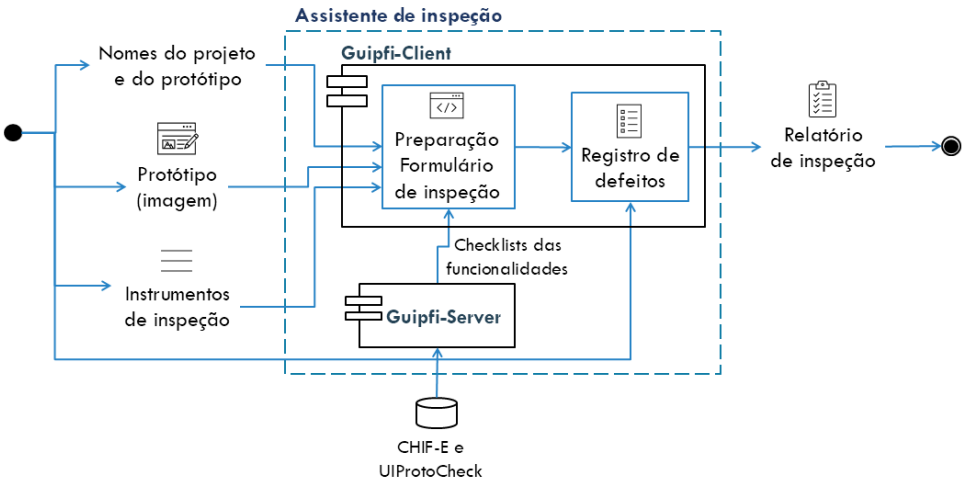


Figura 6.7: Funcionamento do Assistente de Inspeção.

inspeção. A Figura 6.9 apresenta um exemplo de relatório de inspeção em sua versão inicial.

Carregue um ou mais instrumentos de inspeção

Lista Consulta

Nome da ferramenta de inspeção: Lista
Descrição: Exibe uma lista de objetos relacionados a uma entidade do domínio.
Sinônimos: lista vertical, lista horizontal
Explicação: Use este tipo de tela quando quiser ilustrar uma lista de objetos de uma entidade do domínio da aplicação.
Exemplo: Um usuário visualiza uma lista de filmes em um aplicativo de serviço de streaming.
Heurísticas relacionadas: Galeria, Busca, Mestre-Detalhe, Videoconferência

Nome da ferramenta de inspeção: Consulta
Descrição: Exibe valores de atributo de um objeto de uma entidade do domínio da aplicação.
Sinônimos: -
Explicação: Use este tipo de tela para ilustrar a exibição de informações sobre objetos.
Exemplo: Um usuário visualiza informações sobre um livro em um aplicativo de empréstimo de livros.
Heurísticas relacionadas: Mestre-Detalhe, Upload de Arquivo, Perfil de Usuário

Item	Possui relação com a funci...	Categoria do requisito	O seu protótipo atende ao requisito? (Camp...	Descrição do defeito
O protótipo apresenta um conjunto de valores de atributos de um objeto de uma entidade do domínio da aplicação.	Consulta	Obrigatório	Sim Não Não se aplica	Digite aqui a descrição do defeito, se houver
O protótipo apresenta uma lista de objetos de uma entidade do domínio da aplicação.	Lista	Obrigatório	Sim Não Não se aplica	Digite aqui a descrição do defeito, se houver

Figura 6.8: GUIPFI - Heurísticas *Lista* e *consulta* selecionadas como instrumentos de inspeção.

Dados referentes ao 1° checklist preenchido

ID: 67
Item: O protótipo apresenta uma lista de opções de pagamento.
Possui relação com a funcionalidade: Finalizar Compra
Categoria do requisito: Obrigatório
O seu protótipo atende ao requisito? Sim
Descrição do defeito:

ID: 68
Item: O protótipo apresenta um grupo de campos para informar o endereço de entrega.
Possui relação com a funcionalidade: Finalizar Compra
Categoria do requisito: Obrigatório
O seu protótipo atende ao requisito? Não
Descrição do defeito: não possui forma de inserir o endereço de entrega

ID: 69
Item: O protótipo apresenta um grupo de campos para escolha das opções de entrega.
Possui relação com a funcionalidade: Finalizar Compra
Categoria do requisito: Obrigatório
O seu protótipo atende ao requisito? Não
Descrição do defeito: não possui forma de inserir nenhuma informação de entrega

ID: 70
Item: Os rótulos das opções de pagamento estão em um conjunto como este: (cartão de crédito, cartão de débito, boleto bancário, transferência bancária, pagar na entrega), ou seus ícones pertencem a um conjunto de ícones de representam opções de pagamentos.
Possui relação com a funcionalidade: Finalizar Compra
Categoria do requisito: Obrigatório
O seu protótipo atende ao requisito? Sim
Descrição do defeito:

Figura 6.9: GUIPFI - Relatório de inspeção inicial.

6.1.3 Avaliação do desempenho do LLM na sugestão de heurísticas relevantes

Para avaliar a relevância das sugestões de heurísticas oferecidas pelo LLM, foram realizados testes com um conjunto de protótipos previamente rotulados. Este conjunto foi formado por dois protótipos de cada funcionalidade representada pelas 26 heurísticas, totalizando 52 protótipos.

Os protótipos de interface utilizados nesta avaliação foram obtidos em atividades de prototipação de projetos de interface de sistemas de software. A base contém 915 imagens e respectivos *jsons* dos protótipos, que foram rotulados em 31 tópicos. Também foi utilizado o *dataset* EnRICO (Leiva et al., 2020), que foi analisado e reanotado nas categorias de protótipos obtidas no segundo ciclo de design (Capítulo 5). A Figura 6.10 ilustra exemplos de protótipos utilizados nesta avaliação.

Figura 6.10: Exemplos de protótipos de interface e *screenshots* utilizados na avaliação das sugestões de heurísticas do LLM.

Dos 52 protótipos utilizados para obter recomendações de heurísticas, 44 receberam sugestões compatíveis com os rótulos previamente atribuídos a eles, uma taxa de acerto de 84,6%. O modelo se mostrou eficiente também para recomendar heurísticas para telas rotuladas com vários rótulos, como o exibido na Figura 6.11. A sugestão recebida sugere as heurísticas das funcionalidades Menu, Lista e Busca.

"Para inspecionar a tela apresentada na imagem, as seguintes funcio-

Figura 6.11: Exemplo de protótipo que representa mais de uma funcionalidade (Menu, Lista e Busca).

> *nalidades heurísticas são relevantes:*
> *1.* ***Busca (Pesquisa)*** *- A barra de pesquisa no topo sugere um mecanismo para que usuários localizem artigos ou locais dentro do aplicativo. (...)*
> *2.* ***Menu*** *- A tela exibe um menu ou uma barra de categorias (Balões, Velas, Decorações, + Outros), que são formas de navegação pela aplicação.(...)*
> *3.* ***Lista*** *- Os Produtos Populares estão dispostos em uma lista vertical, que é uma forma eficaz de apresentar múltiplos itens para o usuário de modo sequencial.(...) "*

Em alguns casos, o modelo também sugeriu heurísticas de funcionalidades não presentes na tela, mas que possivelmente fariam parte da aplicação. A recomendação abaixo é sobre um protótipo de Carrinho de Compras (Figura 6.12) e a recomendação obtida sugere também as heurísticas Finalizar Compra e Menu, relacionadas à funcionalidade principal.

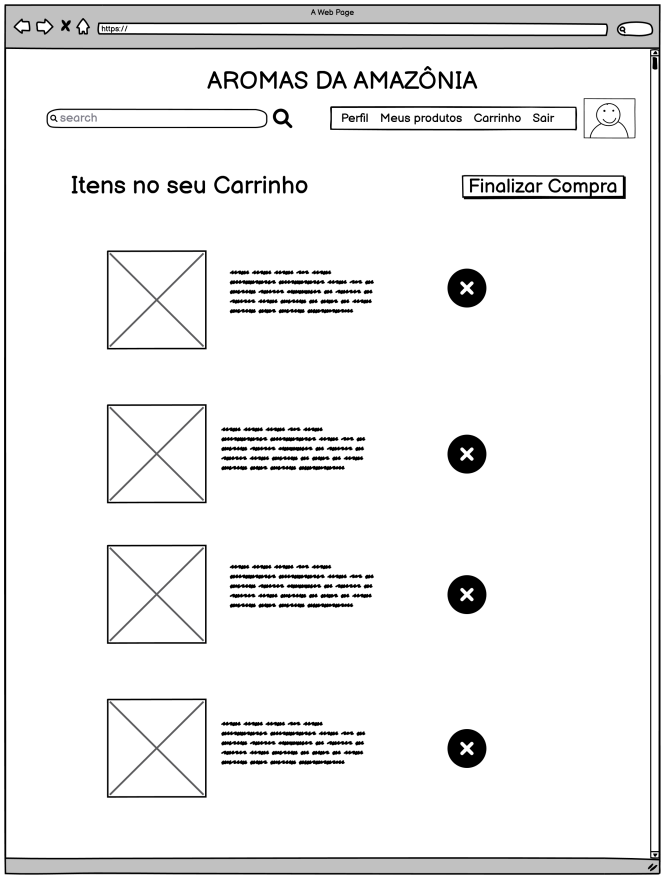


Figura 6.12: Exemplo de protótipo (*Carrinho de Compra*) que recebeu sugestões de heurísticas de funcionalidades não presentes na tela, mas que possivelmente fariam parte da aplicação.

> *"A imagem apresenta uma tela de um aplicativo ou website simulando a página de um carrinho de compras de uma loja online (...). Com base nas heurísticas informadas, as mais relevantes para inspecionar esta tela seriam:*
> *1.* ***Carrinho de Compra****: A tela exibe os itens no carrinho de compras, permitindo que o usuário examine quais produtos selecionou para compra.(...)*
> *2.* ***Finalizar Compra****: Existe um botão de "Finalizar Compra", o que sugere uma interação de encerramento no processo de compra, permitindo que o usuário prossiga para o pagamento e finalize a tran-*

sação. (...)
3. ***Menu****: A parte superior da tela apresenta opções de menu, como "Perfil", "Meus produtos", "Carrinho"e "Sair", permitindo ao usuário navegar entre diferentes áreas do aplicativo ou do site. "*

Os casos em que a recomendação não correspondeu à funcionalidade representada nos protótipos foram:

- Dois do tipo Home, que receberam recomendações de diversas funcionalidades;
- Dois do tipo Mestre-Detalhe, com recomendações das heurísticas Visualizar e Lista;
- Um Editor de Imagens, com recomendação da heurística Galeria;
- Um Cadastro, com recomendação de heurística de Alteração;
- Um protótipo representando um *Perfil de Usuário*, com recomendação das heurísticas Alteração, Upload de Arquivo e Menu.
- Um protótipo de Câmera, onde não foi identificada uma funcionalidade e foram recomendadas heurísticas de usabilidade.

Estes casos foram úteis para indicar pontos a serem melhorados em duas heurísticas: Home e Mestre-Detalhe. No caso da heurística Home, é feita uma caracterização genérica sobre o que significa uma tela inicial de uma aplicação, que pode variar enormemente em termos de conteúdo. Para os exemplos de protótipos de interface desta funcionalidade, o LLM recomendou heurísticas para as diversas funcionalidades apresentadas no protótipo, tais como Menu, Busca, Lista e Galeria.

No caso da heurística Mestre-Detalhe, que representa uma relação entre dados de uma instância de uma entidade do domínio e itens de outra entidade relacionados à primeira, o modelo não reconheceu nos dados este tipo de relação. No entanto, ele recomendou as heurísticas Visualização e Lista, que, combinadas, formam as partes da funcionalidade Mestre-Detalhe, porém, sem a relação de dependência entre elas.

Os resultados obtidos na avaliação do *prompt* elaborado para obtenção de recomendações de instrumentos de inspeção foram considerados suficientemente satisfatórios para implementação da ferramenta GUIPFI. A próxima seção apresenta o planejamento, a execução e os resultados do estudo de avaliação da GUIPFI no contexto da academia.

6.2 Avaliação da GUIPFI na Academia

O primeiro estudo de avaliação da ferramenta GUIPFI foi realizado com o objetivo de analisar seu suporte às atividades de inspeção funcional, envolvendo a recomendação de instrumentos de inspeção e o registro de defeitos. Para isso, o GUIPFI foi utilizado para inspecionar protótipos de interface elaborados por alunos matriculados na disciplina de Tópicos Avançados em Engenharia de Software, cuja ementa abordada foi Engenharia de Software Baseada em LLMs. As seções a seguir apresentam o planejamento e a execução do estudo e os resultados obtidos na avaliação da ferramenta GUIPFI no contexto acadêmico.

6.2.1 Planejamento

O estudo de avaliação do GUIPFI no contexto acadêmico foi realizado com estudantes da disciplina de Tópicos Avançados em Engenharia de Software, na qual estavam matriculados alunos dos cursos de graduação em Engenharia de Software e de pós-graduação em Informática (mestrado e doutorado).

O tema abordado na disciplina foi Engenharia de Software baseada em LLMs. No decorrer do curso, os estudantes, divididos em equipes, realizaram as etapas do ciclo de desenvolvimento de um software utilizando LLMs. As tarefas iniciaram a partir da descrição do software a ser desenvolvido por cada equipe e incluíram a geração de histórias de usuário, de mapa de jornada do usuário, de protótipos de interface, de modelo de dados e do código *back-end*.

O estudo foi realizado após a etapa de elaboração de protótipos de interface. Cada equipe que elaborou um projeto de aplicação teve seus protótipos inspecionados por uma das outras equipes. Para realizar a inspeção, os membros das equipes, no papel de revisores, receberam a especificação e os protótipos de

Tabela 6.1: Designação de equipes de inspetores e desenvolvedores

Inspetores do	**Inspecionaram protótipos do**
Grupo 1	Grupo 2
Grupo 2	Grupo 5
Grupo 5	Grupo 6
Grupo 6	Grupo 7
Grupo 7	Grupo 8
Grupo 8	Grupo 10
Grupo 10	Grupo 11
Grupo 11	Grupo 1

interface da aplicação a ser inspecionada utilizando a ferramenta GUIPFI. A atribuição de papéis de inspetores, em que a inspeção de protótipos de interface é desempenhada por uma equipe diferente da que os desenvolveu, visa simular as atividades desempenhadas no ambiente real, estimulando os participantes a se concentrarem nas atividades de inspeção. Além disso, a atribuição de papéis visa diminuir o viés de aprovação que poderia ocorrer caso os participantes inspecionassem os próprios protótipos. A Tabela 6.1 mostra a designação das equipes desenvolvedoras e revisoras.

Os materiais utilizados no estudo foram: (i) Termo de Consentimento Livre e Esclarecido (TCLE); aula sobre inspeção funcional; (iii) instruções sobre a atividade de inspeção; (iv) a ferramenta GUIPFI; (v) questionário de caracterização dos participantes e avaliação da ferramenta GUIPFI.[2]

O questionário de avaliação é composto por três seções. A primeira seção contém o TCLE, cuja assinatura é necessária para a apresentação das demais seções. A segunda seção (Tabela 6.2) contém perguntas de caracterização dos participantes em relação à experiência profissional e à área de atuação.

A terceira seção do questionário contempla quatro construtos do *Technology Acceptance Model* - TAM 3 (Venkatesh and Bala, 2008) que foram selecionados para avaliar a ferramenta GUIPFI: Facilidade de Uso Percebida *(Perceived Ease*

[2]Pesquisa aprovada pelo Comitê de Ética em Pesquisa - CEP da Universidade Federal do Amazonas - UFAM, sob número CAAE: 82963224.1.0000.5020, em 18/07/2024, Parecer Consubstanciado N. 7.072.722, de 12 de setembro de 2024.

Tabela 6.2: Caracterização do perfil dos participantes

ID	Pergunta
P1	Por favor, indique a sua faixa etária.
P2	Informe, por favor, a sua ocupação.
P3	Por favor, indique quantos anos de experiência você tem na indústria de software.
P4	Se você atua na indústria de software, informe por favor em qual tipo de plataforma você costuma atuar.

Of Use – PEOU), Utilidade Percebida *(Perceived Usefulness – PU)*, Qualidade do Resultado *(Output Quality – OUT)* e Intenção de Uso *(Behavioral Intention – BI)*. Além disso, com o objetivo de mitigar vieses de aceitação da ferramenta, foram incluídas perguntas para avaliar qualitativamente as quatro dimensões do TAM, bem como os pontos fortes e os pontos fracos da ferramenta. A Tabela 6.3 apresenta as perguntas dos quatro construtos utilizados, adaptadas para avaliar a ferramenta GUIPFI. A Tabela 6.4 lista as perguntas complementares do questionário.

6.2.2 Execução

O estudo iniciou-se com uma apresentação aos participantes sobre inspeção funcional e os instrumentos de inspeção CHIF-E e UIProtoCheck. Em seguida, a ferramenta GUIPFI foi apresentada, demonstrando seu funcionamento e as funcionalidades de consulta aos instrumentos de inspeção, recomendações de instrumentos e suporte à inspeção funcional. Ao final da aula, foi realizado um exercício em sala, no qual os participantes receberam a descrição de um sistema de software e três protótipos de interface deste sistema, sendo orientados a escolher um destes para inspecionar funcionalmente utilizando a GUIPFI. Durante a realização do exercício, as dúvidas em relação ao uso da ferramenta foram esclarecidas.

Após a realização do exercício, a atividade de inspeção foi passada aos participantes, sendo dado a eles o prazo de dois dias para realização e envio dos relatórios de inspeção. Também foi solicitado, aos participantes que concordassem em participar da pesquisa, que respondessem o questionário com o TCLE e

Tabela 6.3: Questionário de avaliação da ferramenta GUIPFI - TAM

ID	Pergunta
Utilidade percebida	
U1	Usar a ferramenta GUIPFI melhora meu desempenho na inspeção de protótipos de interface.
U2	Usar o GUIPFI para identificar defeitos funcionais em protótipos de interface melhora a minha produtividade.
U3	Usar o GUIPFI melhora a minha eficácia na detecção de defeitos funcionais em protótipos de interface.
U4	Eu considero a ferramenta GUIPFI útil na inspeção funcional de protótipos de interface.
Facilidade de uso	
F1	Minha interação com a ferramenta GUIPFI foi clara e compreensível.
F2	Interagir com a ferramenta GUIPFI não exigiu muito do meu esforço mental.
F3	Eu considero a ferramenta GUIPFI fácil de utilizar.
F4	Eu acho fácil usar a ferramenta GUIPFI para auxiliar a inspeção de protótipos de interface.
Intenção de uso	
I1	Supondo que eu tivesse acesso à ferramenta GUIPFI, eu pretendo utilizá-la para apoiar a inspeção funcional de protótipos de interface.
I2	Dado que eu tenho acesso à ferramenta GUIPFI, eu prevejo que eu a usaria para inspecionar protótipos de interface de futuros projetos.
I3	Eu planejo usar o GUIPFI nos próximos meses para inspecionar funcionalmente protótipos de interface.
Qualidade das recomendações - Sugerir ferramentas de inspeção	
Q1	A qualidade das recomendações é alta.
Q2	Eu não tive problemas com a qualidade das recomendações.
Q3	Q3 - Eu classifico as recomendações de ferramentas de inspeção como excelentes.

Tabela 6.4: Questionário de avaliação da ferramenta GUIPFI - Perguntas complementares (abertas)

ID	Pergunta
U5	Por favor, descreva em quais situações a ferramenta foi útil durante a inspeção funcional dos protótipos de interface.
F5	Se você teve alguma dificuldade no uso da ferramenta GUIPFI, por favor, descreva aqui.
I4	Você usaria essa ferramenta na prática de engenharia de software? Se não, por quê? Se sim, em quais situações durante o processo de desenvolvimento?
Q4	Por favor, descreva como você utilizou a funcionalidade "Sugerir ferramentas de inspeção"durante a inspeção de protótipos. (aberta)
P5	Agora, indique por favor quais aspectos da ferramenta GUIPFI você considera serem os pontos fortes.
P6	Para finalizar, indique quais aspectos da ferramenta GUIPFI você considera serem os pontos fracos.

as perguntas de avaliação da ferramenta.

6.2.3 Resultados

Os resultados da avaliação da ferramenta GUIPFI foram avaliados através de análise estatística e qualitativa. Na análise estatística, as inspeções foram contabilizadas individualmente por participante e interface inspecionada. As inspeções realizadas por participantes que não assinaram o Termo de Consentimento foram desconsideradas. Foram obtidas 130 inspeções de 62 protótipos de interface e de 21 participantes.

Para caracterizar o perfil dos participantes, buscou-se analisar a proporção de participantes com atuação na indústria de desenvolvimento de software, através da ocupação, do tempo de experiência e da área de atuação. Em relação à ocupação dos participantes, houve 11 estudantes, seis desenvolvedores de software, dois analistas de software, entre outras ocupações, conforme ilustra a Figura 6.13.

Em relação à experiência profissional na área de desenvolvimento de software, 12 dos participantes alegaram ter pouca ou nenhuma experiência. Por outro lado, houve um número considerável de participantes com experiência significativa (9),

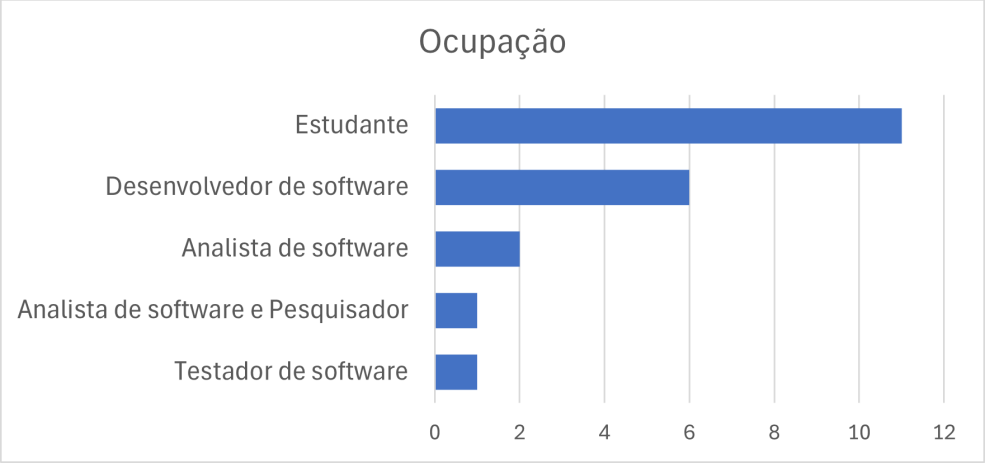


Figura 6.13: Ciclo 3 - Ocupação dos participantes.

sendo sete atuando entre 1 e 5 anos e dois profissionais com 10 a 15 anos de atuação, como mostrado na Figura 6.14.

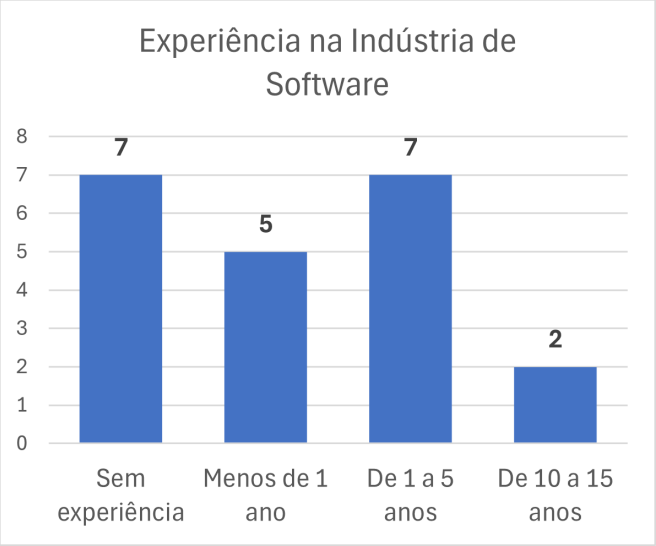


Figura 6.14: Ciclo 3 - Tempo de experiência na indústria de software.

Já em relação à área de atuação na indústria de software, a maioria atua no desenvolvimento Web (50%), seguida pelo desenvolvimento de aplicativos móveis (34%), conforme ilustra a Figura 6.15.

As inspeções foram analisadas através de análise estatística, visando verificar o quanto o recurso de recomendação foi utilizado e se as recomendações foram

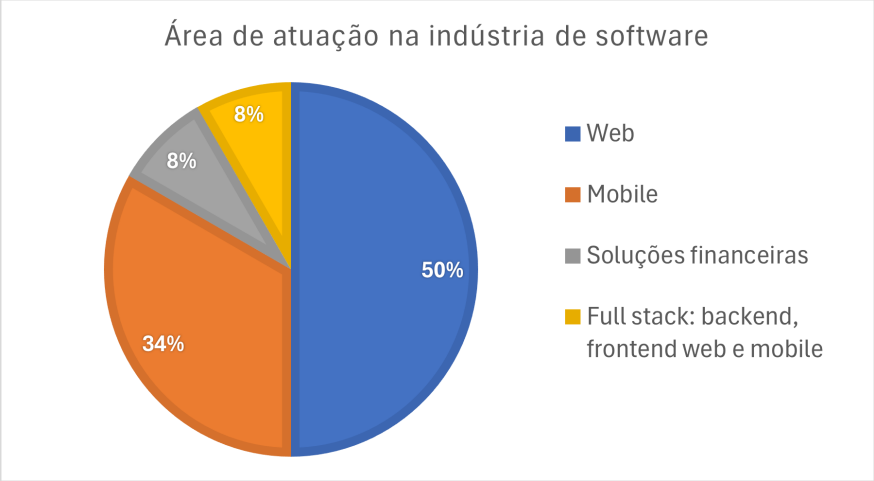


Figura 6.15: Ciclo 3 - Área de atuação na indústria de software.

seguidas. Além disso, buscou-se avaliar o quanto o CHIF-E e o UIProtoCheck foram recomendados ou se foram recomendadas outras heurísticas que não as voltadas para inspeção funcional. Para isso, contabilizou-se para cada inspeção obtida:

- Se usou a funcionalidade de recomendação;
- Quais instrumentos de inspeção foram recomendados pelo recomendador;
- Quais instrumentos de inspeção foram efetivamente usados na inspeção;
- Se foi utilizado algum dos instrumentos de inspeção recomendados pelo recomendador;
- Se usou mais de um instrumento de inspeção;
- Se usou o UIProtoCheck;
- Se foram recomendadas outras heurísticas, além do CHIF-E e do UIProto-Check.

O resultado desta análise é apresentado na Figura 6.16. Do total de 130 inspeções, 82,3% utilizaram alguma recomendação, das quais 78,5% aceitaram pelo

	Quant.	%
Total de inspeções	130	100%
Inspeções que usaram recomendação	107	82,3%
Inspeções que aceitaram alguma das recomendações	84	78,5%
Inspeções que usaram mais de um instrumento de inspeção	57	43,8%
Recomendações com o UIProtoCheck	32	24,6%
Inspeções que usaram o UIProtoCheck	11	8,5%
Recomendações sem nenhuma das funcionalidades ou o UIProtoCheck	15	11,5%
Recomendações de heurísticas não presentes no catálogo	45	34,6%

Figura 6.16: Ciclo 3 - Análise estatística das inspeções funcionais de protótipos de interface.

menos uma delas, o que demonstra uma boa aceitação do recurso de recomendação. Além disso, 43,8% das inspeções aplicaram mais de um instrumento de inspeção, indicando que a possibilidade de uso de diferentes instrumentos foi assimilada pelos participantes. Por outro lado, 34% das recomendações ofereceram heurísticas não presentes no catálogo, e 11,5% das recomendações não envolveram nenhuma das funcionalidades do CHIF-E ou o UIProtoCheck. Esses casos podem estar relacionados à ausência de heurísticas específicas para as funcionalidades analisadas, o que levou o sistema a sugerir alternativas genéricas ou conceitualmente próximas. Tais resultados evidenciam a necessidade de aprimorar o modelo de recomendação, seja por meio do refinamento das descrições textuais das heurísticas, seja pela expansão do catálogo com novas entradas que representem melhor as variações de contexto e terminologia observadas nos protótipos inspecionados.

O resultado da avaliação do construto Utilidade Percebida é apresentado na Figura 6.17. É possível verificar que a maioria dos participantes considerou a ferramenta útil para inspecionar funcionalmente os protótipos de interface (80%), tendo concordado em relação a ganhos de eficácia (80%), produtividade (85%) e desempenho (90%).

Esse resultado é corroborado pela análise qualitativa das questões U5 e P5,

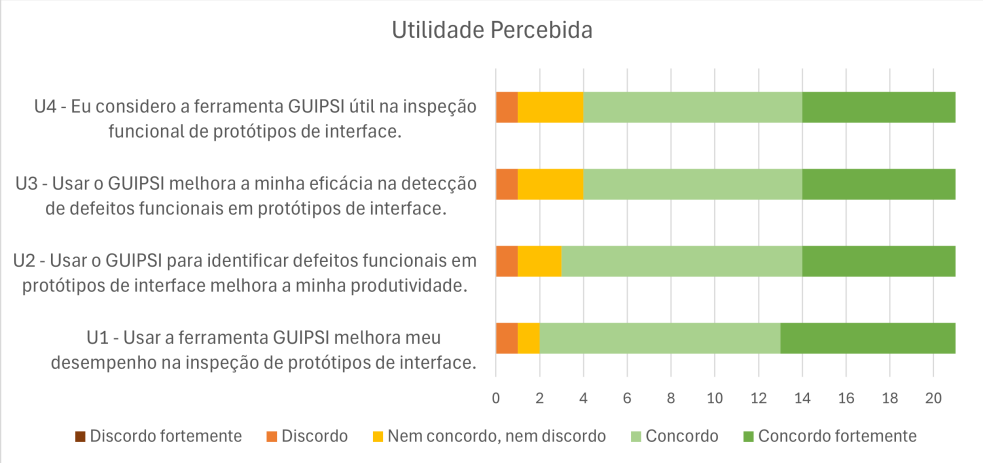


Figura 6.17: Utilidade percebida.

que questionam em quais situações a ferramenta foi útil e seus pontos fortes. A Figura 6.18 apresenta a rede de códigos resultante desta análise.

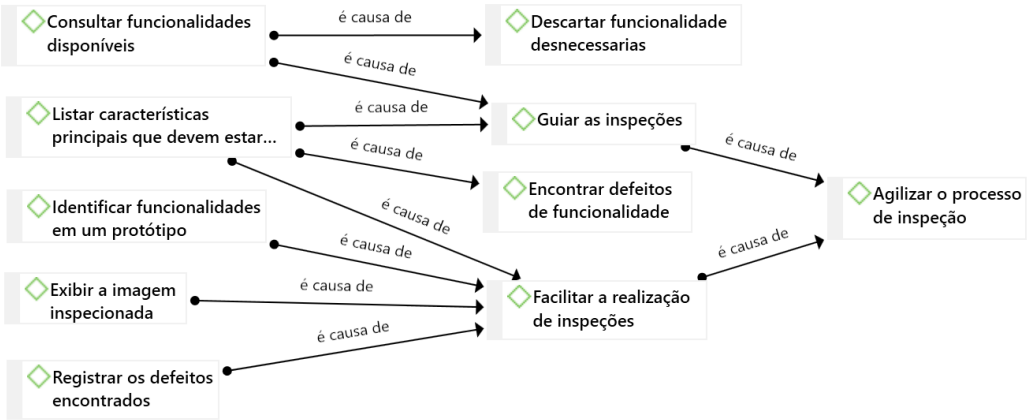


Figura 6.18: Utilidade percebida.

Os participantes indicaram que a ferramenta foi útil para consultar as funcionalidades e seus requisitos, o que ajudou a guiar as inspeções na seleção de heurísticas e na desconsideração de funcionalidades inadequadas para os protótipos inspecionados.

> P19 - *"A linkagem das heurísticas dentro da resposta retornada pelo ChatGPT permite uma rápida consulta sobre as informações de uma determinada heurística."*

> P2 - *"Ela foi útil em me fornecer perguntas para direcionar minha inspeção e assim torná-la mais objetiva e eficiente."*

> P7 - *"Foi útil para analisar e descartar funcionalidades desnecessárias."*

Além disso, os participantes indicaram que o recurso de recomendação utilizando um LLM disponível na ferramenta foi útil para identificar as funcionalidades representadas nos protótipos e que os *checklists* das heurísticas e o formulário para registrar os defeitos encontrados tornaram a inspeção mais prática.

> P3 - *"Foi útil em dois momentos. No primeiro momento, a ferramenta utiliza uma IA para informar quais são as principais heurísticas encontradas em um determinado tipo de tela. Em outra etapa, a ferramenta gera um checklist para o tipo de tela que você está inspecionando; isso agiliza o processo."*

> P10 - *"Torna a inspeção mais prática por conta dos formulários."*

Eles também indicaram que estes recursos, aliados à exibição da imagem inspecionada na própria ferramenta, contribuíram para facilitar a realização das inspeções, agilizando o processo de inspeção de um modo geral.

> P1 - *"Apresentar a imagem da tela sob inspeção no momento do preenchimento do checklist ajuda a acelerar o processo."*

> P17 - *"Facilitou um processo que exigiria um trabalho maior."*

> P13 - *"Apresentou todos os pontos principais que eu deveria inspecionar em cada tela e a inspeção durou menos de 15 minutos."*

Em relação à Facilidade de Uso Percebida, é possível verificar na Figura 6.19 que, embora a maioria dos participantes tenha concordado em relação à Facilidade de Uso, um número significativo de participantes enfrentou dificuldades durante o uso da ferramenta, especialmente em relação ao esforço mental necessário e à clareza das informações.

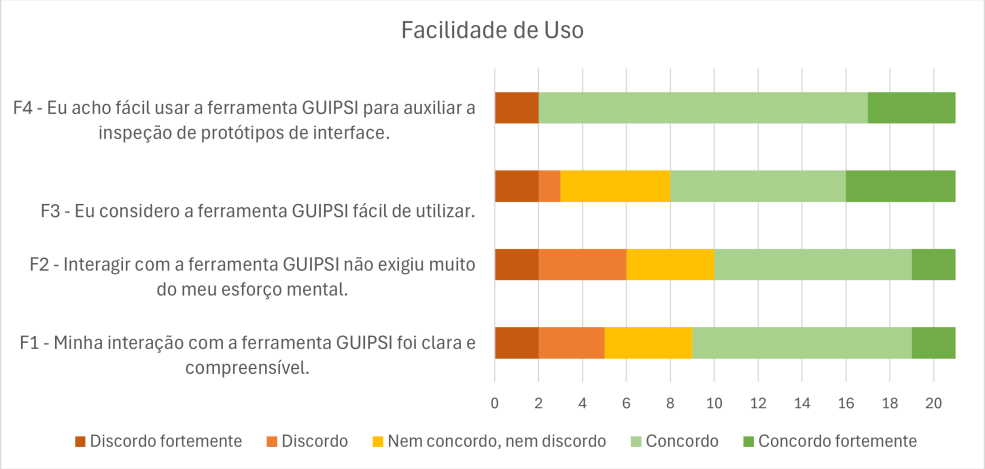


Figura 6.19: Facilidade de uso percebida.

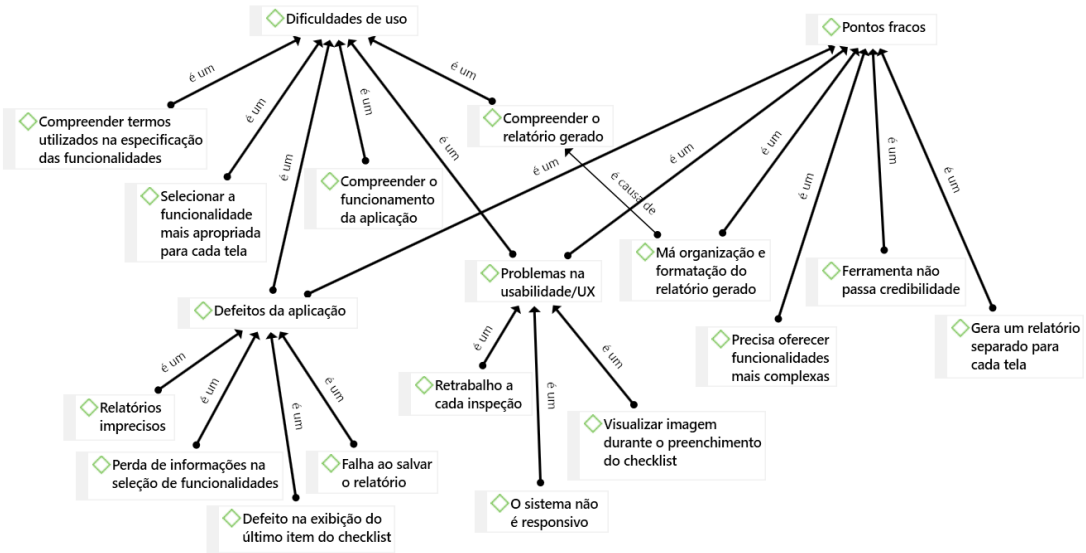


Figura 6.20: Dificuldades encontradas e pontos fracos da ferramenta.

A análise qualitativa das dificuldades encontradas e pontos fracos da ferramenta, realizada a partir de codificação aberta das respostas às questões F5 e P6, é apresentada na rede da Figura 6.20.

Esta análise revelou que os principais problemas enfrentados foram em relação à dificuldade de compreensão do funcionamento da ferramenta, dos termos utilizados na especificação das funcionalidades e do relatório gerado.

> P15 - *"Achei alguns termos um pouco difíceis de entender e algumas perguntas parecem ser muito similares."*

> P10 - *"Achei difícil entender os passos para realizar a inspeção. Foi necessário perguntar como usa."*

> P3 - *"Há informações que eu, como usuário, não consegui entender no relatório."*

Além disso, alguns participantes ainda relataram dificuldade para selecionar instrumentos de inspeção apropriados para as interfaces inspecionadas, apesar do recurso de recomendação de instrumentos de inspeção.

> P12 - *"Algumas telas se encaixavam parcialmente na descrição [das heurísticas] e fiquei confuso sobre qual escolher."*

Outros fatores de influência identificados na dificuldade de uso foram defeitos da aplicação, tais como falha ao preencher defeitos e salvar o relatório, e problemas de usabilidade, como a organização dos componentes na tela e a falta de responsividade da ferramenta.

> P15 - *"Ao tentar selecionar uma outra funcionalidade básica durante o processo de inspeção, todas as avaliações são perdidas."*

> P13 - *"Alguns elementos da tela pareciam não estar bem ajustados para a resolução que eu estava utilizando."*

Em relação à intenção de uso, a maioria dos participantes (81%) respondeu que concorda ou fortemente concorda que prevê que a usaria (I2) ou pretende utilizá-la (I3) durante inspeções funcionais de protótipos de interface. Por outro lado, as respostas foram negativas em relação a planejar utilizar a ferramenta nos próximos meses (I1). Estes resultados estão ilustrados na Figura 6.21.

Para identificar a motivação do resultado da pergunta I1, foram analisadas as respostas à pergunta aberta I4, onde os participantes explicaram em quais situações usariam a ferramenta e por que não usariam. A Figura 6.22 apresenta a rede de códigos resultante desta análise.

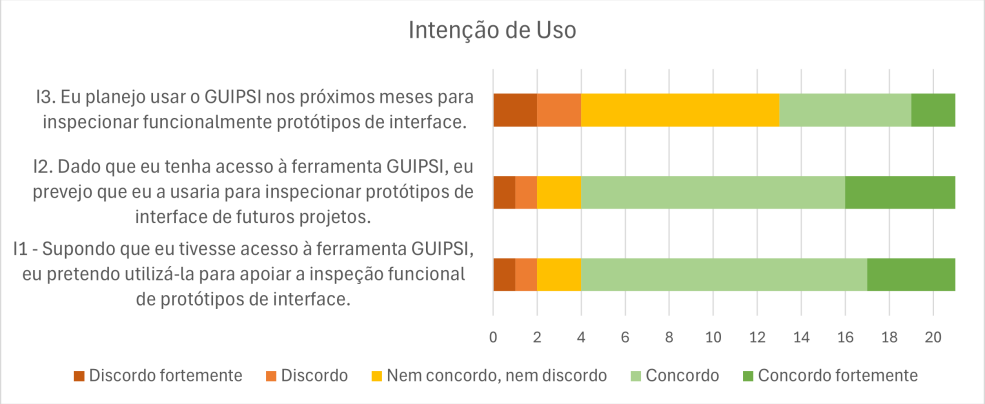


Figura 6.21: Intenção de uso da GUIPFI.

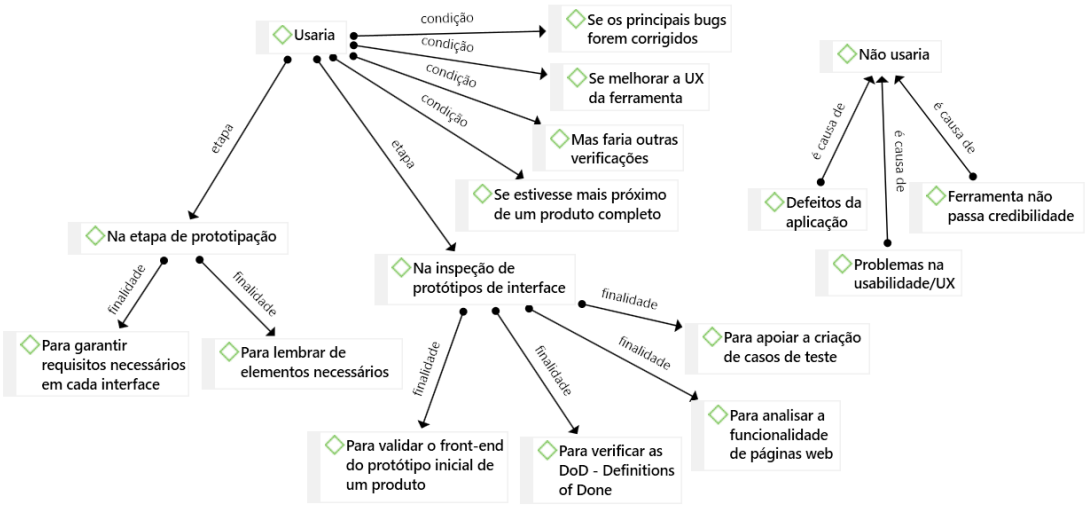


Figura 6.22: Intenção de uso da ferramenta GUIPFI.

Os participantes que alegaram que usariam a ferramenta, o fariam durante a prototipação e na etapa de inspeção de protótipos. Na prototipação, utilizariam a GUIPFI para consultar os elementos necessários à funcionalidade ou garantir que eles estão presentes no protótipo.

> P8 - *"É uma boa maneira de garantir que há tudo o que é necessário para cada interface que estiver sendo criada."*

> P6 - *"A gente usou na parte de prototipação e achei bem útil, porque ali a gente consegue enxergar erros ou algumas coisas que estão faltando."*

Já na etapa de inspeção, a GUIPFI seria utilizada para validar o *front-end* da versão inicial de um produto e para apoiar a criação de casos de teste.

> P7 - *"Usaria para analisar o meu MVP ou para analisar o front-end de começo de um projeto."*

> P3 - *"Usaria como ferramenta para agilizar o processo de teste a fim de validar um produto em estado de protótipo inicial."*

> P15 - *"A ferramenta seria mais um apoio para criação de casos de testes, ou seja, um time de testes iria usar os formulários de avaliação de interfaces para criar casos de testes para validar as funcionalidades nos frameworks de testes que já existem."*

Outros participantes alegaram que a decisão de usar a ferramenta dependeria da correção dos defeitos e de melhorias na experiência de uso.

> P1 - *"Certamente faria uso de alguma ferramenta similar. Porém, a versão utilizada durante esta atividade ainda tem alguns pontos de UX que poderiam ser melhorados a fim de acelerar o desempenho da inspeção."*

> P21 - *"Usaria, se estivesse mais próximo de um produto completo."*

Finalmente, dentre os principais motivos alegados pelos participantes que não usariam a ferramenta, estão os defeitos da aplicação e os problemas de usabilidade.

> P11 - *"Não [usaria], pois acredito que devem existir ferramentas melhores em questão de desenvolvimento e que tenham menos bugs."*

> P16 - *"Não usaria por conta dos problemas na experiência.(...) Precisa melhorar a experiência do usuário."*

O construto Qualidade do Resultado do TAM 3 (Venkatesh and Bala, 2008) foi utilizado neste trabalho para avaliar os resultados produzidos pelo Recomendador. A Figura 6.23 apresenta as respostas dos participantes para as questões Q1, Q2

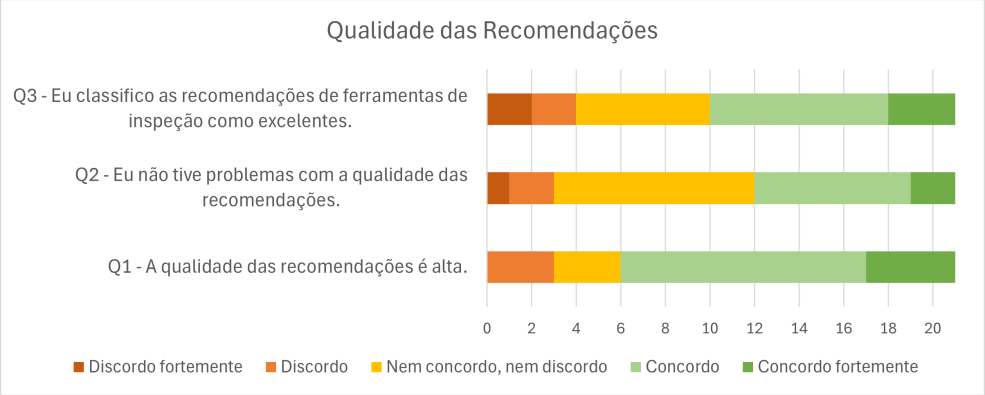


Figura 6.23: Avaliação da qualidade dos resultados - Recomendador.

e Q3. Nela, é possível observar que os participantes tiveram opiniões divididas quanto à qualidade das recomendações, especialmente na questão Q2, que se refere aos problemas encontrados.

As respostas à questão aberta Q4 foram analisadas através de codificação aberta e elucidaram os motivos destas avaliações. A Figura 6.24 apresenta a rede de códigos obtidos nesta análise.

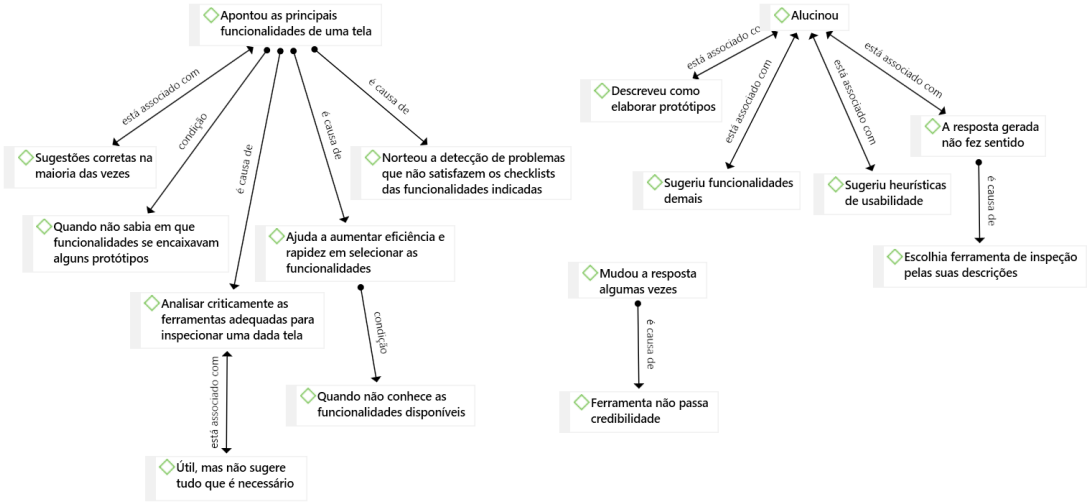


Figura 6.24: Q4 - Qualidade das recomendações de instrumentos de inspeção obtidas com o uso da GUIPFI.

Foram identificados diferentes resultados em relação à qualidade das recomendações. Alguns participantes indicaram que as recomendações apontaram as

principais funcionalidades do protótipo, sendo estas funcionalidades coerentes na maioria das vezes. O recurso foi utilizado quando os participantes não conseguiam identificar a funcionalidade representada, a exemplo do participante P14.

> P14 - *"Eu utilizei para descobrir quais ferramentas eram adequadas."*

O recurso norteou a detecção de problemas nos protótipos relacionados à funcionalidade e foi usado também como uma forma de conferência das heurísticas que o participante julgou adequadas.

> P19 - *"A usei como forma de apoiar a minha análise crítica sobre quais seriam as possíveis heurísticas que deveriam ser inspecionadas em uma dada tela."*

Por outro lado, houve vários participantes que relataram que a ferramenta alucinou de diversas maneiras, seja sugerindo heurísticas demais, recomendando heurísticas de usabilidade, descrevendo formas de prototipar ou mesmo gerando respostas sem sentido.

> P16 - *"Duas vezes o ChatGPT sugeriu heurísticas de Nielsen para a inspeção e não checklists, o que não ajudou em nada."*

> P15 - *"Em uma tentativa, gerou formas de fazer um protótipo em vez de sugerir quais ferramentas de inspeção utilizar."*

Houve ainda situações em que os participantes usaram a recomendação mais de uma vez para a mesma imagem e obtiveram respostas diferentes, o que afetou a confiança dos participantes na ferramenta.

> P20 - *"Algumas vezes ela muda suas respostas, o que faz com que não seja confiável para reprodução."*

De maneira geral, estes resultados mostram que os participantes consideraram a ferramenta útil e têm a intenção de usá-la. Porém, indicaram que ela necessita de ajustes para corrigir defeitos de funcionamento, de usabilidade e de recomendações de instrumentos de inspeção.

6.3 Considerações Finais

A avaliação preliminar do artefato de software GUIPFI, realizada através da sua aplicação no contexto da academia por estudantes de Engenharia de Software, mostrou que, embora tenham relatado algumas dificuldades durante seu uso, os participantes consideraram a ferramenta útil e fácil de utilizar, demonstrando intenção em utilizá-la para realizar inspeções funcionais de protótipos de interface de usuário.

No entanto, alguns pontos do resultado merecem atenção. Em relação aos resultados do Recomendador, verificou-se uma grande quantidade de recomendações de heurísticas de outras fontes não presentes no catálogo, além de recomendações sem nenhum dos instrumentos de inspeção desenvolvidos neste estudo. Além disso, alguns participantes indicaram como fator de diminuição da confiabilidade na ferramenta a variação de respostas nas recomendações de instrumentos de inspeção para o mesmo protótipo. Esta variabilidade já foi relatada na literatura por Zhou et al. (2024), que mostraram em seu estudo sobre a consistência de LLMs que o mesmo *prompt* pode levar a respostas diferentes dependendo da forma como é formulado, mesmo em modelos maiores e com refinamento de instrução.

Estes resultados indicam a necessidade de refinamento do Recomendador visando restringir as recomendações aos instrumentos de inspeção funcional (CHIF-E e UIProtoCheck), através de ajustes na técnica de *prompting* ou na configuração do parâmetro da temperatura do LLM. Ainda assim, a ferramenta oferece outros meios para auxiliar a seleção de instrumentos além da recomendação, tais como a exibição de informações relacionadas às funcionalidades ao selecioná-las ou através de *hiperlinks* para os sites do CHIF-E e do UIProtoCheck.

Além das considerações citadas acima, foram identificados alguns fatores que poderiam ameaçar a validade dos resultados e foram tomadas ações para mitigá-los quando aplicável. Primeiramente, o fato do estudo ter sido realizado com estudantes poderia configurar uma ameaça à representatividade da amostra em relação a profissionais da indústria de software. Como apontado por Falessi et al. (2018), participantes inexperientes podem apresentar resultados significativamente diferentes daqueles de profissionais da indústria de software. No entanto, a pesquisa de caracterização dos participantes mostrou que a maioria deles relatou

já possuir experiência profissional em desenvolvimento de software. Além disso, conforme conclusões de Salman et al. (2015), estudantes podem ser considerados participantes válidos em experimentos de engenharia de software, especialmente em fases iniciais de pesquisa ou quando se avaliam novas técnicas.

Outro fator que pode constituir uma ameaça à validade dos resultados é que os participantes poderiam realizar a atividade apenas para cumprir tarefas da disciplina, sem que houvesse um empenho em simular atividades de inspeção reais. Para mitigar esta ameaça, a atividade foi realizada de maneira anonimizada e apenas com participantes voluntários. O professor da disciplina não interferiu na execução e nos resultados do estudo, e a atividade não foi considerada na avaliação de desempenho na disciplina.

Finalmente, outro fator de ameaça à validade identificado foi em relação aos protótipos inspecionados. Eles foram elaborados pelos estudantes no decorrer da disciplina, com o uso de LLMs, e correspondiam a projetos de aplicações fictícias idealizadas pelos participantes para a realização das atividades. Estes projetos, contudo, foram considerados suficientemente adequados para esta iteração do ciclo de design, que consistiu em uma avaliação preliminar da ferramenta GUIPFI. Após correções de defeitos e ajustes na ferramenta, um novo estudo de avaliação da GUIPFI foi conduzido, desta vez com profissionais da indústria utilizando protótipos de aplicações reais. Seus resultados são apresentados no Capítulo 7.

Capítulo 7

GUIPFI: Estudo de Campo na Indústria de Software

Os resultados da terceira iteração do Ciclo de Design, que avaliou a GUIPFI no contexto da academia, indicaram que a ferramenta foi considerada útil e de fácil utilização. Desta forma, procedeu-se à próxima atividade da metodologia DSR, que consistiu na utilização da ferramenta GUIPFI por profissionais da indústria de desenvolvimento de software durante a inspeção de protótipos de interface de aplicações reais. As próximas seções descrevem as etapas de planejamento deste estudo, sua execução e os resultados obtidos.[1]

7.1 Planejamento

A realização da avaliação da GUIPFI no contexto da indústria foi realizada por profissionais atuantes no desenvolvimento de software, selecionados por conveniência (Baltes and Ralph, 2022). Visando refletir o público-alvo da ferramenta, foram convidados participantes com experiência em engenharia de requisitos, design e testes.

Os materiais utilizados no estudo consistiram em: (i) Termo de Consentimento Livre e Esclarecido (TCLE), (ii) carta-convite para participação no estudo,

[1]Pesquisa aprovada pelo Comitê de Ética em Pesquisa - CEP da Universidade Federal do Amazonas - UFAM, sob o número CAAE: 82963224.1.0000.5020, em 18/07/2024, Parecer Consubstanciado N. 7.072.722, de 12 de setembro de 2024.

(iii) instruções sobre a atividade de inspeção, (iv) a ferramenta GUIPFI e (v) um questionário de caracterização dos participantes e avaliação da ferramenta GUIPFI.

A aceitação da ferramenta GUIPFI foi avaliada através dos construtos Utilidade Percebida, Facilidade de Uso Percebida, Intenção de Uso e Qualidade do Resultado. Estes construtos são originários do modelo TAM de avaliação de aceitação de tecnologias (Venkatesh and Bala, 2008) e são os mesmos utilizados no estudo de avaliação da GUIPFI na academia, razão pela qual o questionário previamente utilizado foi mantido (Tabelas 6.2, 6.3 e 6.4).

Antes da realização do estudo, foram feitos ajustes nas heurísticas e melhorias na ferramenta a partir dos resultados obtidos no estudo de avaliação da GUIPFI no contexto da academia. As próximas seções apresentam as melhorias realizadas na especificação das heurísticas do CHIF-E, na usabilidade, no recurso de recomendação de instrumentos de inspeção e nos relatórios da GUIPFI.

7.1.1 Melhorias nas heurísticas do CHIF-E

Alguns participantes relataram ter dificuldade em entender os itens dos *checklists* presentes nas heurísticas. Desta forma, foram realizados ajustes nos termos utilizados, visando facilitar a compreensão das heurísticas. Foram atualizadas as Funcionalidades Básicas *Alteração, Busca, Cadastro, Galeria, Lista, Mestre-detalhe, Upload de Arquivo e Visualização*, além das Funcionalidades específicas *Home e Perfil de Usuário*. Nelas, foram modificados termos que se referem à especificação e a entidades do domínio da aplicação. O termo *atributo* foi substituído pelos termos *propriedade* e *característica*. Já os itens que se referem a uma entidade do domínio da aplicação agora usam a expressão *entidade que faz parte da especificação da aplicação*.

A Tabela 7.1 ilustra as modificações em requisitos do *checklist* da heurística *Lista*:

Referências a conjuntos, remanescentes da primeira versão das heurísticas, também foram reescritas, como ilustrado na Tabela 7.2 com um item do *checklist* da heurística *Câmera*:

Tabela 7.1: Exemplos de ajustes de termos e expressões na especificação das heurísticas - Lista

Antes	**Depois**
"O protótipo apresenta uma lista de objetos de uma entidade do domínio da aplicação."	*"O protótipo apresenta uma lista de objetos de uma entidade que faz parte da especificação da aplicação."*
"Cada item da lista apresenta valores de atributos do objeto."	*"Cada item da lista apresenta valores que ilustram características ou propriedades dos objetos."*

Tabela 7.2: Exemplo de ajustes de termos e expressões na especificação das heurísticas - Câmera

Antes	**Depois**
"Opcionalmente, o protótipo pode apresentar um botão para habilitar ou desabilitar o flash, cujo ícone pertence a um conjunto de ícones usados para representar botões de flash."	*"Opcionalmente, o protótipo pode apresentar um botão para habilitar ou desabilitar o flash com um ícone que representa botões de flash."*

7.1.2 Melhorias na GUIPFI

Com o objetivo de reduzir possíveis interferências da ferramenta nos resultados da avaliação, foram realizadas modificações decorrentes das observações feitas durante o estudo anterior, conduzido no ambiente acadêmico. As análises das interações dos participantes e dos relatórios gerados pela GUIPFI permitiram identificar defeitos e limitações de usabilidade e desempenho, que poderiam comprometer a interpretação dos resultados na nova etapa. Assim, foram implementados os seguintes ajustes e correções antes da realização do estudo de avaliação com profissionais da indústria de software:

- Correção de perda de dados ao adicionar ou remover *checklists*;
- Correção de perda de dados de inspeção na geração do relatório;

- Remoção de botões de Editar/Parar edição. Exibir campos diretamente no relatório;
- Geração de um arquivo único com o relatório;
- Corrigir defeito de rolagem da tela ao preencher o último item do *checklist*.

Além disso, foram realizadas melhorias na apresentação dos dados do relatório, que incluem a imagem inspecionada e a exibição dos defeitos registrados em forma de tabela. A Figura 7.1 ilustra um relatório de inspeção com as melhorias de apresentação.

Dados Gerais do 1° ciclo de registros

Nome do Projeto: Impacto Financeiro
Nome da Interface: Edicao de Impacto FInanceiro
Arquivo de Imagem: EdicaoImpactoFinanceiro.png - 105725 bytes

Imagem do Protótipo 1: EdicaoImpactoFinanceiro.png - 105725 bytes

Dados do Checklist do 1° ciclo de registros

ID	Item	Categoria do requisito	O protótipo do usuário atende ao requisito?	Descrição do defeito (se houver)
Os itens abaixo possuem relação com a funcionalidade: Cadastro				
11	O protótipo apresenta um formulário de entrada de dados com um ou mais campos cujos valores não estão preenchidos.	Obrigatório	Sim	Não foi atribuído valor ao campo Título.
12	Os campos do formulário representam os atributos da entidade no domínio da aplicação.	Obrigatório	Sim	
13	O formulário tem um botão de cadastro que tem um rótulo que está em um conjunto como este: (criar, enviar, salvar, solicitar inclusão, adicionar, encaminhar, finalizar, confirmar, cadastrar, gerar), ou que tem um ícone que representa a ação de cadastrar.	Obrigatório	Sim	

Figura 7.1: GUIPFI - Relatório de inspeção gerado pela ferramenta.

7.1.3 Ajustes no prompt do Recomendador

Com o objetivo de aumentar a precisão e a relevância das recomendações geradas pelo componente Recomendador, foram realizadas modificações no prompt enviado ao modelo de linguagem (LLM). Essa medida visou reduzir a ocorrência de

respostas imprecisas ou não relacionadas ao conteúdo do catálogo, observadas na avaliação anterior. O ajuste do prompt teve como finalidade refinar o contexto de entrada, orientando o modelo a priorizar heurísticas que integram o CHIF-E ou o UIProtoCheck e a estabelecer correspondências mais coerentes com as funcionalidades descritas nos protótipos.

Na parte inicial, foi introduzido um início amigável, seguido de uma explicação sobre as informações que serão fornecidas. Também foi acrescentada uma melhor conceituação de inspeção funcional e diferenciação de inspeção de usabilidade.

> *Olá, chat! Vou descrever agora um conjunto de heurísticas de funcionalidades recorrentes.*
> *Estas heurísticas descrevem funcionalidades comumente encontradas em sistemas de informação. Elas são usadas para realizar a inspeção funcional de protótipos de interface de usuário de aplicações.*
> *A inspeção funcional verifica se uma funcionalidade está corretamente representada em um protótipo de interface.*
> *A inspeção funcional não leva em consideração problemas de usabilidade.*
> *As heurísticas que descrevem funcionalidades recorrentes estão divididas em dois grupos: Funcionalidades Básicas e Funcionalidades Específicas.*
> *O grupo Funcionalidades Básicas descreve funcionalidades genéricas que existem em sistemas de informação e aplicações diversas. Já o grupo Funcionalidades Específicas descreve funcionalidades mais complexas existentes em domínios de aplicação específicos, tais como comércio eletrônico ou redes sociais.*
> *As Funcionalidades Básicas são: Alteração, Busca, Cadastro, Galeria, Lista, Menu, Mestre-Detalhe, Upload de Arquivo e Visualização.*
> *As Funcionalidades Específicas são: Autenticação, Câmera, Carrinho de Compra, Chat, Configurações, Editor de Imagens, Finalizar Compra, Home, Mapas, Media Player, Notícias, Página de Produto, Perfil de Usuário, Quiz, Registro de Conta, Tutorial, Videoconferência."*

A segunda parte que compõe o *prompt*, com a descrição resumida dos instrumentos de inspeção, permaneceu igual. Abaixo, um exemplo do resumo da

funcionalidade Videoconferência. Após a descrição resumida das heurísticas, o *prompt* agora inclui também uma explicação sobre o UIProtoCheck e uma definição do termo *instrumentos de inspeção funcional.*

> *A funcionalidade Videoconferência Apresenta os elementos de interface que representam uma chamada de vídeo entre pessoas.Um protótipo de videoconferência representa um encontro entre pessoas em diferentes locais mediado por um computador, onde recebem e transmitem dados de áudio e vídeo.Uma situação de uso desta funcionalidade ocorre quando um usuário participa de uma reunião de trabalho usando o aplicativo de videoconferência de sua empresa.*
> ...
> *O checklist genérico UIProtoCheck é indicado para inspeções funcionais de protótipos de interface quando a funcionalidade representada na imagem não corresponde a nenhuma das funcionalidades descritas no conjunto de heurísticas de funcionalidades recorrentes.*
> *A partir de agora, vamos chamar o conjunto de heurísticas de funcionalidades recorrentes e o UIProtoCheck de instrumentos de inspeção funcional.*

A terceira parte do *prompt* permanece a mesma, informando a imagem inspecionada e os nomes do projeto e do protótipo, caso tenham sido informados:

> *Considere a seguinte tela de uma aplicação: <Imagem passada>*;
> *Se o nome do projeto for informado, ele é incluído no prompt:
> *O nome do projeto é: <nome do projeto>*
> *Se o nome do protótipo for informado, ele é incluído no prompt:
> *O nome do protótipo é: <nome do protótipo>*

A quarta parte do *prompt* agora solicita recomendações de *instrumentos de inspeção* e solicita que a recomendação se limite aos instrumentos previamente descritos.

Agora atue como um recomendador de instrumentos de inspeção funcional de protótipos de interface. Considere os instrumentos de inspeção descritos e a imagem informada.
Indique quais os instrumentos de inspeção mais relevantes para inspecionar a imagem. Limite a recomendação aos instrumentos de inspeção funcional descritos.

7.2 Execução

Na execução do estudo, foram enviados convites via e-mail a 18 profissionais atuantes na indústria de desenvolvimento de software, selecionados por conveniência. O convite (Apêndice F) descreveu como os profissionais iriam participar do estudo: utilizando a ferramenta GUIPFI para inspecionar até três protótipos de interface de projetos de aplicações reais de software dos quais tivessem participado e respondendo a um questionário ao final após a realização das inspeções. O convite também informava o tempo estimado do estudo, bem como o caráter voluntário e anônimo da participação.

O questionário respondido ao final do estudo continha, além das perguntas de avaliação da aceitação da tecnologia (TAM), o Termo de Consentimento (Apêndice B), onde foram especificados os objetivos do estudo, os benefícios, riscos e detalhes de anonimato de participantes e dados do estudo.

Anexo ao convite, foram enviadas instruções para a realização do estudo (Apêndice F). As instruções continham os passos de utilização da ferramenta, que consistiam em: (1) informar o nome do projeto, o nome do protótipo e a imagem para a inspeção, (2) escolher quais instrumentos de inspeção utilizar, com auxílio opcional da funcionalidade de recomendação, (3) responder os *checklists* dos instrumentos de inspeção selecionados para cada imagem, (4) salvar os dados para geração do relatório e (5) fazer o download do relatório ao final do estudo. Aos participantes também foi solicitado que enviassem os relatórios gerados pela GUIPFI para a realização da análise estatística durante a avaliação da ferramenta.

Dez profissionais aceitaram o convite e participaram do estudo, sendo quatro analistas de software, dois designers de UI/UX, dois desenvolvedores de software

e dois testadores de software. Todos os desenvolvedores afirmaram trabalhar atualmente no desenvolvimento de aplicações web. Ao final do estudo, foram obtidas 29 inspeções de protótipos. A Tabela 7.3 apresenta as informações de ocupação, tempo de experiência e faixa etária dos profissionais que participaram do estudo. Já a Tabela 7.4 descreve a finalidade das empresas onde trabalham os participantes do estudo.

Tabela 7.3: Ocupação, tempo de experiência e faixa etária dos participantes.

ID	Tempo de Experiência	Faixa Etária	Ocupação	Empresa
PI1	>30	51 a 60	Analista	E1
PI2	10 a 15	41 a 50	Analista	E1
PI3	15 a 20	41 a 50	Designer de UI/UX	E2
PI4	20 a 25	41 a 50	Analista	E2
PI5	25 a 30	41 a 50	Desenvolvedor	E3
PI6	1 a 5	20 a 30	Analista	E1
PI7	10 a 15	31 a 40	Desenvolvedor	E4
PI8	1 a 5	31 a 40	Testador	E5
PI9	5 a 10	20 a 30	Testador	E6
PI10	5 a 10	31 a 40	Designer de UI/UX	E7

Tabela 7.4: Caracterização das empresas onde os participantes trabalham.

ID Empresa	**Descrição**
E1	Empresa de economia mista que desenvolve soluções de tecnologia da informação e comunicação
E2	Instituição pública de defesa dos direitos sociais e individuais
E3	Secretaria de finanças da administração municipal
E4	Instituto de pesquisa, desenvolvimento e inovação
E5	Instituto de pesquisa, desenvolvimento e inovação
E6	Plataforma de cupons de descontos
E7	Empresa de desenvolvimento de tecnologia robótica

As faixas etárias dos participantes do estudo variam de 20 a 60 anos. Já o seu tempo de experiência na indústria de software vai desde um ano até mais de 30 anos. As Figuras 7.2 e 7.3 apresentam graficamente estas informações.

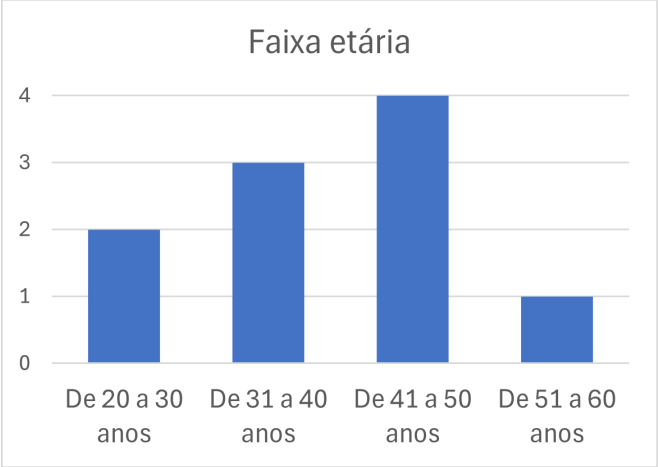


Figura 7.2: Faixa etária dos participantes da indústria.

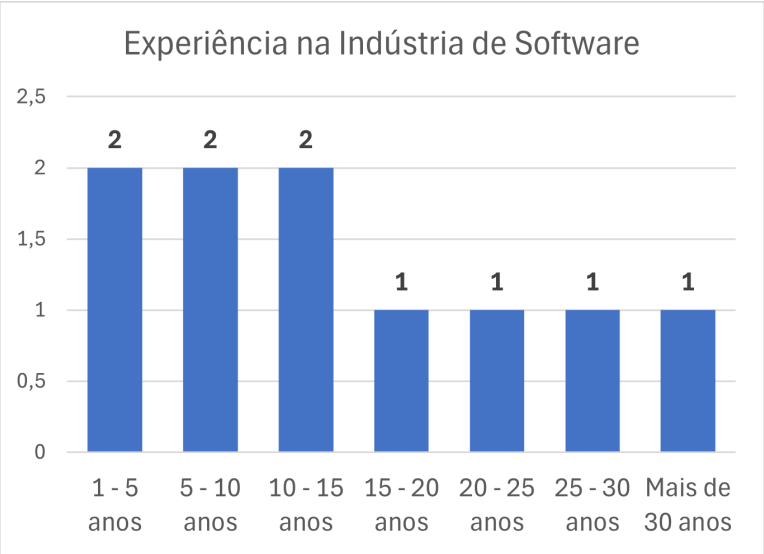


Figura 7.3: Tempo de experiência dos participantes da indústria.

7.3 Resultados

Os relatórios de inspeção obtidos nesta segunda fase de avaliação foram analisados estatisticamente utilizando os mesmos critérios definidos na primeira fase. Foi analisado para cada protótipo o uso ou não do Recomendador, a aceitação das recomendações e o uso de múltiplos instrumentos de inspeção. Além disso, visando avaliar se o refinamento do *prompt* resultou na melhora da qualidade das

recomendações, foi avaliado se o Recomendador recomendou somente as heurísticas do CHIF-E e o UIProtoCheck. A Figura 7.4 apresenta o resultado da análise estatística.

	Quant.	%
Total de inspeções	29	100%
Inspeções que usaram recomendação	22	75,9%
Inspeções que aceitaram alguma das recomendações	20	90,9%
Inspeções que usaram mais de um instrumento de inspeção	25	86,2%
Recomendações com o UIProtoCheck	0	0,0%
Inspeções que usaram o UIProtoCheck	3	10,3%
Recomendações sem nenhuma das funcionalidades ou o UIProtoCheck	0	0,0%
Recomendações de heurísticas não presentes no catálogo	0	0,0%

Figura 7.4: Ciclo 3 - Análise estatística das inspeções realizadas por participantes da indústria.

Os resultados mostraram uma pequena diminuição no percentual de participantes que utilizaram a recomendação de instrumentos de inspeção (de 82,3% para 75,9%), o que pode indicar uma maior facilidade de escolha de instrumentos por profissionais mais experientes. Por outro lado, das 22 inspeções que usaram recomendações, 20 aceitaram as sugestões (90,9%) o que pode indicar uma melhoria na qualidade do resultado do componente Recomendador.

Também houve um grande aumento no percentual de inspeções que usaram mais de um instrumento, de 43,8% na avaliação na academia para 86,2% na indústria.

Com o refinamento do *prompt* de maneira a torná-lo mais objetivo, não foram geradas recomendações sem as funcionalidades do CHIF-E ou heurísticas de usabilidade. No entanto, nenhuma das sugestões do Recomendador indicou o UIProtoCheck. Ainda assim, o *checklist* de propósito geral foi usado na inspeção de três protótipos (10,3% das inspeções).

A avaliação da aceitação da ferramenta GUIPFI, através de construtos do TAM utilizados no questionário, indicou uma boa aceitação pela maioria dos

profissionais, tendo havido uma melhor aceitação em relação à primeira fase. Isso ocorreu possivelmente devido às melhorias realizadas no relatório e na interface da ferramenta. Dois dos profissionais, um da área de design e um da área de testes, deram classificações mais baixas do que as dos demais participantes.

Em relação à avaliação da Utilidade Percebida da ferramenta pelos participantes, nove dos dez participantes concordaram que a ferramenta melhorou seu desempenho (U1), sua eficácia (U3) e a consideraram útil na inspeção de protótipos de interface (U4). A avaliação do construto de Utilidade Percebida é ilustrada na Figura 7.5.

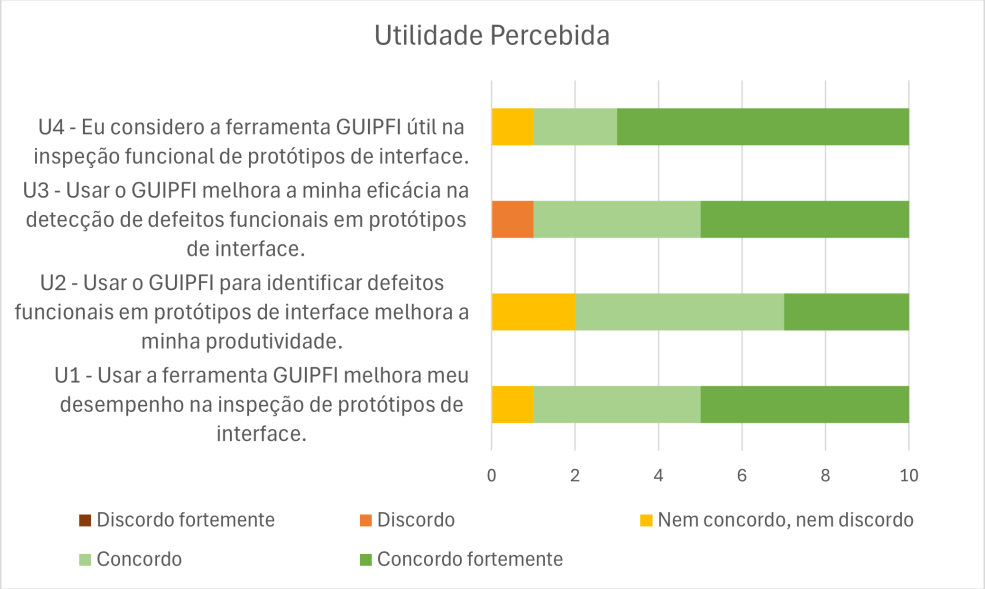


Figura 7.5: Utilidade percebida - participantes da indústria.

Um dos participantes ficou neutro quanto à melhoria da produtividade (U2), tendo concordado em relação à utilidade nos demais aspectos avaliados. Apenas o participante PI3 discordou quanto à melhoria da sua eficácia na detecção de defeitos (U3), ficando neutro nos demais itens avaliados na percepção de utilidade.

Quando questionados sobre quais situações a ferramenta foi útil (questão U5), a maioria dos participantes apontou que ela foi útil para mostrar elementos ausentes nos protótipos, como ilustra a resposta dos participantes PI1 e PI6.

> PI1 - *"A ferramenta me alertou no protótipo de edição que um dos campos não estava preenchido. Quando vou apresentar um protótipo*

a um usuário, muitas vezes essa falta de informação atrapalha o entendimento dele."

PI6 - *"Foi útil em identificar, no meu contexto, campos com rótulos que não representam a informação que será pesquisada ou botões sem descrição. No dia a dia, esse tipo de inconsistência passa batido."*

Outra utilidade indicada pelos profissionais foi a antecipação da detecção de defeitos, evitando que eles se propagassem para as fases posteriores de desenvolvimento do software, ilustrada a seguir pelos comentários dos participantes PI2 e PI9.

PI2 - *"Ter um checklist que nos direciona na elaboração do protótipo nos ajuda a identificar falhas antes de requisitos e funcionais antes de passarmos a demanda para o desenvolvedor."*

PI9 - *"Foi bem útil, pois lista todas as informações obrigatórias para cada tipo de interface, permitindo assim uma avaliação prévia do protótipo criado pelo time de UI/UX."*

Em relação à Facilidade de Uso Percebida, a maioria dos participantes também relatou facilidade em seu uso, conforme ilustra a Figura 7.6. Oito participantes consideraram a ferramenta clara e compreensível (F1), nove afirmaram que ela não exige muito esforço mental (F2), seis consideraram a ferramenta fácil de utilizar (F3) e sete acharam a ferramenta fácil de usar para inspecionar protótipos de interface (F4). Três participantes ficaram neutros sobre a ferramenta ser fácil de usar (F3) e seu uso para inspecionar protótipos (F4). Já o participante PI3 discordou fortemente das afirmações F1 e F3 e discordou da F2.

Ao serem questionados sobre as principais dificuldades encontradas no uso da ferramenta, os participantes relataram alguns problemas relacionados à visualização de elementos. Um deles foi a exibição da caixa de seleção de instrumentos de inspeção, que fica localizada na parte inferior da tela e que não era facilmente percebida pelos participantes, como ilustra a resposta do participante PI9 abaixo. Além disso, esta caixa de seleção ficava fora da área visível da tela após o carregamento da imagem para inspeção, necessitando que o usuário rolasse a tela

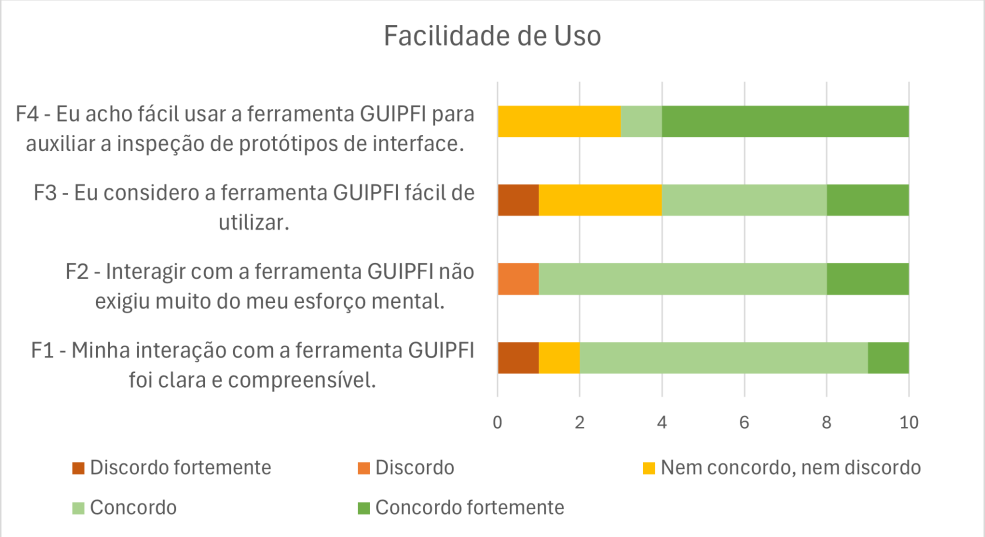


Figura 7.6: Facilidade de uso percebida - participantes da indústria.

para visualizá-la, como relatou o participante PI3. Alguns participantes também relataram ter demorado a perceber os itens dos *checklists* dos instrumentos de inspeção selecionados, carregados na parte inferior da tela, como ilustra a resposta do participante PI1.

> PI9 - *"O campo de selecionar o instrumento de inspeção não é bem claro que é um campo. No início, achei que era rodapé da página."*

> PI3 - *"Na interface apresentada, tive dificuldade em perceber elementos ocultos devido à necessidade de rolagem, destacando um problema de encontrabilidade (findability) em UX."*

> PI1 - *"Não percebi que os itens que eram para eu responder abriam embaixo e que eu tinha que rolar a tela para eles aparecerem."*

Outras dificuldades de uso relatadas por alguns dos profissionais envolvem a compreensão da sequência de ações necessárias para usar a ferramenta, como é o caso do participante PI4, e a similaridade dos botões de ação e *feedbacks* do sistema pouco perceptíveis, relatadas pelo participante PI10.

> PI4 - *"Não ficou muito claro no início se deveria recarregar a tela para um novo protótipo. Usando, ficou claro que, mesmo salvando,*

se recarregar a tela, os dados eram perdidos, então, para rodar uma nova inspeção sem perder os dados da inspeção anterior, é necessário carregar uma nova imagem, clicar novamente em sugerir nova inspeção e remover manualmente os instrumentos de inspeção anteriores."

PI10 - *"Alguns feedbacks do sistema foram rápidos e nem percebi que aconteceram. (...) Os botões de ação são muito similares em termos de design, o que me exigiu uma leitura mais lenta."*

O participante PI8 relatou dificuldades relacionadas à especificação das heurísticas que compõem o CHIF-E e ao UIProtoCheck. Por exemplo, os itens do *checklist* genérico mencionam um "documento de referência", mas em nenhum local da aplicação o usuário é informado de que necessitaria de alguma especificação complementar para inspecionar o protótipo se o UIProtoCheck fosse escolhido:

PI8 - *"Em relação ao checklist genérico: "A funcionalidade representada no protótipo corresponde a uma funcionalidade especificada no documento de referência?", qual é o documento de referência? É o do documento de requisitos? É a proposta do projeto?"*

Os requisitos dos *checklists* das heurísticas do CHIF-E podem ser obrigatórios ou opcionais, indicando se os requisitos das funcionalidades precisam ser necessariamente atendidos pelo protótipo. Durante a inspeção de protótipos com o uso da ferramenta, no entanto, mesmo estes itens opcionais precisam ser marcados com uma das três alternativas disponíveis: Sim, Não e Não se aplica, para evitar que o relatório seja gerado em branco. Esse funcionamento causou confusão na compreensão do significado de itens opcionais.

PI8 - *"O campo da categoria opcional é obrigatório a marcação do item, então eu não entendi as categorias obrigatórias e opcionais."*

Na avaliação da Intenção de Uso, sete dos dez participantes demonstraram intenção de usar a ferramenta, com apenas o participante (PI3) discordando quanto à intenção de usá-la nos próximos meses (questão I3). A Figura 7.7 apresenta um gráfico das respostas dos profissionais às questões I1, I2 e I3.

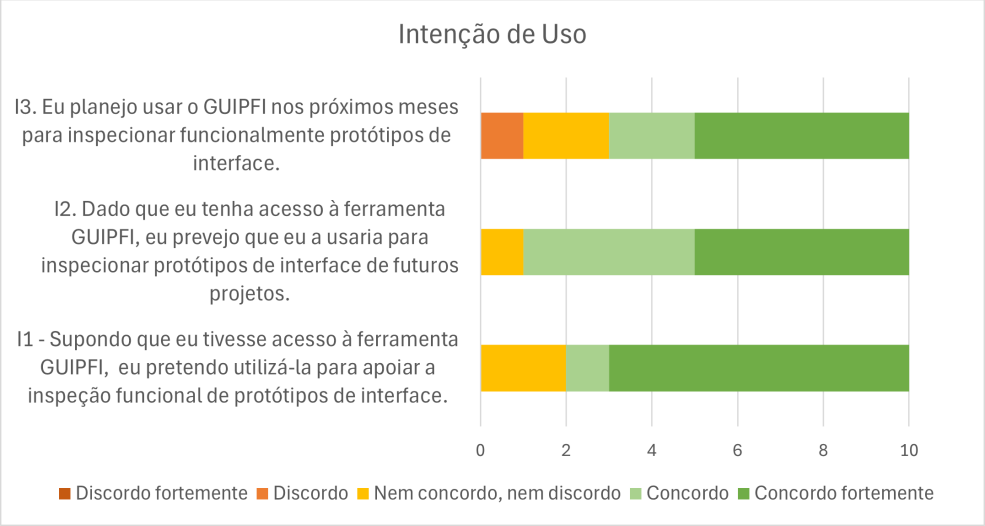


Figura 7.7: Intenção de uso da GUIPFI - profissionais da indústria.

A análise das respostas da questão I4, sobre as situações em que usariam a GUIPSI durante o processo de desenvolvimento, mostrou que os participantes a utilizariam em diversas fases, desde o início da prototipação, passando pela validação com o usuário e pelos testes de interface.

Os participantes PI4, PI6 e PI7 utilizariam a GUIPFI no início de um projeto, durante a prototipação e os testes de novas funcionalidades.

> PI4 - *"Usaria. Durante a fase de protótipo e durante da fase de testes de novas funcionalidades."*

> PI7 - *"Usaria principalmente no início de um projeto, enquanto o protótipo da interface ainda não está finalizado, assim ajudaria a definir melhor o que falta."*

Os participantes PI1, PI2, PI5 e PI9 afirmaram que usariam a ferramenta após a finalização da fase de prototipação, antes do início da codificação.

> PI2 - *"Usaria antes de enviar uma issue para o Desenvolvedor com o objetivo de minimizar possíveis erros ou falta de compreensão por apresentar um protótipo incompleto."*

> PI9 - *"Eu usaria sim a ferramenta principalmente para validar os protótipos criados pelo time de design, pois atualmente no time em que atuo não é feita essa inspeção e acredito que me auxiliaria a melhorar as interfaces antes mesmo de serem desenvolvidas."*

O participante PI1 também a utilizaria para analisar protótipos preliminares utilizados para validar a aplicação com o usuário.

> PI1 - *"Essa ferramenta me seria útil na prototipação que utilizo para validar minha aplicação com o usuário. (...) Gostei mesmo. Se entrar em produção, eu uso. É rápido, simples e ajuda muito... Esses protótipos na hora que estou desenvolvendo já estão na mão, então é muito simples... Joga ali dentro do moedor e já vê se faltou algo. Ótimo."*

Alguns participantes, por outro lado, afirmaram que talvez venham a utilizar a ferramenta, a depender do alinhamento das funcionalidades do projeto com as disponíveis na ferramenta ou de melhorias na interface. Por exemplo, os participantes PI3 e PI10 afirmaram que o fariam durante o desenvolvimento de protótipos de média e alta fidelidade ou para validar o protótipo com usuários da aplicação.

> PI10 - *"O momento ideal seria no desenvolvimento de protótipos de média/alta fidelidade, pois nessa etapa a estrutura dos textos e layout já estariam mais avançados."*

> PI3 - *"Melhorando a interface seria útil para aplicar com usuários na fase de validação do protótipo."*

Já o participante PI8 afirmou que talvez utilizasse a GUIPFI na etapa de teste estático de protótipos de interface.

> PI8 - *"Talvez eu use a ferramenta na hora de realizar o teste estático, onde avalio o protótipo de interface prototipada pela designer."*

O último construto, Qualidade de Saída, avaliou especificamente o recurso de Recomendação de instrumentos de inspeção. As recomendações foram avaliadas positivamente por todos os participantes, tendo o participante PI5 se manifestado

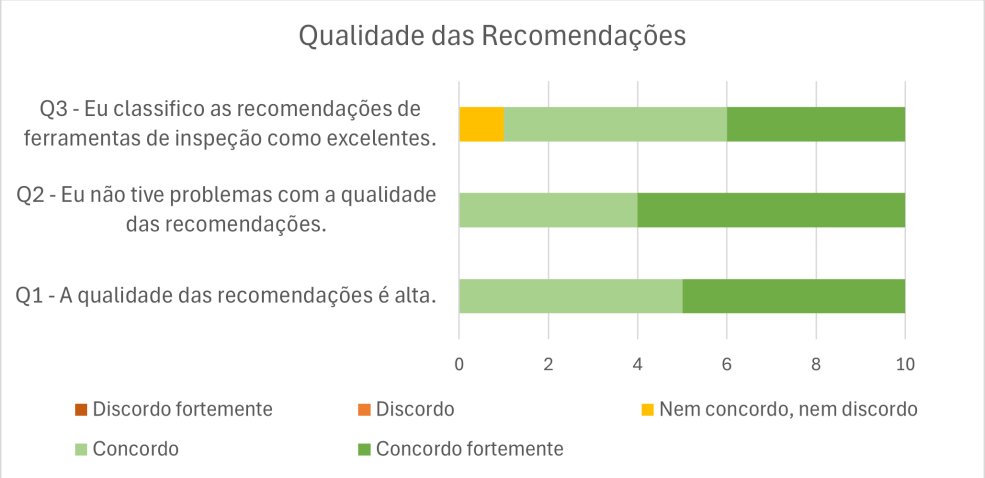


Figura 7.8: Avaliação da qualidade dos resultados do componente Recomendador - profissionais da indústria.

de maneira neutra em relação à excelência das recomendações (Q3). A Figura 7.8 ilustra esses resultados.

Seis profissionais consideraram as recomendações de instrumentos de inspeção adequadas para a imagem informada (PI1, PI2, PI3, PI4, PI8 e PI9). Cinco dos participantes mencionaram que o recurso de recomendação os auxiliou na escolha de instrumentos de inspeção (PI5, PI7, PI8, PI9 e PI10). Já o participante PI6 achou que a resposta poderia ter sido um pouco melhor para um dos protótipos inspecionados.

> PI3 - *"As sugestões foram essenciais, pois estavam fáceis de encontrar e com sugestões relacionadas ao protótipo da imagem."*

> PI9 - *"Eu preenchi os campos e sempre utilizei a funcionalidade de sugerir ferramentas de inspeção para auxiliar nos instrumentos que poderia utilizar para inspecionar os protótipos e as recomendações sempre foram muito assertivas."*

7.4 Ameaças à Validade

Esta seção apresenta os principais fatores que poderiam ameaçar a validade dos resultados obtidos no estudo de avaliação da GUIPFI no contexto da indústria

de software, junto às medidas que foram tomadas para mitigá-los.

A principal ameaça identificada refere-se ao tamanho reduzido da amostra de participantes provenientes da indústria de software, o que pode limitar a generalização dos resultados para outros contextos e populações, influenciando a validade externa do estudo. Uma amostra mais ampla possibilitaria uma compreensão mais aprofundada acerca da influência de fatores como área de atuação e tempo de experiência na aceitação da ferramenta. Para mitigar essa ameaça, buscou-se incluir profissionais de diferentes empresas e com perfis variados de ocupação, de modo a aumentar a diversidade de perspectivas e ampliar a representatividade dos resultados obtidos.

Outra ameaça potencial diz respeito ao baixo controle experimental sobre as condições do estudo. Diferentemente da segunda iteração do Ciclo de Design, não foram utilizados protótipos contendo defeitos previamente inseridos, o que reduziu o controle sobre as variáveis envolvidas e sobre a mensuração direta dos resultados, constituindo-se em uma ameaça à validade interna. Tal decisão metodológica foi intencional, pois objetivou-se aumentar o realismo do experimento e observar o uso espontâneo da ferramenta em contextos autênticos da indústria de software em detrimento de um controle mais rígido das variáveis (Stol and Fitzgerald, 2018).

Uma ameaça relacionada à validade de construto refere-se ao potencial viés de aceitação tecnológica por parte dos participantes, que poderiam manifestar predisposição favorável à ferramenta em função de interesse ou familiaridade com novas tecnologias. Tal viés poderia comprometer a correspondência entre os construtos teóricos pretendidos (percepção de utilidade e aplicabilidade) e as medições efetivamente realizadas. Para mitigar esse risco, os participantes foram orientados a realizar suas avaliações de forma imparcial e fundamentada em suas rotinas profissionais, de modo a assegurar que suas respostas refletissem percepções autênticas sobre a aplicabilidade da ferramenta em situações reais de trabalho.

Por fim, a limitação do tamanho da amostra e a ausência de análises estatísticas mais robustas configuram ameaças à validade de conclusão, uma vez que restringem a força estatística e a precisão das inferências feitas a partir dos dados coletados. Para reduzir o impacto dessa ameaça, as análises quantitativas foram

complementadas por evidências qualitativas, permitindo uma triangulação de dados que reforçou a consistência e a credibilidade das interpretações apresentadas.

7.5 Considerações Finais

A ferramenta GUIPFI foi considerada útil por analistas e desenvolvedores para identificar requisitos ausentes e antecipar a detecção de defeitos. Eles relataram intenção de usá-la no futuro em vários momentos da prototipação, desde as versões preliminares até a fase que antecede a implementação.

Por outro lado, um dos designers não viu muita utilidade na GUIPFI e concentrou sua avaliação na usabilidade da ferramenta. Um dos testadores avaliou aspectos mais amplos da ferramenta além dos quatro construtos do TAM. Sua avaliação envolveu também o fluxo de uso da ferramenta, a usabilidade e as características dos instrumentos de inspeção (CHIF-E e UIProtoCheck), como sinônimos e a indicação de itens opcionais nos *checklists*.

O componente Recomendador foi muito bem avaliado pelos participantes. Dentre os quatro participantes que não usaram o Recomendador em pelo menos uma das inspeções, três eram profissionais com mais de 15 anos de experiência na indústria. Isto pode ser um indício de que, quanto maior a experiência do profissional e a familiaridade com as funcionalidades especificadas, menos necessário será o uso do Recomendador, o que evidencia o caráter instrutivo da ferramenta. Todos os participantes conseguiram realizar a atividade proposta, embora muitos tenham tido dificuldade em perceber o componente de seleção de instrumentos de inspeção. As dificuldades relatadas pelos profissionais indicaram importantes pontos de melhoria na definição das heurísticas e na usabilidade da ferramenta.

Em comparação com a avaliação conduzida no contexto acadêmico (Capítulo 6), o estudo de campo na indústria apresentou resultados mais consistentes quanto à aceitação e à aplicabilidade da GUIPFI. Na academia, a ferramenta foi avaliada por estudantes de Engenharia de Software com pouca experiência profissional, o que contribuiu para uma maior dependência do componente Recomendador e para dúvidas na seleção dos instrumentos de inspeção mais adequados. Já na indústria, com participantes mais experientes e familiarizados com processos de desenvolvimento e validação de requisitos, observou-se uma redução no uso do

Recomendador (de 82,3% para 75,9%), acompanhada de um aumento expressivo no uso combinado de diferentes instrumentos de inspeção (de 43,8% para 86,2%), o que indica uma apropriação mais autônoma da ferramenta pelos profissionais.

Outra diferença relevante refere-se à qualidade das recomendações geradas e à percepção de utilidade da ferramenta. Enquanto no ambiente acadêmico foram observadas recomendações inconsistentes — algumas envolvendo heurísticas não pertencentes ao catálogo —, na indústria esse problema foi praticamente eliminado, possivelmente em razão do refinamento do prompt do Recomendador e da clareza maior das descrições funcionais nas versões aprimoradas do CHIF-E e do relatório da GUIPFI. Além disso, os profissionais avaliaram a ferramenta como útil para antecipar a detecção de defeitos funcionais e identificar requisitos ausentes, destacando seu potencial instrutivo e de apoio à comunicação entre áreas de design, requisitos e testes.

Por fim, a comparação entre os dois contextos evidencia a evolução progressiva do artefato ao longo do ciclo de DSR, demonstrando que as melhorias realizadas após a avaliação acadêmica – como o ajuste da interface, a ampliação das heurísticas e o aprimoramento do mecanismo de recomendação – resultaram em maior clareza, estabilidade e aceitação da ferramenta em um ambiente real de desenvolvimento. Essa evolução reflete o amadurecimento da GUIPFI como artefato científico e tecnológico, validando sua utilidade prática e sua relevância teórica para apoiar a inspeção funcional de protótipos de interface de usuário em diferentes contextos de aplicação.

Capítulo 8

Conclusão

A prototipação de interfaces de usuário é um recurso muito utilizado no desenvolvimento de software para explorar ideias e apresentar soluções, facilitando a comunicação no time de desenvolvimento e com o cliente. Protótipos de interface de usuário têm sido utilizados para auxiliar diversas atividades durante o processo de desenvolvimento de software, tais como a implementação, a elaboração de casos de teste e a implantação. Por este motivo, é importante que eles passem por etapas de garantia da qualidade, de maneira a antecipar a detecção de defeitos, diminuindo seus impactos em fases posteriores.

Pesquisas relacionadas apresentam diversas abordagens e ferramentas de auxílio à prototipação e à inspeção de protótipos de interface. Tais abordagens vão desde a sugestão de alternativas de protótipos, passando por inspeções semióticas, de usabilidade e verificações automatizadas para atender a regras de design. No entanto, as abordagens existentes para identificação de defeitos se concentram em validar aspectos de usabilidade ou de design de produtos de software, deixando de lado o aspecto funcional dos protótipos preliminares de interface de usuário.

Desta forma, a seguinte questão de pesquisa orientou este projeto: *"Como inspecionar protótipos de interface de usuário em relação à sua funcionalidade?"*. O objetivo deste projeto de pesquisa foi o de apoiar a inspeção de protótipos de interface de usuário com foco na funcionalidade, através da elaboração de instrumentos de inspeção que possibilitem antecipar a identificação de defeitos relacionados à funcionalidade.

Para atingir o objetivo da pesquisa, foram definidos os seguintes objetivos específicos:

1. Caracterizar funcionalidades comumente representadas em protótipos de interface de usuário em diferentes domínios de aplicação;

2. Elaborar instrumentos para inspeção de protótipos de interface com foco na sua funcionalidade;

3. Apoiar a identificação de defeitos funcionais através de ferramenta automatizada baseada nos instrumentos de inspeção desenvolvidos nesta pesquisa.

A metodologia de pesquisa adotada foi a DSR (Design Science Research), uma vez que os objetivos definidos visaram ao desenvolvimento de artefatos de apoio à inspeção funcional de protótipos de interface. A primeira atividade da DSR foi a realização de uma pesquisa com profissionais visando caracterizar o uso de prototipação na indústria de software. Este estudo revelou que a prototipação de interfaces de usuário é um recurso amplamente utilizado durante o processo de desenvolvimento de software nas empresas, mas ainda é realizada de maneira informal. Além disso, evidenciou-se a importância de que o papel da prototipação esteja claramente definido no processo de desenvolvimento e que os protótipos elaborados sejam incluídos nas atividades de inspeção, de forma a reduzir ciclos extras de prototipação e, consequentemente, custos de desenvolvimento.

Os resultados do estudo com profissionais da indústria corroboraram a necessidade de identificar defeitos funcionais em protótipos de interface o mais cedo possível, de maneira a evitar sua propagação para fases mais avançadas do processo de desenvolvimento e a possibilitar sua correção antes da fase de implementação.

Todos os objetivos específicos definidos nesta pesquisa foram atingidos através da realização de iterações no Ciclo de Design da metodologia DSR. Para atingir o primeiro objetivo específico, a primeira iteração do Ciclo de Design iniciou-se pela identificação de funcionalidades comumente representadas em protótipos de interface. Para isso, foi realizado um estudo experimental de elaboração de projetos de software com o uso de prototipação de interfaces, no qual os participantes simularam os papéis de cliente e desenvolvedor. Os protótipos obtidos neste estudo

foram analisados e categorizados em funcionalidades, sendo especificada para cada funcionalidade uma heurística descrevendo suas principais características. O resultado, um catálogo com 19 heurísticas de inspeção funcional (CHIF), foi avaliado e refinado através de um grupo focal com especialistas. Os resultados obtidos na avaliação do CHIF mostraram que o uso de heurísticas de design de funcionalidades conhecidas pode auxiliar o design de protótipos de interface, servindo como um guia que pode ser utilizado antes, durante e após o design de interface. Esta etapa atingiu o primeiro objetivo específico desta pesquisa.

Na segunda iteração do Ciclo de Design, o CHIF foi ampliado através da análise de um *dataset* de interfaces de software extraídas de lojas de aplicativos, visando torná-lo mais abrangente de maneira a ser utilizado na inspeção de protótipos de interface preliminares. Além da expansão do catálogo, foi desenvolvido um *checklist* de propósito geral (UIProtoCheck) para inspeção de protótipos de interface que representam funcionalidades não contempladas pelo catálogo de heurísticas. Ambos os instrumentos de inspeção foram avaliados em um estudo experimental que comparou seu desempenho na inspeção de protótipos de interface de projetos de aplicações desenvolvidos por alunos. Os resultados mostraram que ambos os instrumentos são eficazes na identificação de defeitos associados à funcionalidade, sendo cada um mais adequado na identificação de certos tipos de defeitos funcionais. Com o desenvolvimento destes instrumentos de inspeção, atingiu-se o segundo objetivo específico desta pesquisa.

Os resultados da segunda iteração do Ciclo de Design também revelaram a necessidade de auxílio à seleção de instrumentos de inspeção. Visando oferecer uma solução para esta necessidade e dar suporte à realização de inspeções funcionais, foi desenvolvida a GUIPFI, uma ferramenta web que apoia a realização de inspeções funcionais de protótipos de interface através da recomendação de instrumentos de inspeção para protótipos fornecidos, da montagem do formulário de inspeção a partir de instrumentos selecionados e da geração do relatório de inspeção. Para avaliar a GUIPFI, foi conduzido um estudo em duas fases, primeiramente com alunos de graduação em Engenharia de Software e pós-graduação em Informática e depois com profissionais da indústria de software. O objetivo destes estudos foi verificar se os instrumentos de inspeção desenvolvidos viabilizavam a detecção antecipada de defeitos funcionais em protótipos de interface e se a GUIPFI oferecia

suporte adequado à seleção de instrumentos e à realização de inspeções funcionais. Os resultados mostraram que os participantes consideraram a ferramenta útil para mostrar elementos ausentes nos protótipos e antecipar a detecção de defeitos, bem como revelaram intenção de usá-la caso esteja disponível. Eles também indicaram como pontos fortes a facilidade de uso e a sugestão de instrumentos de inspeção e sugeriram melhorias tais como a possibilidade de enviar mais de uma imagem por vez e um recurso para limpar os campos para iniciar uma nova inspeção. Os resultados deste estudo evidenciam a realização do terceiro objetivo específico.

Finalmente, respondendo à questão de pesquisa, este trabalho viabilizou a realização de inspeções de protótipos de interface em relação à sua funcionalidade através do desenvolvimento de instrumentos de inspeção especificamente voltados para este fim e de um artefato de software que faz uso destes instrumentos para oferecer suporte à realização de inspeções funcionais. O primeiro instrumento de inspeção, o CHIF-E, consiste em um catálogo de heurísticas de funcionalidades comumente representadas em protótipos de IU de sistemas de informação. O segundo instrumento de inspeção, o UIProtoCheck, é um *checklist* de propósito geral para inspecionar funcionalmente protótipos de interface que representam funcionalidades que não estão diretamente especificadas através de heurísticas no CHIF-E. Ambos os instrumentos de inspeção funcional integram a ferramenta GUIPFI no apoio à realização assistida de inspeções funcionais de protótipos de interface.

Diferentemente de abordagens de inspeção existentes (Islam et al., 2020; Yang et al., 2021; Duan et al., 2024), os instrumentos de inspeção utilizados na GUIPFI foram elaborados com foco nos requisitos funcionais que devem ser atendidos por protótipos que representam funcionalidades comumente encontradas em sistemas de informação. Os resultados do estudo de avaliação da GUIPFI mostraram que ela pode ser usada tanto no ambiente da academia, em disciplinas como Engenharia de Requisitos com prototipação ou Verificação, Validação e Testes, quanto na inspeção de interfaces de usuário por profissionais da indústria de software de diferentes perfis para melhorar a comunicação com clientes, reduzir o retrabalho e detectar precocemente defeitos relacionados à funcionalidade.

8.1 Contribuições da Pesquisa

A principal contribuição desta pesquisa para a base de conhecimento é uma abordagem de apoio a inspeções funcionais de protótipos de interface de usuário que utiliza instrumentos de inspeção especialmente projetados para este fim. Especificamente, as principais contribuições desta pesquisa são:

1. Evidências experimentais da necessidade de instrumentos de inspeção de protótipos de interface, obtidas através da pesquisa de caracterização de uso da prototipação na indústria de software (De Macedo et al., 2022);

2. O catálogo de heurísticas de inspeção funcional - CHIF-E, que descreve funcionalidades conhecidas, obtidas da análise de um conjunto de protótipos e posteriormente expandido com funcionalidades obtidas de um *dataset* rotulado em tópicos das funcionalidades representadas. O CHIF foi avaliado através de sua utilização em atividades de prototipação de funcionalidades conhecidas, realizadas por especialistas. Sua versão expandida, CHIF-E, foi avaliada em um experimento controlado, na inspeção de protótipos de interface de 8 projetos de software;

3. Um website com o catálogo de heurísticas, nos idiomas português e inglês, disponível em https://sites.google.com/view/design-heuristics;

4. O UIProtoCheck, um *checklist* genérico de inspeção funcional de protótipos de interface, desenvolvido com o objetivo de oferecer um instrumento de inspeção para protótipos que representam funcionalidades não especificadas no CHIF-E, disponível em https://sites.google.com/view/uiprotocheck;

5. Uma base de protótipos de interface categorizadas de acordo com as funcionalidades catalogadas;

6. Um método de inclusão de novas heurísticas no CHIF-E;

7. A ferramenta GUIPFI (Guided User Interface Prototype Functional Inspection), um artefato de software que implementa a abordagem apresentada, oferecendo suporte à inspeção funcional através das funcionalidades

de consulta de instrumentos de inspeção, recomendação de instrumentos de inspeção, registro de defeitos e relatório de inspeção. Disponível em https://guipfi.vercel.app.

8.1.1 Artigos Publicados

- De Macedo, G. T., Fontão, A., and Gadelha, B. (2022a). Prototyping in software quality assurance: A survey with software practitioners. In Proceedings of the XXI Brazilian Symposium on Software Quality, pages 1–10.
- De Macedo, G. T., Viana, L., and Gadelha, B. (2022b). Towards meaning in user interface prototypes. In ICEIS (2), pages 235–246.
- De Macedo, G. T., Fontão, A., and Gadelha, B. (2023). UIProtoCheck: A checklist for semantic inspection of user interface prototypes. In Proceedings of the XXXVII Brazilian Symposium on Software Engineering, SBES '23, page 485–490, New York, NY, USA. Association for Computing Machinery.
- De Macedo, G. T., Fontão, A., and Gadelha, B. (2024). Building soft skills through a role-play-based approach for requirements engineering remote education. Journal of the Brazilian Computer Society, 30(1):1–16.

8.2 Trabalhos Futuros

Esta pesquisa investigou e desenvolveu artefatos de apoio à inspeção funcional de protótipos de interface de usuário. Relacionadas a este tópico, apresentam-se diversas oportunidades de pesquisas futuras que podem expandir e solidificar os resultados existentes sobre inspeção funcional.

- **Ampliação do catálogo de heurísticas para novos domínios de aplicação** – Novas pesquisas podem ser conduzidas visando acrescentar novas funcionalidades ao CHIF-E, avaliando a sua aplicabilidade na identificação de defeitos em interfaces de domínios diversos dos de sistemas de informação. Essa expansão pode ser alcançada através da realização de novos estudos experimentais em domínios como sistemas embarcados, jogos ou sistemas

críticos, para identificar funcionalidades específicas destes domínios. Outra maneira de contribuir para a ampliação das funcionalidades do CHIF-E seria através da criação de uma plataforma de ecossistema de software onde pesquisadores e profissionais possam contribuir com novas heurísticas e exemplos de aplicação.

- **Automatização da identificação de novas funcionalidades** – Pesquisas futuras podem ser desenvolvidas visando automatizar as atividades de identificação de novas funcionalidades e especificação de heurísticas. Técnicas de mineração de dados, processamento de linguagem natural ou LLMs podem ser empregadas no desenvolvimento de mecanismos de mineração de bases de protótipos ou de capturas de telas de aplicações, visando identificar padrões que indiquem a existência de novas funcionalidades. Tais técnicas também podem ser aplicadas na identificação automática das características típicas de novas funcionalidades, apoiando a especificação de suas respectivas heurísticas.

- **Automatização da identificação de defeitos** – Novas pesquisas também podem explorar o potencial de LLMs para investigar se o desempenho de inspeções funcionais pode ser aprimorado através da automatização da identificação de defeitos e geração de relatórios de inspeção, integrando no mecanismo de verificação o catálogo de heurísticas ao contexto específico das aplicações.

- **Modelos integrados de inspeção de interfaces de usuário** – Finalmente, novas pesquisas podem ser conduzidas visando investigar como defeitos funcionais estão relacionados a defeitos de design e de usabilidade. Destas pesquisas podem emergir modelos integrados de inspeção de interfaces de usuário que promovam a melhoria da sua qualidade em diferentes aspectos.

Referências Bibliográficas

Akpinar, M. E. and Yeşilada, Y. (2017). Discovering visual elements of web pages and their roles: users' perception. *Interacting with Computers*, 29(6):845–867.

Baltes, S. and Ralph, P. (2022). Sampling in software engineering research: A critical review and guidelines. *Empirical Software Engineering*, 27(4):94.

Baños Díaz, G. and Zapata Del Río, C. M. d. P. (2018). A proposal of usability heuristics oriented to e-banking websites. In Marcus, A. and Wang, W., editors, *Design, User Experience, and Usability: Theory and Practice*, pages 327–345, Cham. Springer International Publishing.

Barkley, E. F., Cross, K. P., and Major, C. H. (2014). *Collaborative learning techniques: A handbook for college faculty.* John Wiley & Sons.

Bashir, M. S. and Farooq, A. (2019). Euhsa: Extending usability heuristics for smartphone application. *IEEE Access*, 7:100838–100859.

Bayazit, N. (2004). Investigating design: A review of forty years of design research. *Design issues*, 20(1):16–29.

Behutiye, W., Seppänen, P., Rodríguez, P., and Oivo, M. (2020). Documentation of quality requirements in agile software development. In *Proceedings of the Evaluation and Assessment in Software Engineering*, EASE '20, page 250–259, New York, NY, USA. Association for Computing Machinery.

Belikova, S. A., Rogozov, Y. I., and Sviridov, A. S. (2018). Method of user interface design based on semantic approach. In *Computer Science On-line Conference*, pages 311–318. Springer.

Bellomo, S., Nord, R. L., and Ozkaya, I. (2013). Elaboration on an integrated architecture and requirement practice: prototyping with quality attribute focus. In *2013 2nd international workshop on the twin peaks of requirements and architecture (TwinPeaks)*, pages 8–13. IEEE.

Bernal-Cárdenas, C., Moran, K., Tufano, M., Liu, Z., Nan, L., Shi, Z., and Poshyvanyk, D. (2019). Guigle: A gui search engine for android apps. In *2019 IEEE/ACM 41st ICSE-Companion*, pages 71–74. IEEE.

Bjarnason, E., Lang, F., and Mjöberg, A. (2021). A model of software prototyping based on a systematic map. In *Proceedings of the 15th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*, pages 1–11.

Bjarnason, E., Lang, F., and Mjöberg, A. (2023). An empirically based model of software prototyping: a mapping study and a multi-case study. *Empirical Software Engineering*, 28(5):115.

Braun, V. and Clarke, V. (2006). Using thematic analysis in psychology. *Qualitative research in psychology*, 3(2):77–101.

Brhel, M., Meth, H., Maedche, A., and Werder, K. (2015). Exploring principles of user-centered agile software development: A literature review. *Information and software technology*, 61:163–181.

Calle-Escobar, M., Mejía-Gutiérrez, R., Nadeau, J.-P., and Pailhes, J. (2016). Heuristics-based design process. *International Journal on Interactive Design and Manufacturing (IJIDeM)*, 10(4):369–386.

Charfi, S., Ezzedine, H., and Kolski, C. (2015). Rita: a user interface evaluation framework. *Journal of Universal Computer Science*, 21(4):526–560.

Chen, C., Feng, S., Liu, Z., Xing, Z., and Zhao, S. (2020a). From lost to found: Discover missing ui design semantics through recovering missing tags. *Proc. ACM Hum.-Comput. Interact.*, 4(CSCW2).

Chen, J., Chen, C., Xing, Z., Xia, X., Zhu, L., Grundy, J., and Wang, J. (2020b). Wireframe-based ui design search through image autoencoder. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 29(3):1–31.

Chen, J., Swearngin, A., Wu, J., Barik, T., Nichols, J., and Zhang, X. (2022). Towards complete icon labeling in mobile applications. In *CHI Conference on Human Factors in Computing Systems*, pages 1–14.

Cheng, L. C. (2016). The mobile app usability inspection (maui) framework as a guide for minimal viable product (mvp) testing in lean development cycle. In *Proceedings of the 2nd International conference in HCI and UX Indonesia 2016*, pages 1–11.

Chernak, Y. (1996). A statistical approach to the inspection checklist formal synthesis and improvement. *IEEE Transactions on Software Engineering*, 22(12):866–874.

Corbin, J. and Strauss, A. (2014). *Basics of qualitative research: Techniques and procedures for developing grounded theory*. Sage publications.

Da Costa, R. P., Canedo, E. D., De Sousa, R. T., Albuquerque, R. D. O., and Villalba, L. J. G. (2019). Set of usability heuristics for quality assessment of mobile applications on smartphones. *Ieee Access*, 7:116145–116161.

Davison, R., Martinsons, M. G., and Kock, N. (2004). Principles of canonical action research. *Information systems journal*, 14(1):65–86.

De Macedo, G., Fontão, A., and Gadelha, B. (2022). Prototyping in software quality assurance: A survey with software practitioners. In *Proceedings of the XXI Brazilian Symposium on Software Quality*, SBQS 2022, New York, NY, USA. Association for Computing Machinery.

De Macedo, G. T., Fontão, A., and Gadelha, B. (2023). Uiprotocheck: A checklist for semantic inspection of user interface prototypes. In *Proceedings of the XXXVII Brazilian Symposium on Software Engineering*, SBES '23, page 485–490, New York, NY, USA. Association for Computing Machinery.

De Macedo, G. T., Fontão, A., and Gadelha, B. (2024). Building soft skills through a role-play based approach for requirements engineering remote education. *Journal of the Brazilian Computer Society*, 30(1):1–16.

De Macedo, G. T., Viana, L., and Gadelha, B. (2022). Towards meaning in user interface prototypes. In *Proceedings of the 24th International Conference on Enterprise Information Systems - Volume 2: ICEIS*, pages 235–246. INSTICC, SciTePress.

De Mello, R. M., Motta, R. C., and Travassos, G. H. (2016). A checklist-based inspection technique for business process models. In *Business Process Management Forum: BPM Forum 2016, Rio de Janeiro, Brazil, September 18-22, 2016, Proceedings 14*, pages 108–123. Springer.

De Souza, C. S., Leitão, C. F., Prates, R. O., and da Silva, E. J. (2006). The semiotic inspection method. In *Proceedings of VII Brazilian Symposium on Human Factors in Computing Systems*, IHC '06, page 148–157, New York, NY, USA. Association for Computing Machinery.

Deka, B., Huang, Z., Franzen, C., Hibschman, J., Afergan, D., Li, Y., Nichols, J., and Kumar, R. (2017). Rico: A mobile app dataset for building data-driven design applications. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*, pages 845–854.

Dow, S., Fortuna, J., Schwartz, D., Altringer, B., Schwartz, D., and Klemmer, S. (2011). Prototyping dynamics: sharing multiple designs improves exploration, group rapport, and results. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 2807–2816.

Dresch, A., Lacerda, D. P., and Júnior, J. A. V. A. (2015). *Design science research: método de pesquisa para avanço da ciência e tecnologia*. Bookman Editora.

Duan, P., Hartmann, B., Nguyen, K., Li, Y., Hearst, M., and Morris, M. R. (2023). Towards semantically-aware ui design tools: Design, implementation and evaluation of semantic grouping guidelines. In *ICML 2023 Workshop on Artificial Intelligence and Human-Computer Interaction. https://research. google/pubs/-pub52594*.

Duan, P., Warner, J., Li, Y., and Hartmann, B. (2024). Generating automatic feedback on ui mockups with large language models. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*, CHI '24, New York, NY, USA. Association for Computing Machinery.

Easterbrook, S., Singer, J., Storey, M.-A., and Damian, D. (2008). Selecting empirical methods for software engineering research. *Guide to advanced empirical software engineering*, pages 285–311.

Falessi, D., Juristo, N., Wohlin, C., Turhan, B., Münch, J., Jedlitschka, A., and Oivo, M. (2018). Empirical software engineering experts on the use of students and professionals in experiments. *Empirical Software Engineering*, 23:452–489.

Figueiredo, E., Sant'Anna, C., Garcia, A., and Lucena, C. (2012). Applying and evaluating concern-sensitive design heuristics. *Journal of Systems and Software*, 85(2):227–243.

Garcia, A., Silva da Silva, T., and Selbach Silveira, M. (2017). Artifacts for agile user-centered design: a systematic mapping. In *Proceedings of the 50th Hawaii International Conference on System Sciences*.

Gigerenzer, G. (2008). Why heuristics work. *Perspectives on psychological science*, 3(1):20–29.

Glasow, P. A. (2005). Fundamentals of survey research methodology. *Retrieved January*, 18:2013.

Guo, P., Saab, N., Post, L. S., and Admiraal, W. (2020). A review of project-based learning in higher education: Student outcomes and measures. *International Journal of Educational Research*, 102:101586.

Hevner, A. and Chatterjee, S. (2010). Design science research in information systems. In *Design research in information systems*, pages 9–22. Springer.

Hevner, A. R. (2007). A three cycle view of design science research. *Scandinavian journal of information systems*, 19(2):4.

Hevner, A. R., March, S. T., Park, J., and Ram, S. (2004). Design science in information systems research. *MIS quarterly*, 28(1):75–105.

Hollingsed, T. and Novick, D. G. (2007). Usability inspection methods after 15 years of research and practice. In *Proceedings of the 25th Annual ACM International Conference on Design of Communication*, SIGDOC '07, page 249–255, New York, NY, USA. Association for Computing Machinery.

Huang, F., Canny, J. F., and Nichols, J. (2019). Swire: Sketch-based user interface retrieval. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pages 1–10.

Islam, M. N., Bouwman, H., and Islam, A. N. (2020). Evaluating web and mobile user interfaces with semiotics: An empirical study. *IEEE Access*, 8:84396–84414.

Jeong, H. and Hmelo-Silver, C. E. (2016). Seven affordances of computer-supported collaborative learning: How to support collaborative learning? how can technologies help? *Educational Psychologist*, 51(2):247–265.

Kahn, M. J. and Prail, A. (1994). *Formal usability inspections*, page 141–171. John Wiley & Sons, Inc., USA.

Karhapää, P., Behutiye, W., Rodríguez, P., Oivo, M., Costal, D., Franch, X., Aaramaa, S., Choraś, M., Partanen, J., and Abherve, A. (2021). Strategies to manage quality requirements in agile software development: a multiple case study. *Empirical Software Engineering*, 26(2):1–59.

Klie, J.-C., Castilho, R. E. d., and Gurevych, I. (2024). Analyzing dataset annotation quality management in the wild. *Computational Linguistics*, 50(3):817–866.

Kolthoff, K., Bartelt, C., and Ponzetto, S. P. (2020). Gui2wire: rapid wireframing with a mined and large-scale gui repository using natural language requirements. In *Proceedings of the 35th IEEE/ACM International Conference on Automated Software Engineering*, pages 1297–1301.

Kolthoff, K., Bartelt, C., and Ponzetto, S. P. (2023). Data-driven prototyping via natural-language-based gui retrieval. *Automated Software Engineering*, 30(1):13.

Kolthoff, K., Kretzer, F., Bartelt, C., Maedche, A., and Ponzetto, S. P. (2024). Interlinking user stories and gui prototyping: A semi-automatic llm-based approach. In *2024 IEEE 32nd International Requirements Engineering Conference (RE)*, pages 380–388. IEEE.

Käpyaho, M. and Kauppinen, M. (2015). Agile requirements engineering with prototyping: A case study. In *2015 IEEE 23rd International Requirements Engineering Conference (RE)*, pages 334–343.

Laitenberger, O., Atkinson, C., Schlich, M., and El Emam, K. (2000). An experimental comparison of reading techniques for defect detection in uml design documents. *Journal of Systems and Software*, 53(2):183–204.

Lauff, C. A., Knight, D., Kotys-Schwartz, D., and Rentschler, M. E. (2020). The role of prototypes in communication between stakeholders. *Design Studies*, 66:1–34.

Lee, C., Kim, S., Han, D., Yang, H., Park, Y.-W., Kwon, B. C., and Ko, S. (2020). Guicomp: A gui design assistant with real-time, multi-faceted feedback. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–13.

Leiva, L. A., Hota, A., and Oulasvirta, A. (2020). Enrico: A dataset for topic modeling of mobile ui designs. In *22nd International Conference on Human-Computer Interaction with Mobile Devices and Services*, MobileHCI '20, New York, NY, USA. Association for Computing Machinery.

Lelli, V., Blouin, A., and Baudry, B. (2015). Classifying and qualifying gui defects. In *2015 IEEE 8th international conference on software testing, verification and validation (ICST)*, pages 1–10. IEEE.

Li, T. J.-J., Popowski, L., Mitchell, T., and Myers, B. A. (2021). Screen2vec: Semantic embedding of gui screens and gui components. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, New York, NY, USA. Association for Computing Machinery.

Linåker, J., Sulaman, S. M., de Mello, R. M., and Höst, M. (2015). Guidelines for conducting surveys in software engineering. *Lund University*, 50:1–64.

Liu, T. F., Craft, M., Situ, J., Yumer, E., Mech, R., and Kumar, R. (2018). Learning design semantics for mobile apps. In *The 31st Annual ACM Symposium on User Interface Software and Technology*, pages 569–579. ACM.

Lucas, W. and Babaian, T. (2015). The collaborative critique: An inspection method for expert evaluation of user interfaces. *International Journal of Human-Computer Interaction*, 31(11):843–859.

Marenkov, J., Robal, T., and Kalja, A. (2016). A study on immediate automatic usability evaluation of web application user interfaces. In *Databases and Information Systems: 12th International Baltic Conference, DB&IS 2016, Riga, Latvia, July 4-6, 2016, Proceedings 12*, pages 257–271. Springer.

Matteson, M. L., Anderson, L., and Boyden, C. (2016). "soft skills": A phrase in search of meaning. *portal: Libraries and the Academy*, 16(1):71–88.

Melegati, J., Chanin, R., Sales, A., Prikladnicki, R., and Wang, X. (2020). Mvp and experimentation in software startups: a qualitative survey. In *2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pages 322–325.

Moran, K., Bernal-Cárdenas, C., Curcio, M., Bonett, R., and Poshyvanyk, D. (2020). Machine learning-based prototyping of graphical user interfaces for mobile apps. *IEEE Transactions on Software Engineering*, 46(2):196–221.

Moran, K., Li, B., Bernal-Cárdenas, C., Jelf, D., and Poshyvanyk, D. (2018). Automated reporting of gui design violations for mobile apps. In *Proceedings of the 40th International Conference on Software Engineering*, pages 165–175.

Nielsen, J. (1994a). Enhancing the explanatory power of usability heuristics. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pages 152–158.

Nielsen, J. (1994b). *Heuristic evaluation*, page 25–62. John Wiley & Sons, Inc., USA.

Ogedebe, P. M. and Jacob, B. P. (2012). Software prototyping: a strategy to use when user lacks data processing experience. *ARPN Journal of Systems and Software*, 2(6):219.

Packevičius, Š., Barisas, D., Ušaniov, A., Guogis, E., and Bareiša, E. (2018). Text semantics and layout defects detection in android apps using dynamic execution and screenshot analysis. In *International Conference on Information and Software Technologies*, pages 279–292. Springer.

Quiñones, D., Rusu, C., and Rusu, V. (2018). A methodology to develop usability/user experience heuristics. *Computer Standards & Interfaces*, 59:109–129.

Rainer, A. and Wohlin, C. (2022). Recruiting credible participants for field studies in software engineering research. *Information and Software Technology*, 151:107002.

Rocha Silva, T., Winckler, M., and Trætteberg, H. (2019). Ensuring the consistency between user requirements and gui prototypes: A behavior-based automated approach. In *Human-Computer Interaction–INTERACT 2019: 17th IFIP TC 13 International Conference, Paphos, Cyprus, September 2–6, 2019, Proceedings, Part I 17*, pages 644–665. Springer.

Salleh, M. A., Bahari, M., and Zakaria, N. H. (2017). An overview of software functionality service: A systematic literature review. *Procedia Computer Science*, 124:337–344.

Salman, I., Misirli, A. T., and Juristo, N. (2015). Are students representatives of professionals in software engineering experiments? In *2015 IEEE/ACM 37th IEEE international conference on software engineering*, volume 1, pages 666–676. IEEE.

Sauer, C., Jeffery, D., Land, L., and Yetton, P. (2000). The effectiveness of software development technical reviews: a behaviorally motivated program of research. *IEEE Transactions on Software Engineering*, 26(1):1–14.

Sears, A. (1997). Heuristic walkthroughs: Finding the problems without the noise. *International journal of human-computer interaction*, 9(3):213–234.

Sermuga Pandian, V. P., Shams, A., Suleri, S., and Jarke, P. D. M. (2022). Lofi sketch: A large scale dataset of smartphone low fidelity sketches. In *CHI Conference on Human Factors in Computing Systems Extended Abstracts*, pages 1–5.

Sermuga Pandian, V. P., Suleri, S., and Jarke, M. (2021a). Synz: Enhanced synthetic dataset for training ui element detectors. In *26th International Conference on Intelligent User Interfaces-Companion*, pages 67–69.

Sermuga Pandian, V. P., Suleri, S., and Jarke, P. D. M. (2021b). Uisketch: a large-scale dataset of ui element sketches. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–14.

Shah, A. K. and Oppenheimer, D. M. (2008). Heuristics made easy: an effort-reduction framework. *Psychological bulletin*, 134(2):207.

Shull, F., Rus, I., and Basili, V. (2000). How perspective-based reading can improve requirements inspections. *Computer*, 33(7):73–79.

Silva, P. A., Holden, K., and Jordan, P. (2015). Towards a list of heuristics to evaluate smartphone apps targeted at older adults: a study with apps that aim at promoting health and well-being. In *2015 48th Hawaii international conference on system sciences*, pages 3237–3246. IEEE.

Silva, T. R., Hak, J.-L., and Winckler, M. (2017). A behavior-based ontology for supporting automated assessment of interactive systems. In *2017 IEEE 11th International Conference on Semantic Computing (ICSC)*, pages 250–257. IEEE.

Souza, A., Ferreira, B., Valentim, N., Correa, L., Marczak, S., and Conte, T. (2020). Supporting the teaching of design thinking techniques for requirements elicitation through a recommendation tool. *IET Software*, 14(6):693–701.

Spencer, R. (2000). The streamlined cognitive walkthrough method, working around social constraints encountered in a software development company. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pages 353–359.

Stol, K.-J. and Fitzgerald, B. (2018). The abc of software engineering research. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 27(3):1–51.

Storey, M.-A., Engstrom, E., Höst, M., Runeson, P., and Bjarnason, E. (2017). Using a visual abstract as a lens for communicating and promoting design science research in software engineering. In *2017 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*, pages 181–186. IEEE.

Sunkara, S., Wang, M., Liu, L., Baechler, G., Hsiao, Y.-C., Sharma, A., Stout, J., et al. (2022). Towards better semantic understanding of mobile interfaces. *arXiv preprint arXiv:2210.02663*.

Suranto, B. (2015). Software prototypes: Enhancing the quality of requirements engineering process. In *2015 International Symposium on Technology Management and Emerging Technologies (ISTMET)*, pages 148–153. IEEE.

Svensson, R. B. and Regnell, B. (2017). Is role playing in requirements engineering education increasing learning outcome? *Requirements Engineering*, 22(4):475–489.

Swearngin, A., Dontcheva, M., Li, W., Brandt, J., Dixon, M., and Ko, A. J. (2018). Rewire: Interface design assistance from examples. In *Proceedings of the 2018 CHI conference on human factors in computing systems*, pages 1–12.

Thelin, T., Runeson, P., Wohlin, C., Olsson, T., and Andersson, C. (2004). Evaluation of usage-based reading—conclusions after three experiments. *Empirical Software Engineering*, 9:77–110.

Toxboe, A. (2022). User interface design pattern gallery. `http://ui-patterns.com/`.

Valentim, N. M. C. and Conte, T. (2014). Improving a usability inspection technique based on quantitative and qualitative analysis. In *2014 Brazilian Symposium on Software Engineering*, pages 171–180.

Venkatesh, V. and Bala, H. (2008). Technology acceptance model 3 and a research agenda on interventions. *Decision sciences*, 39(2):273–315.

Walker, M., Takayama, L., and Landay, J. (2002). High-fidelity or low-fidelity, paper or computer choosing attributes when testing web prototypes. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 46.

Wang, B., Li, G., Zhou, X., Chen, Z., Grossman, T., and Li, Y. (2021). Screen2words: Automatic mobile ui summarization with multimodal learning. In *The 34th Annual ACM Symposium on User Interface Software and Technology*, pages 498–510.

Wharton, C. (1994). The cognitive walkthrough method: A practitioner's guide. *Usability Inspection Methods, New York*, pages 105–140.

Wilson, C. (2013). *User interface inspection methods: a user-centered design method*. Newnes.

Wohlin, C., Runeson, P., Höst, M., Ohlsson, M. C., Regnell, B., and Wesslén, A. (2012). *Experimentation in software engineering*. Springer Science & Business Media.

Wu, J., Wang, S., Shen, S., Peng, Y.-H., Nichols, J., and Bigham, J. P. (2023). Webui: A dataset for enhancing visual ui understanding with web semantics. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, pages 1–14.

Wu, J., Zhang, X., Nichols, J., and Bigham, J. P. (2021). Screen parsing: Towards reverse engineering of ui models from screenshots. In *The 34th Annual ACM Symposium on User Interface Software and Technology*, UIST '21, page 470–483, New York, NY, USA. Association for Computing Machinery.

Yang, B., Xing, Z., Xia, X., Chen, C., Ye, D., and Li, S. (2021). Don't do that! hunting down visual design smells in complex uis against design guidelines. In *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*, pages 761–772. IEEE.

Yeratziotis, A. and Zaphiris, P. (2018). A heuristic evaluation for deaf web user experience (he4dwux). *International Journal of Human–Computer Interaction*, 34(3):195–217.

Yilmaz, S., Daly, S. R., Seifert, C. M., and Gonzalez, R. (2016). Evidence-based design heuristics for idea generation. *Design Studies*, 46:95–124.

Zang, X., Xu, Y., and Chen, J. (2021). Multimodal icon annotation for mobile applications. In *Proceedings of the 23rd International Conference on Mobile Human-Computer Interaction*, pages 1–11.

Zecheng, H., Sunkara, S., Zang, X., Xu, Y., Liu, L., Wichers, N., Schubiner, G., Lee, R., and Chen, J. (2020). Actionbert: Leveraging user actions for semantic understanding of user interfaces. *arXiv preprint arXiv:2012.12350*.

Zhang, Z., Basili, V., and Shneiderman, B. (1999). Perspective-based usability inspection: An empirical validation of efficacy. *Empirical Software Engineering*, 4(1):43–69.

Zhou, L., Schellaert, W., Martínez-Plumed, F., Moros-Daval, Y., Ferri, C., and Hernández-Orallo, J. (2024). Larger and more instructable language models become less reliable. *Nature*, 634(8032):61–68.

Zowghi, D. and Paryani, S. (2003). Teaching requirements engineering through role playing: Lessons learnt. In *Proceedings. 11th IEEE International Requirements Engineering Conference, 2003.*, pages 233–241. IEEE.

Apêndice A

Catálogo de Heurísticas de Inspeção Funcional - CHIF-E

A.1 Funcionalidades Básicas

Nesta seção são apresentadas as especificações heurísticas para as categorias de protótipo do nível de Funcionalidades Básicas: Alteração (Tabela A.1), Busca (Tabela A.2), Cadastro (Tabela A.3), Galeria (Tabela A.5), Lista (Tabela A.6), Menu (Tabela A.7), Mestre-Detalhe (Tabela A.8), *Upload de Arquivo* (Tabela A.9) e Visualização (Tabela A.4).

A.2 Funcionalidades Específicas

Nesta seção são apresentadas as especificações heurísticas para as categorias de protótipo do nível de Funcionalidades Específicas: Autenticação (Tabela A.10), Câmera (Tabela A.11), Carrinho de Compras (Tabela A.12), Chat (Tabela A.13), Configurações (Tabela A.14), Editor de Imagens (Tabela A.15), Finalizar Compra (Tabela A.16), *Home* (Tabela A.17), Mapas (Tabela A.18), Media Player (Tabela A.19), Notícias (Tabela A.20), Página de Produto (Tabela A.21), Perfil de Usuário (Tabela A.22), Quiz (Tabela A.23), Registro de Conta (Tabela A.24) e Videoconferência (Tabela A.26).

Tabela A.1: Heurística HEU1-01 – Alteração.

ID: HEU1-01

Nome: **Alteração**
Sinônimos: Edição, Atualização
Definição: Apresenta um formulário que mostra os valores de atributos de um objeto para que eles possam ser modificados.
Explicação: Use este tipo de tela para ilustrar a atualização de dados pelo usuário e seu armazenamento no banco de dados da aplicação.
Exemplo: Um usuário altera seu endereço em um site de uma petshop.
Checklist:

1. O protótipo apresenta um formulário de entrada de dados com um ou mais campos.
2. Os campos do formulário representam as propriedades de uma entidade que faz parte da especificação da aplicação.
3. Os campos estão preenchidos com os valores que ilustram características ou propriedades do objeto representado.
4. O formulário tem um botão de atualização com um rótulo que descreve botões de alteração (ex.: salvar, salvar alterações, salvar edição,alterar, atualizar, finalizar, feito, confirmar) ou com um ícone que representa a ação de alteração ( ).

Informações relacionadas:

1. Entidades do domínio da aplicação e suas respectivas propriedades.

Heurísticas relacionadas: Perfil de Usuário, Upload de Arquivo

Tabela A.2: Heurística HEU1-02 – Busca.

ID: HEU1-02

Nome: **Busca**
Sinônimos: Pesquisa
Definição: Representa um mecanismo de busca
Explicação: Use este tipo de tela quando quiser ilustrar como os usuários inserem palavras-chave para localizar informações em uma aplicação de software.
Exemplo: Um usuário procura um livro pelo título em uma livraria online.
Checklist:

1. O protótipo apresenta um formulário de entrada de dados com um ou mais campos para busca.

2. Os campos de busca possuem rótulos que correspondem a propriedades de entidades que fazem parte da especificação da aplicação e que representam critérios da busca. Exemplos: buscar por nome, buscar produto, o que você procura.

3. Opcionalmente, o protótipo apresenta uma lista de objetos de entidades que fazem parte da especificação da aplicação para ilustrar os resultados da busca.

4. O formulário tem um botão de busca com um rótulo que descreve botões de busca (ex.: buscar, busca, pesquisar) ou com um ícone que representa a ação de buscar (ex.:).

Informações relacionadas:

1. Entidades do domínio da aplicação e suas respectivas propriedades.

Heurísticas relacionadas: Lista, Galeria

Tabela A.3: Heurística HEU1-03 – Cadastro.

ID: HEU1-03

Nome: **Cadastro**
Sinônimos: Inclusão, Adição, Registro
Definição: Apresenta um formulário para adicionar um objeto em uma entidade do domínio da aplicação.
Explicação: Use este tipo de tela para ilustrar a entrada de dados pelo usuário que será armazenada no banco de dados da aplicação.
Exemplo: Um funcionário de um site de notícias publica informações sobre as eleições municipais.
Checklist:

1. O protótipo apresenta um formulário de entrada de dados com um ou mais campos cujos valores não estão preenchidos.

2. Os campos do formulário representam as propriedades de uma entidade que faz parte da especificação da aplicação.

3. O formulário tem um botão de cadastro com um rótulo que descreve botões de cadastro (ex.: criar, enviar, salvar, solicitar inclusão, adicionar, encaminhar, finalizar, confirmar, cadastrar, gerar.) ou com um ícone que representa a ação de cadastrar (ex.:).

Informações relacionadas:

1. Entidades do domínio da aplicação e suas respectivas propriedades.

Heurísticas relacionadas: Registro de Conta, Upload de Arquivo, Perfil de Usuário

Tabela A.4: Heurística HEU1-04 – Visualização.

ID: HEU1-04

Nome: **Visualização**
Sinônimos: - Consulta
Definição: Exibe informações características de um objeto de uma entidade do domínio da aplicação.
Explicação: Este tipo de protótipo é usado para ilustrar a exibição de informações sobre objetos.
Exemplo: Um usuário visualiza informações sobre um livro em um aplicativo de empréstimo de livros.
Checklist:

1. O protótipo exibe informações que ilustram propriedades de um objeto de uma entidade que faz parte da especificação da aplicação.

Informações relacionadas:

1. Entidades do domínio da aplicação e suas respectivas propriedades.

Heurísticas relacionadas: Mestre-Detalhe, Upload de Arquivo e Perfil de Usuário

Tabela A.5: Heurística HEU1-05 – Galeria.

ID: HEU1-05

Nome: **Galeria**
Sinônimos:
Definição: Exibe em formato de grade de um conjunto de objetos de uma entidade do domínio.
Explicação: Use este tipo de tela para apresentar uma visão geral de um conjunto de imagens relacionadas a um tema, em formato de grade. Cada célula pode conter informações adicionais sobre um objeto, além da imagem.
Exemplo: Um usuário navega por informações de roupas esportivas femininas em um aplicativo de comércio eletrônico. As roupas são exibidas em forma de grade e cada célula mostra a foto, título e preço de uma peça.
Checklist:

1. O protótipo apresenta um grupo de imagens que ilustram propriedades de uma entidade que compõe a especificação da aplicação.
2. As imagens estão arranjadas em forma de grade.
3. Opcionalmente, cada célula da grade apresenta valores que ilustram outras propriedades do objeto correspondente à imagem.

Informações relacionadas:

1. Entidades do domínio da aplicação e suas respectivas propriedades.

Heurísticas relacionadas: Lista

Tabela A.6: Heurística HEU1-06 – Lista.

ID: HEU1-06

Nome: **Lista**
Sinônimos: Lista vertical, Lista horizontal
Definição: Exibição de um conjunto de objetos relacionados a uma entidade do domínio da aplicação.
Explicação: Uma lista apresenta um conjunto de objetos de uma entidade no domínio da aplicação.
Exemplo: Um usuário visualiza uma lista de filmes em um aplicativo de serviço de *streaming*.
Checklist:

1. O protótipo apresenta uma lista de objetos de uma entidade que faz parte da especificação da aplicação.
2. Cada item da lista apresenta valores que ilustram características ou propriedades dos objetos.
3. Os itens da lista estão organizados na horizontal ou na vertical.
4. Se a lista está organizada em forma de tabela, os títulos das colunas correspondem a nomes de propriedades dos objetos.

Informações relacionadas:

1. Entidades do domínio da aplicação e suas respectivas propriedades.

Heurísticas relacionadas: Galeria, Busca, Mestre-Detalhe, Videoconferência

Tabela A.7: Heurística HEU1-07 – Menu.

ID: HEU1-07

Nome: **Menu**
Sinônimos: barra de ferramentas, barra inferior, barra superior, menu drop-down.
Definição: Exibe uma lista das funcionalidades disponíveis em uma aplicação.
Explicação: Um menu ilustra as ações que um usuário pode realizar em uma aplicação.
Exemplo: Um usuário navega pelos departamentos de uma loja online de venda de roupas.
Checklist:

1. O protótipo apresenta um grupo de elementos em uma lista lateral, barra superior, barra inferior ou na forma de botões.

2. Os rótulos e ícones dos elementos do grupo representam as funcionalidades oferecidas pela aplicação.

Informações relacionadas:

1. Lista das funcionalidades oferecidas pela aplicação.

2. Ícones que representam as funcionalidades da aplicação.

Heurísticas relacionadas: Home, Lista

Tabela A.8: Heurística HEU1-08 – Mestre-Detalhe.

ID: HEU1-08

Nome: **Mestre-Detalhe**
Sinônimos:
Definição: Exibe propriedades de um objeto de uma entidade da aplicação seguido por uma lista de objetos de outra entidade da aplicação relativos ao objeto principal.
Explicação: Um protótipo Mestre-Detalhe é utilizado para ilustrar quando um objeto tem um outro conjunto de objetos relacionados a ele.
Exemplo: Um usuário visualiza dados de um vendedor e a lista de produtos que ele comercializa em um marketplace.
Checklist:

1. O protótipo apresenta valores que ilustram características ou propriedades de um objeto de uma entidade que faz parte da especificação da aplicação (mestre).
2. O protótipo apresenta uma lista de objetos de uma segunda entidade que faz parte da especificação da aplicação (detalhe).
3. A lista de objetos da segunda entidade tem uma relação de dependência com o objeto mestre.

Informações relacionadas:

1. Entidades do domínio da aplicação e suas respectivas propriedades.

Heurísticas relacionadas: Visualização, Lista

Tabela A.9: Heurística HEU1-09 – Upload de Arquivos.

ID: HEU1-09

Nome: **Upload de Arquivo**
Sinônimos: –
Definição: Apresenta um formulário para upload de arquivos.
Explicação: Usado para ilustrar um formulário de upload de arquivos relacionados a uma entidade do domínio da aplicação.
Exemplo: Um usuário de rede social faz o upload da sua foto de perfil.
Checklist:

1. O protótipo apresenta um formulário com um ou mais campos de upload relacionados a propriedades de uma entidade que faz parte da especificação da aplicação.

2. Os rótulos dos campos de upload descrevem uma ação de seleção de arquivos. Exemplos: escolha um arquivo, selecione foto, anexe um arquivo.

3. Os campos de upload têm botões para realizar o upload de arquivos e seus rótulos descrevem ações de upload (ex.: enviar, fazer upload, adicionar) ou seus ícones representam a ação de upload de arquivo (ex.:).

Informações relacionadas:

1. Entidades do domínio da aplicação e suas respectivas propriedades.

Heurísticas relacionadas: Cadastro, Alteração, Visualização

Tabela A.10: Heurística HEU2-01 – Autenticação.

ID: HEU2-01

Nome: **Autenticação**
Sinônimos: Login, Log in
Definição: Formulário para informações de credenciais para acesso a recursos de um software.
Explicação: Um protótipo do tipo Autenticação é utilizado quando se deseja ilustrar que um usuário deve informar suas credenciais para acessar funcionalidades restritas de um software.
Exemplo: Um usuário informa seu e-mail e senha para se conectar a um aplicativo de rede social.
Checklist:

1. O protótipo apresenta um formulário de entrada de dados com um campo para identificação do usuário e um para fornecimento de autenticação;
2. A autenticação pode ser fornecida por caracteres ou por biometria;
3. O rótulo do campo de identificação de usuário é comumente utilizado para descrever campos de identificação do usuário. Exemplos: id, login, e-mail, email, identificação, cpf, digite seu cpf, nome, usuário, matrícula, digite o login, nome de usuário, telefone.
4. O rótulo do campo de autenticação é comumente utilizado para descrever campos de autenticação. Exemplos: Senha, Digite a senha.
5. O formulário tem um botão de autenticação com um rótulo que descreve botões de autenticação (exemplos: login, log in, logar, conectar, entrar, entrar com google, entrar com facebook, logar com a conta do gmail, continuar, fazer login) ou com um ícone representa a ação de autenticação (exemplos:).
6. Opcionalmente, o protótipo pode apresentar um link para recuperação de acesso;
7. Opcionalmente, o protótipo pode apresentar opções de autenticação com plataformas externas à aplicação.

Informações relacionadas: –
Heurísticas relacionadas: –

Tabela A.11: Heurística HEU2-02 – Câmera.

ID: HEU2-02

Nome: **Câmera**
Sinônimos: –
Definição: Representa o uso da câmera para capturar imagens para uma aplicação.
Explicação: Use este tipo de tela quando quiser ilustrar que um usuário precisa incluir uma foto na aplicação utilizando a câmera.
Exemplo: m usuário tira uma foto usando a câmera do telefone celular e a publica em um aplicativo de rede social.
Checklist:

1. O protótipo apresenta uma área para representar a imagem que está sendo capturada.

2. O protótipo apresenta um botão para capturar imagem e seu ícone representa uma ação de captura de imagens. Exemplos:

3. O protótipo apresenta um botão para cancelar a ação com um ícone representa o cancelamento de uma ação. Exemplos:

4. Opcionalmente, o protótipo pode apresentar um botão para alternar entre câmeras frontais e traseiras com um ícone que representa a ação de alternar entre câmeras. Exemplos:

5. Opcionalmente, o protótipo pode apresentar um botão para habilitar ou desabilitar o flash com um ícone que representa ação de ativar/desativar o flash. Exemplos:

Informações relacionadas: –
Heurísticas relacionadas: Editor de Imagens

Tabela A.12: Heurística HEU2-03 – Carrinho de Compra.

ID: HEU2-03

Nome: **Carrinho de Compra**
Sinônimos: Carrinho
Definição: Lista de produtos ou serviços selecionados para compra em uma aplicação de e-commerce.
Explicação: Em uma aplicação de comércio eletrônico, um carrinho de compras representa a possibilidade de que os usuários selecionem um ou mais produtos para compra em diferentes quantidades e que armazenem os produtos para compra posterior.
Exemplo: Um usuário compra mantimentos em um aplicativo de supermercado.
Checklist:

1. O protótipo apresenta uma lista de produtos de uma entidade que faz parte da especificação da aplicação.
   (a) Cada item da lista apresenta a descrição do produto.
   (b) Cada item da lista possui um campo de texto para entrada da quantidade.
   (c) Cada item da lista apresenta um rótulo indicando o preço do item e um rótulo indicando o subtotal do item.
   (d) Cada item da lista apresenta um link ou um ícone para remover o item.
2. O protótipo apresenta um rótulo indicando o preço total dos produtos.
3. O protótipo apresenta um botão de Finalizar Compra com um rótulo que descreve botões de finalização de compra (exemplos: finalizar, finalizar compra, proceder ao pagamento, confirmar, confirmar pedido, fechar pedido) ou com um ícone que representa a ação de finalizar a compra (exemplos:).

Informações relacionadas:

1. Entidades do domínio da aplicação e suas respectivas propriedades.

Heurísticas relacionadas: Lista, Finalizar Compra

Tabela A.13: Heurística HEU2-04 – Chat.

ID: HEU2-04

Nome: **Chat**
Sinônimos:
Definição: Funcionalidade de troca de mensagens instantâneas entre usuários de uma aplicação.
Explicação: Um protótipo de Chat representa como usuários podem interagir entre si por troca de mensagens usando a aplicação.
Exemplo: Um usuário troca mensagens de texto com um amigo em um aplicativo de rede social.
Checklist:

1. O protótipo apresenta uma lista de mensagens de texto.
2. As mensagens de texto enviadas são alinhadas à direita no protótipo.
3. As mensagens de texto recebidas são alinhadas à esquerda no protótipo.
4. O protótipo apresenta um campo de texto para envio de novas mensagens.
5. O rótulo do campo de texto é comumente utilizado para descrever campos para envio de mensagens em chat. Exemplos: começar conversa, digite uma mensagem, digite algo, digite sua mensagem.
6. O protótipo tem um botão de envio de mensagem com um rótulo que descreve botões de envio de mensagem (ex. enviar) ou o seu ícone representa a ação de enviar mensagem (exemplos:).

Informações relacionadas: –
Heurísticas relacionadas: Videoconferência

Tabela A.14: Heurística HEU2-05 – Configurações.

ID: HEU2-05

Nome: **Configurações**
Sinônimos: Preferências
Definição: Representa as configurações de preferência de uso de uma aplicação.
Explicação: Um protótipo do tipo Configurações representa as diferentes maneiras de configurar uma aplicação.
Exemplo: Um usuário escolhe que tipo de notificações receber de um aplicativo de previsão do tempo.
Checklist:

1. O título do protótipo é um termo comumente usado para descrever telas de configuração. Exemplos: Configurações, Notificações, Preferências.

2. O protótipo apresenta um ou mais itens correspondentes às opções personalizáveis da aplicação.

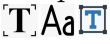
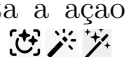
3. Opcionalmente, as opções de configuração podem ser agrupadas em seções ou sub-telas.

Informações relacionadas: –
Heurísticas relacionadas: Alteração

Tabela A.15: Heurística HEU2-06 – Editor de Imagens.

ID: HEU2-06

Nome: **Editor de Imagens**
Sinônimos: –
Definição: Funcionalidade de edição de imagens.
Explicação: Use este tipo de protótipo de tela para representar uma interface de edição de imagens.
Exemplo: Um usuário edita uma foto da câmera para cortar as bordas.
Checklist:

1. O protótipo contém uma área para representar a imagem editada.
2. O protótipo apresenta um botão Salvar com um ícone que representa a ação de Salvar. Exemplos:
3. O protótipo contém um botão com um rótulo comumente usado para descrever botões de cancelar ou com um ícone que representa a ação *cancelar alterações*. Exemplos:
4. O protótipo apresenta pelo menos um dos seguintes botões:
   (a) O protótipo apresenta um botão Crop com um ícone que representa a ação de cortar imagens. Exemplos:
   (b) O protótipo apresenta um botão redimensionar com um ícone que representa a ação de redimensionar imagens. Exemplos:
   (c) O protótipo apresenta um botão Girar com um ícone que representa a ação de girar imagens. Exemplos:
   (d) O protótipo apresenta um botão Reverse com um rótulo que descreve a ação de restaurar imagens (Restaurar, Reverter), ou com um ícone que representa a ação de reverter mudanças em imagens. Exemplos:
   (e) O protótipo apresenta um botão para inserir texto na imagem com um ícone que representa a ação de inserir texto nas imagens. Exemplos: 
   (f) O protótipo apresenta um botão para adicionar efeitos na imagem com um ícone que representa a ação de adicionar efeitos visuais nas imagens. Exemplos: 
   (g) O protótipo apresenta um botão Compartilhar com um ícone que representa a ação de compartilhamento. Exemplos:
   (h) O protótipo apresenta um botão Menu com um ícone que representa um menu. Exemplos:

Informações relacionadas: –
Heurísticas relacionadas: Câmera

Tabela A.16: Heurística HEU2-07 – Finalizar Compra.

ID: HEU2-07

Nome: **Finalizar Compra**
Sinônimos: –
Definição: Representa a definição de opções de pagamento, envio e cobrança em aplicações e-commerce.
Explicação: Este tipo de protótipo ilustra as ações necessárias na conclusão de uma compra online, tais como coleta detalhes de pagamento, envio e cobrança.
Exemplo: Um usuário finaliza seu pedido em um site de vendas de perfumes, escolhendo um cartão de crédito como forma de pagamento, informando o endereço de entrega e selecionando o frete expresso oferecido pelo site.
Checklist:

1. O protótipo apresenta uma lista de opções de pagamento.
2. O protótipo apresenta um grupo de campos para informar o endereço de entrega.
3. O protótipo apresenta um grupo de campos para escolha das opções de entrega.
4. As opções de pagamento são precedidas por rótulos que as descrevem (exemplos: cartão de crédito, cartão de débito, pix, boleto bancário, transferência bancária, pagar na entrega) ou por ícones que as representam (exemplos:).
5. O protótipo apresenta um botão de envio de pedido com um rótulo que descreve botões de conclusão de compra (ex.: fazer pedido, enviar seu pedido, pagar agora, confirmar pedido) ou com um ícone que representa a ação de finalização de compra (ex.:).

Informações relacionadas: –
Heurísticas relacionadas: Carrinho de Compra

Tabela A.17: Heurística HEU2-08 – Home.

ID: HEU2-08

Nome: **Home**
Sinônimos: Página Inicial
Definição: Apresenta as informações mais relevantes de uma aplicação em uma variedade de formatos.
Explicação: Este protótipo é utilizado para ilustrar a página inicial de uma aplicação, dando uma visão geral dos serviços ou produtos oferecidos.
Exemplo: Usuários visualizam destaques, promoções e itens mais vendidos em um aplicativo de venda de óculos.
Checklist:

1. O protótipo apresenta listas ou galerias de objetos de entidades que fazem parte da especificação da aplicação.

2. As listas apresentadas representam destaques de recursos oferecidos pelo aplicativo.

Informações relacionadas:

1. Entidades do domínio da aplicação e suas respectivas propriedades.

Heurísticas relacionadas: Lista, Galeria

Tabela A.18: Heurística HEU2-09 – Mapas.

ID: HEU2-09

Nome: **Mapas**
Sinônimos: –
Definição: Busca de locais e visualização em mapas.
Explicação: Um protótipo Mapa é usado para representar busca geográfica e exibição.
Exemplo: O site de uma empresa exibe uma mapa com a localização de suas lojas físicas.
Checklist:

1. O protótipo apresenta uma área de imagem para representar o mapa.

2. Opcionalmente, o protótipo:

   (a) Apresenta um campo de pesquisa precedido por um rótulo que pertence a um conjunto de rótulos usados para descrever campos de termos de pesquisa. Exemplos: Pesquise, Busque.

   (b) Apresenta um Botão de Pesquisa cujo rótulo pertence a um conjunto de rótulos usados para descrever botões de pesquisa (Buscar, Pesquisar) ou cujo ícone pertence a um conjunto de ícones usados para representar botões de pesquisa. Exemplos:

   (c) Apresenta um botão para exibir a localização atual cujo ícone pertence a um conjunto de ícones que representam botões de localização. Exemplos:

Informações relacionadas: –
Heurísticas relacionadas: Busca

Tabela A.19: Heurística HEU2-10 – Media Player.

ID: HEU2-10

Nome: **Media Player**
Sinônimos: –
Definição: Reprodutor de música ou vídeo.
Explicação: Um reprodutor de mídia ilustra os componentes de áudio ou vídeo embutidos no aplicativo.
Exemplo: O usuário reproduz uma música usando um aplicativo de streaming de música.
Checklist:

1. O protótipo apresenta uma área para representar um media player.
2. O protótipo apresenta um botão para reproduzir a mídia cujo ícone pertence a um conjunto de ícones que representam os botões de reprodução da mídia. Exemplos:
3. O protótipo apresenta um botão para pausar a mídia cujo ícone pertence a um conjunto de ícones utilizados para representar os botões de pausa da mídia. Exemplos:
4. O protótipo apresenta um botão para controlar o volume da mídia cujo ícone pertence a um conjunto de ícones que representam os botões de volume da mídia. Exemplos:
5. Opcionalmente, o protótipo:
   (a) Apresenta um botão para pular para a próxima mídia cujo ícone pertence a um conjunto de ícones usados para representar botões de pular mídia. Exemplos:
   (b) Apresenta um botão para pular à mídia anterior cujo ícone pertence a um conjunto de ícones usados para representar os botões da mídia anterior. Exemplos:
   (c) Apresenta uma lista de mídias para reproduzir.

Informações relacionadas: –
Heurísticas relacionadas: Videoconferência, Lista

Tabela A.20: Heurística HEU2-11 – Notícias.

ID: HEU2-11

Nome: **Notícias**
Sinônimos: –
Definição: Representa uma lista de notícias.
Explicação: Um protótipo do tipo Notícias apresenta uma lista de eventos recentes e relevantes.
Exemplo: Um usuário lê notícias em um portal de notícias.
Checklist:

1. O protótipo apresenta uma lista de objetos de uma entidade Notícia do domínio da aplicação.
2. Cada item da lista apresenta o título da notícia.
3. Cada item da lista apresenta o texto da notícia.
4. Cada item da lista apresenta um texto representando a data e a hora da notícia.
5. Opcionalmente, cada item da lista apresenta uma imagem.

Informações relacionadas: –
Heurísticas relacionadas: Lista

Tabela A.21: Heurística HEU2-12 – Página de Produto.

ID: HEU2-12

Nome: **Página de Produto**
Sinônimos: Detalhes
Definição: Apresenta informação detalhada sobre um produto.
Explicação: Este tipo de protótipo apresenta informação detalhada sobre um produto em catálogos ou aplicações de e-commerce.
Exemplo: Um usuário visualiza um sapato e seus detalhes de cores e tamanhos disponíveis em um site de comércio eletrônico de calçados. Ele também verifica o preço, o material que compõe o produto e as instruções de limpeza.
Checklist:

1. O protótipo apresenta as informações de título, imagem, preço e descrição do produto.

2. Opcionalmente, o protótipo apresenta o preço de venda, opções de tamanho e cores, fotos adicionais, quantidade em estoque, prazo de entrega, formulário para entrada da quantidade do item ou uma lista de produtos relacionados.

3. O protótipo tem um botão de Adicionar ao Carrinho com um rótulo usado para descrever botões de Adicionar ao Carrinho (exemplos: alugar, comprar, fazer reserva, solicitar, selecionar, adicionar ao carrinho) ou com um ícone que descreve botões de Adicionar ao Carrinho.

   Exemplos:

Informações relacionadas: –
Heurísticas relacionadas: Visualização, Mestre-Detalhe

Tabela A.22: Heurística HEU2-13 – Perfil de Usuário.

ID: HEU2-13

Nome: **Perfil de Usuário**
Sinônimos: Perfil
Definição: Apresenta informações sobre a conta do usuário que está usando a aplicação.
Explicação: Um protótipo de Perfil de Usuário mostra informações sobre um usuário e suas preferências em relação ao aplicativo de software.
Exemplo: Um usuário define a restrição de acesso para conteúdo impróprio para idade em um aplicativo de serviço de *streaming*.
Checklist:

1. O protótipo apresenta informações que representam atributos de uma entidade de Usuário que faz parte da especificação da aplicação.

2. O protótipo apresenta informações que representam as preferências do usuário e opções de personalização relacionadas à conta do usuário.

3. Opcionalmente, o protótipo apresenta essas informações como um formulário de entrada para ilustrar como os usuários podem adicionar ou editar informações.

Informações relacionadas:

1. Entidades do domínio da aplicação e suas respectivas propriedades.

Heurísticas relacionadas: Cadastro, Alteração, Visualização

Tabela A.23: Heurística HEU2-05 – Quiz.

ID: HEU2-14

Nome: **Quiz**
Sinônimos: –
Definição: Um questionário com respostas corretas pré-definidas.
Explicação: Um protótipo do tipo Quiz representa interfaces onde o usuário deve acertar a resposta correta.
Exemplo: Um professor usa um Quiz para avaliar a aprendizagem dos estudantes.
Checklist:

1. O protótipo apresenta um texto para representar a pergunta.
2. O protótipo apresenta um campo para representar a resposta, que pode ser uma lista de respostas alternativas para seleção ou um campo de texto.
3. O protótipo apresenta um botão de envio com um rótulo usado para descrever botões de submissão da resposta (exemplos: Verificar, Enviar, Submeter) ou com um ícone que descreve botões de submissão da resposta (exemplos:).

Informações relacionadas: –
Heurísticas relacionadas: Lista

Tabela A.24: Heurística HEU2-15 – Registro de Conta.

ID: HEU2-15

Nome: **Registro de Conta**
Sinônimos: Sign in, sign up, registro de usuário
Definição: Criação de uma conta de usuário para acesso a funcionalidades de um software.
Explicação: Ilustra a criação de uma conta de acesso feita por um usuário de um software para acessar recursos personalizados ou restritos.
Exemplo: Um usuário se cadastra em um site de um restaurante para pedir uma pizza.
Checklist:

1. O protótipo apresenta um formulário de entrada de dados com um ou mais campos cujos valores não estão preenchidos.
2. Os campos do formulário representam os atributos da entidade que faz parte da especificação da aplicação relativa a usuários.
3. O formulário tem um campo para identificação do usuário, um campo para senha e um campo para redigitar a senha.
4. O rótulo do campo de identificação de usuário é usado para descrever campos de identificação do usuário. Exemplos: id, login, e-mail, email, identificação, cpf, nome, usuário, matrícula, nome de usuário, telefone.
5. O rótulo do campo de senha é usado para descrever campos de senha. Exemplos: senha, digite a senha.
6. O rótulo do campo de redigitar a senha é usado para descrever campos de redigitação de senha. Exemplos: redigite a sua senha, confirmar senha, confirme sua senha.
7. O formulário tem um botão de registro e seu rótulo descreve botões de registro de conta (exemplos: cadastrar, confirmar, confirmar cadastro, cadastrar-se, finalizar, continuar, criar conta, finalizar cadastro, concluir, registrar) ou seu ícone representa botões de registro de conta. Exemplos: .

Informações relacionadas:

1. Entidades do domínio da aplicação e suas respectivas propriedades.

Heurísticas relacionadas: Cadastro

Tabela A.25: Heurística HEU2-16 – Tutorial.

ID: HEU2-16

Nome: **Tutorial**
Sinônimos: Onboarding
Definição: Um protótipo do tipo Tutorial representa uma tela de ajuda ao usuário durante o primeiro uso de um aplicativo.
Explicação: Um protótipo de Tutorial representa telas planejadas para mostrar aos usuários como utilizar o aplicativo, suas principais funcionalidades e benefícios.
Exemplo: Um protótipo do tipo Tutorial mostra como começar a usar um aplicativo de troca de mensagens.
Checklist:

1. Um protótipo de Tutorial deve ter pelo menos uma das seguintes características:
   (a) Oferecer opções iniciais de personalização para um início rápido.
   (b) Mostrar a funcionalidade da chave do aplicativo e uma dica para começar a usá-la.
   (c) Apresentar uma tela Modal que mostra uma dica e pergunta se o usuário quer saber mais a respeito.
   (d) Apresentar uma tela explicativa mostrando os principais recursos do aplicativo.
   (e) Apresentar um carrossel mostrando os principais benefícios do aplicativo.

Informações relacionadas: –
Heurísticas relacionadas: Configurações

Tabela A.26: Heurística HEU2-17 – Videoconferência.

ID: HEU2-17

Nome: **Videoconferência**
Sinônimos:
Definição: Funcionalidade de videoconferência.
Explicação: Um protótipo de videoconferência representa um encontro entre pessoas em diferentes locais mediados por um computador, onde recebem e transmitem dados de áudio e vídeo.
Exemplo: Um usuário participa de uma reunião de trabalho usando o aplicativo de videoconferência de sua empresa.
Checklist:

1. O protótipo apresenta um componente Video Player em sua parte central.
2. O protótipo apresenta um ou mais componentes da webcam para representar os participantes da videoconferência.
3. O protótipo apresenta um botão para ativar ou desativar a câmera e o seu ícone representa botões de câmera. Exemplos: .
4. O protótipo apresenta um botão para ativar ou desativar o microfone e o seu ícone representa botões de microfone. Exemplos: .
5. O protótipo apresenta um botão para encerrar chamada e o seu ícone representa botões de chamada. Exemplos: .
6. Opcionalmente, o protótipo apresenta uma lista de participantes.
7. Opcionalmente, o protótipo apresenta a funcionalidade de chat.

Informações relacionadas: –
Heurísticas relacionadas: Chat, Lista

Apêndice B

Termo de Consentimento Livre e Esclarecido – TCLE

Prezado Participante,

Você está sendo convidado(a) a participar de uma pesquisa que avaliará a utilidade de uma ferramenta de apoio à inspeção funcional de protótipos de interface de usuário. Sua participação nesta pesquisa não é obrigatória. Os pesquisadores responsáveis pelo estudo poderão fornecer quaisquer esclarecimentos sobre este projeto.

Título da Pesquisa: Uma Abordagem Assistida de Inspeção Funcional de Protótipos de Interface de Software

Pesquisador Responsável: Gretchen Torres de Macedo Endereço: Universidade Federal do Amazonas Instituto de Computação Av. Rodrigo Otávio, nº 6.200, Campus Universitário Senador Arthur Virgílio Filho. Setor Norte do Campus Universitário Manaus - AM - CEP 69.077 – 000, e-mail gtmacedo@icomp.ufam.edu.br.

Orientador: Bruno Freitas Gadelha Endereço: Universidade Federal do Amazonas Instituto de Computação Av. Rodrigo Otávio, nº 6.200, Campus Universitário Senador Arthur Virgílio Filho. SETOR NORTE do Campus Universitário Manaus - AM - CEP 69.077 – 000, e-mail bruno@icomp.ufam.edu.br.

1. **OBJETIVO GERAL:** O principal objetivo deste projeto de pesquisa é reduzir o tempo e esforço de desenvolvimento de software através da detecção antecipada de defeitos, ainda na fase de elicitação de requisitos e

prototipação da interface. Pretende-se atingi-lo através de uma abordagem de inspeção de protótipos de interface com foco na funcionalidade.

2. **OBJETIVOS ESPECÍFICOS:** Desenvolver uma abordagem de inspeção funcional de protótipos de interface; Avaliar o desempenho da abordagem desenvolvida em relação a eficácia e eficiência na detecção de defeitos relacionados à funcionalidade.

3. **RISCOS:** (a) Toda pesquisa realizada com seres humanos apresenta riscos. No entanto, os riscos apresentados nesta pesquisa são mínimos, pois as tecnologias utilizadas não alteram aspectos fisiológicos, psicológicos ou sociais dos participantes. As atividades não tratam de aspectos relacionados à pessoa em si, mas sim às técnicas e processos utilizados. Ressaltamos que em nenhuma atividade será usada amostra biológica, pois a pesquisa será realizada com seres humanos e não em seres humanos. Ainda assim, existe o risco de que o participante se sinta intimidado a participar das atividades de pesquisa. Quanto a isso, informamos que a participação nesta pesquisa é totalmente voluntária. Se em algum momento da condução dos estudos, o participante sentir-se constrangido devido à não compreensão das perguntas, de termos ou expressões utilizadas, o pesquisador responsável poderá ser consultado para esclarecer quaisquer dúvidas. Aos participantes é assegurado o direito de não realizar qualquer atividade com a qual não se sinta confortável e de não responder a qualquer pergunta das entrevistas e questionários, caso abordem algum tópico sensível. Também é garantido ao participante o direito de se retirar da atividade de pesquisa a qualquer momento que desejar. Neste caso, os dados coletados deste participante até o momento da desistência serão desconsiderados. O participante também poderá solicitar, a qualquer momento posterior à coleta de dados dos estudos, a exclusão de seus dados e informações prestadas durante os estudos, de forma integral ou parcial. Caso o participante venha a sofrer algum dano/abalo psicológico decorrente da participação nas atividades da pesquisa, o encaminharemos ao Centro de Serviço de Psicologia Aplicada da UFAM (CSPA). Informamos que sua identidade será mantida em sigilo. O questionário é preenchido de maneira anônima, sem dados pessoais de identificação do respondente, não

sendo possível ao pesquisador identificar os participantes que o responderam. Havendo registro fotográfico ou gravações, a anonimização dos participantes será garantida através do tratamento das imagens. Eventuais informações prestadas pelo participante nas respostas que possam vir a identificá-lo serão removidas. Informações sensíveis que venham a ser prestadas serão ocultadas, pelos pesquisadores, das respostas e do material elaborado pelos participantes. Os resultados da pesquisa serão analisados e publicados, mas sua identidade não será divulgada, sendo guardada em sigilo. Os dados obtidos serão anonimizados e analisados de maneira agrupada. Os artefatos que venham a ser elaborados poderão ser utilizados no desenvolvimento da ferramenta que é objeto do projeto de pesquisa. Estes dados serão utilizados somente para fins de pesquisa científica, mantendo o sigilo acerca de sua identificação.

4. **CONFIDENCIALIDADE:** Toda informação coletada nesta pesquisa possui caráter confidencial, e seu nome e o da organização não serão identificados de modo algum, a não ser em caso de autorização explícita para esse fim.

5. **BENEFÍCIOS:** Através desta pesquisa, espera-se oferecer como benefício da atividade de inspeção funcional de protótipos de interface, que os participantes consigam identificar defeitos contidos em protótipos de interface de usuário de maneira mais rápida. Além disso, espera-se oferecer como benefício geral da pesquisa um software de apoio à inspeção funcional de protótipos de interface, que possa auxiliar equipes de desenvolvimento de software na detecção antecipada de defeitos relacionados às funcionalidades representadas nos protótipos e, ocasionando a diminuição do tempo e esforço de desenvolvimento necessários à correção de defeitos.

6. **PARTICIPAÇÃO:** Você deverá ser um profissional que atue na área de desenvolvimento de sistemas para responder ao questionário ou participar das entrevistas. Nas atividades de elaboração de protótipos de baixa fidelidade da interface de usuário e nos ciclos de melhoria do artefato de software desenvolvido, você deverá ser estudante de graduação dos cursos

de Engenharia de Software ou Ciência da Computação da UFAM ou de outra instituição de ensino superior no Brasil. O participante deverá possuir conhecimento prévio sobre engenharia de requisitos e prototipação. Além disso, deve possuir a maioridade (no mínimo 18 anos). Nas atividades de inspeção de protótipos de interface de usuário utilizando o artefato de software desenvolvido, você deverá ser estudante de graduação dos cursos de Engenharia de Software ou Ciência da Computação da UFAM ou de outra instituição de ensino superior no Brasil. O participante deverá possuir conhecimento prévio sobre Engenharia de Requisitos e Prototipação de Interface. Além disso, deve possuir a maioridade (no mínimo 18 anos).

7. **PROCEDIMENTOS:** Você participará da pesquisa através de uma das seguintes maneiras: 1) através da resposta a um questionário sobre o uso de prototipação no seu ambiente de trabalho, 2) da elaboração de protótipos de baixa fidelidade da interface de aplicações ou; 3) através de uma atividade de inspeção de protótipos de interface de usuário utilizando a ferramenta de software desenvolvida e da avaliação desta por meio de questionários e entrevistas. Todos os dados fornecidos em entrevistas e questionários serão descartados após a tabulação dos dados, assim como as informações prestadas durante o uso e avaliação da ferramenta. Os protótipos de tela serão utilizados para elaboração da ferramenta e serão descartados ao final dos ciclos de desenvolvimento.

8. **CUSTOS:** Todos os materiais necessários para realização da pesquisa serão fornecidos pelos próprios pesquisadores, de forma que não acarrete custos para os participantes. Eventuais despesas, tais como transporte e alimentação, ou danos materiais que venham a ocorrer ao participante decorrente de sua participação nos estudos, serão reembolsados integralmente e em espécie ao final da atividade de pesquisa, como prevê o item IV.3.g, da Resolução CNS nº466 de 2012. Além disso, está assegurado o direito a indenizações e qualquer tipo de assistência necessária para reparação a qualquer prejuízo causado pela pesquisa, segundo a resolução CNS nº 466 de 2012, IV.3.h, IV.4.c e V.7.

9. Para qualquer outra informação, o (a) Sr.(a) poderá entrar em contato com a pesquisadora responsável, professor orientador nos contatos acima mencionados ou poderá entrar em contato com o Comitê de Ética em Pesquisa – CEP/UFAM, na Rua Teresina, 495, Adrianópolis, telefone (92) 3305-1181 ramal 2004, celular (92) 9171-2496, e-mail: cep.ufam@gmail.com.

Consentimento Pós–Informação

Eu, [nome do participante], fui informado sobre o que o pesquisador quer fazer e porque precisa da minha colaboração, e entendi a explicação. Por isso, eu concordo em participar do projeto, sabendo que não vou ganhar nada e que posso sair quando quiser. Este documento é emitido em duas vias que serão ambas assinadas por mim e pelo pesquisador, ficando uma via com cada um de nós.

Declaração de Consentimento: Li ou alguém leu para mim as informações contidas neste documento antes de assinar este termo de consentimento. Declaro que toda a linguagem técnica utilizada na descrição deste estudo de pesquisa foi explicada satisfatoriamente e que recebi respostas para todas as minhas dúvidas. Confirmo também que recebi uma cópia deste Termo de Consentimento Livre e Esclarecido. Compreendo que sou livre para me retirar do estudo em qualquer momento, sem qualquer penalidade. Declaro ter mais de 18 anos e dou meu consentimento de livre e espontânea vontade para participar deste estudo.

Caso o participante esteja impossibilitado de assinar:
Eu [nome do auxiliar do participante], abaixo assinado, confirmo a leitura do presente termo na integra para o(a) participante [nome do participante], o(a) qual declarou na minha presença a compreensão plena e aceitação em participar desta pesquisa, o qual utilizou a sua impressão digital (abaixo) para confirmar a participação.

Apêndice C

Artefatos do Primeiro Ciclo de Design

Este apêndice apresenta o cenário do problema apresentado aos especialistas durante a atividade de avaliação do catálogo de Heurísticas de Design.

C.1 Cenário da Avaliação das Heurísticas por Especialistas

A empresa Crash Motos comercializa motos novas e usadas. Ela solicitou à nossa empresa o desenvolvimento de um *website* para venda de suas motos online para todo o Brasil. Além de motos, o site também comercializará acessórios e peças de reposição.

O website deve apresentar uma página inicial com destaques e promoções. As promoções serão os produtos que possuem preço atual inferior ao preço original. Os destaques serão produtos específicos selecionados pela empresa para que sejam exibidos na página inicial. As motos estarão organizadas em categorias: *custom*, *sport*, *off-road*, *scooter*. Outras categorias poderão surgir conforme os modelos comercializados.

O site deve oferecer uma opção de busca de motos e acessórios. As listas de motos poderão ser filtradas por suas características: marca, modelo, cilindradas, preço, quilometragem.

Ao visualizar uma moto, o site deverá exibir seu preço, preço original, marca, modelo, quilometragem, categoria e cilindradas. Também devem ser exibidos os acessórios compatíveis com o modelo. Será possível informar o CEP para verificar o preço do frete e prazo estimado de entrega dos produtos.

O sistema deve oferecer a funcionalidade de chat, para que os usuários tirem dúvidas sobre motos e acessórios com vendedores especializados.

Ao selecionar uma moto ou acessório para compra, o sistema deve exibir uma página de carrinho de compra, onde os produtos pré-selecionados serão listados. Esta página deve mostrar uma lista de sugestões de produtos relacionados aos que estão no carrinho. Além disso, esta página oferecerá uma opção para concluir a compra.

Na página de finalização da compra, o site deve apresentar ao usuário as opções de pagamento. Também deve ser solicitado ao usuário o endereço para entrega dos produtos.

Para realizar uma compra no site, é necessário que o usuário se cadastre ou realize o login caso já possua cadastro. Os dados para cadastro de usuário são: nome completo, CPF ou CNPJ, e-mail, senha e data de nascimento. É necessário também que o usuário cadastre pelo menos um endereço de entrega.

Além do *website*, deve ser desenvolvido um sistema de apoio para que a empresa Crash Motos possa cadastrar os dados que serão exibidos no site: motos, acessórios, categorias, destaques, tabela de frete e prazos de entrega. O cadastro de motos deve registrar marca, modelo, ano, quilometragem, cilindradas. Deve permitir também o cadastro de acessórios e peças relacionados ao modelo.

C.2 Roteiro de Atividades do Grupo Focal

1. Escolha 3 das seguintes funcionalidades e elabore um protótipo de interface para cada uma delas. Utilize o catálogo de heurísticas de design para auxiliar na prototipação das funcionalidades escolhidas.

   Link para catálogo: https://sites.google.com/view/design-heuristics/

2. Sugestão: utilize a ferramenta Balsamiq (https://balsamiq.com/wireframes/desktop/)

3. Após elaborar os protótipos, responda ao questionário acessando o link: https://forms.gle/Aj5M9iXynQHQkgke6

Funcionalidades para prototipação (selecione três):
Sistema de apoio

- Categorias
- Acessórios
- Motos
- Destaques
- Tabela de frete e prazos de entrega

Website

- Login
- Cadastro de usuário
- Página inicial
- Busca
- Página de produto
- Carrinho de compra
- Finalização da compra
- Chat

Apêndice D

Artefatos do Segundo Ciclo de Design

D.1 Protótipos de interface inspecionados no experimento controlado

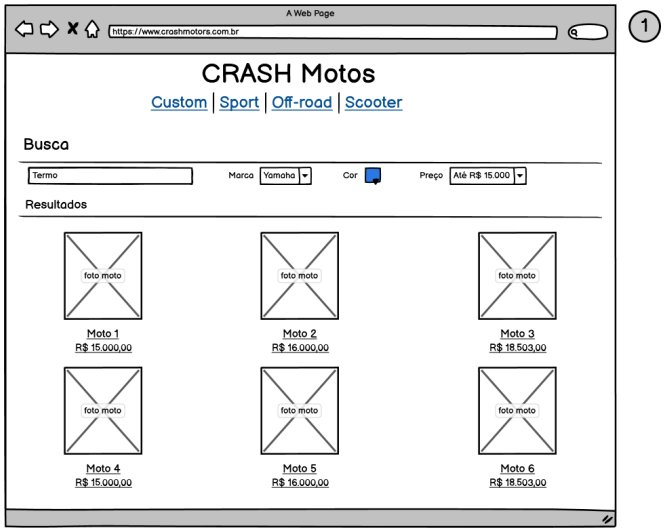


Figura D.1: Protótipo de IU usado no estudo - Página de Busca.

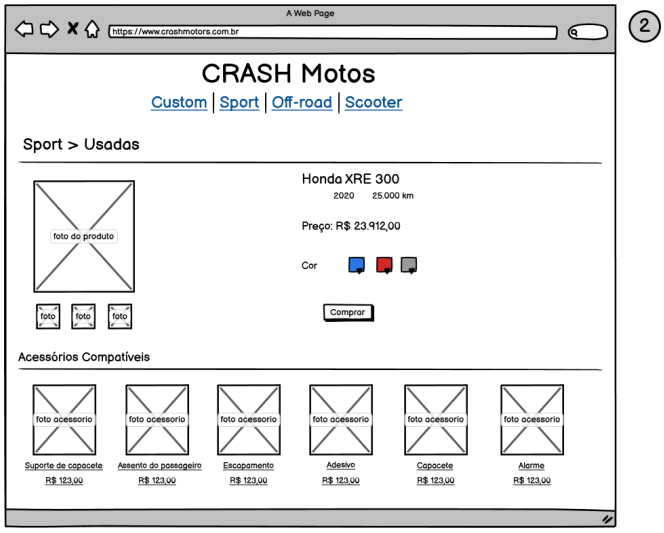


Figura D.2: Protótipo de IU usado no estudo - Página de Produto.

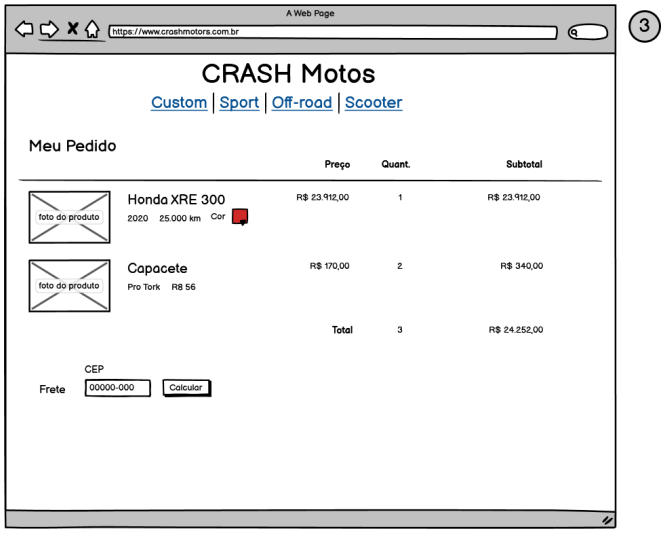


Figura D.3: Protótipo de IU usado no estudo - Página de Carrinho de Compra.

D.2 Instruções para condução da inspeção

Inspeção de Funcionalidades Conhecidas

Nesta atividade, você realizará a inspeção de protótipos. Esta inspeção é baseada no aspecto funcional, ou seja, você vai tentar identificar o tipo de funcionalidade representada nos protótipos e problemas que afetem a sua funcionalidade.

Processo de inspeção de funcionalidades conhecidas

O processo a ser seguido nesta atividade é uma adaptação de processo proposto por Sauer et al. (2000). As etapas deste processo estão descritas abaixo.

1. Detecção de defeitos - individual

Nesta etapa, cada membro do grupo analisa cada protótipo de maneira individual.

a. Anote seu nome e o horário de início.

b. Leia o cenário.

c. Selecione um protótipo de interface para inspecionar.

d. Identifique a funcionalidade do protótipo.

e. Utilizando a ferramenta de inspeção fornecida (*checklist* ou catálogo de heurísticas), verifique se a funcionalidade representada está de acordo com a especificação.

f. Anote os defeitos encontrados na planilha individual. g. Repita esses passos para os demais protótipos.

h. Anote o horário de término.

2. Coleção de defeitos

Nesta etapa, a equipe faz a consolidação dos problemas e remoção de duplicados.

a. A equipe se reúne e anota seus nomes e o horário de início da reunião na Planilha de Grupo.

b. A equipe anota os problemas encontrados na Planilha de Grupo, tomando o cuidado de remover os problemas duplicados.

3. Discriminação de defeitos

Nesta etapa, a equipe se reúne para revisar os problemas consolidados e classificá-los em Defeitos ou Falso-positivos.

a. A equipe analisa cada problema.

b. Após a análise, a equipe anota a classificação de cada problema na coluna Classificação final da Planilha de Grupo, junto com a justificativa desta classificação.

c. Após analisar e classificar todos os problemas, a equipe anota o horário de término da reunião.

Apêndice E

Artefatos do Terceiro Ciclo de Design

E.1 Instruções para Realização da Atividade de Inspeção

Tabela E.1: Designação de equipes de inspetores e desenvolvedores

Inspetores do	**Inspecionaram protótipos do**
Grupo 1	Grupo 2
Grupo 2	Grupo 5
Grupo 5	Grupo 6
Grupo 6	Grupo 7
Grupo 7	Grupo 8
Grupo 8	Grupo 10
Grupo 10	Grupo 11
Grupo 11	Grupo 1

- A atividade de inspeção é **individual**. Cada membro da equipe vai fazer a inspeção funcional de cada um dos protótipos de interface do projeto recebido. Por exemplo, cada integrante do Grupo 1 vai inspecionar cada um dos protótipos de tela do projeto do Grupo 2.

- Obtenha os protótipos de interface e a especificação do projeto no fórum da atividade.
- Prazo para entrega: 24/10/2024
- Deve ser utilizada a ferramenta GUIPSI para inspecionar funcionalmente as telas da sua aplicação. Para isso, deve-se seguir os seguintes passos:
    1. Abrir a aplicação, disponível em https://guipsi.vercel.app/
    2. Escolher a opção “Inspecionar protótipo de interface”
    3. Na tela de inspeção, preencher o nome do projeto e o nome da tela a ser inspecionada.
    4. Carregar a tela para inspeção na aplicação utilizando o botão Carregar Imagem.
    5. Opcionalmente, é possível obter uma recomendação utilizando o botão “Sugerir ferramentas de inspeção”.
    6. Selecionar uma ou mais ferramentas de inspeção na caixa “Carregue uma ou mais ferramentas de inspeção”. Os itens do *checklist* das ferramentas selecionadas serão carregados abaixo da caixa de seleção.
    7. Clicar no botão Editar Dados
    8. Realizar a inspeção do protótipo, respondendo a cada item do *checklist* na coluna “O seu protótipo atende ao requisito?”, e anotando os defeitos encontrados na coluna “Descrição do defeito”.
    9. Ao finalizar a inspeção, pressionar o botão "Parar edição".
    10. Em seguida, pressionar o botão “Salvar dados no relatório”.
    11. Após salvar os dados, pressionar o botão “Baixar relatório”.
    12. Repetir este procedimento para as demais telas do projeto.
- Enviar os relatórios gerados (1 para cada tela) no colabweb
- Responder o TCLE e o questionário (link disponível no colabweb)

E.2 Material da Aula - Inspeção Funcional de Protótipos de Interface de Usuário

Protótipos de Interface de Usuário

- Modelos preliminares da interface de um software
- Usados para
  - Demonstrar ideias
  - Explorar soluções
  - Facilitar comunicação
  - Alinhar expectativas
- Também usados para:
  - Auxiliar estimativas
  - Definição de sprints
  - Referência para desenvolvimento
  - Apoiar atividades de teste

Protótipos de UI - Exemplos

Inspeção Funcional de Protótipos de Interface de Usuário

- Checar se o protótipo representa corretamente a funcionalidade
- Funcionalidade descrita em algum documento de referência:
  - Requisitos
  - Histórias de usuário
  - Mapa de jornada do usuário
- Dois instrumentos:
  1. **Catálogo de Funcionalidades Recorrentes**
  2. **UIProtoCheck – checklist de uso geral**

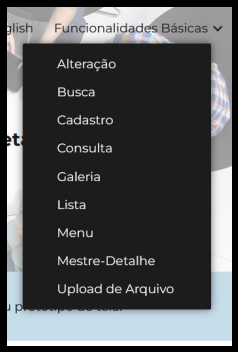
Instrumentos de Inspeção Funcional

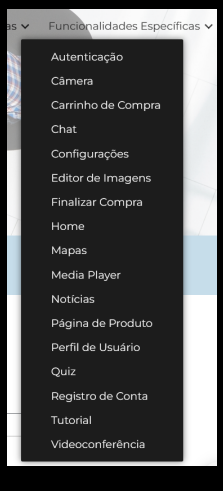
Catálogo de Funcionalidades Recorrentes: https://sites.google.com/view/design-heuristics

Instrumentos de Inspeção Funcional

Catálogo de Funcionalidades Recorrentes: https://sites.google.com/view/design-heuristics

1. Escolher a funcionalidade de corresponde ao protótipo
2. Inspecionar de acordo com o checklist da funcionalidade escolhida

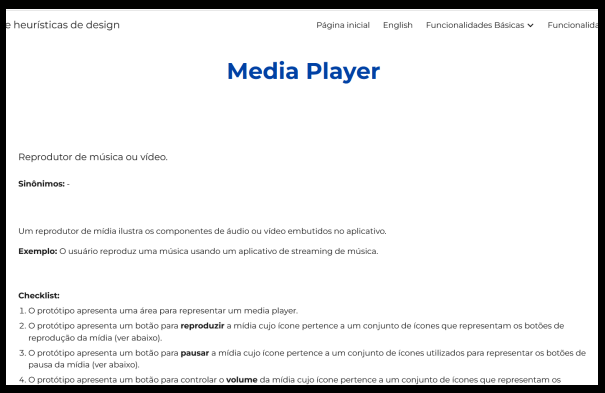




Instrumentos de Inspeção Funcional

Catálogo de Funcionalidades Recorrentes:
https://sites.google.com/view/design-heuristics

1. Escolher a funcionalidade de corresponde ao protótipo
2. Inspecionar de acordo com o checklist da funcionalidade escolhida



e heurísticas de design

Página inicial English Funcionalidades Básicas Funcionalida

Media Player

Reprodutor de música ou vídeo.

Sinônimos: -

Um reprodutor de mídia ilustra os componentes de áudio ou vídeo embutidos no aplicativo.

Exemplo: O usuário reproduz uma música usando um aplicativo de streaming de música.

Checklist:

1. O protótipo apresenta uma área para representar um media player.
2. O protótipo apresenta um botão para **reproduzir** a mídia cujo ícone pertence a um conjunto de ícones que representam os botões de reprodução da mídia (ver abaixo).
3. O protótipo apresenta um botão para **pausar** a mídia cujo ícone pertence a um conjunto de ícones utilizados para representar os botões de pausa da mídia (ver abaixo).
4. O protótipo apresenta um botão para controlar o **volume** da mídia cujo ícone pertence a um conjunto de ícones que representam os

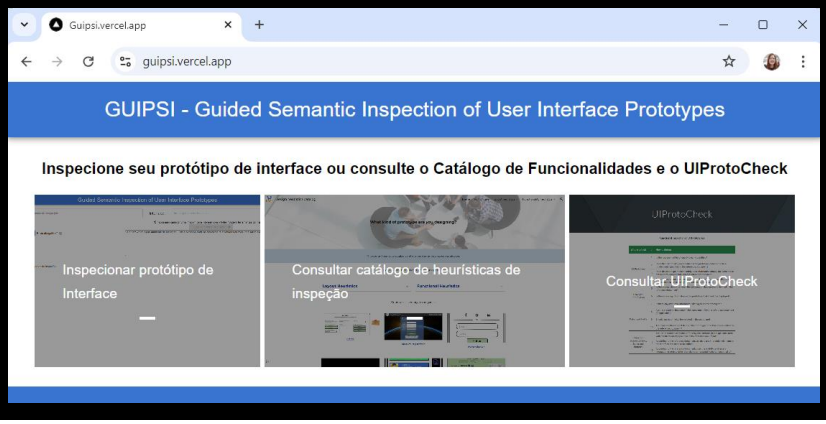
Instrumentos de Inspeção Funcional

UIProtoCheck – checklist de propósito geral pra inspeção funcional de protótipos de interface:
https://sites.google.com/view/uiprotocheck

Where to look	✓	How to detect
UI Prototype	1	Is the represented functionality clearly identified?
	2	Does the functionality represented in the prototype correspond to a functionality specified in the reference document?
	3	Does the prototype present ambiguous information among its elements or between its components and the reference document?
	4	Does the prototype use names, terms, and descriptions according to the corresponding functionality in the reference document?
Displayed Information	5	Is there consistency between the information displayed and defined in the reference document?
	6	Is there missing information in the prototype that should be displayed?
	7	Is there any additional information displayed on the prototype?
Data Input Fields	8	Are the prototype data entry fields consistent with the reference document (if applicable)?
	9	Should any input fields be included in the prototype?
	10	Are there additional input fields in the prototype other than those defined in the reference document?
Interaction elements (links, icons, and buttons)	11	Are the interaction elements offered by the interface prototype consistent with the functionality specified in the reference document?
	12	According to the reference document, are there any interaction elements in the interface that need to be added?
	13	According to the reference document, does the prototype offer any interaction elements other than those corresponding to its functionality?

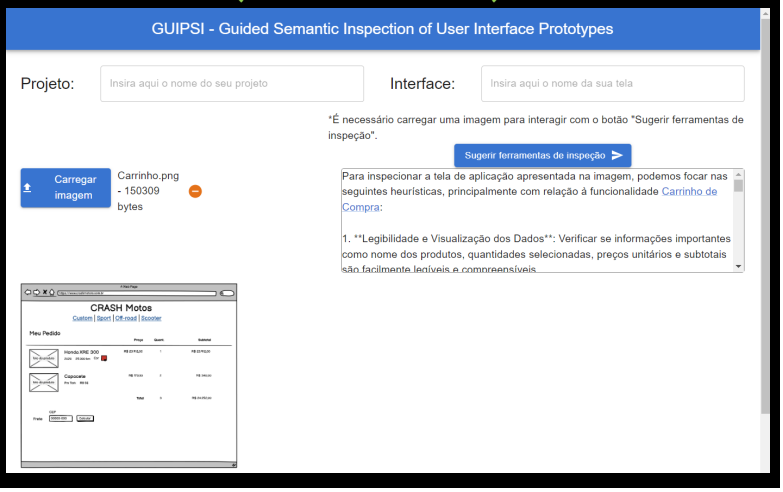
Ferramenta de Apoio à Inspeção Funcional – GUIPSI (usar Chrome)

- https://guipsi.vercel.app/



Ferramenta de Apoio à Inspeção Funcional - GUIPSI (usar Chrome)

- https://guipsi.vercel.app/



Ferramenta de Apoio à Inspeção Funcional - GUIPSI (usar Chrome)

- https://guipsi.vercel.app/

Carregue uma ou mais ferramentas de inspeção

Carrinho de Compra | Lista | UIProtoCheck

Nome da ferramenta de inspeção: Carrinho de Compra
Descrição: Lista de produtos ou serviços selecionados para compra em uma aplicação de e-commerce.
Sinônimos: Carrinho
Explicação: Um carrinho de compras permite que os usuários comprem mais de um item por vez, mais de uma instância de um item e armazenem os itens para comprar mais tarde.
Exemplo: Um usuário compra mantimentos em um aplicativo de supermercado.
Heurísticas relacionadas: Lista, Finalizar Compra

Nome da ferramenta de inspeção: Lista
Descrição: Exibe uma lista de objetos relacionados a uma entidade do domínio.
Sinônimos: lista vertical, lista horizontal
Explicação: Use este tipo de tela quando quiser ilustrar uma lista de objetos de uma entidade do domínio da aplicação.
Exemplo: Um usuário visualiza uma lista de filmes em um aplicativo de serviço de streaming.
Heurísticas relacionadas: Galeria, Busca, Mestre-Detalhe, Videoconferência

Nome da ferramenta de inspeção: UIProtoCheck
Descrição: Checklist genérico.
Sinônimos: -
Explicação: Use esse checklist para uma funcionalidade que não tenha checklist específico.
Exemplo: -
Heurísticas relacionadas: -

Ferramenta de Apoio à Inspeção Funcional - GUIPSI (usar Chrome)

https://guipsi.vercel.app/

Sinônimos: -
Explicação: Use esse checklist para uma funcionalidade que não tenha checklist específico.
Exemplo: -
Heurísticas relacionadas: -

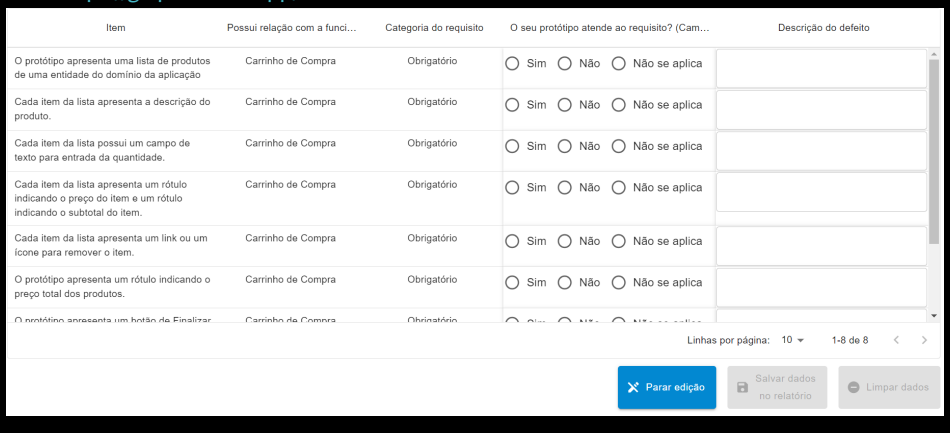
Item	Possui relação com a funcionali...	Categoria do requisito	O seu protótipo atende ao requisito? (Campo obri...	Descrição do defeito
O protótipo apresenta uma lista de objetos de uma entidade do domínio da aplicação.	Lista	Obrigatório	Usuário não respondeu	
Cada item da lista apresenta valores de atributos do objeto.	Lista	Obrigatório	Usuário não respondeu	
O protótipo apresenta uma lista de produtos de uma entidade do domínio da aplicação	Carrinho de Compra	Obrigatório	Usuário não respondeu	
Cada item da lista apresenta a descrição do produto.	Carrinho de Compra	Obrigatório	Usuário não respondeu	
Cada item da lista possui um campo de texto para entrada da quantidade.	Carrinho de Compra	Obrigatório	Usuário não respondeu	
Cada item da lista apresenta um rótulo indicando o preço do item e um rótulo indicando o subtotal do item.	Carrinho de Compra	Obrigatório	Usuário não respondeu	
Cada item da lista apresenta um link ou um ícone	Carrinho de Compra	Obrigatório	Usuário não respondeu	

Linhas por página: 10 1-10 de 23

Editar dados | Salvar dados no relatório | Limpar dados

Ferramenta de Apoio à Inspeção Funcional - GUIPSI (usar Chrome)

https://guipsi.vercel.app/



Item	Possui relação com a funci...	Categoria do requisito	O seu protótipo atende ao requisito? (Cam...	Descrição do defeito
O protótipo apresenta uma lista de produtos de uma entidade do domínio da aplicação	Carrinho de Compra	Obrigatório	Sim Não Não se aplica	
Cada item da lista apresenta a descrição do produto.	Carrinho de Compra	Obrigatório	Sim Não Não se aplica	
Cada item da lista possui um campo de texto para entrada da quantidade.	Carrinho de Compra	Obrigatório	Sim Não Não se aplica	
Cada item da lista apresenta um rótulo indicando o preço do item e um rótulo indicando o subtotal do item.	Carrinho de Compra	Obrigatório	Sim Não Não se aplica	
Cada item da lista apresenta um link ou um ícone para remover o item.	Carrinho de Compra	Obrigatório	Sim Não Não se aplica	
O protótipo apresenta um rótulo indicando o preço total dos produtos.	Carrinho de Compra	Obrigatório	Sim Não Não se aplica	

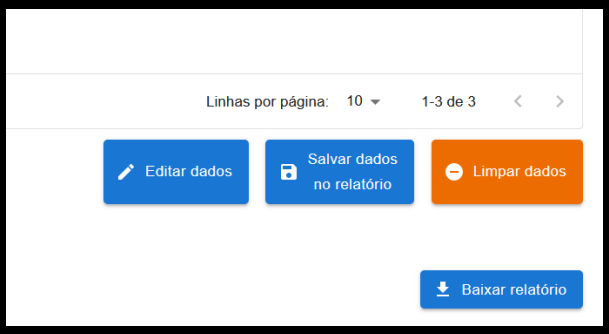
Linhas por página: 10 1-8 de 8

Parar edição | Salvar dados no relatório | Limpar dados

Ferramenta de Apoio à Inspeção Funcional - GUIPSI (usar Chrome)

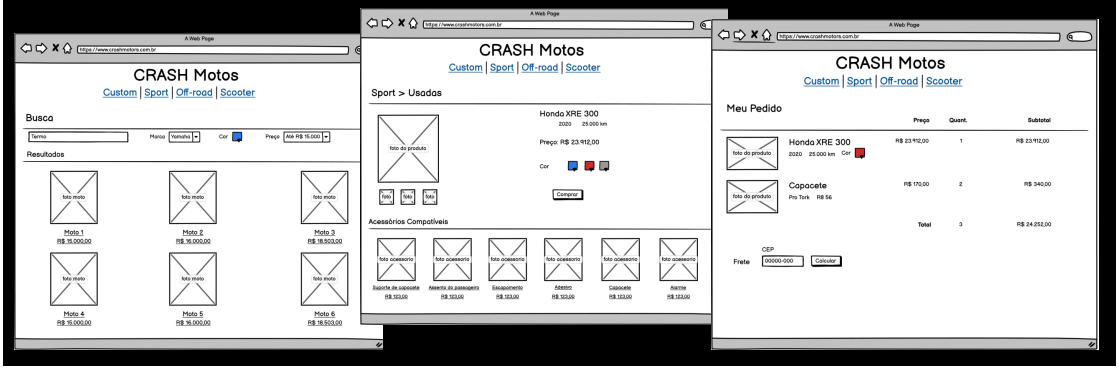
- https://guipsi.vercel.app/

1. Editar dados
2. Preencher relatório
3. Parar edição
4. Salvar dados no relatório
5. Baixar relatório



Exercício

Exercício: Inspeção de protótipos de interface



Atividade: Inspeção Funcional de protótipos de interface

- A inspeção é individual – cada integrante da equipe vai inspecionar cada uma das telas do projeto atribuído. Utilize o material no item "Fórum: Coloque aqui a Especificação e os protótipos de tela da sua aplicação" no colabweb.
  - Ex.: cada integrante do Grupo 1 vai inspecionar cada um dos protótipos de tela do projeto do Grupo 2
- **Prazo de entrega: 24/10/2024 23h59**

1. Inspecionar e salvar os relatórios de inspeção – 1 por tela
2. Enviar relatórios no colabweb (zip) – Item "Atividade :: Inspeção funcional de protótipos de interface"
3. Responder o questionário de avaliação da ferramenta - https://forms.gle/XtUk3TC53vqStJou8

Os Inspetores do	Vão inspecionar os protótipos do
Grupo 1	Grupo 2
Grupo 2	Grupo 5
Grupo 5	Grupo 6
Grupo 6	Grupo 7
Grupo 7	Grupo 8
Grupo 8	Grupo 10
Grupo 10	Grupo 11
Grupo 11	Grupo 1

Apêndice F

Artefatos do Estudo de Validação na Indústria

F.1 Carta Convite

Olá, tudo bem?

Gostaria de te convidar a participar de um estudo de avaliação da ferramenta GUIPFI. Ela serve para realizar inspeções funcionais rápidas em protótipos de interface de usuário.

Em quê consiste o estudo:

- Você vai selecionar três protótipos de interface de um projeto de software real no qual tenha atuado e utilizar a ferramenta para inspecioná-los.
- Após as inspeções, você responderá a um questionário de avaliação.

A duração estimada do estudo é de **20 a 25 minutos** para as inspeções e **15 minutos** para o questionário.

Seguem em anexo as instruções de inspeção. Agradeço imensamente sua participação!

Observação: Sua participação é voluntária e seus dados serão anonimizados. Os protótipos usados na inspeção serão descartados. Mais detalhes estão especificados no Termo de Consentimento incluído no questionário.

F.2 Instruções para realização do estudo

Inspeção Funcional de Protótipos de Interface de Usuário

- Olá! Obrigada por participar do estudo de avaliação da ferramenta **GUIPFI** !
- Você pode utilizar esta ferramenta para inspecionar seus protótipos de interface e encontrar defeitos funcionais.
- Seu funcionamento é bastante simples. Você coloca seu protótipo na ferramenta e escolhe um ou mais checklists para inspecioná-la.
- A ferramenta oferece checklists para funcionalidades básicas, específicas e um checklist genérico, caso seu protótipo não represente uma das funcionalidades.
- Se você estiver em dúvida sobre quais checklists usar para inspecionar seu protótipo, pode usar o recurso "Sugerir instrumentos de inspeção". Ele vai analisar a imagem que você colocou e indicar os checklists adequados.

Vamos lá!

Roteiro dos passos

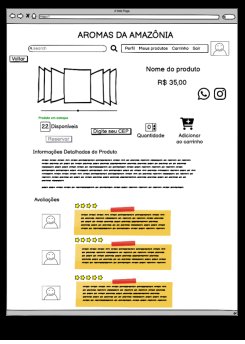
O estudo vai ser realizado em 10 passos, que são detalhados nos próximos slides!

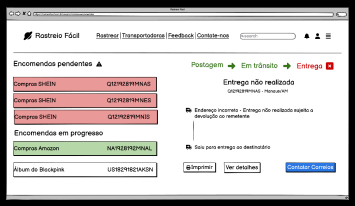
1. Selecionar 3 protótipos
2. Acessar a ferramenta e clicar na opção “Inspecionar protótipo de interface”
3. Preencher os nomes do projeto e do protótipo
4. Carregar a imagem
5. Obter sugestão (opcional)
6. Selecionar instrumentos de inspeção
7. Preencher o checklist
8. Salvar os dados e repetir passos 1 a 7 para os outros 2 protótipos
9. Baixar e enviar o relatório de inspeção para gtmacedo@gmail.com
10. Responder o questionário: https://forms.gle/deiJNHnA65q2QByw6

1. Preparação

Para começar, selecione três imagens de protótipos de tela de um projeto de software real

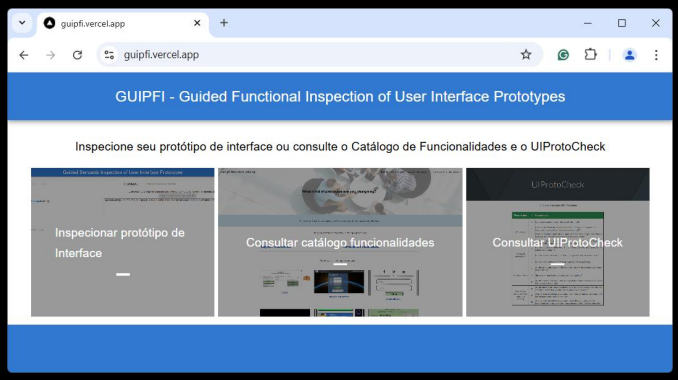
- Pode estar em desenvolvimento ou já ter sido desenvolvido





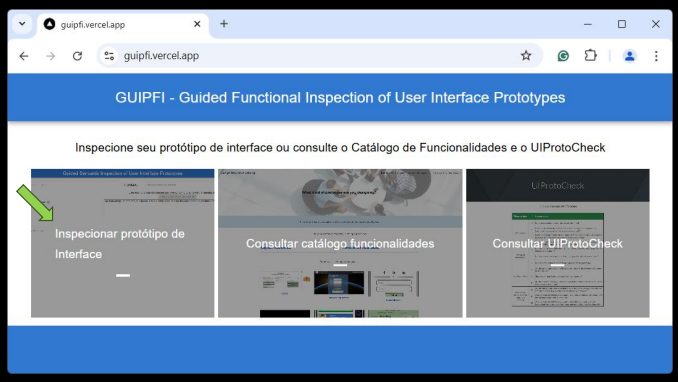
2. Acesse a ferramenta GUIPFI

https://guipfi.vercel.app/



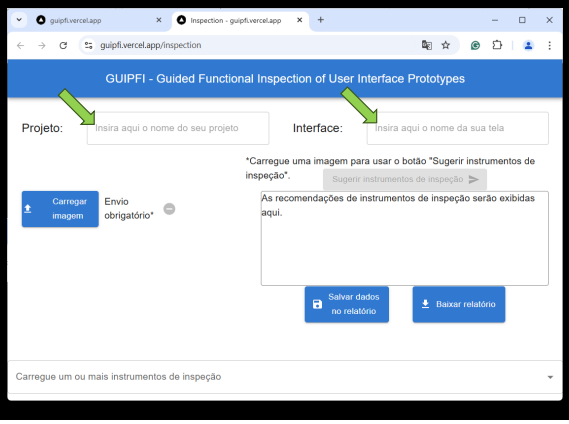
2. Acesse a ferramenta GUIPFI

- Clique na opção “Inspecionar protótipo de Interface”



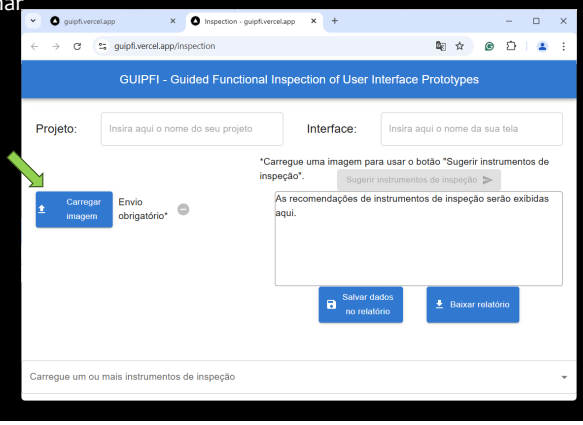
3. Nomes do projeto e protótipo

- Preencha o nome do seu projeto e o título do protótipo que você vai inspecionar



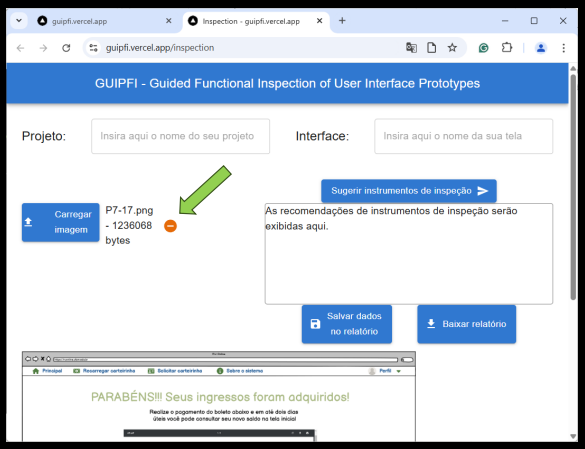
4. Carregar imagem

- Clique no botão “Carregar imagem” e informe a imagem do protótipo que você vai inspecionar



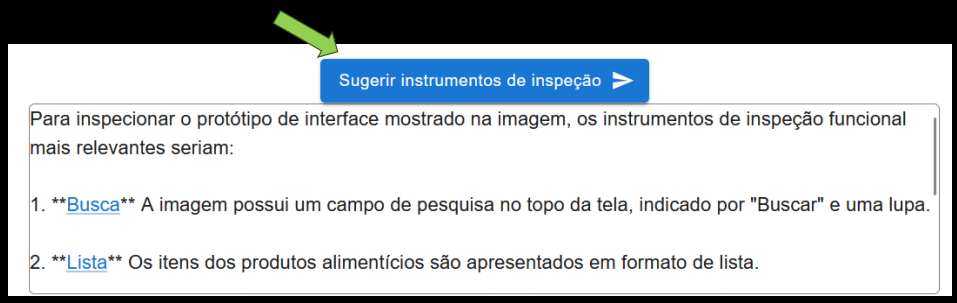
4. Carregar imagem

- Se quiser trocar de imagem, use o botão



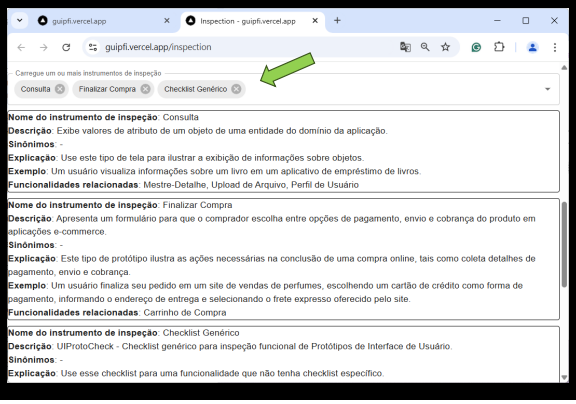
5. Obter sugestão (opcional)

- Se quiser obter recomendações de checklists para inspecionar seu protótipo , use o botão “Sugerir instrumentos de inspeção”



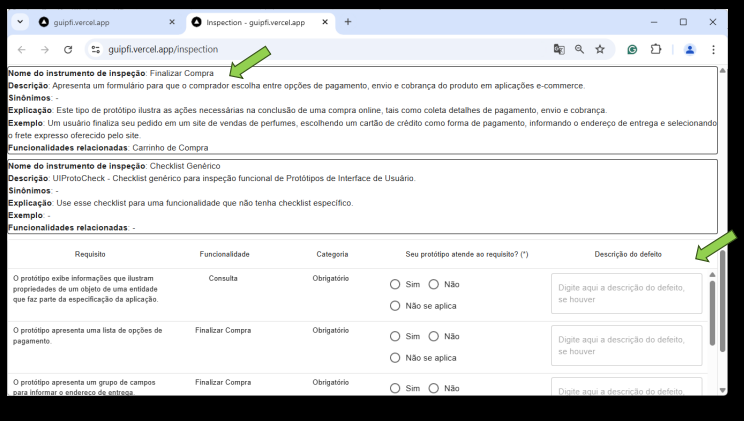
6. Seleção de instrumentos de inspeção

- Escolha um ou mais instrumentos de inspeção na caixa “Carregue um ou mais instrumentos de inspeção” para inspecionar o seu protótipo



6. Seleção de instrumentos de inspeção

A ferramenta vai exibir um resumo de cada instrumento e os itens do checklist logo abaixo:



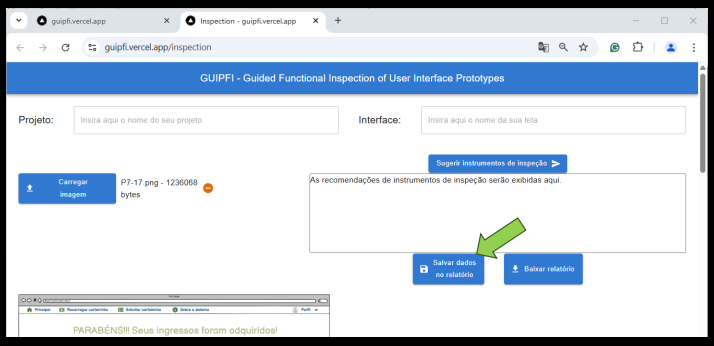
7. Preencha o checklist

- Preencha cada item do checklist informando se o protótipo atende ao requisito e a descrição do defeito, caso encontrado

Item	Possui relação com a funcio…	Categoria do requisito	O seu protótipo atende ao requisito? (Campo…	Descrição do defeito
O formulário tem um botão de busca cujo rótulo está em um conjunto como este: (buscar, busca, pesquisar), ou que tem um ícone que representa a ação de buscar.	Busca	Obrigatório	◯ Sim ◯ Não ◯ Não se aplica	Digite aqui a descrição do defeito, se houver
O protótipo apresenta uma lista de objetos de uma entidade da aplicação.	Lista	Obrigatório	◯ Sim ◯ Não ◯ Não se aplica	Digite aqui a descrição do defeito, se houver
Cada item da lista apresenta valores de atributos dos objetos.	Lista	Obrigatório	◯ Sim ◯ Não ◯ Não se aplica	Digite aqui a descrição do defeito, se houver
O protótipo apresenta uma lista elementos em uma lista lateral, barra superior, barra inferior ou na forma de botões.	Menu	Obrigatório	◯ Sim ◯ Não	Digite aqui a descrição do defeito, se houver

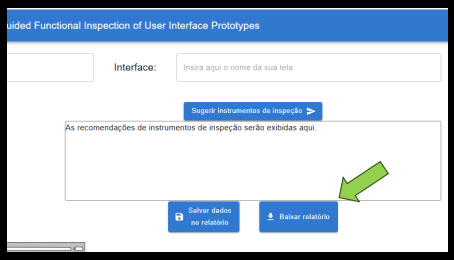
8. Salvar

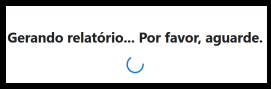
- Clique no botão “Salvar dados no relatório”
- Repita os passos 1 a 8 para a segunda e a terceira inspeção



9. Baixar e Enviar o relatório

- Após realizar e salvar a 3ª inspeção, clique no botão “Baixar relatório”





- Envie o relatório para “gtmacedo@icomp.ufam.edu.br”

10. Preencher o questionário

- Para finalizar, precisamos saber a sua opinião sobre a ferramenta GUIPFI
- Por favor, preencha o questionário disponível em:

https://forms.gle/deiJNHnA65q2QByw6

Sua participação é muito importante!
Muito obrigada!