



UNIVERSIDADE DO ESTADO DO AMAZONAS
ESCOLA SUPERIOR DE TECNOLOGIA
CURSO DE ENGENHARIA DE COMPUTAÇÃO



**INTEGRAÇÃO ENTRE *DESIGN THINKING* E CICLO DE VIDA DO
DESENVOLVIMENTO DE SOFTWARE: PROPOSTA DE UM MODELO
CONVERGENTE APLICADO AO DESENVOLVIMENTO DE UMA
PLATAFORMA WEB**

NICOLAS FERNANDES DE LIMA

Manaus – AM

2026

NICOLAS FERNANDES DE LIMA

**INTEGRAÇÃO ENTRE *DESIGN THINKING* E CICLO DE VIDA DO
DESENVOLVIMENTO DE SOFTWARE: PROPOSTA DE UM MODELO
CONVERGENTE APLICADO AO DESENVOLVIMENTO DE UMA PLATAFORMA
WEB**

Trabalho de Conclusão de Curso apresentado à banca avaliadora do Curso de Engenharia de Computação, da Escola Superior de Tecnologia, da Universidade do Estado do Amazonas, como pré-requisito para obtenção do título de Engenheiro de Computação.

Orientadora: Dra. Áurea Hiléia da Silva Melo

Manaus – AM

2026

Ficha Catalográfica

Ficha catalográfica elaborada automaticamente de acordo com os dados fornecidos pelo(a) autor(a).
Sistema Integrado de Bibliotecas da Universidade do Estado do Amazonas.

L732i	Lima, Nicolas Fernandes de Integração entre design thinking e ciclo de vida do desenvolvimento de software : proposta de um modelo convergente aplicado ao desenvolvimento de uma plataforma web / Nicolas Fernandes de Lima. Manaus : [s.n], 2026. 125 f.: color.; 21.0 cm. TCC - Graduação em Engenharia de Computação- Universidade do Estado do Amazonas, Manaus, 2026. Inclui Bibliografia. Orientador: Melo, Áure Hiléia da Silva. 1. Engenharia de software. 2. Design thinking. 3. SDLC-software development life cycle. 4. Engenharia de requisitos. 5. Prototipação. I. Melo, Áure Hiléia da Silva (Orient.) II. Universidade do Estado do Amazonas. III. Título CDU(1997)62:004
-------	--



FOLHA DE APROVAÇÃO

INTEGRAÇÃO ENTRE DESIGN THINKING E CICLO DE VIDA DO DESENVOLVIMENTO DE SOFTWARE: PROPOSTA DE UM MODELO CONVERGENTE APLICADO AO DESENVOLVIMENTO DE UMA PLATAFORMA WEB

NICOLAS FERNANDES DE LIMA

Trabalho de Conclusão de Curso defendido e aprovado pela banca avaliadora constituída pelos professores:

Profª Dra. Áurea Hiléia da S. Melo
Presidente

Prof. Dr. Luis Cuevas Rodrigues
Avaliador

Profª. Dr. Ana Carolina Oran Fonseca
Avaliador

Manaus 22 de junho de 2026

Resumo

Este trabalho investigou como o *Design Thinking* pode ser incorporado às atividades iniciais do ciclo de vida do desenvolvimento de software (SDLC), com foco na compreensão do problema, na elicitación e no refinamento de requisitos, na concepção da solução, na prototipação e na validação conceitual. Para isso, realizou-se uma análise teórica de abordagens de *Design Thinking* e de suas convergências com práticas consolidadas do SDLC, a partir da qual foi proposto o *Modelo Convergente DT-SDLC*. O modelo foi aplicado no estudo de caso do *TCCMatch*, uma plataforma web de mediação de orientações de TCC, por meio de um estudo de caso orientado à produção de artefatos como mapa de stakeholders, personas, jornada do usuário, quadro CSD, requisitos, histórias de usuário, priorização funcional e protótipos. Os resultados indicaram que a integração entre *Design Thinking* e SDLC contribuiu para ampliar a compreensão do contexto de uso, reduzir ambiguidades na definição da solução, fortalecer a rastreabilidade entre achados qualitativos e requisitos e apoiar a validação antecipada de decisões de projeto. Concluiu-se que essa convergência se mostrou metodologicamente produtiva para projetos centrados no usuário, sobretudo em sistemas web que demandam equilíbrio entre empatia, criatividade, formalização e rigor técnico.

Palavras-Chave: Engenharia de Software, Design Thinking, SDLC, Engenharia de Requisitos, Prototipação, TCCMatch.

Abstract

This work investigated how *Design Thinking* can be incorporated into the initial activities of the Software Development Life Cycle (SDLC), focusing on problem understanding, requirements elicitation and refinement, solution conception, prototyping, and conceptual validation. To this end, a theoretical analysis of *Design Thinking* approaches and their convergences with consolidated SDLC practices was conducted, from which the *DT-SDLC Convergent Model* was proposed. The model was applied in the case study of *TCCMatch*, a web platform for mediating undergraduate thesis supervision, through a case study oriented toward the production of artifacts such as stakeholder map, personas, user journey, CSD board, requirements, user stories, functional prioritization, and prototypes. The results indicated that the integration between *Design Thinking* and SDLC contributed to broadening the understanding of the usage context, reducing ambiguities in the definition of the solution, strengthening traceability between qualitative findings and requirements, and supporting the early validation of design decisions. It was concluded that this convergence proved to be methodologically productive for user-centered projects, especially in web systems that require a balance between empathy, creativity, formalization, and technical rigor.

Keywords: Software Engineering, Design Thinking, SDLC, Requirements Engineering, Prototyping, TCCMatch.

Sumário

Lista de Quadros	vii
Lista de Figuras	x
1 Introdução	1
1.1 Justificativa	3
1.2 Objetivos	5
1.2.1 Objetivo Geral	5
1.2.2 Objetivos Específicos	5
1.3 Metodologia	6
1.3.1 Etapa 1: Planejamento e fundamentação teórica	6
1.3.2 Etapa 2: Análise de relação entre DT e SDLC	7
1.3.3 Etapa 3: Construção do modelo metodológico	7
1.3.4 Etapa 4: Aplicação do modelo no estudo de caso	8
1.3.5 Etapa 5: Análise e fechamento metodológico	9
1.4 Organização do trabalho	9
2 Fundamentação Teórica e Trabalhos Correlatos	11
2.1 Engenharia de Software	11
2.2 Design Thinking	15
2.3 Trabalhos Correlatos	31
3 Definição do Modelo Convergente DT-SDLC	37
3.1 Visão geral	37

3.2	Estudo dos modelos de Design Thinking	38
3.3	SDLC x Design Thinking	41
3.4	Proposição do Modelo Convergente DT-SDLC	52
4	Aplicação do Modelo Convergente no Estudo de Caso <i>TCCMatch</i>	71
4.1	Visão geral	71
4.2	Contexto do caso e delineamento qualitativo	72
4.3	Aplicação do Modelo Convergente DT-SDLC no Protótipo	76
4.4	Resultados e discussão	116
	Considerações Finais	120
4.5	Trabalhos Futuros	122

Lista de Quadros

2.1	Resumo de trabalhos correlatos sobre a integração entre DT e SDLC	33
2.2	Inserção de etapas e técnicas de Design Thinking em atividades do SDLC nos trabalhos correlatos	34
3.1	Estrutura do Design Thinking segundo Brown (2009)	39
3.2	Estrutura do Design Thinking segundo Lindberg et al. (2011)	39
3.3	Estrutura do Design Thinking segundo Vianna et al. (2012)	40
3.4	Comparação entre abordagens de Design Thinking	40
3.5	Relação entre a abordagem de Brown e atividades do SDLC	43
3.6	Relação entre a abordagem de Lindberg et al. e atividades do SDLC	44
3.7	Relação entre a abordagem de Vianna e atividades do SDLC	47
3.8	Resumo comparativo entre etapas do Design Thinking e atividades gerais do SDLC	49
3.9	Estrutura operacional do Modelo Convergente DT-SDLC	54
3.10	Critérios para seleção dos artefatos do SDLC	56
4.1	Perfil dos stakeholders considerados no estudo de caso	72
4.2	Aplicação das etapas do Modelo Convergente DT-SDLC no estudo de caso . . .	73
4.3	Mapa de artefatos selecionados na aplicação do Modelo Convergente DT-SDLC .	76
4.4	Inventário dos artefatos do SDLC utilizados no protótipo	79
4.5	Achados qualitativos consolidados na Imersão	82
4.6	Documento de visão e escopo preliminar do protótipo <i>TCCMatch</i>	83
4.7	Atores do sistema e perfis de acesso considerados no protótipo	83
4.8	Mapa textual de stakeholders do sistema <i>TCCMatch</i>	84
4.9	Necessidades e restrições iniciais derivadas da imersão	85

4.10	Requisitos iniciais/candidatos derivados da imersão	86
4.11	Personas analíticas elaboradas para o sistema <i>TCCMatch</i>	89
4.12	Jornada atual do aluno no processo de busca por orientação	91
4.13	Rastreabilidade entre insights qualitativos e requisitos do sistema	92
4.14	Requisitos consolidados do <i>TCCMatch</i> após a organização dos achados	92
4.15	Modelo de domínio textual do protótipo <i>TCCMatch</i>	93
4.16	Critérios norteadores e regras preliminares para o protótipo	95
4.17	Cardápio de ideias gerado na etapa de Ideação	96
4.18	Matriz de posicionamento das funcionalidades propostas	96
4.19	Histórias de usuário priorizadas no backlog inicial do <i>TCCMatch</i>	97
4.20	Diagrama textual de casos de uso do protótipo <i>TCCMatch</i>	97
4.21	Especificação conceitual dos casos de uso principais do <i>TCCMatch</i>	98
4.22	Backlog inicial priorizado para o MVP do <i>TCCMatch</i>	99
4.23	Arquitetura funcional projetada para o MVP do <i>TCCMatch</i>	99
4.24	Mapa de navegação conceitual do protótipo <i>TCCMatch</i>	102
4.25	Diagrama textual de atividades do fluxo principal do <i>TCCMatch</i>	103
4.26	Especificação preliminar das telas prototipadas no <i>TCCMatch</i>	110
4.27	Modelo conceitual de dados preliminar do protótipo <i>TCCMatch</i>	110
4.28	Critérios de aceitação conceitual para avaliação do protótipo	111
4.29	Roteiro de validação qualitativa aplicado ao protótipo do <i>TCCMatch</i>	111
4.30	Resumo qualitativo da validação conceitual do protótipo	112
4.31	Backlog refinado após a validação conceitual do protótipo	115
4.32	Comparação analítica entre Engenharia de Requisitos tradicional e Modelo Con- vergente DT-SDLC	118
4.33	Síntese dos resultados obtidos com os artefatos do modelo convergente	119

Lista de Figuras

2.1	Ciclo de vida do desenvolvimento de software (SDLC)	14
2.2	Espaços de <i>Inspiration</i> , <i>Ideation</i> e <i>Implementation</i> no Design Thinking	19
2.3	Relação entre espaço do problema, espaço da solução e alinhamento iterativo no Design Thinking	21
2.4	Esquema representativo das etapas do processo de Design Thinking segundo Vianna et al	23
2.5	Relação entre profundidade do conhecimento e técnicas de imersão no Design Thinking	26
3.1	Imersão	59
3.2	Análise	62
3.3	Ideação	65
3.4	Prototipação	68
3.5	Ciclo operacional do Modelo Convergente DT-SDLC	70
4.1	Reenquadramento de foco do protótipo <i>TCCMatch</i>	81
4.2	Mapa visual de stakeholders do protótipo <i>TCCMatch</i>	85
4.3	Diagrama de afinidades do protótipo <i>TCCMatch</i>	87
4.4	Mapa conceitual do processo de orientação do protótipo <i>TCCMatch</i>	88
4.5	Esquema visual das personas do protótipo <i>TCCMatch</i>	89
4.6	Mapa de empatia do aluno concluinte indeciso	90
4.7	Linha de dor da jornada do aluno no processo de busca por orientação	91
4.8	Modelo de domínio visual do protótipo <i>TCCMatch</i>	94

4.9	Quadro de brainstorming do protótipo <i>TCCMatch</i>	95
4.10	Diagrama visual de casos de uso do protótipo <i>TCCMatch</i>	98
4.11	Arquitetura funcional visual do protótipo <i>TCCMatch</i>	100
4.12	Organização funcional do MVP do protótipo <i>TCCMatch</i>	101
4.13	Fluxograma de navegação principal do protótipo <i>TCCMatch</i>	102
4.14	Diagrama visual de atividades do protótipo <i>TCCMatch</i>	104
4.15	Storyboard do fluxo do aluno no protótipo <i>TCCMatch</i>	105
4.16	Wireframe da tela de busca do protótipo <i>TCCMatch</i>	107
4.17	Wireframe da tela de detalhe da oportunidade do protótipo <i>TCCMatch</i>	108
4.18	Wireframe do painel da coordenação do protótipo <i>TCCMatch</i>	109
4.19	Protótipo: tela de busca de orientadores do <i>TCCMatch</i>	113
4.20	Protótipo: tela de detalhes da oportunidade do <i>TCCMatch</i>	113
4.21	Protótipo: tela de acompanhamento da solicitação do <i>TCCMatch</i>	114
4.22	Protótipo: painel da coordenação do <i>TCCMatch</i>	114

Introdução

O desenvolvimento de software consolidou-se, nas últimas décadas, como um dos pilares da transformação digital e da modernização dos processos organizacionais, desempenhando papel essencial na automatização de atividades e na geração de vantagem competitiva (PRESSMAN, 2016; SOMMERVILLE, 2019). Contudo, mesmo com metodologias consolidadas e ferramentas de apoio cada vez mais sofisticadas, os desafios relacionados à concepção e à construção de soluções digitais permanecem recorrentes, especialmente nas etapas iniciais do ciclo de vida do desenvolvimento de software (SDLC) (BOEHM, 1988; WIEGERS; BEATTY, 2013). Dificuldades na elicitação precisa de requisitos, falhas na comunicação entre equipes técnicas e stakeholders e a tradução incompleta das necessidades dos usuários em artefatos técnicos continuam figurando entre as principais causas de retrabalho e de desperdício de recursos em projetos (POHL, 2010; PRESSMAN, 2016).

Grande parte desses problemas decorre da natureza dinâmica e complexa dos contextos em que os sistemas são concebidos. Organizações modernas operam em ambientes de constante mudança, o que exige das equipes de engenharia de software uma capacidade de adaptação e interpretação contínua de novas demandas, nem sempre contemplada pelos modelos tradicionais (SOMMERVILLE, 2019). Embora metodologias como o ciclo em cascata, o incremental e as abordagens ágeis tenham evoluído para mitigar riscos, elas frequentemente enfrentam obstáculos quando se trata de compreender a fundo o problema e antecipar, nas fases de análise e concepção, o impacto das soluções sobre a experiência do usuário (BOEHM, 2006; NORMAN, 2013).

Além disso, a crescente pressão por produtividade e previsibilidade em prazos e custos faz com que muitas equipes priorizem a execução em detrimento da investigação (PRESSMAN, 2016). A busca por métricas objetivas e por documentação estruturada, embora necessária para

a gestão de projetos, pode afastar os times da imersão qualitativa que permitiria compreender o contexto real de uso do sistema (WIEGERS; BEATTY, 2013). Esse distanciamento entre o que é projetado e o que realmente é necessário reflete-se em produtos pouco intuitivos, de baixa aceitação e que frequentemente exigem revisões após o lançamento (NIELSEN, 1994; NORMAN, 2013).

Outro ponto crítico está na transição entre o levantamento de requisitos e a prototipação. Nessa fase, as representações produzidas, como diagramas, histórias de usuário e especificações formais, nem sempre capturam as nuances do problema ou as necessidades subjetivas dos usuários (POHL, 2010; PRESSMAN, 2016). Essa lacuna de entendimento entre analistas, desenvolvedores e clientes impacta diretamente a qualidade do protótipo e, consequentemente, do produto final, tornando o processo de engenharia mais suscetível a falhas de concepção (SOMMERVILLE, 2019).

Embora a Engenharia de Software tenha atingido elevado grau de maturidade em termos de processos e padrões técnicos, ainda há limitações significativas quando o desenvolvimento depende de fatores humanos, criativos e empáticos (BROWN, 2009). As ferramentas disponíveis são eficazes para gerenciar escopo, risco e qualidade, mas pouco abrangentes quando se trata de compreender comportamentos, motivações e experiências dos usuários (VIANNA et al., 2012). Essa carência se manifesta especialmente nas fases de análise, concepção e prototipação, nas quais o entendimento do problema e a geração de soluções deveriam ocorrer de forma mais colaborativa e iterativa (BROWN, 2009; VIANNA et al., 2012).

Dessa forma, compreender como as práticas do SDLC podem ser ampliadas por abordagens que estimulem empatia, experimentação e validação precoce de ideias torna-se fundamental para o aprimoramento do processo de desenvolvimento (NORMAN, 2013; VIANNA et al., 2012). A partir dessa perspectiva, surge a necessidade de investigar metodologias complementares capazes de reduzir as lacunas entre o planejamento técnico e o entendimento das reais necessidades humanas, promovendo um processo mais integrado, eficiente e inovador (BROWN, 2009; PRESSMAN, 2016).

Nesse contexto, este trabalho insere-se na discussão sobre a integração entre *Design Thinking*

e SDLC, com foco nas fases iniciais do SDLC. A pesquisa propõe um modelo metodológico voltado à compreensão do problema, à identificação das necessidades dos usuários, à definição de requisitos e à prototipação, utilizando o desenvolvimento conceitual do *TCCMatch* como estudo de caso para orientar a aplicação da proposta.

1.1 Justificativa

A Engenharia de Software consolidou-se ao longo das últimas décadas como uma área estruturada, metodologicamente robusta e essencial para o desenvolvimento de sistemas de alta qualidade (PRESSMAN; MAXIM, 2019). Entretanto, apesar de sua maturidade técnica, ainda persiste um distanciamento entre os processos formais do SDLC e as dimensões humanas envolvidas na criação de soluções digitais (BOEHM, 1988). Em muitas organizações, o foco em métricas, cronogramas, documentação e controle de escopo é necessário para conduzir o projeto, mas nem sempre é suficiente para compreender o contexto real de uso, as motivações dos usuários e as necessidades que ainda não foram traduzidas em requisitos (BROWN, 2009; VIANNA et al., 2012).

Essa lacuna é especialmente crítica nas fases iniciais do SDLC, quando o problema ainda está sendo compreendido, os requisitos estão em formação, alternativas de solução são discutidas e os primeiros protótipos começam a materializar decisões de projeto. Nesse momento, decisões mal fundamentadas podem se transformar em requisitos frágeis, fluxos pouco aderentes ao usuário e protótipos que apenas formalizam uma compreensão incompleta do problema (POHL, 2010; PRESSMAN, 2016). Por isso, este trabalho justifica a incorporação do *Design Thinking* nesse recorte específico do SDLC: da compreensão inicial do problema até a prototipação conceitual.

A escolha desse recorte é relevante porque aplicar o *Design Thinking* antes de sua articulação com o SDLC poderia gerar percepções, ideias e registros criativos sem conexão suficiente com os artefatos de engenharia, como requisitos, escopo, backlog, modelos, fluxos e critérios de aceitação. Nesse caso, o DT poderia permanecer como uma etapa exploratória isolada, com menor capacidade de orientar tecnicamente o desenvolvimento. Por outro lado, aplicar o DT apenas depois da prototipação deslocaria sua contribuição para um momento mais corretivo,

quando várias decisões de problema, escopo, requisitos e solução já teriam sido assumidas. Assim, parte do potencial de empatia, reenquadramento, ideação e validação precoce seria perdido ou utilizado apenas para ajustar decisões já consolidadas.

Dessa forma, integrar o *Design Thinking* ao SDLC até a fase de prototipação permite atuar no momento em que ainda é possível qualificar a compreensão do problema e transformar achados qualitativos em artefatos úteis ao desenvolvimento. O DT contribui com imersão, empatia, cocriação, organização de insights, ideação e validação conceitual; o SDLC, por sua vez, contribui com estrutura, formalização, rastreabilidade e continuidade técnica. A integração proposta, portanto, não pretende substituir o SDLC nem estender o DT para todas as fases posteriores de implementação, testes técnicos, implantação e manutenção. Seu objetivo é fortalecer o trecho em que decisões conceituais ainda podem ser revistas com menor custo e maior impacto sobre a aderência da solução (HEHN; MENDEZ, 2021; VETTERLI et al., 2013).

A existência deste trabalho se justifica, portanto, pela necessidade de estruturar essa integração de maneira metodológica. Não basta afirmar que o DT pode apoiar o desenvolvimento de software; é necessário explicitar onde ele entra no SDLC, por que sua aplicação é mais pertinente até a prototipação, quais técnicas podem ser mobilizadas e quais artefatos podem ser gerados ou aprimorados. A relevância da pesquisa está em propor um modelo que conecte práticas centradas no usuário a produtos reconhecíveis do processo de desenvolvimento, preservando a flexibilidade do DT sem perder a organização exigida pela Engenharia de Software.

Além da contribuição teórica, o estudo propõe uma aplicação concreta por meio do desenvolvimento conceitual do *TCCMatch* até a fase de prototipação. Esse estudo de caso permite avaliar exploratoriamente como a integração entre *Design Thinking* e SDLC pode apoiar a passagem entre problema, necessidades dos usuários, requisitos, alternativas de solução e protótipos, evidenciando a utilidade do modelo no recorte em que esta pesquisa se concentra.

1.2 Objetivos

1.2.1 Objetivo Geral

O presente trabalho teve como objetivo propor um modelo de aplicação do DT nas etapas iniciais do SDLC, visando aprimorar os processos de compreensão do problema, identificação das necessidades dos usuários e definição de requisitos, bem como avaliar sua aplicabilidade por meio do desenvolvimento de um protótipo de um sistema web denominado *TCCMatch*, voltado ao apoio da mediação das orientações de trabalhos de conclusão de curso da UEA.

1.2.2 Objetivos Específicos

Para atingir o objetivo geral proposto, o trabalho estabeleceu os seguintes objetivos específicos:

- Identificar, na literatura, abordagens de Design Thinking, SDLC e engenharia de requisitos, levantando conceitos, métodos e práticas aplicáveis às fases iniciais do SDLC.
- Analisar as convergências e divergências entre o Design Thinking e as fases iniciais do SDLC, com foco em elicitação de requisitos, refinamento e prototipação;
- Definir um modelo de aplicação do DT para apoiar a execução das etapas iniciais do SDLC;
- Levantar e analisar as necessidades dos potenciais usuários do sistema *TCCMatch*, utilizando técnicas fundamentadas no modelo proposto;
- Desenvolver o protótipo *TCCMatch* como estudo de caso do modelo proposto;
- Avaliar a aplicabilidade e as contribuições do modelo por meio da análise dos resultados obtidos com o desenvolvimento do protótipo;

1.3 Metodologia

Para atingir os objetivos propostos, este estudo adotou uma abordagem de natureza qualitativa, de caráter exploratório e aplicado, organizada como um estudo de caso conceitual orientado à construção de um protótipo para validação do modelo. A pesquisa qualitativa foi escolhida por permitir a compreensão de fenômenos complexos e contextuais, enquanto a abordagem exploratória possibilitou investigar formas de integração entre o *Design Thinking* e as fases iniciais do SDLC (GIL, 2002; YIN, 2015). O caráter aplicado do estudo justificou-se pela utilização do *TCCMatch* como contexto de experimentação metodológica, permitindo avaliar conceitualmente como princípios de *Design Thinking* podem apoiar a compreensão do problema, a organização de requisitos, a ideação e a prototipação nas fases iniciais do SDLC.

O trabalho foi estruturado como um fluxo metodológico sequencial. A função desse fluxo é orientar a produção da pesquisa, indicando o que deve ser realizado em cada momento, quais processos compõem cada etapa e qual produto parcial deve estar disponível para que a etapa seguinte possa ser executada.

1.3.1 Etapa 1: Planejamento e fundamentação teórica

Esta etapa consiste em organizar a base inicial do trabalho, delimitando o tema, o problema, os objetivos e os conceitos necessários para sustentar a pesquisa.

Processos que compõem a etapa:

- delimitar o tema, o problema de pesquisa, a justificativa, o objetivo geral, os objetivos específicos e o escopo do estudo;
- levantar referências sobre Engenharia de Software, SDLC, Engenharia de Requisitos, *Design Thinking*, prototipação e trabalhos correlatos;
- organizar os conceitos fundamentais que serão utilizados na análise do problema e na construção da proposta metodológica;
- registrar as relações iniciais entre desenvolvimento de software, compreensão do usuário,

levantamento de requisitos e prototipação.

Produto esperado: base teórica organizada, com conceitos, categorias e referências suficientes para orientar a análise comparativa e a definição do modelo metodológico.

1.3.2 Etapa 2: Análise de relação entre DT e SDLC

Esta etapa consiste em examinar como o *Design Thinking* pode se relacionar com as fases iniciais do SDLC, sem definir antecipadamente quais técnicas ou artefatos serão utilizados.

Processos que compõem a etapa:

- estabelecer critérios de comparação entre abordagens de *Design Thinking* e atividades do SDLC;
- mapear as fases iniciais do SDLC relacionadas à compreensão do problema, levantamento de requisitos, análise, concepção e prototipação;
- identificar, de forma analítica, pontos de aproximação, limites e possíveis lacunas entre DT e SDLC;
- organizar as relações encontradas em uma estrutura comparativa que sirva de apoio para a etapa seguinte.

Produto esperado: quadro analítico de relações entre DT e SDLC, indicando critérios para orientar a construção do modelo sem antecipar conclusões sobre sua aplicação.

1.3.3 Etapa 3: Construção do modelo metodológico

Esta etapa consiste em transformar a análise anterior em um modelo de aplicação, definindo uma sequência de trabalho capaz de orientar projetos de software nas fases iniciais do ciclo de vida.

Processos que compõem a etapa:

- definir as etapas internas do modelo e a ordem em que elas devem ser executadas;

- estabelecer a finalidade de cada etapa do modelo em relação ao SDLC;
- indicar critérios para selecionar técnicas, registros e artefatos conforme o objetivo e o escopo da aplicação;
- prever mecanismos de rastreabilidade e retroalimentação entre informações coletadas, decisões tomadas e produtos gerados.

Produto esperado: modelo metodológico estruturado, contendo etapas, finalidade de cada etapa, critérios de aplicação e orientação para registro dos produtos gerados durante o processo.

1.3.4 Etapa 4: Aplicação do modelo no estudo de caso

Esta etapa consiste em aplicar o modelo metodológico em um caso delimitado, utilizando-o como roteiro para conduzir a compreensão do problema, a organização das informações, a concepção da solução e a prototipação.

Processos que compõem a etapa:

- caracterizar o contexto do estudo de caso e os grupos de stakeholders envolvidos;
- levantar informações sobre o problema, as necessidades, as restrições e as oportunidades do contexto analisado;
- organizar os dados coletados em registros que permitam compreender o problema e apoiar a definição de requisitos;
- estruturar alternativas de solução com base nas informações organizadas;
- representar a solução proposta por meio de protótipos e registros adequados ao nível conceitual da pesquisa.

Produto esperado: documentação da aplicação do modelo no estudo de caso, contendo os registros e produtos gerados em cada etapa do fluxo metodológico.

1.3.5 Etapa 5: Análise e fechamento metodológico

Esta etapa consiste em estabelecer e aplicar critérios para examinar a aplicação do fluxo metodológico, considerando a passagem entre problema, requisitos, solução e prototipação.

Processos que compõem a etapa:

- aplicar critérios de coerência entre o problema delimitado, os dados organizados, as decisões de solução e os protótipos produzidos;
- conferir a rastreabilidade entre os registros gerados nas etapas do fluxo metodológico;
- registrar pontos observados durante a aplicação conceitual do modelo;
- registrar limitações do percurso metodológico e indicar possibilidades de continuidade para trabalhos futuros.

Produto esperado: registro analítico da aplicação, contendo critérios utilizados, pontos observados, limitações do percurso e encaminhamentos possíveis para continuidade da pesquisa.

Essa organização metodológica funciona como roteiro de produção do trabalho: primeiro estabelece a base teórica, depois relaciona DT e SDLC, em seguida estrutura o modelo, aplica o modelo em um caso delimitado e, por fim, analisa o percurso realizado.

1.4 Organização do trabalho

Além deste capítulo introdutório, no qual são apresentados a contextualização do problema, a justificativa, os objetivos e o percurso metodológico da pesquisa, este trabalho está organizado em mais três capítulos de desenvolvimento e uma seção final de encerramento.

O Capítulo 2 apresenta a fundamentação teórica e os trabalhos correlatos. Nele são discutidos conceitos de Engenharia de Software, SDLC, Engenharia de Requisitos, prototipação e Design Thinking, além de estudos que tratam da aproximação entre DT e desenvolvimento de software. Ao final, esse capítulo consolida a base conceitual necessária para compreender por que a integração entre práticas centradas no usuário e atividades do SDLC pode contribuir para as fases iniciais de projetos de software.

O Capítulo 3 define o *Modelo Convergente DT-SDLC*. Inicialmente, são analisadas abordagens de Design Thinking propostas por Brown (2009), Lindberg et al. (2011) e Vianna et al. (2012), observando suas etapas, técnicas e níveis de formalização. Em seguida, essas abordagens são relacionadas às atividades iniciais do SDLC, permitindo identificar pontos de convergência, limites de aplicação e critérios para a composição do modelo. O capítulo conclui com a proposição do modelo metodológico, estruturado para apoiar a compreensão do problema, a organização de requisitos, a ideação e a prototipação conceitual.

O Capítulo 4 apresenta a aplicação do modelo no estudo de caso do *TCCMatch*. Nesse capítulo são descritos o contexto do problema, os stakeholders considerados, os artefatos produzidos em cada etapa do modelo e a forma como técnicas de DT foram articuladas a produtos do SDLC. Também são apresentados os resultados e a discussão da aplicação, destacando como o modelo contribuiu para transformar achados qualitativos em requisitos, decisões de escopo, fluxos, protótipos e critérios de validação conceitual.

Por fim, as Considerações Finais retomam os objetivos da pesquisa, sintetizam as principais contribuições teóricas e práticas do trabalho, apresentam as limitações do estudo e indicam possibilidades de trabalhos futuros. Essa parte encerra a monografia ao avaliar o percurso realizado e apontar caminhos para o aprimoramento do modelo e para a evolução do protótipo *TCCMatch*.

Fundamentação Teórica e Trabalhos Correlatos

Este capítulo apresenta os fundamentos teóricos que embasaram o desenvolvimento deste trabalho, com ênfase nas fases iniciais do ciclo de vida do desenvolvimento de software e nas contribuições do Design Thinking para compreender problemas, organizar requisitos, conceber soluções e validar protótipos. Assim, os modelos de processo da Engenharia de Software são apresentados de forma sintética, apenas na medida em que ajudam a contextualizar a proposta de integração desenvolvida nos capítulos seguintes.

2.1 Engenharia de Software

A Engenharia de Software, conforme Sommerville (2019) e Pressman e Maxim (2020), constitui um conjunto de métodos, técnicas e práticas organizadas com o objetivo de desenvolver sistemas de software de forma estruturada, garantindo qualidade, rastreabilidade e previsibilidade. Seu desenvolvimento histórico está diretamente associado à necessidade de controle sobre sistemas cada vez mais complexos, especialmente a partir da chamada “crise do software” nas décadas de 1960 e 1970 (PRESSMAN; MAXIM, 2020).

Nesse contexto, surgiram estruturas de ciclo de vida e modelos de processo que organizam o desenvolvimento em etapas, artefatos e mecanismos de validação, permitindo maior controle sobre o produto de software e sobre as decisões tomadas ao longo do projeto (SOMMERVILLE, 2019; PRESSMAN; MAXIM, 2019).

SDLC: ciclo de vida do desenvolvimento de software

O *Software Development Life Cycle* (SDLC), ou ciclo de vida do desenvolvimento de software, representa uma estrutura conceitual utilizada para organizar o desenvolvimento de sistemas desde a identificação inicial do problema até sua manutenção e evolução em ambiente operacional. Diferentemente de um modelo específico de processo, como cascata, incremental ou ágil, o SDLC descreve um conjunto de etapas fundamentais que podem ser executadas de forma sequencial, iterativa ou incremental, conforme o contexto do projeto, o grau de incerteza dos requisitos e as necessidades de controle da organização (SOMMERVILLE, 2019; PRESSMAN; MAXIM, 2019).

No âmbito da Engenharia de Software, o SDLC é relevante porque estabelece uma visão integrada do desenvolvimento, relacionando atividades técnicas, documentos produzidos, decisões de projeto e mecanismos de verificação. Assim, cada etapa não deve ser compreendida apenas como uma fase operacional, mas como um momento de produção e refinamento de artefatos que sustentam a rastreabilidade, a comunicação entre stakeholders e a qualidade do sistema (POHL, 2010; WIEGERS; BEATTY, 2013).

Embora o SDLC possa abranger planejamento, requisitos, análise, projeto, implementação, testes, implantação e manutenção, este trabalho concentra-se nas fases iniciais do ciclo, pois é nelas que se definem o problema, o escopo, os atores envolvidos, as necessidades dos usuários, os requisitos e as primeiras decisões conceituais da solução. A Figura 2.1 apresenta essa organização geral.

- **Planejamento e estudo de viabilidade:** Define a oportunidade ou problema que justifica o software, identifica stakeholders, restrições, riscos e objetivos iniciais, e produz artefatos como visão do produto, escopo preliminar e análise inicial de viabilidade (PRESSMAN; MAXIM, 2019; SOMMERVILLE, 2019).
- **Elicitação e especificação de requisitos:** Busca compreender necessidades, expectativas e restrições dos usuários e demais partes interessadas por meio de entrevistas, observação, análise documental, workshops e outras técnicas de levantamento. Seus produtos incluem

requisitos funcionais e não funcionais, regras de negócio, histórias de usuário, casos de uso e critérios de aceitação (SOMMERVILLE, 2019).

- **Análise e modelagem:** Organiza e refina os requisitos levantados, avaliando consistência, prioridade, completude e viabilidade. Nessa etapa, necessidades ainda abertas são transformadas em modelos, fluxos, matriz de priorização e matriz de rastreabilidade, reduzindo ambiguidades antes do projeto da solução (POHL, 2010; PRESSMAN; MAXIM, 2019).
- **Projeto da solução:** Traduz os requisitos analisados em decisões iniciais de arquitetura, módulos, fluxos, interfaces e experiência de uso. Para o recorte deste trabalho, essa etapa é tratada principalmente em nível conceitual, por meio de wireframes, protótipos e especificações preliminares de telas (SOMMERVILLE, 2019).
- **Etapas posteriores:** Implementação, testes, implantação e manutenção completam o ciclo de vida, mas não constituem o foco empírico deste estudo. Elas são mencionadas para situar a continuidade do SDLC, uma vez que o modelo proposto atua principalmente antes da codificação, apoiando a qualidade dos insumos que seguirão para etapas técnicas posteriores (SOMMERVILLE, 2019; PRESSMAN; MAXIM, 2019).

Desse modo, o ponto central para este trabalho não é descrever exhaustivamente todas as atividades do SDLC, mas destacar como documentação, rastreabilidade e validação conceitual podem qualificar as decisões tomadas antes da implementação (PRESSMAN; MAXIM, 2019).

Compreendido o SDLC como estrutura geral do desenvolvimento, importa observar que os modelos de processo organizam suas etapas de maneiras diferentes. Como o objetivo deste trabalho está concentrado nas fases iniciais, os modelos são retomados apenas de forma sintética, destacando sua relação com requisitos, análise, ideação e validação antecipada. Essa delimitação permite comparar as abordagens sem deslocar o foco para atividades técnicas posteriores, como codificação, implantação ou manutenção. Assim, a revisão dos modelos serve principalmente para evidenciar como cada forma de organização do processo lida com incerteza, participação dos usuários e refinamento progressivo da solução.



Figura 2.1: Ciclo de vida do desenvolvimento de software (SDLC). **Fonte:** Elaborado pelo autor com apoio do ChatGPT (2026).

Modelos de processo do SDLC

O modelo em cascata representa uma organização sequencial do desenvolvimento, na qual requisitos, análise, projeto, implementação, testes e manutenção tendem a ser tratados como fases sucessivas. Sua contribuição para este trabalho está em evidenciar a importância da documentação e da definição prévia de requisitos; sua limitação, por outro lado, está na baixa flexibilidade diante de mudanças e incertezas nas fases iniciais (SOMMERVILLE, 2019; PRESSMAN; MAXIM, 2019).

O modelo incremental organiza o desenvolvimento em entregas parciais, permitindo priorizar subconjuntos de requisitos e obter feedback progressivo. Essa lógica é relevante para a proposta desta pesquisa porque reforça a necessidade de ciclos de aprendizagem, refinamento e validação antes de consolidar a solução final (SOMMERVILLE, 2019; PRESSMAN; MAXIM, 2019).

O modelo em V mantém uma relação explícita entre etapas de especificação e atividades de

verificação e validação. Embora seja mais associado a contextos que exigem controle formal, ele contribui para este trabalho ao reforçar que requisitos e critérios de aceitação precisam ser definidos de modo rastreável desde o início do ciclo (SOMMERVILLE, 2019).

O modelo espiral, por sua vez, destaca a gestão de riscos e a avaliação iterativa em projetos com maior incerteza. Sua relação com este estudo está na valorização da prototipação e da avaliação antecipada como formas de reduzir riscos antes de decisões técnicas mais custosas (PRESSMAN; MAXIM, 2019).

Metodologias Ágeis

As metodologias ágeis enfatizam ciclos curtos, adaptação a mudanças, colaboração com stakeholders e entrega incremental de valor. No contexto deste trabalho, interessa sobretudo sua aproximação com práticas de priorização, feedback e validação contínua, que dialogam com o Design Thinking nas etapas de compreensão do problema e concepção da solução.

O Scrum contribui principalmente por organizar requisitos em backlog, priorizar funcionalidades e promover revisão frequente com stakeholders. Já o Extreme Programming (XP) é mais voltado a práticas técnicas de implementação, como testes, refatoração e integração contínua; por isso, é mencionado apenas como parte do panorama da Engenharia de Software, sem aprofundamento, uma vez que o estudo não aborda a codificação do sistema (SOMMERVILLE, 2019).

Em síntese, os modelos de processo e métodos ágeis mostram que o SDLC pode variar quanto à linearidade, iteração, controle e participação dos usuários. Para esta pesquisa, contudo, o aspecto mais relevante é a preparação das fases iniciais: compreender o problema, organizar requisitos, priorizar funcionalidades, prototipar alternativas e validar conceitualmente a solução antes da implementação.

2.2 Design Thinking

O Design Thinking (DT) pode ser compreendido como uma abordagem de inovação centrada no ser humano, voltada à resolução de problemas complexos por meio de processos colaborativos,

exploratórios e iterativos. Nessa perspectiva, o DT é apresentado como uma forma de articular as necessidades das pessoas com aquilo que é tecnicamente factível e organizacionalmente viável, deslocando o foco da simples construção de artefatos para a formulação de soluções com significado para os usuários (BROWN, 2008, 2009). Assim, o DT não deve ser entendido apenas como um conjunto de técnicas criativas, mas como uma lógica de investigação e experimentação aplicada a problemas pouco estruturados.

No contexto do SDLC, essa abordagem ganha relevância sobretudo nas fases iniciais, em que a compreensão do problema, a identificação de necessidades implícitas e a validação precoce de soluções são determinantes para a qualidade do produto final. O DT contribui para ampliar a empatia com o usuário, favorecer a colaboração multidisciplinar e estruturar ciclos de aprendizagem a partir de protótipos e feedback (LINDBERG et al., 2011; VIANNA et al., 2012). Em complemento, estudos voltados à Engenharia de Requisitos indicam que o DT pode apoiar a elicitación, a definição e a validação de requisitos, especialmente em cenários marcados por incerteza, inovação e mudança frequente (VETTERLI et al., 2013; HEHN; MENDEZ, 2021).

Seu uso é, portanto, especialmente indicado em contextos nos quais há incerteza sobre os requisitos, necessidade de inovação ou dificuldade em compreender o comportamento dos usuários. Nesses casos, o DT atua como uma abordagem complementar aos processos tradicionais do SDLC, uma vez que não se concentra diretamente na implementação técnica do sistema, mas na compreensão aprofundada do problema e na validação conceitual das soluções propostas (BROWN, 2009; VIANNA et al., 2012).

Para a construção teórica desta seção, foram selecionadas três perspectivas complementares sobre o Design Thinking. Brown é adotado como referência central por representar uma das formulações mais difundidas do DT no campo da inovação e da estratégia organizacional. Embora o conceito não possa ser atribuído a um único autor, pois resulta de uma trajetória mais ampla dos estudos e práticas de design, Brown contribuiu para sistematizar e popularizar o DT ao descrevê-lo como uma abordagem centrada no ser humano, capaz de articular necessidades das pessoas, possibilidades tecnológicas e viabilidade de negócio. Sua inclusão, portanto, permite estabelecer a base conceitual e estratégica do método, especialmente no que se refere à inovação

em produtos, serviços, processos e experiências (BROWN, 2008, 2009).

Lindberg, Meinel e Wagner foram selecionados por oferecerem uma leitura acadêmica do DT, vinculada ao debate científico sobre desenvolvimento de sistemas e inovação em tecnologia da informação. O estudo desses autores, publicado na obra *Design Thinking: Understand – Improve – Apply*, associada ao programa de pesquisa HPI-Stanford em Design Thinking, discute o DT como um conceito relevante para o desenvolvimento de TI, relacionando-o ao pensamento de engenharia, à colaboração interdisciplinar, à aprendizagem iterativa e a métodos próximos ao desenvolvimento ágil. Essa perspectiva é importante para este trabalho porque aproxima o Design Thinking do contexto acadêmico da Engenharia de Software e contribui para compreendê-lo de forma mais conceitualizada no processo de desenvolvimento (LINDBERG et al., 2011).

Vianna et al., por sua vez, foram escolhidos por apresentarem uma abordagem aplicada e operacional, direcionada à inovação em negócios reais. Diferentemente de uma formulação predominantemente conceitual, sua obra organiza o DT em fases, técnicas e ferramentas práticas, como imersão, análise e síntese, ideação e prototipação, favorecendo sua aplicação em projetos organizacionais. Essa contribuição é especialmente relevante para o presente estudo porque oferece instrumentos que podem ser incorporados às fases iniciais do SDLC, apoiando o levantamento de necessidades, a organização de informações qualitativas, a geração de alternativas e a validação inicial de soluções (VIANNA et al., 2012).

Assim, Brown contribui com a base conceitual e estratégica do Design Thinking; Lindberg, Meinel e Wagner contribuem com sua interpretação acadêmica e sua aproximação com o desenvolvimento de TI; e Vianna et al. contribuem com uma orientação prática voltada à aplicação em negócios. A seguir, apresenta-se um resumo sobre a abordagem do DT de cada autor.

Abordagem de Brown (2009)

O Design Thinking é apresentado por Brown como uma abordagem de inovação centrada no ser humano, orientada a conciliar necessidades das pessoas, factibilidade tecnológica e viabilidade de negócio. Nessa formulação, o design deixa de atuar apenas no refinamento estético de produtos para participar da definição de serviços, processos, estratégias e experiências. Essa

concepção é particularmente relevante para o SDLC, pois desloca a atenção para a compreensão do contexto de uso e para a formulação do problema antes da construção técnica da solução.

Um dos aspectos mais importantes dessa abordagem é a caracterização do processo como um sistema de espaços, e não como uma sequência rígida de etapas. Brown descreve o Design Thinking como um fluxo iterativo no qual as equipes transitam e retornam entre espaços de exploração, desenvolvimento e refinamento, em vez de seguir um encadeamento linear fechado. Sob a ótica do SDLC, esse ponto é importante porque aproxima o levantamento de requisitos e a concepção inicial da solução de práticas mais abertas de observação, prototipação e aprendizagem contínua.

No que se refere à aplicação, a abordagem de Brown mostra-se particularmente adequada a contextos de inovação em produtos, serviços e modelos de negócio, sobretudo quando o problema ainda não está claramente definido. No entanto, considerando os critérios comparativos adotados neste trabalho, trata-se de uma formulação mais conceitual porque organiza o processo em três espaços amplos, sem explicitar subetapas nem vincular de modo prescritivo técnicas, artefatos e entregas intermediárias a cada espaço. Por isso, sua contribuição para contextos que exigem documentação, rastreabilidade e rigor processual tende a ser mais indireta.

A abordagem proposta por Brown organiza o Design Thinking em três espaços principais: *Inspiration*, *Ideation* e *Implementation*. Esse fluxo é mostrado na Figura 2.2.

- **Inspiration:** Corresponde às circunstâncias, problema, oportunidade ou ambos, que motivam a busca por soluções. Nesse espaço, Brown enfatiza observação direta, aproximação com o contexto de uso, atenção a usuários extremos e descrição das jornadas dos clientes, além do trabalho interdisciplinar desde o início do projeto.

O resultado desse espaço é a ampliação da compreensão do problema, das restrições e das oportunidades percebidas no contexto investigado.

- **Ideation:** Refere-se ao processo de geração, desenvolvimento e teste de ideias que podem levar a soluções. Brown associa esse espaço a práticas como *brainstorming*, construção de cenários, *storytelling*, elaboração de esboços e prototipação rápida, sempre com ciclos sucessivos de teste e refinamento.

O principal resultado desse espaço é um conjunto de alternativas desenvolvidas e avaliadas conceitualmente, ainda em nível exploratório, mas já com maior concretude.

- **Implementation:** Nesse espaço, o foco desloca-se para o encaminhamento da solução em direção ao mercado ou à adoção organizacional. Brown associa esse momento à construção de recursos para implementação, à comunicação da proposta para o negócio, ao desenho da estratégia de difusão e à execução da visão concebida.

O resultado esperado é a definição de um caminho de concretização da solução, o que, no contexto do SDLC, se aproxima mais do planejamento de adoção e viabilização organizacional do que da codificação do software em si.

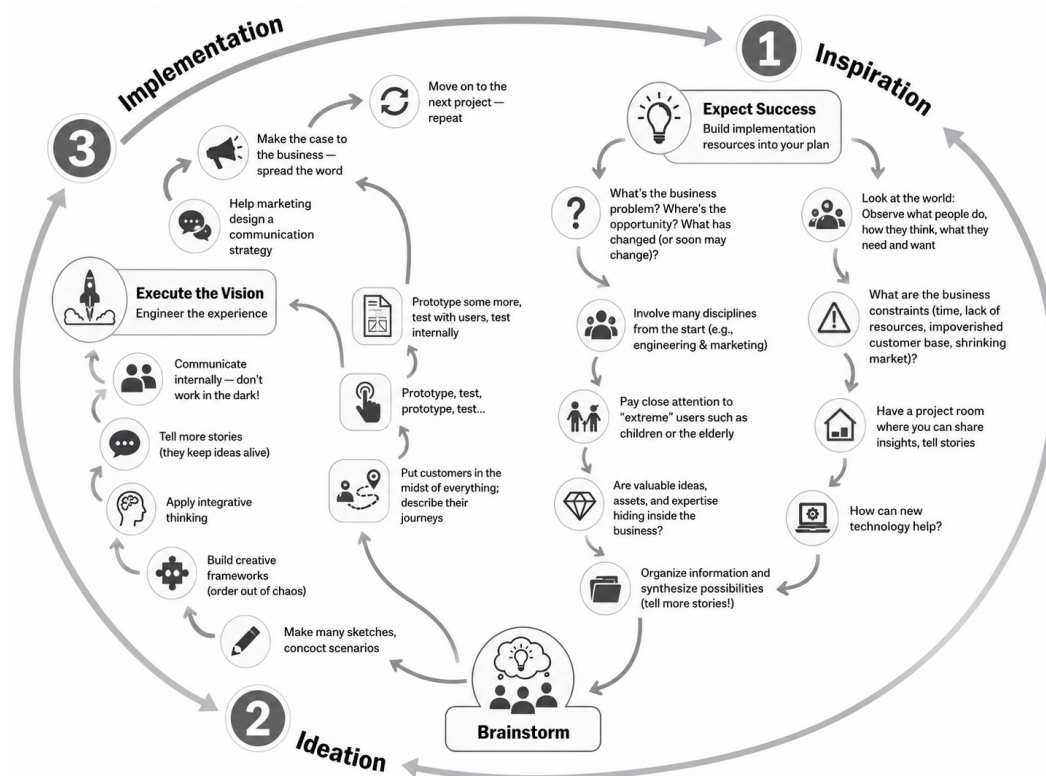


Figura 2.2: Espaços de *Inspiration*, *Ideation* e *Implementation* no Design Thinking. **Fonte:** Brown (2008).

Essa abordagem é amplamente utilizada em contextos de inovação; entretanto, sua formulação permanece mais resumida e menos prescritiva do que modelos que explicitam subetapas, artefatos e formas mais detalhadas de organizar as informações.

Abordagem de Lindberg et al. (2011)

A abordagem proposta por Lindberg et al. apresenta o Design Thinking como um conceito promissor para o desenvolvimento em tecnologia da informação, ao destacar sua capacidade de ampliar a inovação em contextos de engenharia e de se articular com abordagens afins, como o desenvolvimento ágil. Por essa razão, a proposta mostra-se especialmente relevante para estudos que buscam aproximar o DT do campo da Engenharia de Software.

Em vez de descrever o processo como uma sequência fixa de cinco etapas, os autores o estruturam a partir da distinção entre espaço do problema e espaço da solução e o resumem em três características básicas: exploração do problema, exploração da solução e alinhamento iterativo entre ambos. Além disso, o texto enfatiza a diversidade interdisciplinar da equipe como condição importante para que esse modo de trabalho se torne efetivo em projetos de TI.

No plano operacional, o capítulo menciona um fluxo de trabalho expresso por pares de ação como observar e organizar, idealizar e prototipar, revisar e refinar, além de exemplos de práticas como *inquiries*, entrevistas, observações, autoexperimentos, *brainstorming*, *sketching* e prototipação. Assim, sua principal força está menos em prescrever uma sequência detalhada de fases e mais em explicitar como o pensamento de design pode contribuir para lidar com *wicked problems*, integrar usuários ao processo e favorecer aprendizagem iterativa em desenvolvimento de TI. A relação entre exploração do problema, exploração da solução e alinhamento iterativo é apresentada na Figura 2.3.

- **Exploração do espaço do problema:** Corresponde ao esforço de compreender o problema de modo amplo e multiperspectivo, por meio de investigação prática e aproximação com situações reais de uso. Nessa dimensão, o texto menciona recursos como *inquiries*, entrevistas, observações e autoexperimentos.

O principal resultado é uma compreensão mais rica do contexto e dos usuários, ainda aberta e sujeita a reformulação.

- **Exploração do espaço da solução:** Refere-se à geração paralela de alternativas e à sua elaboração progressiva com apoio de práticas como *brainstorming*, *sketching* e prototipa-

ção. A ênfase está em aprender sobre a solução enquanto ela é concebida, e não apenas em selecionar rapidamente uma resposta única.

O principal resultado é um conjunto de conceitos de solução representados de forma tangível e comparável.

- **Alinhamento iterativo entre problema e solução:** Corresponde ao movimento de revisar e refinar continuamente as ideias à luz de feedback de usuários, clientes e especialistas. Nessa lógica, os protótipos funcionam como meios de comunicação e aprendizagem, permitindo revisar o entendimento do problema e ajustar o caminho da solução.

O principal resultado é o refinamento progressivo da proposta a partir de ciclos de feedback e revisão.

- **Diversidade interdisciplinar:** Embora não configure uma fase do processo, esse elemento aparece como condição estruturante da abordagem. Os autores destacam equipes interdisciplinares, perfis *T-shaped* e colaboração intensiva como fatores que ampliam a capacidade de lidar com problemas ambíguos em contextos de TI.

Esse aspecto reforça a relevância da abordagem para o SDLC, sobretudo em projetos que exigem articulação entre perspectivas técnicas, organizacionais e de uso.

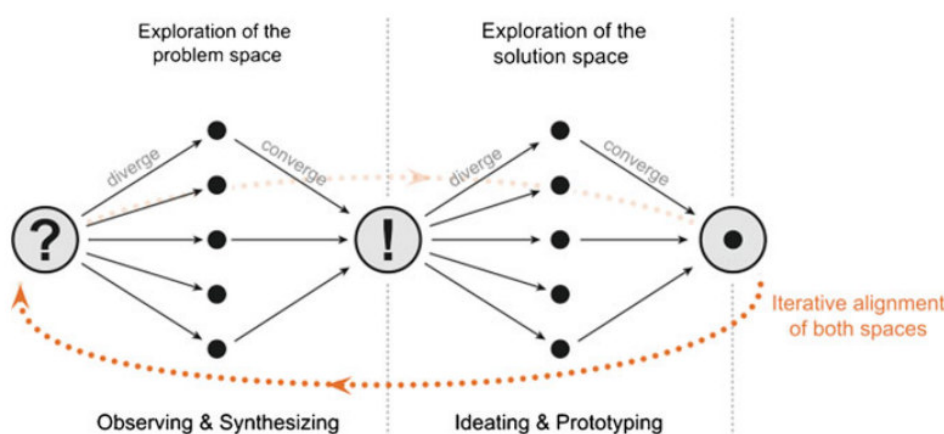


Figura 2.3: Relação entre espaço do problema, espaço da solução e alinhamento iterativo no Design Thinking.
Fonte: Lindberg et al. (2011).

Em comparação com Brown, a contribuição de Lindberg et al. não está em enumerar mais etapas, mas em explicitar com maior ênfase a coevolução entre problema e solução, o papel da

interdisciplinaridade e a relação do DT com o contexto específico do desenvolvimento de TI. Ainda assim, por não oferecer um detalhamento prescritivo extenso de artefatos e mecanismos de documentação, a abordagem também requer complementação quando aplicada a processos formais de engenharia de software.

Abordagem de Vianna et al. (2012)

A abordagem estruturada em quatro etapas, imersão, análise e organização, ideação e prototipação, foi desenvolvida com o objetivo de tornar o Design Thinking mais aplicável em contextos organizacionais e de inovação em negócios. Em comparação com as abordagens anteriores, sua principal característica é a ênfase operacional: os autores apresentam não apenas as fases do processo, mas também técnicas, ferramentas e exemplos de aplicação, configurando uma proposta mais sistematizada e diretamente orientada à prática. Esse aspecto é particularmente relevante para o SDLC, pois aproxima o DT de uma lógica metodológica mais compatível com processos estruturados.

Outro ponto central da proposta de Vianna et al. está na valorização da investigação qualitativa como fundamento para a formulação de soluções. A imersão permite compreender usuários, stakeholders, processos e contexto; a análise e organização transformam dados dispersos em oportunidades e direcionadores de projeto; a ideação explora alternativas; e a prototipação valida hipóteses de modo iterativo. Nesse modelo, o uso de artefatos como cartões de insight, diagrama de afinidades, personas, mapa de empatia, jornada do usuário, blueprint e protótipos funciona como ponte entre a investigação e a tomada de decisão, contribuindo para organizar informações e orientar a construção das soluções.

Os autores também atribuem forte importância ao caráter experimental do Design Thinking. Os protótipos não têm como objetivo representar uma solução finalizada, mas viabilizar aprendizado rápido, discussão com usuários e refinamento progressivo das alternativas propostas. Dessa forma, o erro é incorporado como parte do processo de aprendizagem, reduzindo riscos associados à implementação de soluções inadequadas e favorecendo uma melhor aderência entre problema, contexto e solução. Além disso, o livro explicita um encadeamento metodológico importante: a Imersão divide-se em Imersão Preliminar e Imersão em Profundidade; os

dados coletados são convertidos em cartões de insight e organizados na fase de Análise e Organização; as alternativas são geradas e discutidas na Ideação; e, por fim, são tangibilizadas na Prototipação. Esse fluxo é mostrado na Figura 2.4. Por isso, a abordagem de Vianna mostra-se especialmente útil em projetos que exigem não apenas criatividade, mas também organização do processo investigativo e capacidade de transformação dos achados em ações concretas, pois combina etapas nomeadas, subetapas explícitas e artefatos de organização e prototipação diretamente associados ao processo.

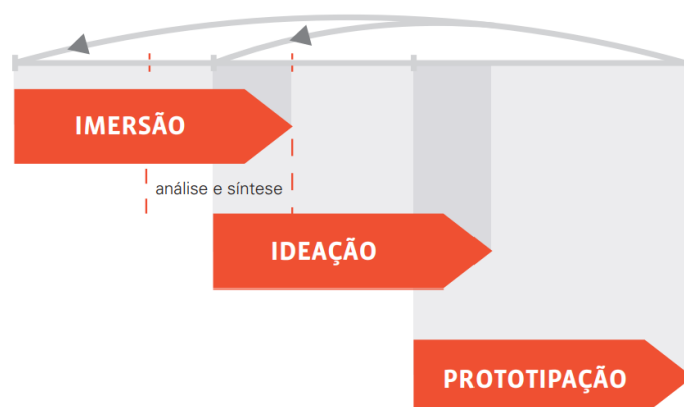


Figura 2.4: Esquema representativo das etapas do processo de Design Thinking segundo Vianna et al. **Fonte:** Vianna et al. (2012).

- **Imersão:** Em Vianna et al. (2012), a Imersão é explicitamente dividida em duas subetapas: *Imersão Preliminar* e *Imersão em Profundidade*. A primeira visa reenquadrar o problema, delimitar o escopo e identificar perfis e temas relevantes; a segunda aprofunda a compreensão do contexto de uso e das necessidades dos atores investigados.

As principais técnicas incluem:

- **Reenquadramento.** *Como usar:* consiste em revisar coletivamente o desafio inicial, reformulando perguntas, limites e perspectivas do projeto para evitar que a equipe permaneça presa a uma definição prematura do problema. *Quando usar:* é mais adequado no início da imersão preliminar, quando o problema ainda está amplo, pouco delimitado ou excessivamente orientado por suposições prévias. *Resultado obtido:* gera uma definição mais clara do foco investigativo e orienta a seleção dos aspectos do contexto que devem ser aprofundados.

- **Pesquisa exploratória.** *Como usar:* envolve levantamentos iniciais do contexto por meio de contatos preliminares, observações introdutórias e identificação dos atores, temas e restrições mais relevantes. *Quando usar:* é indicada quando a equipe ainda precisa construir uma visão geral do cenário antes de aprofundar a investigação. *Resultado obtido:* produz um panorama inicial do problema, dos stakeholders e das oportunidades de investigação.
- **Pesquisa desk.** *Como usar:* baseia-se na coleta e análise de dados secundários, como documentos, relatórios, estudos, referências de mercado, soluções existentes e tendências do domínio. *Quando usar:* torna-se útil quando se deseja complementar a investigação de campo com informações já disponíveis e reduzir lacunas iniciais de conhecimento. *Resultado obtido:* reúne referências comparativas e informações contextuais que ajudam a sustentar a compreensão preliminar do problema.
- **Entrevistas.** *Como usar:* devem ser conduzidas com usuários, especialistas e demais atores relevantes por meio de perguntas abertas, estimulando relatos sobre experiências, dificuldades, expectativas e práticas cotidianas. *Quando usar:* são apropriadas na imersão em profundidade, quando a equipe precisa captar percepções detalhadas e significados atribuídos ao problema pelos participantes. *Resultado obtido:* produz relatos qualitativos ricos, capazes de revelar dores, motivações e necessidades explícitas ou latentes.
- **Cadernos de sensibilização.** *Como usar:* consistem em instrumentos entregues aos participantes para que registrem atividades, impressões, sentimentos ou ocorrências ao longo de determinado período. *Quando usar:* são recomendados quando se deseja acompanhar comportamentos distribuídos no tempo e captar aspectos da experiência que não emergem facilmente em entrevistas pontuais. *Resultado obtido:* gera registros autorrelatados que ampliam a compreensão sobre hábitos, rotinas e percepções dos usuários.
- **Sessão generativa.** *Como usar:* promove encontros mediados nos quais os participantes expressam percepções e ideias por meio de estímulos visuais, associação

de palavras, montagem de materiais ou atividades criativas. *Quando usar:* é útil quando a equipe precisa acessar conteúdos subjetivos, repertórios simbólicos e interpretações difíceis de verbalizar diretamente. *Resultado obtido:* produz material interpretativo que ajuda a revelar valores, expectativas e significados associados ao contexto investigado.

- **Um dia na vida.** *Como usar:* consiste em acompanhar ou reconstruir a rotina do usuário ao longo de um período significativo, buscando compreender suas ações, interações, decisões e pontos de tensão. *Quando usar:* é indicada quando o problema depende fortemente da sequência de atividades e da experiência cotidiana dos atores envolvidos. *Resultado obtido:* permite visualizar a experiência do usuário em contexto e identificar oportunidades de melhoria ao longo de sua jornada real.
- **Sombra.** *Como usar:* baseia-se na observação direta do participante em seu ambiente natural, acompanhando suas ações com mínima interferência para perceber comportamentos, improvisações e obstáculos concretos. *Quando usar:* é especialmente relevante quando há diferença entre o que o usuário diz e o que efetivamente faz no contexto de uso. *Resultado obtido:* revela práticas tácitas, necessidades não verbalizadas e fatores contextuais que dificilmente seriam captados apenas por relato.

O resultado dessa etapa é um conjunto amplo de dados qualitativos sobre o problema, ainda não estruturados, mas já delimitados por perfis, contextos e oportunidades de investigação. Como mostram Vianna et al. (2012), as técnicas de imersão também podem ser compreendidas segundo o nível de profundidade do conhecimento que se busca obter sobre os usuários e o contexto investigado, como mostra a Figura 2.5.

- **Análise e Organização:** Nesta etapa, os dados coletados são organizados e interpretados com o objetivo de identificar padrões e transformar informações em conhecimento aplicável ao projeto.

As principais técnicas incluem:

- **Cartões de insight.** *Como usar:* consistem em registrar achados relevantes da imer-

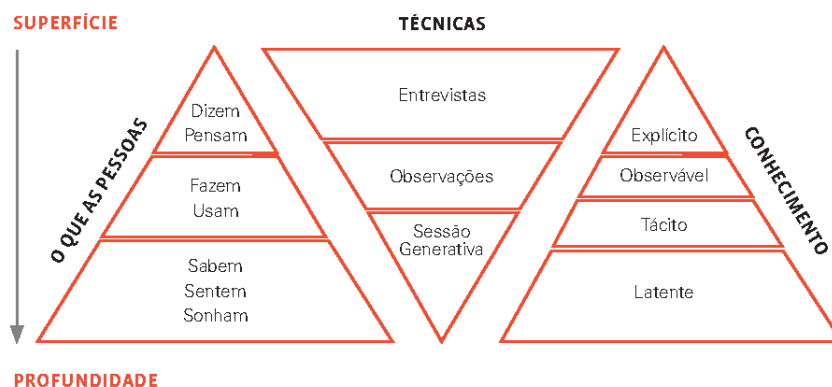


Figura 2.5: Relação entre profundidade do conhecimento e técnicas de imersão no Design Thinking. **Fonte:** Vianna et al. (2012).

são em unidades resumidas, normalmente contendo observação, interpretação e evidência associada. *Quando usar:* são indicados logo após a coleta de dados, quando a equipe precisa tornar visíveis os achados mais significativos e preparar sua organização posterior. *Resultado obtido:* gera um conjunto manejável de insights que preserva evidências importantes da investigação.

- **Diagrama de afinidades.** *Como usar:* organiza os cartões de insight por proximidade temática, recorrência ou relação de sentido, permitindo agrupar dados dispersos em blocos interpretativos. *Quando usar:* deve ser empregado quando a quantidade de informações coletadas já exige organização visual e agrupamento colaborativo. *Resultado obtido:* produz padrões e macrotemas que ajudam a estruturar a leitura do problema.
- **Mapa conceitual.** *Como usar:* representa visualmente relações entre conceitos, atores, fatores e consequências envolvidos no contexto investigado. *Quando usar:* é útil quando o problema possui múltiplas variáveis e a equipe precisa compreender dependências e conexões entre os elementos analisados. *Resultado obtido:* fornece uma visão estruturada do problema e de suas relações internas, apoiando a formulação de direcionadores de projeto.
- **Critérios norteadores.** *Como usar:* traduzem os aprendizados da análise em pa-

râmetros que devem orientar a avaliação e a seleção das soluções futuras. *Quando usar:* são definidos após a organização dos dados, quando o grupo já dispõe de base suficiente para transformar achados em princípios de decisão. *Resultado obtido:* estabelece referências claras para julgar aderência, relevância e coerência das propostas geradas.

- **Personas.** *Como usar:* consistem na construção de perfis representativos baseados em padrões identificados na pesquisa, descrevendo objetivos, comportamentos, dificuldades e expectativas. *Quando usar:* são adequadas quando a equipe precisa traduzir dados abstratos em perfis concretos que orientem requisitos e decisões de design. *Resultado obtido:* produz referências de usuários que facilitam a comunicação interna e mantêm o foco no público-alvo do projeto.
- **Mapa de empatia.** *Como usar:* organiza percepções sobre o que o usuário pensa, sente, diz e faz, relacionando comportamentos observados a dimensões cognitivas e emocionais. *Quando usar:* é útil quando se deseja aprofundar a compreensão da experiência do usuário para além de necessidades estritamente funcionais. *Resultado obtido:* oferece uma leitura mais humana e integrada da experiência, apoiando decisões centradas no usuário.
- **Jornada do usuário.** *Como usar:* descreve as etapas percorridas pelo usuário em sua interação com um serviço, processo ou sistema, evidenciando ações, pontos de contato, emoções e dificuldades. *Quando usar:* torna-se especialmente relevante quando o problema envolve fluxos de uso, transições entre etapas e momentos críticos da experiência. *Resultado obtido:* permite identificar gargalos, oportunidades de melhoria e pontos de intervenção prioritários.
- **Blueprint.** *Como usar:* articula a jornada do usuário com processos internos, atores de retaguarda e suportes necessários para a entrega da experiência. *Quando usar:* é indicado quando a solução depende da coordenação entre interação do usuário e funcionamento interno do serviço ou sistema. *Resultado obtido:* produz uma visão integrada entre frente de uso e bastidores operacionais, apoiando a definição mais

completa da solução.

O principal resultado dessa etapa é a definição estruturada do problema e a produção de direcionadores claros para a geração e avaliação de soluções.

- **Ideação:** A etapa de ideação é responsável pela geração, discussão e seleção preliminar de alternativas de solução com base nos achados organizados.

As técnicas incluem:

- **Brainstorming.** *Como usar:* promove sessões de geração livre de ideias, suspendendo julgamentos iniciais para estimular variedade, associação e expansão do repertório criativo. *Quando usar:* é mais adequado quando o problema já foi delimitado e a equipe precisa abrir o leque de alternativas antes de selecionar caminhos. *Resultado obtido:* gera um conjunto amplo de possibilidades de solução, que serve de base para discussão e refinamento posterior.
- **Workshop de cocriação.** *Como usar:* reúne equipe, usuários e outros participantes relevantes em atividades colaborativas de geração e discussão de propostas. *Quando usar:* é indicado quando se deseja combinar conhecimentos distintos e aumentar a aderência das soluções ao contexto de uso. *Resultado obtido:* produz alternativas construídas de forma compartilhada, com maior diversidade de perspectivas e maior legitimidade junto aos envolvidos.
- **Cardápio de ideias.** *Como usar:* organiza as propostas geradas em formato visual e comparável, facilitando o registro, a comunicação e a combinação entre alternativas. *Quando usar:* é útil após sessões de ideação, quando há muitas ideias e a equipe precisa sistematizá-las antes de priorizar. *Resultado obtido:* oferece um repertório estruturado de opções de solução que pode ser retomado e refinado nas etapas seguintes.
- **Matriz de posicionamento.** *Como usar:* compara as alternativas segundo critérios como relevância, viabilidade, impacto ou aderência aos direcionadores definidos na

análise. *Quando usar:* deve ser empregada no momento de selecionar e priorizar propostas para prototipação. *Resultado obtido:* favorece a escolha argumentada das ideias com maior potencial de desenvolvimento.

Essa etapa permite explorar diferentes abordagens de solução sem perder o vínculo com os critérios e evidências produzidos nas fases anteriores.

- **Prototipação:** A prototipação tem como objetivo tangibilizar e testar as soluções propostas por meio de representações que antecipam a experiência de uso.

As principais técnicas incluem:

- **Protótipo em papel.** *Como usar:* consiste em representar interfaces, fluxos ou elementos da solução com desenhos simples e materiais de baixa fidelidade. *Quando usar:* é indicado nas fases iniciais de prototipação, quando a equipe precisa testar rapidamente a lógica de uso sem alto custo de produção. *Resultado obtido:* permite verificar estrutura, navegação e compreensão geral da proposta antes de qualquer implementação mais elaborada.
- **Modelo de volume.** *Como usar:* materializa a solução em três dimensões ou em forma física simplificada para explorar aspectos espaciais, formais ou de manipulação. *Quando usar:* é mais adequado quando a solução envolve componentes físicos, disposição espacial ou interação com objetos concretos. *Resultado obtido:* oferece percepção tangível da forma e do funcionamento geral da proposta.
- **Encenação.** *Como usar:* simula a experiência de uso por meio da representação de situações, interações e comportamentos entre usuários e solução. *Quando usar:* torna-se útil quando se deseja testar serviços, fluxos de atendimento ou interações complexas entre múltiplos atores. *Resultado obtido:* evidencia problemas de uso, comunicação e sequência de ações ainda antes da implementação.
- **Storyboard.** *Como usar:* apresenta visualmente, em sequência narrativa, as situações de uso da solução e os principais pontos de contato da experiência. *Quando*

usar: é indicado quando a equipe precisa comunicar o funcionamento da proposta e discutir cenários de uso de forma rápida e acessível. *Resultado obtido*: produz uma narrativa clara da experiência esperada, apoiando validação e alinhamento entre os participantes.

- **Protótipo de serviços.** *Como usar*: representa a solução em termos de interação entre usuário, pontos de contato e processos de suporte, mesmo quando o serviço ainda não existe plenamente. *Quando usar*: é relevante quando a proposta depende da articulação entre interface, atendimento, processos e bastidores operacionais. *Resultado obtido*: permite testar a lógica do serviço e antecipar ajustes necessários na experiência e na operação.

O foco está na validação rápida com usuários e stakeholders, permitindo identificar problemas, recolher aprendizados e refinar a solução antes da implementação.

No contexto prático, essa abordagem é amplamente aplicada em projetos que exigem compreensão aprofundada do usuário e validação de soluções antes da implementação. Sua estrutura permite conduzir o processo de forma iterativa, mantendo o foco no problema, na experiência do usuário e na geração de valor para o contexto organizacional.

Entretanto, sua aplicação demanda maior esforço inicial na coleta e análise de dados, além de exigir participação ativa dos usuários e disponibilidade da equipe para conduzir atividades de investigação e organização dos achados de modo consistente.

Como síntese da fundamentação sobre Design Thinking, a seleção de Brown (2009), Lindberg et al. (2011) e Vianna et al. (2012) permite observar três níveis complementares de formalização da abordagem. Brown (2009) contribui com uma formulação conceitual ampla, organizada em macroespaços de inovação e útil para compreender a centralidade humana do DT. Lindberg et al. (2011) aproximam o DT do contexto de tecnologia da informação, destacando a coevolução entre problema e solução, a interdisciplinaridade e os ciclos de alinhamento iterativo. Vianna et al. (2012), por sua vez, oferecem maior detalhamento operacional, ao organizar etapas, subetapas, técnicas e formas de registro que favorecem sua articulação com processos estruturados. Essa

leitura fundamenta a etapa seguinte do trabalho: transformar os referenciais discutidos neste capítulo em critérios de comparação, relações com o SDLC e decisões metodológicas para a definição do modelo convergente.

2.3 Trabalhos Correlatos

Diversos estudos têm investigado a integração entre Design Thinking (DT) e SDLC, especialmente em suas fases iniciais, com foco na elicitação, análise e validação de requisitos.

Uma aplicação do DT à Engenharia de Requisitos mostra que a abordagem pode ser utilizada para ampliar a comunicação com stakeholders e apoiar o refinamento de necessidades de usuários nas fases iniciais do SDLC (JUNIOR; BARROS; ROCHA, 2013). Na mesma direção, argumenta-se que a Engenharia de Requisitos tradicional, pensada para sistemas mais estáveis, precisa incorporar práticas de DT para responder a contextos marcados por rápida mudança, maior proximidade com o usuário e necessidade de prototipação frequente (VETTERLI et al., 2013).

No campo empírico, um estudo relata a modernização de dois sistemas reais com apoio de DT e mostra que os protótipos atuaram como artefatos relevantes de comunicação entre participantes do projeto (CANEDO; COSTA, 2018). Nesse estudo, 24 de 29 participantes responderam ao questionário, sendo que 75% consideraram os protótipos importantes para suas atividades e 92% avaliaram positivamente a experiência de usá-los para apoiar funcionalidades do sistema (CANEDO; COSTA, 2018). Em estudo posterior, a análise de respostas de 59 profissionais de equipes ágeis brasileiras identificou que brainstorming e prototipação estão entre as técnicas de DT mais utilizadas no desenvolvimento, enquanto entrevistas com usuários e prototipação aparecem com destaque na elicitação de requisitos (CANEDO et al., 2020).

Em uma perspectiva de mapeamento e sistematização, um estudo sobre a aplicação do DT no desenvolvimento de software aponta a existência de múltiplos modelos, técnicas e combinações possíveis, ressaltando tanto a flexibilidade quanto a heterogeneidade da abordagem (SOUZA; FERREIRA; CONTE, 2017). Foram observados 11 modelos de DT, 55 técnicas e 10 ferramentas de software, concluindo-se que não há um modelo absoluto aplicável a todos os problemas e que

as técnicas costumam ser combinadas conforme o contexto do projeto (SOUZA; FERREIRA; CONTE, 2017). Essa constatação reforça a necessidade de critérios mais claros para seleção e adaptação metodológica.

Em relação aos benefícios e limites da abordagem, um estudo sobre o uso do DT na elicitação de requisitos em ambientes ágeis identifica 13 desafios estudados, dos quais oito apresentaram forte evidência de contribuição da abordagem, três evidência parcial e dois não permitiram conclusão segura, com destaque para dificuldades relacionadas a requisitos não funcionais, uso de artefatos e mudanças de requisitos (MARTINS et al., 2019). De forma convergente, uma revisão sistemática mostra que a aplicação do DT à Engenharia de Requisitos pode trazer vieses quando concentra esforços excessivos no usuário final e deixa em segundo plano outros stakeholders e aspectos técnicos essenciais, o que reforça a necessidade de equilíbrio entre empatia e disciplina de engenharia (PINTO; SIQUEIRA, 2020).

Estudos mais recentes avançam na proposição de modelos explícitos de integração. Argumenta-se que o modo de trabalho human-centered do DT pode complementar de maneira significativa a Engenharia de Requisitos, mas destaca-se que a combinação entre as duas abordagens não deve ser improvisada. Os resultados apontam para a necessidade de integração orientada por artefatos e por estratégias de adaptação entre práticas exploratórias e estruturadas (HEHN; MENDEZ, 2021). Em linha semelhante, evidencia-se que a participação ativa dos usuários e a validação recorrente das soluções contribuem para maior aderência entre software e contexto de uso, reduzindo retrabalho e ampliando o potencial inovador das soluções desenvolvidas (PALMA; ARAÚJO; SOUZA, 2022).

Para organizar os principais estudos utilizados como referência nesta pesquisa, o Quadro 2.1 apresenta uma síntese dos focos, conclusões e implicações metodológicas identificadas na literatura sobre a integração entre DT e SDLC. Como o objetivo deste trabalho não foi apenas indicar que DT e SDLC podem ser combinados, mas explicitar onde essa combinação ocorre no ciclo de desenvolvimento, também se tornou necessário observar em que etapa do SDLC cada autor posiciona etapas e técnicas do DT, como realiza essa inserção e por qual motivo metodológico.

Quadro 2.1: Resumo de trabalhos correlatos sobre a integração entre DT e SDLC

Estudo	Foco	Principais conclusões	Implicação para este trabalho
Vetterli et al. (2013) (VETTERLI et al., 2013)	Relação entre DT e Engenharia de Requisitos	Argumentam que a Engenharia de Requisitos precisa incorporar práticas mais exploratórias e centradas no usuário para responder a contextos de maior incerteza, mudança e inovação.	Reforça a pertinência de aproximar DT e SDLC nas fases iniciais do desenvolvimento.
Junior et al. (2013) (JUNIOR; BARROS; ROCHA, 2013)	Aplicação do DT à Engenharia de Requisitos	Mostram que técnicas de empatia e prototipação ampliam a comunicação com stakeholders e ajudam a refinar necessidades ainda ambíguas nas fases iniciais do SDLC.	Sustenta o uso do DT como apoio à elicitación e ao refinamento de requisitos.
Canedo e Costa (2018) (CANEDO; COSTA, 2018)	Estudo de caso sobre DT em requisitos ágeis	Relatam a modernização de dois sistemas reais com participação ativa de <i>designer</i> , observando que os protótipos melhoraram a comunicação no projeto; 75% dos participantes consideraram esses artefatos importantes para suas atividades e 92% avaliaram positivamente a experiência de uso dos protótipos.	Acrescenta evidência empírica de que prototipação e colaboração interdisciplinar fortalecem as fases iniciais do SDLC.
Souza et al. (2017) (SOUZA; FERREIRA; CONTE, 2017)	Mapeamento sistemático sobre DT no desenvolvimento de software	Identificam 11 modelos de DT, 55 técnicas e 10 ferramentas de apoio, concluindo que o DT é adaptável e não segue necessariamente uma ordem fixa entre fases.	Indica que o modelo proposto deve combinar flexibilidade metodológica com critérios explícitos de aplicação.
Pereira et al. (2018) (PEREIRA; RUSSO, 2018)	Revisão sistemática sobre DT e métodos ágeis	Indicam que práticas de DT aproximam usuários finais e equipe de desenvolvimento e melhoram qualidade e usabilidade do software.	Evidencia a utilidade do DT como complemento a processos já estruturados de desenvolvimento.
Martins et al. (2019) (MARTINS et al., 2019)	Desafios do DT na elicitación de requisitos	Identificam forte evidência de contribuição do DT em desafios ligados a envolvimento de stakeholders, definição, validação e priorização de requisitos, mas apontam limitações em requisitos não funcionais, artefatos e mudanças de requisitos.	Mostra que a integração proposta precisa manter disciplina de engenharia além da empatia e da cocriação.
Pinto e Siqueira (2020) (PINTO; SIQUEIRA, 2020)	Revisão sistemática sobre problemas do DT para ER	Indicam que o DT pode introduzir viés quando concentra excessivamente a atenção no usuário final e deixa em segundo plano outros stakeholders e aspectos técnicos da solução.	Justifica a necessidade de uma proposta híbrida que preserve visão humana sem perder abrangência analítica.
Hehn et al. (2020) (HEHN et al., 2020)	Integração entre DT e Engenharia de Requisitos	Propõem três abordagens de integração entre DT e ER e destacam que a combinação deve ser adaptada ao contexto e orientada pelos artefatos produzidos em cada etapa.	Apoia a construção de um fluxo com rastreabilidade entre achados, requisitos, decisões e protótipos.
Parizi et al. (2022) (PARIZI et al., 2022)	Mapeamento sistemático sobre uso de DT no desenvolvimento de software	Ao avaliar 127 estudos, identificam 3 estratégias de integração, 16 modelos, 85 técnicas e 7 critérios de seleção, concluindo que a escolha de técnicas ainda é um problema aberto.	Reforça a importância de explicitar critérios e relações entre etapas no modelo convergente proposto.

Estudo	Foco	Principais conclusões	Implicação para este trabalho
Palma et al. (2022) (PALMA; ARAÚJO; SOUZA, 2022)	Aplicação de DT ao desenvolvimento de software	Propõem uma metodologia de desenvolvimento baseada em práticas de DT e concluem que a abordagem favorece imersão no domínio, geração de ideias relevantes e colaboração com usuários, contribuindo para soluções mais inovadoras.	Reforça a relevância de manter o usuário no centro do processo sem perder a organização metodológica do SDLC.
Kahan et al. (2024) (KAHAN; GENERO; OLIVEROS, 2024)	Refinamento de processo de elicitação baseado em DT	Confirmam a utilidade de um processo estruturado em <i>Empathise</i> , <i>Synthesise</i> e <i>Ideate</i> , com destaque para o papel da empatia e para as lições aprendidas no refinamento do processo.	Sustenta a necessidade de detalhar operacionalmente como o DT é convertido em procedimento aplicável no SDLC.

A partir dessa síntese, observou-se que a integração do DT ao SDLC não se limita à adoção isolada de técnicas como empatia, ideação, prototipação e validação. O aspecto mais relevante está em compreender em quais etapas do processo de software essas práticas foram inseridas e de que modo produziram diferencial metodológico, seja na compreensão do problema, na elicitação e organização de requisitos, na concepção de alternativas ou na validação conceitual de soluções. Essa leitura foi essencial para o presente trabalho porque permitiu comparar não apenas quais técnicas foram usadas, mas em que momento do SDLC o DT agregou valor, com qual função e com qual justificativa. O Quadro 2.2 sistematiza esse ponto.

Quadro 2.2: Inserção de etapas e técnicas de Design Thinking em atividades do SDLC nos trabalhos correlatos

Estudo	Etapas do SDLC priorizada	Etapas/técnicas de DT inserida	Como e por que a integração é realizada
Vetterli et al. (2013) (VETTERLI et al., 2013)	Engenharia de Requisitos e definição inicial do problema	Práticas exploratórias, observação, compreensão do usuário e prototipação	O DT é posicionado antes da estabilização dos requisitos, para ampliar a investigação do problema e reduzir a dependência de especificações prematuras em contextos de alta incerteza.
Junior et al. (2013) (JUNIOR; BARROS; ROCHA, 2013)	Elicitação, análise e refinamento de requisitos	Empatia, ideação e prototipação	As práticas de DT são usadas como apoio à comunicação com stakeholders e à transformação de necessidades ambíguas em requisitos mais compreensíveis, porque a ER precisa lidar com expectativas ainda pouco estruturadas.
Canedo e Costa (2018) (CANEDO; COSTA, 2018)	Levantamento de requisitos em projetos ágeis de modernização	Participação de designer, protótipos e validação com participantes	O DT é inserido no início do desenvolvimento como meio de produzir protótipos que funcionam como artefatos de comunicação e validação, reduzindo ambiguidades antes da implementação.

Estudo	Etapa do SDLC priorizada	Etapa/técnica de DT inserida	Como e por que a integração é realizada
Canedo et al. (2020) (CANEDO et al., 2020)	Requisitos e planejamento em projetos ágeis	Entrevistas com usuários, <i>brainstorming</i> e prototipação	As técnicas aparecem associadas à elicitación e à discussão de funcionalidades em equipes ágeis, pois favorecem entendimento compartilhado, colaboração e validação rápida das ideias.
Souza et al. (2017) (SOUZA; FERREIRA; CONTE, 2017)	Diferentes atividades do desenvolvimento de software	Modelos, técnicas e ferramentas variadas de DT	O estudo mostra que a inserção não é fixa: técnicas de DT são selecionadas conforme o problema, a etapa do projeto e o objetivo da equipe, justificando a necessidade de critérios explícitos de adaptação.
Pereira et al. (2018) (PEREIRA; RUSSO, 2018)	Desenvolvimento ágil, especialmente descoberta, requisitos e validação de produto	Participação do usuário, ideação colaborativa e prototipação	O DT é tratado como complemento aos métodos ágeis, antecipando descoberta e validação de necessidades para melhorar qualidade, usabilidade e aderência do software ao contexto de uso.
Martins et al. (2019) (MARTINS et al., 2019)	Elicitación de requisitos	Envolvimento de stakeholders, definição, priorização e validação com apoio de técnicas de DT	A integração é justificada pela capacidade do DT de mitigar desafios clássicos da elicitación, embora os autores indiquem que requisitos não funcionais, artefatos e mudanças ainda exigem disciplina metodológica do SDLC.
Pinto e Siqueira (2020) (PINTO; SIQUEIRA, 2020)	Engenharia de Requisitos	Empatia e prototipação centradas no usuário, analisadas criticamente	O estudo não propõe uma inserção operacional única; ele alerta que o DT deve ser integrado à ER com controle sobre stakeholders secundários e restrições técnicas, para evitar viés excessivo no usuário final.
Hehn et al. (2020) (HEHN et al., 2020)	Engenharia de Requisitos orientada ao ser humano	Estratégias de integração entre DT e ER, apoiadas por artefatos intermediários	O DT é incorporado à ER por abordagens ajustáveis ao contexto do projeto; a integração é feita por artefatos que conectam achados exploratórios, necessidades humanas e requisitos rastreáveis.
Parizi et al. (2022) (PARIZI et al., 2022)	Atividades do desenvolvimento de software em geral	Estratégias, modelos e técnicas de DT mapeados na literatura	A inserção é analisada como problema de seleção metodológica: a técnica de DT deve ser escolhida conforme atividade do SDLC, maturidade da equipe, tipo de problema e objetivo esperado.
Palma et al. (2022) (PALMA; ARAÚJO; SOUZA, 2022)	Desenvolvimento de software com ênfase nas fases iniciais do SDLC	Imersão, geração de ideias, colaboração com usuários e prototipação	O DT é organizado como metodologia de apoio à compreensão do domínio e à concepção da solução, porque favorece inovação e maior aderência entre software e necessidades reais.
Kahan et al. (2024) (KAHAN; GENERO; OLIVEROS, 2024)	Processo de elicitación de requisitos	<i>Empathise, Synthesise e Ideate</i>	As etapas de DT são estruturadas como procedimento de elicitación, permitindo transformar empatia e síntese em insumos para requisitos; a justificativa é tornar a aplicação do DT mais operacional e repetível na ER.

A leitura conjunta desses estudos indica que o DT tende a produzir maior valor quando empregado como abordagem complementar ao SDLC, e não como substituta dos processos

do ciclo de vida. Os relatos empíricos apontam ganhos na aproximação com stakeholders, na compreensão do contexto de uso e na validação inicial de soluções por protótipos (CANEDO; COSTA, 2018; CANEDO et al., 2020; PALMA; ARAÚJO; SOUZA, 2022). Em paralelo, os estudos secundários reforçam que a literatura ainda é marcada por elevada heterogeneidade, com múltiplos modelos, técnicas e estratégias de integração, além de desafios associados à seleção de técnicas, aos requisitos não funcionais, aos artefatos e ao equilíbrio entre perspectivas humanas e restrições técnicas (SOUZA; FERREIRA; CONTE, 2017; MARTINS et al., 2019; PINTO; SIQUEIRA, 2020; PARIZI et al., 2022).

De modo geral, os estudos indicam que o Design Thinking contribui principalmente nas fases iniciais do SDLC, em especial na compreensão do problema, na elicitação de requisitos, na geração de alternativas e na validação conceitual de soluções. Entretanto, os resultados também evidenciam que sua aplicação isolada não resolve integralmente demandas de documentação, rastreabilidade, tratamento de requisitos não funcionais e controle técnico do processo. Assim, a literatura aponta que o maior potencial do DT no SDLC está menos na substituição de práticas consolidadas e mais na sua integração complementar aos processos tradicionais de desenvolvimento. Para esta pesquisa, esse achado orientou a construção do modelo híbrido: cada técnica de DT deve ser vinculada a uma etapa do SDLC, a um artefato esperado e a uma justificativa funcional, evitando uma integração genérica ou apenas discursiva.

Definição do Modelo Convergente DT-SDLC

3.1 Visão geral

Este capítulo tem como objetivo definir o *Modelo Convergente DT-SDLC*, dando continuidade ao percurso metodológico apresentado no Capítulo 1 e à fundamentação teórica consolidada no Capítulo 2. Enquanto o capítulo anterior reuniu os conceitos, abordagens e trabalhos correlatos que sustentam a pesquisa, este capítulo concentra-se em transformar essa base em uma proposta metodológica de integração entre *Design Thinking* e as fases iniciais do SDLC.

Seguindo a metodologia definida para o trabalho, a organização do capítulo corresponde às etapas de análise da relação entre DT e SDLC e de construção do modelo metodológico. Assim, as seções foram estruturadas para apresentar a condução do processo de definição do modelo: primeiro, são sistematizados os modelos de Design Thinking considerados na pesquisa; em seguida, são analisadas suas relações com atividades do SDLC; por fim, é apresentada a estrutura operacional do modelo convergente.

A Seção 3.2 organiza a comparação entre abordagens de Design Thinking com foco nos critérios necessários à definição metodológica. A Seção 3.3 relaciona essas abordagens às atividades iniciais do SDLC, identificando pontos de aproximação, limites e possibilidades de integração. A Seção 3.4 formaliza o *Modelo Convergente DT-SDLC*, descrevendo suas etapas, diretrizes de aplicação, artefatos esperados e mecanismos de retroalimentação e rastreabilidade.

Desse modo, o capítulo não retoma a fundamentação teórica de forma ampla, mas utiliza os referenciais já apresentados para justificar a construção do modelo. O objetivo é explicitar como a análise conceitual conduzida até aqui foi convertida em uma proposta estruturada, aplicável às

fases de compreensão do problema, organização de requisitos, ideação, prototipação e validação conceitual de soluções de software.

3.2 Estudo dos modelos de Design Thinking

O *Design Thinking* (DT) consolidou-se como uma abordagem relevante para inovação e resolução de problemas complexos, sendo amplamente explorado em diferentes contextos, inclusive no desenvolvimento de software. Entretanto, a diversidade de modelos e frameworks disponíveis torna essencial compreender suas particularidades, a fim de selecionar o mais adequado ao propósito deste estudo, que consistiu na integração entre as práticas do DT e as fases iniciais do SDLC até a prototipação. Para evitar avaliações apenas impressionistas, a comparação deste capítulo adota três critérios observáveis na descrição de cada modelo: quantidade de etapas explicitadas, presença ou ausência de subetapas nomeadas e quantidade de técnicas ou recursos operacionais explicitados na sistematização apresentada neste trabalho. Com base nesses critérios, considera-se mais formalizada a abordagem que explicita melhor a decomposição do processo e os meios operacionais de execução.

Uma das abordagens mais difundidas de Design Thinking caracteriza-se por uma estrutura conceitual baseada em três macroetapas: *Inspiration*, *Ideation* e *Implementation* (BROWN, 2009). Esse modelo enfatiza a empatia com o usuário, a geração de ideias e a experimentação como mecanismos centrais para o desenvolvimento de soluções inovadoras. No contexto deste trabalho, a etapa *Implementation* é interpretada como materialização e experimentação inicial da solução, e não como codificação do software. No recorte analítico adotado neste estudo, essa abordagem é classificada como menos formalizada porque explicita apenas três macroetapas, não discrimina subetapas e, na sistematização do Quadro 3.1, reúne sete técnicas exemplificadas sem vinculá-las a entregas intermediárias específicas. Por esse motivo, o modelo oferece diretrizes gerais de processo, mas menor decomposição operacional das atividades.

Outro modelo, amplamente associado à *Stanford d.school*, organiza o Design Thinking em cinco etapas: *Empathize*, *Define*, *Ideate*, *Prototype* e *Test* (LINDBERG et al., 2011). Essa

Quadro 3.1: Estrutura do Design Thinking segundo Brown (2009)

Fase	Subetapas	Quantidade Técnicas	Exemplos de Técnicas
Inspiration	Não explicitadas	2	Observação, entrevistas
Ideation	Não explicitadas	3	Brainstorming, storytelling, mapas de jornada
Implementation	Não explicitadas	2	Prototipação, testes com usuários

estrutura oferece maior clareza sequencial do que a formulação de Brown porque separa explicitamente compreensão do usuário, definição do problema, ideação, prototipação e teste em etapas distintas, facilitando sua compreensão e aplicação por equipes multidisciplinares. Entretanto, sob os critérios comparativos adotados neste capítulo, a abordagem permanece com formalização intermediária: embora detalhe melhor o encadeamento do processo, não explicita subetapas e, na sistematização do Quadro 3.2, associa sete técnicas distribuídas pelas fases. Assim, o ganho está sobretudo na sequência das etapas, e não no detalhamento operacional interno.

Quadro 3.2: Estrutura do Design Thinking segundo Lindberg et al. (2011)

Fase	Subetapas	Quantidade Técnicas	Exemplos de Técnicas
Empathize	Não explicitadas	2	Observação, entrevistas
Define	Não explicitadas	1	Resumo do problema
Ideate	Não explicitadas	2	Brainstorming, geração de ideias
Prototype	Não explicitadas	1	Protótipos rápidos
Test	Não explicitadas	1	Testes com usuários

Outra abordagem apresenta um modelo estruturado em quatro etapas: Imersão, Análise e Organização, Ideação e Prototipação (VIANNA et al., 2012). Diferentemente das abordagens anteriores, esse modelo apresenta maior formalização no recorte desta pesquisa porque, além de definir quatro etapas, explicita subetapas, como imersão preliminar e em profundidade, e nomeia um conjunto amplo de recursos metodológicos ao longo do processo. Na sistematização do Quadro 3.3, foi possível mapear vinte e cinco recursos operacionais entre técnicas e subetapas explicitadas no livro, o que reforça seu caráter prescritivo. Na etapa de Imersão, distinguem-se subetapas voltadas à investigação inicial e aprofundada do contexto do problema. A fase de Análise e Organização ordena e interpreta os dados coletados, permitindo a geração de insights.

A etapa de Ideação contempla a geração e seleção de alternativas de solução, enquanto a fase de Prototipação estrutura a construção e a validação conceitual das propostas desenvolvidas. Essa combinação amplia a rastreabilidade entre coleta, interpretação e validação, aproximando a abordagem de processos que exigem maior explicitação metodológica, como mostra o Quadro 3.3.

Quadro 3.3: Estrutura do Design Thinking segundo Vianna et al. (2012)

Fase	Subetapas	Quantidade Técnicas	Exemplos de Técnicas
Imersão	Preliminar e em profundidade	8	Reenquadramento, pesquisa exploratória, pesquisa desk, entrevistas, cadernos de sensibilização, sessão generativa, um dia na vida, sombra
Análise e Organização	Organização e interpretação dos dados	8	Cartões de insight, diagrama de afinidades, mapa conceitual, critérios norteadores, personas, mapa de empatia, jornada do usuário, blueprint
Ideação	Geração e seleção de ideias	4	Brainstorming, workshop de cocriação, cardápio de ideias, matriz de posicionamento
Prototipação	Baixa e alta fidelidade	5	Protótipo em papel, modelo de volume, encenação, storyboard, protótipo de serviços

Com base nas análises individuais apresentadas anteriormente, torna-se possível estabelecer uma comparação direta entre as abordagens de Design Thinking consideradas neste estudo, considerando aspectos relacionados à estruturação das etapas, presença de subetapas e diversidade de técnicas associadas. Essa comparação permite resumir as principais características dos modelos analisados, evidenciando suas diferenças em termos de detalhamento processual.

Quadro 3.4: Comparação entre abordagens de Design Thinking

Critério	Brown (2009)	Lindberg et al. (2011)	Vianna et al. (2012)
Quantidade de fases	3	5	4
Subetapas definidas	Não	Não	Sim
Recursos mapeados	7	7	25
Fase de análise/organização	Não	Parcial (<i>Define</i>)	Sim

O Quadro 3.4 evidencia que os três modelos analisados não se diferenciam apenas pela quantidade de fases, mas também pelo grau de detalhamento metodológico que oferecem para

orientar a condução de um projeto. Brown (2009) apresenta uma estrutura mais sintética, organizada em três macrofases, adequada para comunicar a lógica geral do Design Thinking e reforçar princípios como empatia, ideação e experimentação. Entretanto, por não explicitar subetapas nem uma fase específica de análise e organização dos achados, sua aplicação em projetos de software tende a exigir complementações quando o objetivo é transformar dados de investigação em requisitos e artefatos mais estruturados.

Lindberg et al. (2011), por sua vez, avançam em relação a Brown ao distribuir o processo em cinco momentos, separando empatia, definição, ideação, prototipação e teste. Essa decomposição torna o fluxo mais claro e facilita sua aplicação por equipes multidisciplinares. Ainda assim, o Quadro 3.4 mostra que o modelo também não explicita subetapas e apresenta o mesmo número de recursos mapeados em Brown (sete), o que sugere uma contribuição mais associada à clareza sequencial do processo do que ao detalhamento das técnicas e dos artefatos necessários para sua execução.

Já Vianna et al. (2012) apresentam maior nível de operacionalização, pois combinam quatro fases, subetapas definidas, etapa própria de análise e organização e maior variedade de recursos metodológicos mapeados (vinte e cinco). Essa diferença é relevante para a continuidade da análise, pois a integração entre Design Thinking e SDLC demanda mecanismos para coletar dados, organizar interpretações, gerar insights, estruturar requisitos, conceber alternativas e validar protótipos. Dessa forma, a leitura do Quadro 3.4 não encerra a escolha metodológica, mas indica elementos que serão retomados na seção seguinte, quando as abordagens forem relacionadas às atividades iniciais do SDLC.

3.3 SDLC x Design Thinking

Trazendo como contexto as técnicas e fases citadas na seção anterior, busca-se esclarecer a afinidade entre o uso do DT e o SDLC de forma ampla, tomando as atividades descritas na literatura sobre SDLC como referência analítica (SOMMERVILLE, 2019). Assim, a comparação parte de um ciclo que envolve compreensão do problema e engenharia de requisitos, análise e refinamento de requisitos, concepção da solução, projeto, implementação, validação, implan-

tação, evolução e manutenção. Neste momento, não se assume antecipadamente que o DT se aplica apenas a uma parte desse ciclo; a análise busca verificar onde há correspondência direta, onde há aproximação apenas indireta e onde não há aderência metodológica suficiente.

Nesta seção, a análise é conduzida em duas partes complementares. Na primeira, são examinadas separadamente as abordagens de Brown, Lindberg et al. e Vianna et al., considerando suas fases, técnicas associadas e possíveis justificativas de aproximação com atividades do SDLC. Os quadros individuais registram principalmente as correspondências diretas entre cada abordagem de DT e as atividades do SDLC. Na segunda parte, essas leituras são consolidadas em um quadro comparativo mais amplo, que inclui também atividades sem correspondência direta, permitindo justificar a delimitação do modelo apenas após a comparação.

Brown (2009)

Na abordagem proposta por Brown (BROWN, 2009), o *Design Thinking* aparece como um processo amplo, iterativo e fortemente orientado à inovação centrada no ser humano. Seu principal interesse para esta análise não está na formalização do método, mas na capacidade de ampliar a compreensão do problema, estimular a geração de alternativas e favorecer a experimentação inicial da solução. Isso faz com que a abordagem seja particularmente útil em cenários exploratórios, nos quais a equipe ainda precisa construir entendimento sobre o contexto de uso e abrir o espaço de possibilidades antes de convergir para uma definição mais precisa.

O Quadro 3.5 apresenta essa leitura, relacionando as fases de Brown às técnicas associadas a cada momento e às atividades do SDLC com as quais há maior aproximação direta.

A associação de *Inspiration* à compreensão do problema e à engenharia de requisitos decorre do fato de Brown enfatizar observação, entrevistas e contato direto com o contexto de uso. Essa leitura encontra respaldo nos trabalhos correlatos que apontam a necessidade de aproximar a Engenharia de Requisitos de práticas mais exploratórias, centradas no usuário e capazes de lidar com necessidades ainda ambíguas (VETTERLI et al., 2013). Assim, a justificativa de uso dessa etapa está na abertura inicial do problema e na identificação de stakeholders e restrições, ainda que Brown não detalhe mecanismos de registro e rastreabilidade.

Quadro 3.5: Relação entre a abordagem de Brown e atividades do SDLC

Fase	Técnicas associadas	Atividade do ciclo de vida relacionada	Justificativa de uso
Inspiration	Observação, entrevistas	Engenharia de requisitos, análise de domínio, identificação de stakeholders	Sua aproximação se justifica por apoiar a leitura inicial do contexto e das necessidades dos usuários, embora ofereça pouca orientação para registro, organização e rastreabilidade.
Ideation	Brainstorming, storytelling, mapas de jornada	Concepção da solução, modelagem conceitual, definição inicial de funcionalidades	Foi considerada útil para discutir alternativas e esboçar possibilidades de solução, ainda que a priorização e a análise permaneçam pouco formalizadas.
Implementation	Prototipação, testes com usuários	Projeto conceitual, fluxos de interface, protótipos e validação conceitual	Foi interpretada como espaço de experimentação conceitual, adequado à coleta de feedback antes da implementação técnica, sem equivaler à codificação ou à validação formal do sistema.

No caso de *Ideation*, a correlação com a concepção da solução e com a modelagem conceitual foi adotada porque essa fase amplia o espaço de alternativas antes da convergência. Técnicas como *brainstorming*, *storytelling* e mapas de jornada dialogam com achados de estudos que observam o uso recorrente de brainstorming, colaboração com usuários e práticas de ideação em projetos de software orientados por DT (CANEDO et al., 2020). Ainda assim, a justificativa é tratada com ressalva, pois estudos sobre desafios do DT na Engenharia de Requisitos indicam dificuldades relacionadas à priorização, aos artefatos e ao tratamento de requisitos não funcionais (MARTINS et al., 2019).

Já *Implementation* foi relacionada ao projeto conceitual, à prototipação e à validação conceitual porque, na leitura adotada neste trabalho, esse espaço representa a materialização inicial da solução para experimentação com usuários. Essa interpretação é apoiada por estudos nos quais protótipos aparecem como artefatos relevantes de comunicação, validação inicial e alinhamento entre participantes do projeto (CANEDO; COSTA, 2018). Por isso, a justificativa de uso não a equipara à implementação técnica do software, mas ao momento em que fluxos e interfaces ganham forma suficiente para coleta de feedback, sem abranger codificação, integração técnica ou validação formal do sistema.

Desse modo, a abordagem de Brown mostra maior aderência às atividades do SDLC ligadas à abertura do problema, à geração inicial de alternativas e à experimentação conceitual. Em

diálogo com os trabalhos correlatos, seu uso pode ser justificado como apoio exploratório às fases iniciais do SDLC, mas com menor densidade metodológica quando o processo exige organização analítica, documentação e rastreabilidade mais rigorosas (PARIZI et al., 2022).

Lindberg et al. (2011)

Na abordagem de Lindberg et al. (LINDBERG et al., 2011), amplamente associada à tradição da *Stanford d.school*, o destaque está na clareza sequencial do método. A organização em *Empathize*, *Define*, *Ideate*, *Prototype* e *Test* torna o processo didaticamente mais inteligível e favorece sua aplicação por equipes multidisciplinares. No recorte desta pesquisa, isso é especialmente relevante porque oferece uma ponte conceitual mais explícita entre momentos de empatia, definição do problema, geração de alternativas, materialização da solução e validação conceitual.

O Quadro 3.6 apresenta essa articulação entre as fases da abordagem, as técnicas associadas e as atividades do SDLC com as quais há maior aproximação direta.

Quadro 3.6: Relação entre a abordagem de Lindberg et al. e atividades do SDLC

Fase	Técnicas associadas	Atividade do ciclo de vida relacionada	Justificativa de uso
Empathize	Observação, entrevistas	Engenharia de requisitos, análise de domínio, identificação de stakeholders	Sua utilização se justifica como ponto de entrada para compreender usuários, contexto e expectativas dos atores, ainda que a coleta seja menos estruturada para cenários complexos.
Define	Resumo do problema, reformulação do desafio	Análise e refinamento de requisitos, definição do problema	Mostra-se pertinente para delimitar o problema e alinhar a equipe sobre o foco do projeto, embora apresente menor aprofundamento analítico e documental.
Ideate	Brainstorming, geração de ideias	Concepção da solução, definição inicial de funcionalidades	Foi relacionada à concepção da solução por favorecer discussão colaborativa de alternativas, sem oferecer, isoladamente, controle suficiente para priorização técnica.
Prototype	Protótipos rápidos	Projeto conceitual, arquitetura da informação, fluxos de interface	Justifica-se como representação preliminar da solução, útil para discutir arquitetura da informação e fluxos antes da construção técnica.
Test	Testes com usuários	Validação conceitual do protótipo	Foi considerada adequada para aprendizagem qualitativa com usuários sobre o protótipo, sem abranger testes técnicos ou homologação do software implementado.

A fase *Empathize* foi relacionada à compreensão do problema e à engenharia de requisitos porque se concentra na aproximação inicial com usuários, contexto e expectativas dos atores envolvidos. Essa justificativa dialoga com estudos que indicam ganhos do DT na comunicação com stakeholders, na compreensão do contexto de uso e na identificação de necessidades ainda ambíguas (VETTERLI et al., 2013). No entanto, a coleta permanece menos detalhada do que em processos formais de especificação, o que exige complementação por práticas de engenharia quando houver necessidade de documentação mais rigorosa.

Em *Define*, a aproximação com análise e refinamento de requisitos se justifica pela passagem da coleta para a definição do problema. Nessa etapa, Lindberg et al. estruturam a reformulação do desafio e a explicitação do foco do projeto, o que se aproxima, no SDLC, do momento de organizar achados, delimitar escopo e alinhar conceitualmente a equipe. Trabalhos correlatos reforçam essa leitura ao apontar que a integração entre DT e Engenharia de Requisitos precisa ser orientada por artefatos e por adaptação ao contexto, evitando que a empatia permaneça apenas como atividade exploratória sem continuidade metodológica (HEHN et al., 2020).

A correlação de *Ideate* com a concepção da solução decorre do fato de a fase concentrar a geração e a discussão colaborativa de alternativas. A justificativa de uso é sustentada por estudos que identificam brainstorming, colaboração interdisciplinar e participação de usuários como práticas recorrentes em projetos de software que incorporam DT (CANEDO et al., 2020). Ainda assim, mantém-se a ressalva de que a abordagem não oferece, por si só, critérios suficientes para priorização técnica ou detalhamento mais estruturado do sistema.

Na fase *Prototype*, a relação com projeto conceitual, arquitetura da informação e fluxos de interface foi estabelecida porque é nesse momento que as alternativas passam a ser representadas de forma tangível. Essa aproximação é coerente com estudos empíricos nos quais os protótipos funcionaram como artefatos de comunicação entre participantes do projeto e como apoio à compreensão de funcionalidades antes da construção técnica (CANEDO; COSTA, 2018). O vínculo com o SDLC ocorre, portanto, no plano da estruturação preliminar da solução.

Por fim, *Test* foi associado à validação conceitual do protótipo por se tratar de um momento de aprendizagem com usuários a partir do uso da solução representada. A justificativa é refor-

çada por estudos que associam validação recorrente, participação ativa dos usuários e avaliação de protótipos à maior aderência entre software e contexto de uso (CANEDO; COSTA, 2018). Ainda assim, essa etapa não é tratada como teste técnico, verificação formal ou homologação do software implementado.

Em resumo, a abordagem de Lindberg et al. se destaca por tornar mais explícita a transição entre atividades do ciclo de vida ligadas à compreensão do problema, organização dos achados, ideação, prototipação e validação conceitual. A literatura correlata apoia essa sequência como percurso didático e útil nas fases iniciais do SDLC, embora estudos de mapeamento também indiquem que a escolha de técnicas e a integração com artefatos de engenharia continuam sendo desafios metodológicos (PARIZI et al., 2022).

Vianna et al. (2012)

Já a abordagem de Vianna et al. (VIANNA et al., 2012) apresenta um direcionamento mais operacional, com detalhamento mais consistente de etapas, subetapas e técnicas. Por isso, tende a se mostrar particularmente adequada a contextos em que a integração entre DT e SDLC precisa ocorrer de forma mais aplicada, com maior capacidade de transformar dados qualitativos em artefatos, requisitos e decisões de projeto. Em vez de atuar apenas como estrutura conceitual, a proposta de Vianna oferece um repertório metodológico mais diretamente utilizável em projetos reais.

O Quadro 3.7 organiza essa relação entre fases, técnicas associadas e justificativas de uso no SDLC, evidenciando sua aproximação com compreensão do problema, refinamento de requisitos e prototipação orientada à validação.

A fase de *Imersão* foi relacionada à compreensão do problema, à engenharia de requisitos e à identificação de stakeholders porque Vianna explicita um conjunto de técnicas voltadas à investigação inicial e aprofundada do contexto. A justificativa de uso dessa etapa é reforçada por estudos que apontam a necessidade de práticas exploratórias, centradas no usuário e capazes de melhorar a comunicação com stakeholders nas atividades iniciais da Engenharia de Requisitos (VETTERLI et al., 2013). Técnicas como pesquisa desk, entrevistas, observação e mapa de

Quadro 3.7: Relação entre a abordagem de Vianna e atividades do SDLC

Fase	Técnicas associadas	Atividade do ciclo de vida relacionada	Justificativa de uso
Imersão	Pesquisa desk, entrevistas, observação, mapa de stakeholders	Engenharia de requisitos, análise de domínio, identificação de stakeholders	Sua adoção se justifica por estruturar a investigação do contexto, dos atores e das necessidades que orientam os requisitos iniciais.
Análise e Organização	Mapa de empatia, jornada do usuário, cartões de insight	Análise e refinamento de requisitos, definição do problema, critérios iniciais	Mostra-se adequada ao recorte por organizar achados e transformá-los em requisitos, critérios e delimitações dependentes da qualidade dos dados coletados.
Ideação	Brainstorming, workshop de cocriação, cardápio de ideias, matriz de posicionamento	Concepção da solução, modelagem conceitual, priorização funcional	Foi considerada pertinente por articular geração e seleção de alternativas com necessidades já identificadas, ainda exigindo avaliação técnica complementar.
Prototipação	Protótipos de baixa e alta fidelidade, testes com usuários	Projeto conceitual, fluxos de interface, validação conceitual	Justifica-se por tornar fluxos e decisões de interface avaliáveis antes da implementação, sem substituir a construção técnica ou a validação formal do sistema.

stakeholders, portanto, são interpretadas como meios para qualificar o diagnóstico do domínio antes da definição dos requisitos.

Em *Análise e Organização*, a correlação com análise e refinamento de requisitos decorre diretamente do propósito da etapa: transformar dados dispersos em interpretações, padrões e direcionadores de projeto. Técnicas como mapa de empatia, jornada do usuário e cartões de insight encontram apoio em estudos que destacam a necessidade de integrar DT e Engenharia de Requisitos por meio de artefatos, rastreabilidade e adaptação contextual (HEHN et al., 2020). Ao mesmo tempo, a literatura também indica que essa organização não deve ignorar requisitos não funcionais, stakeholders secundários ou restrições técnicas (PINTO; SIQUEIRA, 2020).

A fase de *Ideação* foi associada à concepção da solução e à modelagem conceitual porque combina geração de alternativas com mecanismos de organização e seleção das propostas. Essa justificativa dialoga com estudos que identificam brainstorming, cocriação e participação de usuários como práticas recorrentes no uso de DT em desenvolvimento de software (CANEDO et al., 2020). O uso de cardápio de ideias e matriz de posicionamento também responde a uma lacuna apontada por mapeamentos sistemáticos: a necessidade de explicitar critérios para seleção de técnicas e alternativas, em vez de aplicar o DT de forma genérica (PARIZI et al., 2022).

Na *Prototipação*, a relação com projeto conceitual, fluxos de interface e validação conceitual se sustenta pelo papel da etapa na materialização da solução. A justificativa de uso é especialmente apoiada por estudos empíricos que mostram protótipos como artefatos de comunicação, aprendizagem e avaliação preliminar com usuários (CANEDO; COSTA, 2018). Assim, protótipos em papel, storyboards e protótipos navegáveis são considerados pertinentes para avaliar decisões de interface e fluxo antes da implementação técnica, sem substituir testes formais do software.

Por essa combinação entre investigação, organização dos achados, ideação e prototipação, a abordagem de Vianna apresenta articulação mais detalhada com as atividades iniciais do SDLC, sobretudo quando o objetivo é converter achados de campo em requisitos e protótipos melhor justificados. A leitura dos trabalhos correlatos oferece apoio a essa escolha porque aponta ganhos na aproximação com stakeholders, na elicitação e refinamento de requisitos, na ideação e na validação conceitual por protótipos, mas também reforça a necessidade de critérios, rastreabilidade e equilíbrio entre perspectivas humanas e restrições técnicas (CANEDO et al., 2020). Assim, essas conclusões são utilizadas aqui apenas como evidências de apoio: o DT é tratado como abordagem complementar ao SDLC, especialmente nas atividades anteriores à implementação técnica, e não como substituto de todo o SDLC.

Resumo comparativo da relação entre DT e SDLC

Após a análise individual das três abordagens, o Quadro 3.8 passa a ter a função de reunir, em uma mesma estrutura, a leitura comparativa construída nesta seção sobre a relação entre DT e SDLC. Seu objetivo não é estabelecer equivalência rígida entre os autores, mas organizar, em um único plano, quais etapas de cada abordagem podem ser aproximadas das atividades gerais do SDLC e sob quais condições essa aproximação é interpretada.

Na leitura do Quadro 3.8, as primeiras linhas indicam que as três abordagens se aproximam, com intensidades diferentes, das atividades iniciais do SDLC. Em Lindberg et al., essa aproximação ocorre por meio de *Empathize*, *Define* e *Ideate*; em Brown, por *Inspiration* e *Ideation*; e em Vianna, por *Imersão*, *Análise e Organização* e *Ideação*. Essas relações são estabelecidas porque tais fases ampliam a leitura do contexto, apoiam a identificação de stakeholders,

organizam necessidades e favorecem a formulação inicial de alternativas de solução, aspecto também destacado em estudos que aproximam DT e Engenharia de Requisitos (VETTERLI et al., 2013).

Quadro 3.8: Resumo comparativo entre etapas do Design Thinking e atividades gerais do SDLC

Atividades do SDLC	Lindberg et al. (2011)	Tim Brown (2009)	Vianna (2012)	Justificativa de uso
Compreensão do problema / Engenharia de Requisitos	<i>Empathize</i>	<i>Inspiration</i>	<i>Imersão</i>	A aproximação se justifica por reunir práticas úteis à análise de domínio, identificação de stakeholders e levantamento inicial de necessidades.
Análise e refinamento de requisitos	<i>Define</i>	X	<i>Análise e Organização</i>	O uso é pertinente quando os achados precisam ser organizados em critérios, delimitações e requisitos mais claros.
Concepção da solução e modelagem conceitual	<i>Ideate</i>	<i>Ideation</i>	<i>Ideação</i>	O uso é pertinente para explorar alternativas e discutir, em nível conceitual, possibilidades de solução e funcionalidades iniciais.
Projeto, prototipação e validação conceitual	<i>Prototype / Test</i> (com ressalva)	<i>Implementation</i> (com ressalva)	<i>Prototipação</i>	A relação é considerada válida no nível conceitual, pois permite discutir fluxos, protótipos e feedback antes da implementação técnica.
Implementação técnica	X	X	X	Não há correspondência direta com codificação, integração de componentes ou construção executável do produto final.
Validação e testes de software	X	X	X	Não há correspondência direta com verificação técnica, testes funcionais, testes de integração ou homologação.
Implantação e operação inicial	X	X	X	Não há correspondência direta com disponibilização em ambiente real ou transição para uso produtivo.
Evolução e manutenção	X	X	X	Não há correspondência direta com correção de defeitos, adaptação tecnológica ou sustentação continuada.

A comparação também mostra diferenças entre os autores. Brown se aproxima da abertura exploratória do problema e da geração de possibilidades, mas não explicita uma etapa própria de análise e refinamento de requisitos. Lindberg et al. tornam mais clara a passagem entre empatia, definição, ideação, prototipação e teste, embora com menor detalhamento instrumental. Vianna et al., por sua vez, apresentam maior decomposição operacional ao combinar investigação, organização dos achados, ideação e prototipação com técnicas mais variadas.

A linha referente a projeto, prototipação e validação conceitual mostra uma zona intermediária de aproximação entre DT e SDLC. *Prototype, Implementation* com ressalva e *Prototipação* permitem materializar fluxos, interfaces e representações preliminares da solução, favorecendo avaliação conceitual e feedback de usuários sobre o protótipo (CANEDO; COSTA, 2018). Contudo, essa validação pertence ao nível da prototipação: quando a atividade do SDLC passa a envolver validação e testes de software implementado, não há correspondência direta com as técnicas de DT analisadas.

As linhas finais do quadro completam a comparação com o SDLC mais amplo. Implementação técnica, validação e testes de software, implantação/operação inicial e evolução/manutenção aparecem com **X** para todos os autores porque não há correspondência direta entre as etapas do DT analisadas e atividades como codificação, integração de componentes, verificação técnica, testes funcionais, homologação, disponibilização em ambiente real, correção de defeitos, adaptação tecnológica ou sustentação continuada do sistema. Essa constatação não é tomada como ponto de partida, mas resulta da comparação apresentada no quadro.

Desse modo, o Quadro 3.8 permitiu justificar a delimitação adotada no modelo convergente: o uso do DT mostrou-se mais pertinente para compreensão do problema, organização de requisitos, concepção da solução, prototipação e validação conceitual do protótipo, enquanto as atividades de implementação técnica, validação e testes de software, implantação e manutenção permanecem como etapas do SDLC sem correspondência direta nas técnicas de DT analisadas. Essa delimitação é coerente com mapeamentos que indicam maior presença do DT nas atividades iniciais e de apoio à concepção em projetos de software (PARIZI et al., 2022). A abordagem de Vianna et al. (2012) foi a que ofereceu maior adequação para uso na continuidade

deste trabalho, pois apresenta uma sequência completa para as fases de interesse, explicita uma etapa própria de análise e organização dos achados, possui subetapas definidas e reúne maior quantidade de técnicas operacionais.

Essa adequação torna-se mais evidente quando as técnicas propostas por Vianna et al. são observadas em relação às necessidades específicas das fases iniciais do SDLC. Na Engenharia de Requisitos, técnicas como reenquadramento, pesquisa exploratória, pesquisa desk, entrevistas, cadernos de sensibilização, sessão generativa, um dia na vida e sombra ampliam a compreensão do domínio, dos stakeholders, das tarefas reais e das necessidades explícitas e implícitas dos usuários. Com isso, a etapa de Imersão beneficia o SDLC ao reduzir a distância entre a visão técnica da equipe e o contexto efetivo de uso do sistema, tornando o levantamento de requisitos menos dependente apenas de declarações formais dos usuários (JUNIOR; BARROS; ROCHA, 2013).

Na etapa de análise e refinamento de requisitos, técnicas como cartões de insight, diagrama de afinidades, mapa conceitual, critérios norteadores, personas, mapa de empatia, jornada do usuário e blueprint permitem organizar os dados coletados, identificar padrões, consolidar perfis de uso, mapear fluxos e estabelecer direcionadores para a solução. Essas técnicas beneficiam o SDLC porque transformam informações qualitativas dispersas em insumos mais estruturados para definição de requisitos, priorização de necessidades, delimitação do problema e rastreabilidade entre evidências de campo e decisões de projeto (HEHN et al., 2020).

Na concepção da solução e na modelagem conceitual, técnicas como brainstorming, workshop de cocriação, cardápio de ideias e matriz de posicionamento favorecem a geração, combinação, discussão e seleção de alternativas. Para o SDLC, esse conjunto contribui para associar requisitos a possibilidades de solução, discutir funcionalidades com stakeholders, comparar propostas e apoiar decisões iniciais de escopo e arquitetura da informação antes do detalhamento técnico (CANEDO et al., 2020). Por fim, na fase de projeto, prototipação e validação conceitual, técnicas como protótipo em papel, modelo de volume, encenação, storyboard e protótipo de serviços permitem materializar fluxos, interfaces e experiências de uso de forma preliminar. Esse benefício é central para o SDLC porque antecipa feedback, revela inconsistências de entendimento,

reduz retrabalho e apoia ajustes antes da implementação técnica (CANEDO; COSTA, 2018).

Portanto, a escolha por Vianna et al. não decorre apenas da existência de uma fase de prototipação ou de sua proximidade geral com o Design Thinking, mas da capacidade de suas técnicas cobrirem de modo mais contínuo o percurso que vai da investigação do problema à validação conceitual da solução. Em comparação com Brown e Lindberg et al., a abordagem de Vianna oferece maior detalhamento operacional para apoiar a passagem entre dados de campo, requisitos, critérios de decisão, alternativas de solução e protótipos avaliáveis, o que a torna mais apropriada para fundamentar o modelo convergente proposto neste trabalho.

Assim, a comparação realizada nesta seção não conduz à combinação indistinta das contribuições de Brown, Lindberg et al. e Vianna et al. Brown e Lindberg et al. são utilizados como referências comparativas para compreender diferentes níveis de abstração do Design Thinking; entretanto, a base metodológica adotada para a proposição seguinte é a abordagem de Vianna et al., por ser a mais completa e operacional para integrar investigação, organização de requisitos, ideação e prototipação no contexto das fases iniciais do SDLC.

3.4 Proposição do Modelo Convergente DT-SDLC

A partir da análise comparativa realizada neste capítulo, o *Modelo Convergente DT-SDLC* passa a ser apresentado como uma proposta metodológica de integração entre técnicas de *Design Thinking* e atividades das fases iniciais do SDLC. O modelo não propõe substituir o SDLC por um processo de DT, nem tratar o DT como um fluxo paralelo aplicado externamente ao desenvolvimento. A proposta consiste em utilizar as técnicas de Vianna et al. (VIANNA et al., 2012) como mecanismos de apoio à execução de atividades próprias do SDLC, especialmente na compreensão do domínio, elicitação e análise de requisitos, especificação preliminar, modelagem do domínio, modelagem de casos de uso, definição de arquitetura funcional, organização de fluxos de navegação, prototipação e validação conceitual.

A escolha de Vianna et al. como fundamento do modelo decorre da conclusão construída na seção anterior. Entre as abordagens analisadas, essa é a que oferece maior detalhamento operacional, pois organiza o Design Thinking em quatro etapas, explicita subetapas, apresenta

maior variedade de técnicas e inclui uma fase própria de análise e organização dos achados. Por essa razão, Brown (2009) e Lindberg et al. (2011) permanecem como referências comparativas para compreensão do campo, mas não definem a estrutura central da proposta.

Desse modo, a convergência proposta ocorre pela sobreposição controlada entre duas dimensões. A primeira dimensão é o SDLC, responsável por orientar os produtos técnicos esperados no recorte conceitual do projeto, como documento de visão, requisitos funcionais, requisitos não funcionais, regras de negócio, modelo de domínio, histórias de usuário, diagrama de casos de uso, especificação de casos de uso, diagrama de atividades, arquitetura funcional, mapa de navegação, protótipos e critérios de validação. A segunda dimensão é o conjunto de técnicas de DT sistematizadas por Vianna et al., utilizadas para qualificar a forma como esses artefatos são produzidos, tornando-os mais sensíveis ao contexto de uso, às necessidades dos stakeholders e às evidências coletadas na investigação.

Com base nisso, o *Modelo Convergente DT-SDLC* é estruturado em quatro etapas integradas às fases iniciais do SDLC: *Imersão, Análise, Ideação e Prototipação*. A retroalimentação não é tratada como uma etapa independente, mas como um princípio transversal do fluxo: os resultados de uma atividade podem retornar às anteriores sempre que a análise, a modelagem, a prototipação ou a validação revelarem inconsistências, lacunas ou novas necessidades.

Estrutura operacional do modelo

O modelo proposto deve ser compreendido como uma integração operacional entre o SDLC e as técnicas de Vianna et al. Assim, cada etapa do modelo parte de uma atividade típica do SDLC e utiliza técnicas de DT para qualificar sua execução. A Imersão não substitui a Engenharia de Requisitos, mas amplia a capacidade de identificar atores, dores, restrições e necessidades implícitas. A Análise não substitui a especificação, mas apoia a transformação de achados qualitativos em requisitos, regras, critérios, modelo de domínio e rastreabilidade. A Ideação não substitui a modelagem nem o projeto preliminar, mas contribui para transformar requisitos em alternativas de solução, histórias de usuário, casos de uso, diagrama de atividades, arquitetura funcional, mapa de navegação e decisões de escopo. A Prototipação não substitui projeto detalhado, implementação ou testes, mas permite avaliar conceitualmente interfaces,

navegação e aderência da solução antes da codificação.

Para preservar a coerência metodológica, o modelo adota três princípios. O primeiro é a *centralidade do SDLC*, segundo o qual os artefatos técnicos continuam sendo produtos necessários do processo. O segundo é a *instrumentalização pelo DT*, segundo o qual as técnicas de Vianna et al. são utilizadas para melhorar a qualidade da investigação, da interpretação, da ideação e da validação conceitual. O terceiro é a *rastreabilidade entre evidência, requisito, modelo e protótipo*, segundo o qual cada funcionalidade, história de usuário, caso de uso, atividade, módulo funcional, tela ou percurso de navegação deve poder ser relacionado a necessidades, evidências ou critérios identificados nas etapas anteriores.

O Quadro 3.9 apresenta a estrutura operacional do *Modelo Convergente DT-SDLC*, explicitando que as técnicas de DT são aplicadas sobre fases e artefatos do SDLC, e não utilizadas como substitutas dessas fases. A leitura do quadro deve ser feita de forma horizontal: cada linha relaciona uma atividade do ciclo de vida à base correspondente em Vianna et al., às técnicas selecionadas, à função de integração e aos artefatos esperados. Dessa forma, evidencia-se que o modelo não cria um processo paralelo ao SDLC.

Quadro 3.9: Estrutura operacional do Modelo Convergente DT-SDLC

Fase do SDLC	Base em Vianna et al.	Técnicas de DT aplicadas	Integração realizada	Artefatos do SDLC gerados
Engenharia de Requisitos	Imersão preliminar e Imersão em profundidade	Reenquadramento, pesquisa exploratória, pesquisa desk, entrevistas, observação, sombra e mapa de stakeholders	Investigação do domínio, identificação de atores, dores, restrições, objetivos e necessidades explícitas e implícitas	Documento de visão, escopo preliminar, mapa de stakeholders, atores do sistema, necessidades, restrições e requisitos candidatos
Análise e Especificação de Requisitos	Análise e Organização	Cartões de insight, diagrama de afinidades, mapa conceitual, critérios norteadores, personas, mapa de empatia, jornada do usuário e blueprint	Organização dos achados, classificação de requisitos, identificação de regras de negócio, critérios de usabilidade, conceitos do domínio e rastreabilidade	Especificação preliminar de requisitos, requisitos funcionais e não funcionais, regras de negócio, modelo de domínio, matriz de rastreabilidade, personas e jornada do usuário

Fase do SDLC	Base em Vianna et al.	Técnicas de DT aplicadas	Integração realizada	Artefatos do SDLC gerados
Modelagem e Concepção da Solução	Ideação	Brainstorming, workshop de cocriação, cardápio de ideias e matriz de posicionamento	Transformação de requisitos em alternativas de solução, funcionalidades, interações, atividades, arquitetura funcional, fluxos e prioridades do MVP	Histórias de usuário, backlog conceitual, diagrama de casos de uso, especificação de casos de uso, diagrama de atividades, arquitetura funcional, mapa de navegação e decisões de escopo
Projeto Preliminar e Avaliação de Protótipo	Prototipação	Protótipo em papel, storyboard, encenação, protótipo de serviços, wireframes e protótipo navegável	Materialização da solução para avaliação conceitual de arquitetura da informação, interfaces, navegação, dados e aderência aos requisitos	Wireframes, especificação preliminar de telas, protótipo navegável, critérios de aceitação, roteiro de validação, relatório de feedback e backlog refinado

O Quadro 3.9 não deve ser interpretado como uma exigência de geração integral de todos os artefatos listados. Ele funciona como uma visão de referência para orientar quais produtos podem emergir em cada fase inicial do SDLC quando apoiada por técnicas de DT. Em uma aplicação concreta do modelo, a equipe deve selecionar apenas os artefatos que respondem às necessidades do projeto, ao grau de incerteza do problema, aos stakeholders envolvidos e ao nível de validação pretendido. No recorte deste trabalho, são tratados como essenciais para sustentar o protótipo: documento de visão, requisitos funcionais, requisitos não funcionais, regras de negócio, modelo de domínio, histórias de usuário, diagrama e especificação de casos de uso, diagrama de atividades, arquitetura funcional e mapa de navegação. Artefatos de implementação detalhada, como DER completo, modelo físico de banco de dados, código, plano de testes funcionais ou documentação de implantação, permanecem fora do escopo conceitual.

Para tornar esse recorte explícito, o modelo adota critérios de seleção de artefatos, apresentados no Quadro 3.10. Esses critérios ajudam a evitar tanto a informalidade excessiva, que comprometeria a rastreabilidade, quanto a produção documental desnecessária, que poderia tornar a aplicação do modelo mais pesada do que o necessário. Dessa forma, a seleção dos artefatos passa a funcionar como mecanismo de controle metodológico, mantendo apenas os registros que contribuem para explicar uma necessidade, apoiar uma decisão de projeto ou validar algum aspecto da solução proposta.

Quadro 3.10: Critérios para seleção dos artefatos do SDLC

Critério	Diretriz de seleção	Implicação para a aplicação do modelo
Necessidade do projeto	Produzir o artefato quando ele responder a uma decisão real de compreensão, especificação, modelagem ou validação.	Priorizar artefatos que demonstrem a viabilidade conceitual da solução, como documento de visão, requisitos, modelo de domínio, casos de uso, arquitetura funcional, mapa de navegação e protótipos.
Rastreabilidade	Manter artefatos suficientes para relacionar evidências, requisitos, regras, histórias, casos de uso, atividades, módulos e protótipo.	Registrar a origem dos requisitos e dos ajustes de interface sem exigir documentação completa de implementação.
Escopo da solução	Ajustar o nível de detalhamento ao recorte funcional e ao estágio de maturidade da proposta.	Focar no MVP e nos fluxos principais, deixando artefatos técnicos completos para etapas posteriores.
Validação pretendida	Selecionar artefatos compatíveis com o tipo de validação buscada: conceitual, funcional, técnica ou operacional.	Quando a validação for conceitual, priorizar wireframes, protótipo navegável, roteiro de validação e feedback.
Esforço e viabilidade	Evitar artefatos que aumentem custo documental sem contribuição direta para a decisão atual.	Não exigir todos os diagramas, modelos ou documentos do ciclo de vida, apenas aqueles necessários para sustentar a proposta.

Técnicas de DT e correlação com fases iniciais do SDLC

A aplicação do modelo requer que cada etapa seja conduzida como parte de um encadeamento metodológico do SDLC. Em consonância com a lógica operacional de Vianna et al. (VIANNA et al., 2012), as técnicas de DT são utilizadas para enriquecer atividades já reconhecidas no desenvolvimento de software, e não para criar uma trilha independente. Desse modo, cada etapa explicita sua propositividade dentro do ciclo de vida: compreender o contexto para eliciar requisitos, organizar achados para especificar o sistema, gerar alternativas para modelar a solução e materializar protótipos para validar conceitualmente decisões antes da implementação. As práticas do SDLC asseguram formalização, rastreabilidade e coerência entre problema, requisitos, modelos e protótipo, mas a quantidade de artefatos gerados deve ser proporcional ao objetivo e ao escopo da aplicação.

Imersão

A *Imersão* constitui o ponto de entrada do modelo. Sua base está nas subetapas de Imersão Preliminar e Imersão em Profundidade propostas por Vianna et al. (VIANNA et al., 2012). A entrada dessa etapa é uma demanda inicial ainda pouco estruturada, formada por uma necessidade institucional, uma oportunidade de melhoria, uma dor percebida pelos usuários ou uma hipótese preliminar de sistema.

- **Propósito da etapa:** compreender o domínio do problema antes da definição da solução,

de modo que o SDLC se inicie com evidências sobre atores, necessidades, restrições, processos existentes e lacunas de informação. A etapa apoia a elicitação inicial, a análise de domínio e a definição preliminar de escopo porque reduz a dependência de requisitos declarados de forma abstrata ou prematura. Essa integração está alinhada à visão de Vetterli et al., que defendem práticas mais exploratórias para a Engenharia de Requisitos em contextos incertos, e aos trabalhos de Hehn et al. e Kahan et al., nos quais o DT é tratado como apoio estruturado para transformar empatia e investigação em insumos de requisitos (VETTERLI et al., 2013; HEHN et al., 2020; KAHAN; GENERO; OLIVEROS, 2024). Ao mesmo tempo, considera-se a ressalva de Pinto e Siqueira, segundo a qual a centralidade no usuário deve ser equilibrada com outros stakeholders e restrições técnicas (PINTO; SIQUEIRA, 2020).

- **Itens operacionais:** delimitar o problema inicial; identificar stakeholders; levantar documentos e fontes secundárias; observar o processo atual; coletar percepções dos usuários; registrar dores, expectativas e restrições; formular uma primeira visão de escopo; derivar requisitos iniciais ainda sujeitos a análise e refinamento.
 - **Artefatos do SDLC gerados:** documento de visão, delimitação inicial do problema, mapa de stakeholders, atores do sistema, registros de entrevistas e observações, síntese das necessidades iniciais, restrições identificadas, requisitos iniciais ou candidatos, descrição do contexto de uso e escopo preliminar.
 - **Técnicas e diretrizes de uso:** considerando a sistematização de Vianna et al. e os cuidados apontados nos trabalhos correlatos, as técnicas de Imersão devem ser aplicadas como apoio à elicitação e à análise de domínio, sempre vinculando evidências coletadas aos artefatos do SDLC (VIANNA et al., 2012; VETTERLI et al., 2013; HEHN et al., 2020; PINTO; SIQUEIRA, 2020).
1. **Reenquadramento:** auxilia a revisar a formulação inicial do problema e a evitar que o projeto parta de uma solução previamente assumida. Beneficia a delimitação do problema, o documento de visão e o escopo preliminar. Como cuidado, deve ser

registrado com critérios claros para não gerar mudanças arbitrárias de foco ou perda de alinhamento com a demanda institucional.

2. **Pesquisa exploratória:** amplia o entendimento inicial sobre usuários, processos, dores e oportunidades, beneficiando a análise de domínio e a identificação de stakeholders. Como cuidado, seus achados devem ser tratados como hipóteses iniciais, e não como requisitos finais, pois a literatura correlata ressalta a necessidade de validação e rastreabilidade antes da especificação (HEHN et al., 2020; KAHAN; GENERO; OLIVEROS, 2024).
3. **Pesquisa desk:** apoia a coleta de documentos, normas, sistemas similares e restrições do domínio, contribuindo para descrição do contexto, glossário inicial e restrições do sistema. Como cuidado, não deve substituir o contato com stakeholders, pois documentos podem não refletir práticas reais de uso.
4. **Entrevistas:** captam necessidades declaradas, expectativas, conflitos e critérios de valor percebidos pelos stakeholders, fortalecendo registros de elicitação, atores do sistema e síntese das necessidades iniciais. Como cuidado, devem envolver perfis diversos para evitar viés de um único usuário ou grupo, problema destacado por Pinto e Siqueira ao discutir integrações centradas excessivamente no usuário final (PINTO; SIQUEIRA, 2020).
5. **Cadernos de sensibilização:** registram experiências, tarefas e percepções ao longo do tempo, beneficiando a compreensão de rotinas que não aparecem em reuniões formais. Como cuidado, exigem orientação simples e acompanhamento, para que os registros não se tornem superficiais ou difíceis de converter em evidências úteis para requisitos.
6. **Sessão generativa:** estimula stakeholders a expressarem percepções, dificuldades e expectativas por meios visuais ou colaborativos, apoiando a identificação de necessidades implícitas e oportunidades. Como cuidado, os resultados devem ser consolidados em evidências rastreáveis, evitando que opiniões isoladas sejam convertidas diretamente em escopo.

7. **Um dia na vida:** permite observar tarefas, interrupções, exceções e dependências do contexto real, beneficiando a descrição do contexto de uso e a identificação de restrições operacionais. Como cuidado, a observação deve preservar o escopo da investigação e não generalizar comportamentos de um único participante para todos os perfis.
8. **Sombra:** acompanha o usuário durante a execução de atividades, revelando comportamentos não verbalizados e dificuldades práticas que podem impactar requisitos funcionais e não funcionais. Como cuidado, deve ser combinada com entrevista ou validação posterior, pois a interpretação do observador pode introduzir vieses.

A Figura 3.1 sintetiza essa etapa como uma atividade de Engenharia de Requisitos enriquecida por técnicas de Imersão, evidenciando a passagem da investigação contextual para atores, necessidades, restrições, requisitos candidatos, visão do sistema e escopo preliminar.

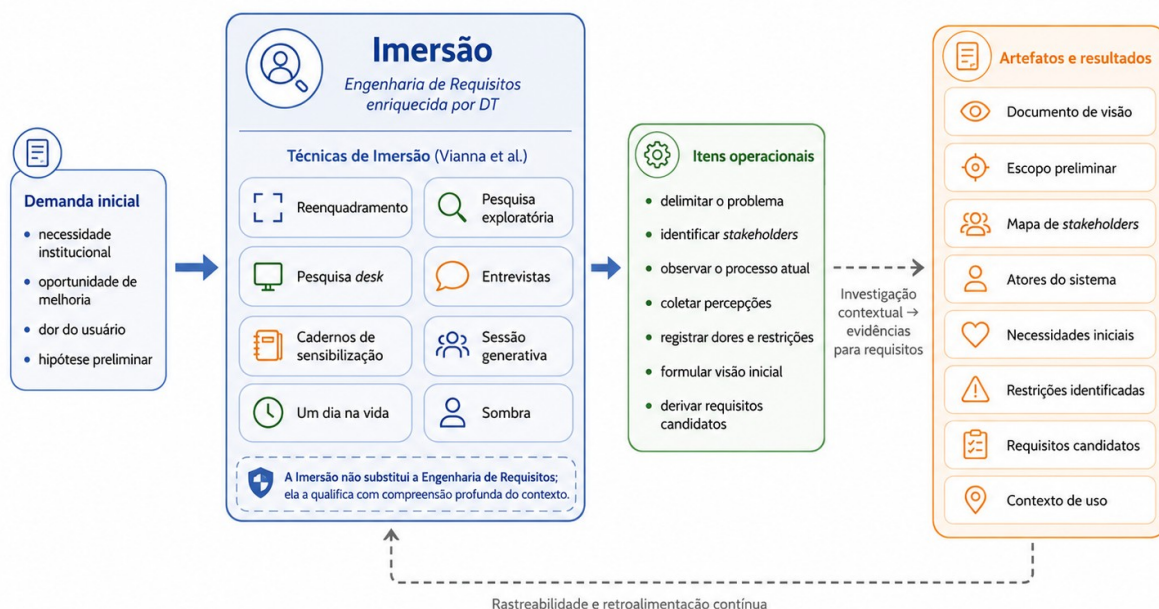


Figura 3.1: Imersão. **Fonte:** Elaborado pelo autor com apoio do ChatGPT (2026).

Análise

A etapa de *Análise* corresponde à fase de Análise e Organização de Vianna et al. Sua função é transformar dados qualitativos obtidos na imersão em artefatos formais ou semiformalizados de requisitos. Essa etapa é central para a convergência DT-SDLC porque cria a ponte entre pesquisa com usuários e especificação do sistema.

- **Propósito da etapa:** interpretar os dados coletados e convertê-los em elementos especificáveis do SDLC, ligando evidências da imersão a requisitos, regras de negócio, critérios norteadores e decisões iniciais de projeto. Essa etapa ajuda o fluxo de desenvolvimento porque cria uma transição controlada entre a investigação qualitativa e a especificação preliminar do sistema, mantendo rastreabilidade entre achados, necessidades, requisitos e critérios de decisão. Hehn et al. destacam a importância de artefatos intermediários para integrar DT e Engenharia de Requisitos, enquanto Martins et al. alertam que requisitos não funcionais, mudanças e artefatos precisam continuar submetidos à disciplina metodológica do SDLC (HEHN et al., 2020; MARTINS et al., 2019). Assim, a organização de insights não substitui a análise de requisitos; ela qualifica sua base empírica e reduz ambiguidades antes da modelagem.
- **Itens operacionais:** selecionar registros relevantes da imersão; agrupar achados por afinidade; identificar tensões, necessidades e oportunidades; construir perfis representativos de usuários; mapear jornadas; diferenciar certezas, suposições e dúvidas; derivar requisitos funcionais, não funcionais e regras de negócio.
- **Artefatos do SDLC gerados:** especificação preliminar de requisitos, requisitos funcionais, requisitos não funcionais, regras de negócio, restrições, critérios de usabilidade, modelo de domínio, matriz de rastreabilidade, personas, jornada do usuário, quadro CSD e glossário do domínio.
- **Técnicas e diretrizes de uso:** nesta etapa, as técnicas de Vianna et al. devem organizar evidências da Imersão em artefatos analisáveis, mantendo a disciplina de requisitos indi-

cada por Martins et al. e a lógica de artefatos intermediários defendida por Hehn et al. (VIANNA et al., 2012; HEHN et al., 2020; MARTINS et al., 2019).

1. **Cartões de insight:** sintetizam evidências relevantes em unidades analisáveis, beneficiando a matriz de rastreabilidade e a justificativa de requisitos. Como cuidado, cada cartão deve indicar origem, contexto e evidência associada, evitando que interpretações da equipe sejam confundidas com necessidades comprovadas.
2. **Diagrama de afinidades:** agrupa achados recorrentes e transforma dados dispersos em categorias de requisitos, regras e restrições. Como cuidado, a categorização deve ser revisada por mais de um participante ou validada com stakeholders, para reduzir subjetividade na leitura dos dados.
3. **Mapa conceitual:** explicita relações entre conceitos do domínio, apoiando glossário, regras de negócio e modelo de domínio. Como cuidado, deve diferenciar conceitos do negócio, funcionalidades do sistema e entidades de dados, evitando misturar análise do problema com desenho prematuro da solução.
4. **Critérios norteadores:** convertem padrões da análise em parâmetros de priorização, usabilidade e decisão de escopo. Como cuidado, precisam ser poucos, verificáveis e rastreáveis a evidências, pois critérios genéricos não ajudam a selecionar funcionalidades nem a orientar o protótipo.
5. **Personas:** consolidam perfis representativos de usuários e ajudam a qualificar atores, necessidades e prioridades. Como cuidado, não devem substituir stakeholders reais nem ocultar usuários secundários, restrições técnicas ou requisitos organizacionais, conforme alertam Pinto e Siqueira (PINTO; SIQUEIRA, 2020).
6. **Mapa de empatia:** organiza percepções, dores, expectativas e motivações dos usuários, contribuindo para requisitos de usabilidade, comunicação e experiência. Como cuidado, deve ser sustentado por dados coletados e não por suposições da equipe.
7. **Jornada do usuário:** evidencia etapas, pontos de dor, oportunidades e momentos críticos do processo, beneficiando fluxos, requisitos e critérios de aceitação. Como

cuidado, deve representar variações importantes do uso, para não transformar um fluxo idealizado no único cenário do sistema.

8. **Blueprint:** relaciona experiência do usuário, processos internos e suporte operacional, contribuindo para requisitos de serviço, dependências organizacionais e restrições do sistema. Como cuidado, deve explicitar fronteiras entre o que será implementado no software e o que depende de processo, equipe ou infraestrutura externa.

A Figura 3.2 ilustra a transformação dos achados da imersão em requisitos especificados, regras de negócio, critérios de usabilidade e matriz de rastreabilidade.

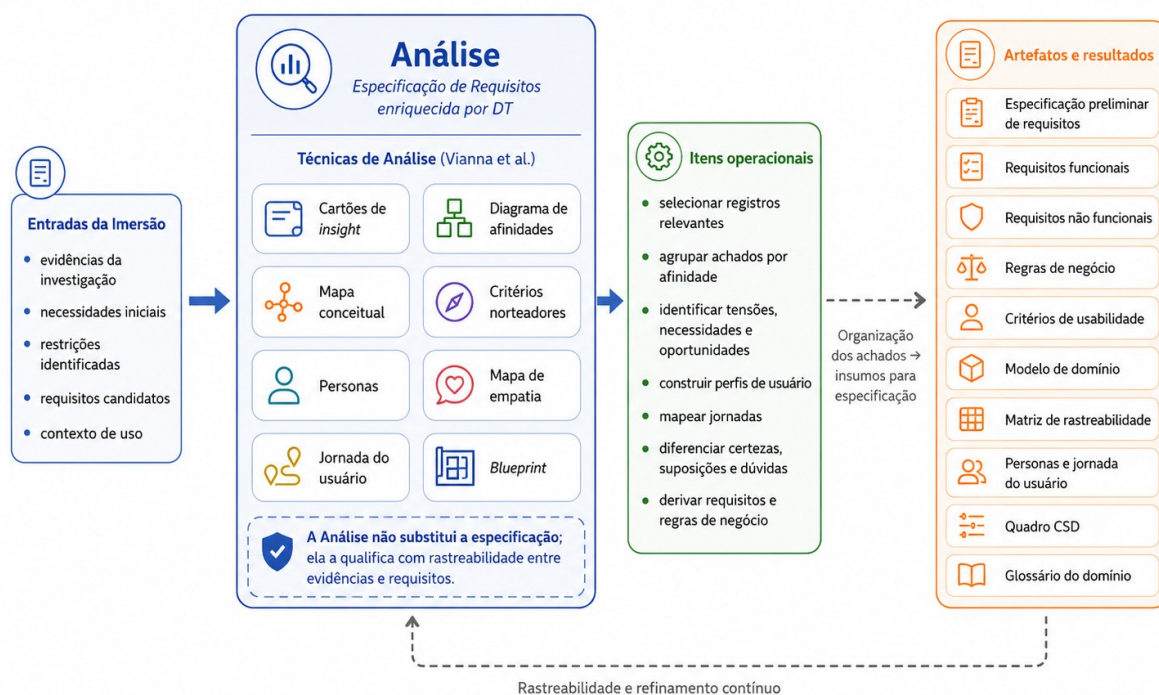


Figura 3.2: Análise. **Fonte:** Elaborado pelo autor com apoio do ChatGPT (2026).

Ideação

Na *Ideação*, a entrada é composta por requisitos especificados, regras de negócio, critérios norteadores, personas, jornadas e restrições identificadas na etapa anterior. A criatividade não ocorre de forma livre e desconectada, mas orientada pelos artefatos do SDLC já estruturados. Essa característica diferencia a etapa de uma sessão genérica de geração de ideias, pois as alternativas precisam responder ao problema delimitado, aos requisitos e aos modelos preliminares

do sistema.

- **Propósito da etapa:** transformar requisitos, critérios e restrições em alternativas de solução que possam avançar no SDLC como funcionalidades, interações, fluxos, casos de uso e decisões preliminares de escopo. A etapa ajuda o desenvolvimento porque evita que a modelagem conceitual surja apenas de preferências da equipe, vinculando cada proposta aos achados e requisitos já organizados. Canedo et al. observam que práticas como entrevistas, *brainstorming* e prototipação favorecem colaboração e discussão de funcionalidades em projetos ágeis, enquanto Pereira et al. destacam o DT como complemento para aproximar usuários e equipe de desenvolvimento (CANEDO et al., 2020; PEREIRA; RUSSO, 2018). Em perspectiva mais ampla, Souza et al. e Parizi et al. reforçam que a escolha das técnicas deve depender da etapa do desenvolvimento, do problema e do objetivo esperado, o que justifica a seleção orientada por artefatos nesta proposta (SOUZA; FERREIRA; CONTE, 2017; PARIZI et al., 2022).
- **Itens operacionais:** revisar critérios norteadores; formular desafios de solução; promover geração divergente de ideias; combinar alternativas complementares; avaliar aderência aos requisitos; selecionar funcionalidades iniciais; identificar atores e casos de uso; especificar casos de uso principais; organizar diagrama de atividades, arquitetura funcional e mapa de navegação; definir prioridades para o MVP e para o protótipo.
- **Artefatos do SDLC gerados:** alternativas de solução, funcionalidades candidatas, histórias de usuário, backlog conceitual, diagrama de casos de uso, especificação de casos de uso, diagrama de atividades, arquitetura funcional, mapa de navegação, matriz de priorização e decisões de escopo para prototipação.
- **Técnicas e diretrizes de uso:** na Ideação, as técnicas devem converter requisitos e critérios em alternativas de solução, mas sem romper a rastreabilidade com as evidências anteriores. Essa orientação dialoga com Vianna et al., com a percepção de Canedo et al. sobre colaboração em projetos ágeis e com os mapeamentos que defendem selecionar técnicas

conforme etapa, objetivo e contexto do desenvolvimento (VIANNA et al., 2012; CANEDO et al., 2020; SOUZA; FERREIRA; CONTE, 2017; PARIZI et al., 2022).

1. **Brainstorming:** amplia o repertório de alternativas e gera funcionalidades candidatas associadas aos requisitos. Beneficia backlog conceitual, histórias de usuário, casos de uso e possibilidades de fluxo. Como cuidado, deve ser precedido por critérios norteadores e finalizado com registro das relações entre ideia, requisito e evidência, evitando propostas criativas sem justificativa no SDLC.
2. **Workshop de cocriação:** aproxima stakeholders da definição da solução, permitindo discutir histórias de usuário, regras, casos de uso, atividades, prioridades e percursos de navegação. Como cuidado, precisa de facilitação para equilibrar vozes, evitar dominância de participantes e preservar restrições técnicas, aspecto coerente com as ressalvas sobre viés e abrangência da integração DT-SDLC (PINTO; SIQUEIRA, 2020; MARTINS et al., 2019).
3. **Cardápio de ideias:** organiza alternativas em conjuntos comparáveis, facilitando a identificação de funcionalidades complementares, redundantes ou incompatíveis. Beneficia backlog conceitual, arquitetura funcional, decisões de escopo e preparação do MVP. Como cuidado, cada ideia deve conter objetivo, requisito relacionado, impacto esperado e dependências, para não se tornar apenas uma lista informal de sugestões.
4. **Matriz de posicionamento:** avalia alternativas por impacto, viabilidade, aderência aos requisitos, esforço e prioridade, apoiando matriz de priorização, arquitetura funcional, escopo para prototipação e seleção do MVP. Como cuidado, os critérios devem ser explícitos e acordados, pois a priorização sem critérios pode reproduzir preferências da equipe ou pressões de stakeholders mais influentes.

A etapa de ideação também cumpre uma função de controle metodológico dentro do modelo proposto. Ao exigir que as ideias geradas sejam confrontadas com requisitos, critérios norteadores, histórias de usuário, casos de uso, diagrama de atividades, arquitetura funcional e mapa de navegação, reduz-se o risco de selecionar funcionalidades apenas por preferência da

equipe ou por facilidade técnica imediata. Desse modo, a criatividade é preservada, mas passa a operar dentro de limites definidos pelo problema investigado e pela especificação preliminar do sistema.

Essa orientação é particularmente relevante em projetos de software, nos quais decisões tomadas nas fases iniciais tendem a influenciar o escopo, a arquitetura funcional, a navegação, o modelo de dados e o esforço de prototipação. Ao transformar requisitos em histórias de usuário, casos de uso, atividades, módulos funcionais, percursos de navegação e funcionalidades candidatas, a etapa cria uma transição mais clara entre especificação e projeto conceitual. A Figura 3.3 representa essa dinâmica ao evidenciar a passagem dos requisitos e critérios organizados para casos de uso, fluxos, backlog e decisões iniciais de priorização.

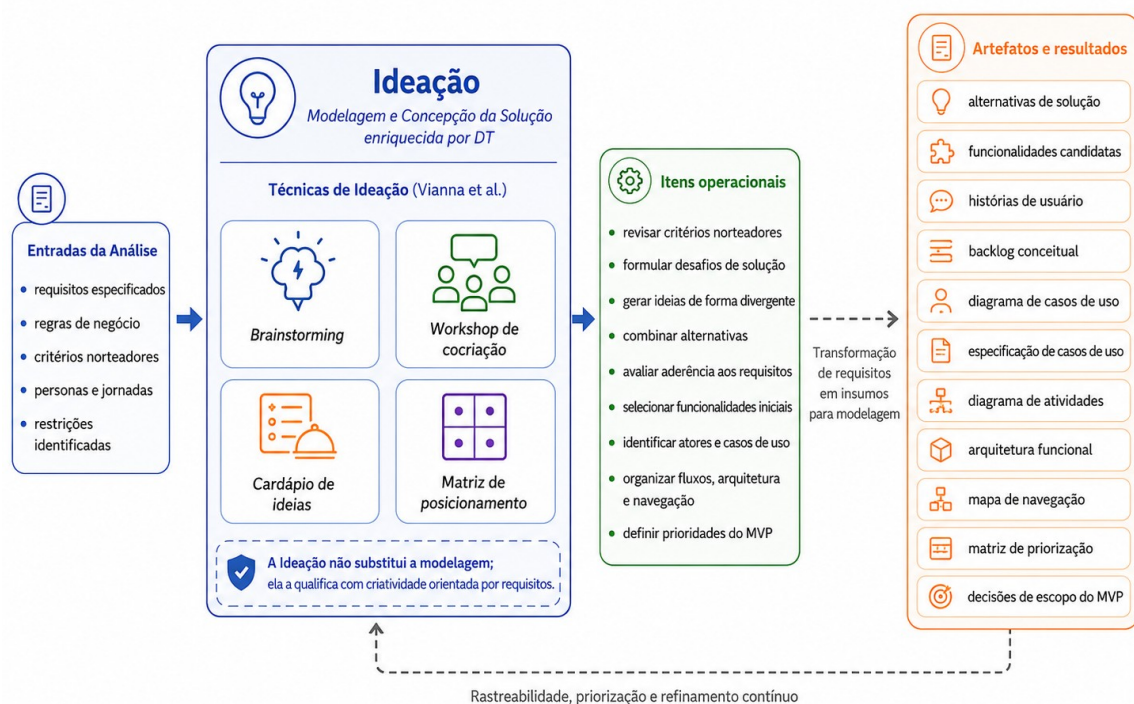


Figura 3.3: Ideação. **Fonte:** Elaborado pelo autor com apoio do ChatGPT (2026).

Prototipação

A *Prototipação* corresponde à etapa de Prototipação de Vianna et al., integrada ao projeto preliminar de software. Sua entrada é formada pelas funcionalidades priorizadas, histórias de

usuário, casos de uso, diagrama de atividades, arquitetura funcional, mapa de navegação, critérios de usabilidade e modelo de domínio. Sua função é materializar a solução em representações tangíveis para que usuários e stakeholders possam compreendê-la, avaliá-la e indicar ajustes antes da implementação técnica.

- **Propósito da etapa:** tornar a solução visível e testável em nível conceitual para que o SDLC avance da modelagem preliminar para a validação de entendimento antes da codificação. A etapa apoia projeto de interface, arquitetura da informação, modelagem básica de dados e revisão de requisitos porque permite observar se fluxos, telas, linguagem, informações solicitadas e funcionalidades priorizadas são compreendidos pelos usuários e stakeholders. Canedo e Costa relatam que protótipos funcionam como artefatos de comunicação e validação em requisitos ágeis, enquanto Pereira et al. associam a prototipação à melhoria de usabilidade e aderência ao contexto de uso (CANEDO; COSTA, 2018; PEREIRA; RUSSO, 2018). A ressalva metodológica permanece: como indicado na seção anterior, trata-se de validação conceitual do protótipo, e não de verificação técnica, testes funcionais, homologação ou validação do sistema implementado.
- **Itens operacionais:** selecionar fluxos prioritários; definir telas ou cenas principais; estruturar mapa de navegação; relacionar telas a casos de uso, histórias de usuário e diagrama de atividades; verificar aderência ao modelo de domínio e à arquitetura funcional; construir protótipos de baixa ou média fidelidade; simular interações; observar compreensão dos usuários; registrar feedback; revisar requisitos, fluxos, dados e decisões de interface.
- **Artefatos do SDLC gerados:** arquitetura da informação, mapa de navegação, wireframes, protótipos de baixa ou média fidelidade, protótipo navegável, especificação preliminar de telas, roteiro de validação, critérios de aceitação conceitual, relatório de feedback, lista de problemas identificados, ajustes em requisitos e backlog refinado.
- **Técnicas e diretrizes de uso:** na Prototipação, as técnicas devem materializar decisões de requisitos, fluxos, interface e dados para validação conceitual antes da implementação. Essa aplicação segue Vianna et al. e dialoga com Canedo e Costa, Pereira et al. e

Canedo et al., que apontam a prototipação como artefato de comunicação, colaboração e validação inicial em projetos de software (VIANNA et al., 2012; CANEDO; COSTA, 2018; PEREIRA; RUSSO, 2018; CANEDO et al., 2020).

1. **Protótipo em papel:** materializa rapidamente telas, fluxos e alternativas de interação, beneficiando wireframes iniciais, mapa de navegação e revisão de requisitos. Como cuidado, deve ser usado para aprendizagem rápida, e não como especificação final de interface ou prova de viabilidade técnica.
2. **Modelo de volume:** quando aplicável, permite representar elementos tangíveis, estruturas físicas ou relações espaciais envolvidas na solução. No contexto de software web, seu uso tende a ser indireto, apoiando a compreensão de objetos, documentos, ambientes ou dispositivos relacionados ao processo. Como cuidado, deve ser adotado apenas quando houver benefício claro para requisitos ou contexto de uso, evitando artefatos sem relação com o sistema.
3. **Encenação:** simula papéis, entradas, respostas e exceções, favorecendo revisão de fluxos, regras de negócio, casos de uso, diagrama de atividades e requisitos de interação. Como cuidado, a simulação deve registrar variações, dúvidas e decisões resultantes, para que o aprendizado seja incorporado aos artefatos do SDLC.
4. **Storyboard:** representa cenários de uso em sequência, ajudando a relacionar histórias de usuário, casos de uso, diagrama de atividades, telas e eventos principais. Como cuidado, deve incluir cenários alternativos ou pontos de falha relevantes, pois narrativas lineares podem ocultar exceções importantes para o projeto.
5. **Protótipo de serviços:** evidencia relações entre usuário, processo, suporte e sistema, contribuindo para arquitetura da informação, dependências operacionais e requisitos de apoio. Como cuidado, deve separar responsabilidades do software e responsabilidades organizacionais, para não atribuir ao sistema problemas que dependem de processo ou gestão.
6. **Wireframes e protótipos navegáveis:** permitem avaliar consistência de interface,

sequência de navegação, dados necessários e compreensão dos usuários, gerando relatório de feedback, ajustes no backlog, refinamento do mapa de navegação, revisão da arquitetura funcional e decisões de interface. Como cuidado, não substituem testes funcionais, verificação técnica ou homologação do sistema implementado; seu papel é validar entendimento e aderência conceitual antes da codificação.

A Figura 3.4 apresenta essa etapa ao mostrar a materialização dos casos de uso, atividades, arquitetura funcional, mapa de navegação e requisitos priorizados em protótipos avaliáveis, gerando feedback para a versão revisada da solução.

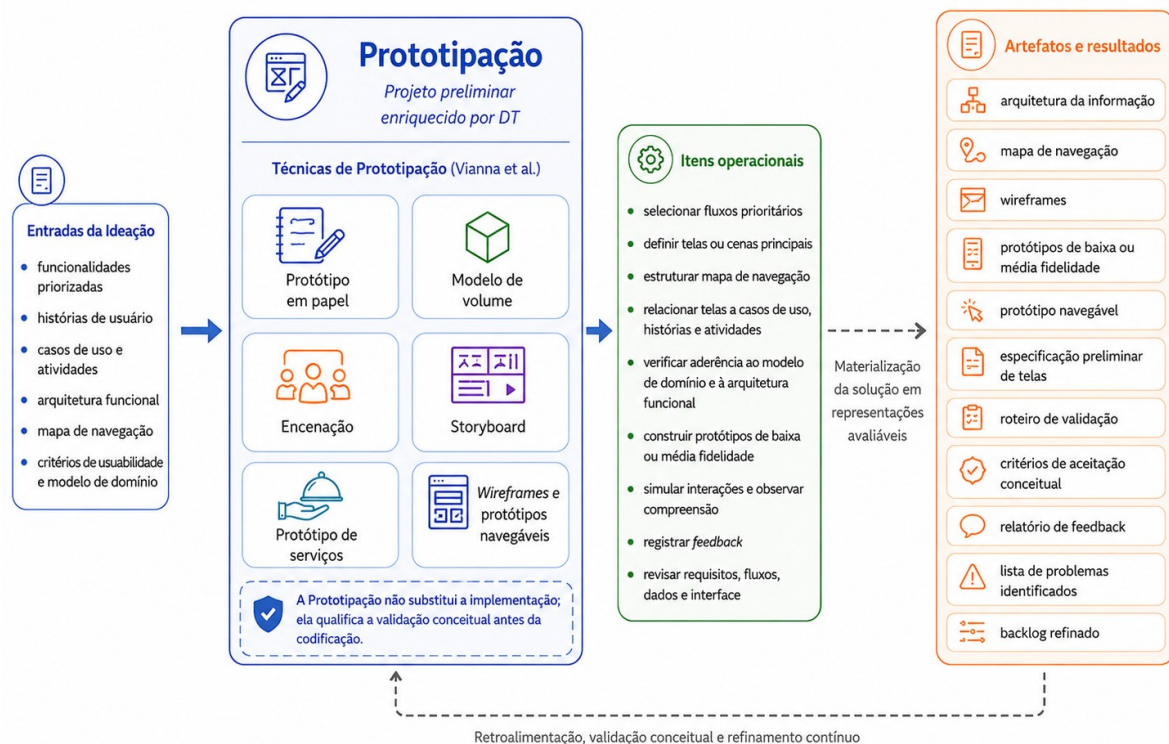


Figura 3.4: Prototipação. **Fonte:** Elaborado pelo autor com apoio do ChatGPT (2026).

Retroalimentação e rastreabilidade do fluxo

Embora a retroalimentação não componha uma etapa independente em Vianna et al., ela é incorporada ao modelo como princípio transversal de integração com o SDLC. Sempre que a análise de requisitos, o modelo de domínio, a definição de casos de uso, a arquitetura funcional, o

diagrama de atividades, o mapa de navegação ou a validação conceitual revelar inconsistências, lacunas ou novas necessidades, o fluxo pode retornar à atividade mais adequada. Assim, um problema percebido no protótipo pode exigir ajuste em fluxos, revisão de requisitos, alteração em casos de uso, refinamento do mapa de navegação, revisão da arquitetura funcional ou nova investigação com stakeholders.

Essa retroalimentação deve ser controlada por práticas do SDLC, como registro de decisões, atualização documental, análise de impacto, revisão de escopo e manutenção da rastreabilidade entre evidências, requisitos, modelos e protótipos. Desse modo, o modelo preserva a flexibilidade própria do Design Thinking sem perder a formalização necessária ao desenvolvimento de software.

Diretrizes de aplicação do modelo em projetos de software

Para que o *Modelo Convergente DT-SDLC* possa abranger diferentes propostas de desenvolvimento de software, sua aplicação deve seguir diretrizes de adaptação e controle. A primeira diretriz é selecionar, para cada atividade do SDLC, as técnicas de Vianna et al. mais adequadas à complexidade do projeto, à disponibilidade dos stakeholders, ao grau de incerteza do problema e ao nível de documentação exigido. O modelo não pressupõe o uso simultâneo de todas as técnicas de DT, nem a geração de todos os artefatos possíveis do ciclo de vida; pressupõe a escolha coerente daqueles que melhor qualificam a compreensão, especificação, modelagem e validação conceitual do projeto.

A segunda diretriz é preservar a rastreabilidade entre achado, insight, requisito, regra de negócio, modelo de domínio, história de usuário, caso de uso, diagrama de atividades, arquitetura funcional, mapa de navegação, decisão de interface e protótipo. Toda funcionalidade representada no protótipo deve possuir vínculo com necessidades identificadas na imersão, requisitos especificados e modelos preliminares da solução. Do mesmo modo, todo ajuste gerado na validação conceitual deve ser reincorporado aos artefatos anteriores, evitando contradições entre investigação, especificação, modelagem e solução. A rastreabilidade, nesse contexto, não significa excesso documental, mas suficiência de registros para explicar por que determinada decisão foi tomada.

A terceira diretriz é reconhecer o limite de escopo do modelo. A proposta é particularmente adequada às fases iniciais do SDLC, nas quais a compreensão do problema, a negociação entre stakeholders, a especificação de requisitos, a modelagem conceitual e a definição preliminar da solução exercem forte impacto sobre o produto final. No contexto deste trabalho, isso significa sua aplicação até a prototipação de uma plataforma web, sem avançar para codificação, testes técnicos, implantação ou manutenção do sistema. Portanto, artefatos orientados à implementação detalhada, implantação ou sustentação só devem ser previstos quando forem necessários à decisão conceitual do projeto.

Desse modo, a proposição apresentada oferece uma integração operacional entre técnicas de Vianna et al. e atividades do SDLC, com foco na produção seletiva de artefatos do ciclo de vida. A Figura 3.5 sintetiza esse encadeamento ao apresentar as fases do SDLC apoiadas por técnicas de DT, seus principais artefatos e a retroalimentação transversal que permite revisar decisões sempre que surgirem novas necessidades, inconsistências ou oportunidades.

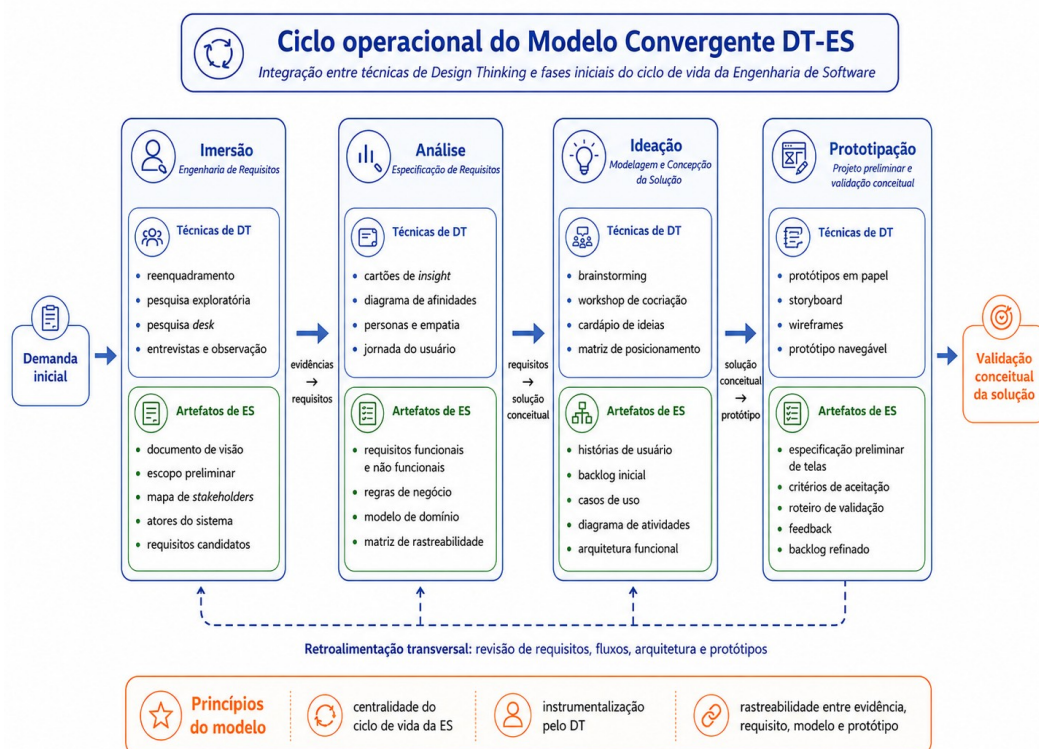


Figura 3.5: Ciclo operacional do Modelo Convergente DT-SDLC. **Fonte:** Elaborado pelo autor com apoio do ChatGPT (2026).

Aplicação do Modelo Convergente no Estudo de Caso *TCCMatch*

4.1 Visão geral

Este capítulo tem como objetivo apresentar a aplicação do *Modelo Convergente DT-SDLC* no estudo de caso do *TCCMatch*, dando continuidade ao percurso metodológico descrito no Capítulo 1 e ao modelo definido no Capítulo 3. Enquanto o capítulo anterior formalizou as etapas, diretrizes, artefatos e mecanismos de rastreabilidade do modelo, este capítulo concentra-se em demonstrar como essa estrutura orientou o desenvolvimento conceitual de uma plataforma web voltada à mediação entre alunos e professores no processo de definição de orientações de Trabalhos de Conclusão de Curso (TCC).

Seguindo a metodologia definida para o trabalho, a organização do capítulo corresponde às etapas de aplicação do modelo no estudo de caso e de análise dos resultados obtidos. Assim, as seções foram estruturadas para apresentar a condução da aplicação: primeiro, caracteriza-se o contexto do caso e o delineamento qualitativo utilizado; em seguida, discutem-se as ameaças à validade do estudo; depois, descreve-se a produção dos artefatos do protótipo conforme as etapas do modelo; por fim, discutem-se os resultados gerados pela aplicação.

A Seção 4.2 apresenta o contexto do *TCCMatch*, os stakeholders considerados e a estratégia de aplicação do modelo. A Seção 4.3 explicita as ameaças à validade consideradas na interpretação dos resultados. A Seção 4.4 descreve a execução do *Modelo Convergente DT-SDLC* no protótipo, organizando os artefatos produzidos nas etapas de *Imersão*, *Análise*, *Ideação* e *Prototipação*. A Seção 4.5 reúne os resultados e a discussão da aplicação, destacando as contribuições do modelo para a passagem entre achados qualitativos, requisitos, decisões de escopo,

fluxos, protótipos e critérios de validação conceitual.

Desse modo, o capítulo não busca demonstrar uma implementação completa do sistema, nem avançar para testes técnicos, implantação ou manutenção. O objetivo é evidenciar como o modelo proposto pode ser utilizado para orientar, de forma estruturada e rastreável, as fases iniciais do SDLC em uma situação concreta de desenvolvimento conceitual, mantendo o foco na compreensão do problema, na organização dos requisitos, na concepção da solução e na validação conceitual do protótipo.

4.2 Contexto do caso e delineamento qualitativo

O sistema concebido, denominado *TCCMatch*, foi pensado como uma plataforma web capaz de aproximar alunos e professores a partir de compatibilidade temática, disponibilidade de orientação e interesses declarados pelos usuários. A proposta parte do entendimento de que o processo atual de formação das duplas de orientação costuma ocorrer de forma dispersa, apoiado em conversas informais, troca de mensagens, indicações entre colegas e busca fragmentada por informações sobre docentes, linhas de pesquisa e vagas abertas.

Para fins de aplicação do modelo, considerou-se um curso de graduação com aproximadamente 150 alunos ativos, dos quais cerca de 35 encontram-se em fase de pré-projeto ou matrícula em TCC, além de 20 docentes, sendo 10 com atuação recorrente em orientação. A amostra qualitativa construída para o estudo de caso foi composta por participantes representativos de três grupos de stakeholders, conforme o Quadro 4.1.

Quadro 4.1: Perfil dos stakeholders considerados no estudo de caso

Grupo	Quantidade	Caracterização
Alunos concluintes	12	Estudantes em fase de definição de tema, busca por orientador ou tentativa de ajuste de proposta inicial de TCC
Professores orientadores	6	Docentes com linhas de pesquisa ativas, experiências distintas com orientação e diferentes níveis de disponibilidade
Coordenação do curso	1	Responsável pelo acompanhamento de vagas, aceite de orientações e monitoramento do andamento inicial dos TCCs

Além dos stakeholders centrais, o delineamento do caso considerou fontes documentais de informação, como formulário de matrícula em TCC, planilha de controle de orientações, mensagens recorrentes enviadas por alunos à coordenação e modelos institucionais de distribuição de vagas. A utilização dessas fontes permitiu ampliar a compreensão do problema para além das manifestações individuais dos participantes, reforçando o caráter qualitativo da análise.

Estratégia de aplicação do modelo

A aplicação do modelo no protótipo foi organizada segundo as quatro etapas do *Modelo Convergente DT-SDLC*, cada uma associada a uma necessidade das fases iniciais do SDLC. Na *Imersão*, as técnicas de investigação apoiaram a identificação de atores, dores, restrições e necessidades ainda não formalizadas. Na *Análise*, as técnicas de interpretação converteram dados qualitativos em requisitos, critérios e artefatos rastreáveis. Na *Ideação*, as técnicas de geração e seleção de alternativas apoiaram a concepção da solução. Na *Prototipação*, as representações preliminares permitiram avaliar fluxos e interfaces antes da implementação técnica.

O Quadro 4.2 sintetiza como as técnicas selecionadas em cada etapa do modelo foram utilizadas no caso e qual benefício produziram para as fases iniciais do SDLC. A seleção apresentada prioriza os artefatos necessários ao protótipo, e não a cobertura completa de todos os documentos possíveis de um processo de desenvolvimento.

Quadro 4.2: Aplicação das etapas do Modelo Convergente DT-SDLC no estudo de caso

Etapa do modelo	Técnicas aplicadas no caso	Benefício para o SDLC
Imersão	Reenquadramento, pesquisa desk, entrevistas, observação do fluxo vigente e mapa de stakeholders	Apoio à compreensão do domínio, identificação de atores, levantamento inicial de necessidades e delimitação do problema
Análise	Cartões de insight, diagrama de afinidades, personas, mapa de empatia, jornada do usuário, mapa conceitual e critérios norteadores	Transformação de achados qualitativos em requisitos, critérios de usabilidade, prioridades e artefatos rastreáveis
Ideação	Brainstorming orientado, workshop de cocriação, cardápio de ideias e matriz de posicionamento	Geração, comparação e seleção de alternativas de solução coerentes com os requisitos levantados
Prototipação	Protótipos em papel, storyboards, wireframes e protótipos navegáveis de baixa e média fidelidade	Materialização dos fluxos e interfaces para avaliação conceitual, coleta de feedback e redução de retrabalho antes da implementação

Desse modo, o capítulo aplica precisamente a lógica defendida na definição do modelo: o *Modelo Convergente DT-SDLC* é empregado como instrumento de transição entre investigação qualitativa, requisitos, decisões de projeto e protótipos avaliáveis. Os artefatos do SDLC, como requisitos, histórias de usuário, backlog e fluxos de navegação, aparecem como produtos selecionados do percurso metodológico do modelo, e não como atividades isoladas, substitutas da abordagem de DT ou lista documental obrigatória.

Ameaças à validade

A interpretação dos resultados deste estudo de caso deve considerar algumas ameaças à validade decorrentes do delineamento adotado, do escopo do protótipo e da natureza qualitativa da avaliação. Essas limitações não anulam a contribuição do trabalho, mas delimitam o alcance das conclusões apresentadas e indicam cuidados necessários para futuras aplicações do *Modelo Convergente DT-SDLC*.

Em primeiro lugar, destaca-se a realização de um único estudo de caso. A aplicação foi conduzida no contexto do *TCCMatch*, uma plataforma voltada à mediação entre alunos, professores e coordenação no processo de definição de orientações de TCC. Embora esse caso seja adequado para demonstrar a integração entre *Design Thinking* e SDLC em um problema realista, marcado por múltiplos stakeholders e necessidades pouco formalizadas, os resultados não devem ser generalizados automaticamente para todos os tipos de sistemas, domínios organizacionais ou equipes de desenvolvimento. A contribuição principal, nesse sentido, está na demonstração analítica da aplicabilidade do modelo em um contexto específico, e não na comprovação estatística de sua eficácia universal.

Outra ameaça refere-se à ausência de implementação técnica completa do sistema. O estudo concentrou-se nas fases iniciais do SDLC, abrangendo compreensão do problema, análise de requisitos, ideação, prototipação e validação conceitual. Dessa forma, não foram avaliados aspectos próprios das etapas posteriores do ciclo de vida, como arquitetura executável, codificação, testes funcionais, implantação, desempenho, segurança, manutenção e evolução em ambiente real. Como consequência, as conclusões obtidas dizem respeito à qualidade da estruturação conceitual da solução e à rastreabilidade entre achados, requisitos e protótipos, não ao

comportamento operacional de um sistema implementado.

A validação também apresenta limites por ter caráter predominantemente qualitativo. Os artefatos produzidos foram analisados com base na coerência entre necessidades identificadas, requisitos formulados, decisões de escopo, fluxos de navegação e protótipos. Esse tipo de avaliação é adequado ao objetivo exploratório do trabalho, pois permite compreender percepções, ambiguidades, expectativas e ajustes conceituais antes da implementação. No entanto, por não envolver métricas quantitativas amplas, experimentos controlados ou comparação sistemática com outros modelos, os resultados devem ser interpretados como evidências qualitativas de adequação metodológica, e não como mensuração definitiva de desempenho, produtividade ou redução de retrabalho.

Por fim, considera-se a possibilidade de viés do pesquisador. Como a aplicação do modelo, a seleção dos artefatos, a interpretação dos achados qualitativos e a organização dos requisitos foram conduzidas no âmbito da própria pesquisa, existe o risco de que decisões analíticas tenham sido influenciadas pelas expectativas do autor em relação à utilidade do modelo proposto. Para reduzir essa ameaça, buscou-se explicitar o encadeamento entre evidências, artefatos e decisões de projeto, utilizar diferentes fontes de informação no delineamento do caso e manter a rastreabilidade entre as etapas do modelo. Ainda assim, aplicações futuras com avaliadores externos, maior diversidade de participantes e validação em ambientes reais de desenvolvimento poderiam fortalecer a confiabilidade dos resultados.

Portanto, as ameaças à validade indicam que os resultados deste capítulo devem ser compreendidos como uma demonstração fundamentada da viabilidade conceitual do *Modelo Convergente DT-SDLC*, e não como uma generalização definitiva. O valor do estudo está em mostrar como o modelo pode apoiar a passagem estruturada entre investigação qualitativa, requisitos, ideação e prototipação, preservando a necessidade de novas aplicações, implementação técnica e avaliações empíricas complementares. Ainda assim, as limitações identificadas contribuem para tornar mais transparente o alcance da pesquisa e evitar interpretações excessivamente amplas sobre os resultados obtidos. Dessa forma, a discussão das ameaças à validade reforça o caráter exploratório do estudo e delimita com maior precisão sua contribuição metodológica.

4.3 Aplicação do Modelo Convergente DT-SDLC no Protótipo

A aplicação do *Modelo Convergente DT-SDLC* no protótipo *TCCMatch* foi conduzida como um fluxo seletivo de produção de artefatos. Em cada etapa, os instrumentos de *Design Thinking* foram utilizados para ampliar a compreensão do contexto, enquanto os artefatos do SDLC organizaram os achados em elementos úteis para especificação, priorização e prototipação. Assim, a seção não descreve apenas atividades executadas, mas explicita o que foi gerado, qual resultado foi obtido, por que esse artefato foi suficiente para o protótipo e como cada produto alimentou a etapa seguinte.

O Quadro 4.3 apresenta a visão geral dos artefatos produzidos ou projetados ao longo da construção do protótipo, mantendo a sequência definida pelo modelo convergente e distinguindo os artefatos efetivamente necessários daqueles que ficariam para etapas posteriores do SDLC.

Quadro 4.3: Mapa de artefatos selecionados na aplicação do Modelo Convergente DT-SDLC

Etapa do modelo	Artefatos de DT usados	Artefatos do SDLC selecionados	Justificativa do recorte
Imersão	Reenquadramento, mapa de stakeholders, mapa de dores, registros de observação e síntese de entrevistas	Documento de visão sintético, problema delimitado, atores do sistema, restrições iniciais, necessidades observadas, requisitos iniciais/candidatos e escopo preliminar	Suficientes para delimitar o problema e orientar requisitos iniciais do protótipo
Análise	Cartões de insight, diagrama de afinidades, personas, mapa de empatia, jornada do usuário, mapa conceitual e critérios norteadores	Requisitos funcionais e não funcionais, regras de negócio, critérios de usabilidade, matriz de rastreabilidade e escopo do MVP	Necessários para transformar evidências em decisões rastreáveis
Ideação	Brainstorming orientado, workshop de cocriação, cardápio de ideias e matriz de posicionamento	Alternativas de solução, histórias de usuário, backlog inicial priorizado, módulos do sistema e fluxo principal de navegação	Adequados para definir MVP e preparar a prototipação
Prototipação	Storyboards, protótipos em papel, wireframes, protótipo navegável e registro de feedback	Especificação preliminar de telas, modelo conceitual de dados, critérios de aceitação, roteiro de validação, backlog refinado e ajustes de requisitos	Compatíveis com validação conceitual, sem exigir implementação

Artefatos de maior detalhamento técnico, como arquitetura física, especificação completa de componentes, modelo lógico ou físico de banco de dados, plano de testes funcionais, documentação de implantação e plano de manutenção, foram deliberadamente excluídos do escopo

por não serem necessários à validação conceitual do protótipo. Caso o *TCCMatch* avance para desenvolvimento implementável, esses artefatos deverão ser produzidos em etapas posteriores do SDLC.

Critérios de seleção dos artefatos

A seleção dos artefatos não foi definida apenas pela disponibilidade de técnicas de *Design Thinking* ou pela lista tradicional de documentos do SDLC. O critério principal foi a aderência de cada artefato ao objetivo do estudo de caso: compreender o problema, transformar evidências qualitativas em requisitos, apoiar decisões de escopo e permitir validação conceitual antes da implementação. Assim, foram priorizados artefatos capazes de produzir rastreabilidade entre achados, necessidades, requisitos e protótipos, mantendo baixo custo de elaboração e utilidade direta para a etapa seguinte do modelo.

Também foram considerados quatro critérios complementares. O primeiro foi a capacidade de representar a perspectiva dos stakeholders, evitando que a solução fosse definida apenas pela visão técnica do pesquisador. O segundo foi a capacidade de organizar dados qualitativos em informações acionáveis para requisitos. O terceiro foi a utilidade do artefato para tomada de decisão, especialmente na priorização do MVP. O quarto foi a suficiência para validação conceitual, isto é, a possibilidade de avaliar coerência, fluxo e valor percebido sem exigir codificação ou infraestrutura técnica.

Com base nesses critérios, os principais artefatos foram selecionados pelas seguintes razões:

- **Mapa de stakeholders:** foi utilizado porque o problema envolve alunos, professores e coordenação, com interesses e responsabilidades diferentes. Esse artefato permitiu explicitar quem seria afetado pela solução e quais relações deveriam ser consideradas na definição dos requisitos.
- **Mapa de empatia:** foi selecionado para aprofundar a compreensão das percepções, dores, expectativas e inseguranças dos usuários, especialmente dos alunos em busca de orientação e dos professores que recebem solicitações. Sua função foi transformar aspectos subjetivos da experiência em critérios de usabilidade e necessidades observáveis.

- **Personas:** foram utilizadas para sintetizar perfis representativos de usuários e evitar que os requisitos fossem descritos de forma genérica. Elas apoiaram a definição de funcionalidades coerentes com diferentes objetivos, como busca por orientador, gestão de disponibilidade e acompanhamento pela coordenação.
- **Jornada do usuário:** foi escolhida porque o problema do *TCCMatch* não ocorre em um único ponto de interação, mas em uma sequência de busca, contato, espera, resposta e acompanhamento. A jornada permitiu identificar gargalos, momentos de incerteza e oportunidades de intervenção do sistema ao longo do processo.
- **Cartões de insight e diagrama de afinidades:** foram empregados para agrupar evidências qualitativas recorrentes e reduzir dispersão analítica. Esses artefatos ajudaram a transformar observações, entrevistas e registros do contexto em categorias úteis para requisitos e decisões de escopo.
- **Requisitos, regras de negócio e critérios norteadores:** foram selecionados por constituírem a ponte entre a investigação qualitativa e o SDLC. Eles formalizam o que o sistema deve apoiar, quais restrições precisam ser respeitadas e quais critérios devem orientar as decisões de interface e priorização.
- **Histórias de usuário:** foram escolhidas para traduzir requisitos em enunciados centrados em valor, relacionando ator, necessidade e objetivo. Esse formato facilitou a passagem entre análise e ideação, pois permitiu discutir funcionalidades a partir da perspectiva de quem utilizará o sistema.
- **Backlog priorizado:** foi utilizado para organizar as histórias e funcionalidades conforme relevância para o MVP. Sua inclusão permitiu distinguir elementos essenciais para validação conceitual daqueles que poderiam ser tratados em versões futuras.
- **Fluxos, wireframes e protótipos:** foram selecionados por materializarem decisões de navegação e interface sem exigir implementação. Esses artefatos permitiram avaliar se os

requisitos e histórias de usuário estavam coerentemente representados na experiência proposta.

- **Critérios de aceitação e roteiro de validação:** foram incluídos para orientar a avaliação conceitual do protótipo. Eles ajudaram a verificar se as telas, fluxos e funcionalidades propostas respondiam às necessidades levantadas e se havia ajustes necessários antes de uma implementação futura.

Dessa forma, a seleção dos artefatos buscou equilibrar profundidade qualitativa e objetividade metodológica. Artefatos voltados à empatia e interpretação do contexto foram usados para compreender o problema; artefatos de requisitos e backlog foram usados para formalizar decisões; e artefatos de prototipação foram usados para validar a solução em nível conceitual.

Para deixar explícita a cobertura dos artefatos utilizados, o Quadro 4.4 consolida todos os artefatos do SDLC selecionados para o protótipo, indicando onde aparecem no capítulo e qual função cumprem no fluxo do modelo. Essa síntese complementa o Quadro 4.3, pois detalha os produtos efetivamente gerados sem ampliar o escopo para documentação técnica de implementação.

Quadro 4.4: Inventário dos artefatos do SDLC utilizados no protótipo

Etapa	Artefato selecionado	Representação no capítulo	Função no ciclo de vida
Imersão	Documento de visão e escopo preliminar	Quadro 4.6	Delimitar problema, objetivo e fronteiras do protótipo
Imersão	Atores do sistema e perfis de acesso	Quadro 4.7	Definir quem interage com a solução e quais responsabilidades serão prototipadas
Imersão	Necessidades e restrições iniciais	Quadro 4.9	Converter achados qualitativos em condicionantes para requisitos
Imersão	Requisitos iniciais/candidatos	Quadro 4.10	Registrar hipóteses de requisitos que serão refinadas na análise
Análise	Requisitos funcionais e não funcionais	Quadro 4.14	Formalizar capacidades esperadas e critérios de qualidade conceitual

Etapa	Artefato selecionado	Representação no capítulo	Função no ciclo de vida
Análise	Regras de negócio e critérios norteadores	Quadro 4.16	Orientar limites de funcionamento, priorização e decisões de interface
Análise	Modelo de domínio	Quadro 4.15 e Figura 4.8	Representar conceitos centrais, relações e informações do domínio acadêmico
Análise	Matriz de rastreabilidade	Quadro 4.13	Relacionar evidências, requisitos e artefatos derivados
Ideação	Alternativas de solução e priorização	Quadros 4.17 e 4.18	Selecionar funcionalidades aderentes ao problema e ao MVP
Ideação	Histórias de usuário e backlog inicial	Quadros 4.19 e 4.22	Transformar requisitos em unidades de trabalho conceituais
Ideação	Diagrama e especificação de casos de uso	Quadros 4.20 e 4.21; Figura 4.10	Relacionar atores, objetivos de interação e cenários principais do sistema
Ideação	Arquitetura funcional e mapa de navegação	Quadro 4.23, Figura 4.11, Quadro 4.24 e Figura 4.13	Organizar responsabilidades, módulos e percursos antes das telas
Ideação	Diagrama de atividades	Quadro 4.25 e Figura 4.14	Descrever o fluxo dinâmico entre busca, solicitação, decisão docente e acompanhamento
Prototipação	Especificação preliminar de telas	Quadro 4.26	Vincular interfaces a objetivos de uso e requisitos
Prototipação	Modelo conceitual de dados	Quadro 4.27	Identificar entidades e informações necessárias sem avançar para banco físico
Prototipação	Critérios de aceitação, roteiro de validação e feedback	Quadros 4.28, 4.29 e 4.30	Avaliar aderência conceitual antes de implementação
Prototipação	Backlog refinado e ajustes de requisitos	Quadro 4.31	Registrar retroalimentação do protótipo para os artefatos anteriores

Essa visão de inventário funciona como síntese de rastreabilidade dos artefatos selecionados no protótipo, conectando evidências da imersão, requisitos, backlog, módulos, telas e feedback de validação. Com isso, o capítulo explicita que a aplicação do modelo não produziu documentos isolados, mas uma sequência de artefatos encadeados, suficiente para sustentar a validação conceitual proposta.

Imersão

A primeira etapa teve como finalidade compreender como ocorre, na prática, a aproximação entre aluno e orientador, quais são os principais pontos de atrito do processo atual e quais necessidades emergem antes da especificação formal do sistema. O resultado esperado dessa fase, de acordo com o modelo, não era ainda definir telas ou funcionalidades finais, mas produzir uma base qualitativa suficiente para sustentar a Engenharia de Requisitos e explicitar requisitos iniciais ainda sujeitos a análise.

O primeiro artefato produzido foi o *reenquadramento do problema*. A formulação inicial “como criar uma plataforma para escolher orientadores” foi substituída por uma formulação mais aderente ao contexto observado: “como reduzir assimetria de informação, insegurança e falta de rastreabilidade no processo de formação das orientações de TCC”. Esse reenquadramento alterou o foco da solução: em vez de tratar o sistema apenas como mecanismo de escolha, o protótipo passou a tratá-lo como instrumento de mediação, transparência e acompanhamento.

A Figura 4.1 apresenta o reenquadramento de foco realizado na etapa de imersão, evidenciando a passagem de uma formulação centrada apenas na criação de uma plataforma para uma formulação orientada à redução da assimetria de informação, da insegurança e da falta de rastreabilidade no processo de orientação.

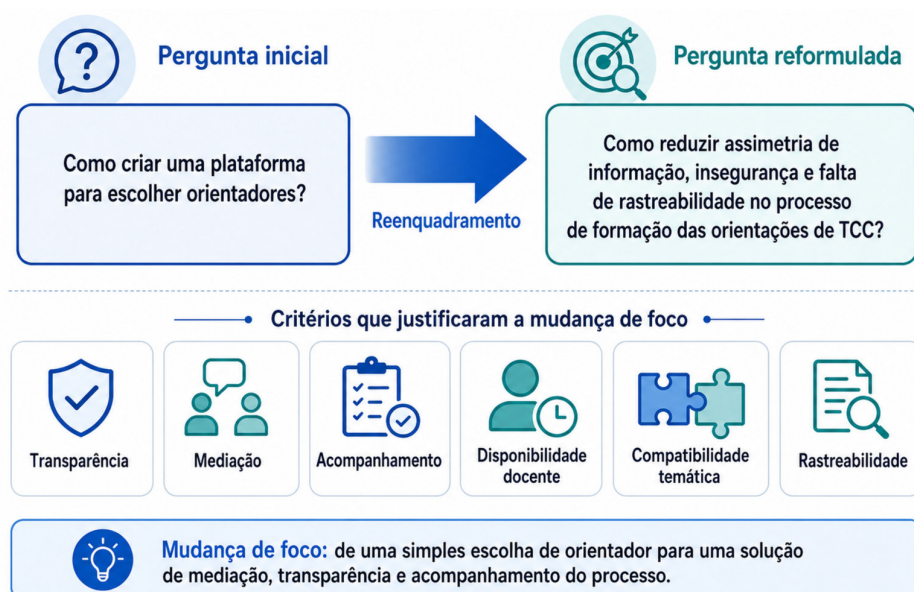


Figura 4.1: Reenquadramento de foco do protótipo *TCCMatch*. **Fonte:** Elaborado pelo autor com apoio do ChatGPT (2026).

A pesquisa desk foi utilizada para examinar documentos acadêmicos e registros administrativos associados ao processo de orientação, como formulários de matrícula em TCC, planilhas de controle de orientações, mensagens recorrentes enviadas à coordenação e modelos institucionais de distribuição de vagas. Esse levantamento gerou um inventário inicial de informações necessárias ao sistema, incluindo dados de aluno, perfil docente, linha de pesquisa, disponibilidade, situação da solicitação e status da orientação.

As entrevistas e observações do fluxo vigente complementaram a pesquisa desk. Entre os alunos, os relatos evidenciaram insegurança no momento de escolher um orientador, desconhecimento sobre áreas de atuação dos professores e dificuldade de acompanhar respostas. Entre os professores, emergiu a percepção de sobrecarga informacional, especialmente pelo recebimento de pedidos desalinhados com suas linhas de pesquisa. Do ponto de vista da coordenação, o principal problema identificado foi a baixa visibilidade sobre alunos ainda sem orientação e docentes próximos do limite de vagas.

O Quadro 4.5 apresenta os principais achados qualitativos estruturados nesta etapa e suas implicações iniciais para o sistema.

Quadro 4.5: Achados qualitativos consolidados na Imersão

Stakeholder	Achado predominante	Implicação para o sistema
Alunos	Insegurança quanto a tema, professor adequado e disponibilidade de vagas	Necessidade de transparência sobre perfis docentes, áreas e status de orientação
Alunos	Dificuldade de iniciar contato e acompanhar retorno de docentes	Necessidade de canal estruturado de manifestação de interesse e acompanhamento
Professores	Recebimento de pedidos desalinhados com a linha de pesquisa	Necessidade de filtragem temática e melhor qualificação da solicitação
Professores	Baixa visibilidade sobre o perfil acadêmico do aluno antes do aceite	Necessidade de mini perfil do estudante e resumo da proposta
Coordenação	Ausência de visão consolidada do processo de formação das duplas	Necessidade de painel de monitoramento e acompanhamento de vagas

Com base nesses achados, foi elaborado um documento de visão e escopo preliminar para delimitar a finalidade do protótipo antes da definição detalhada dos requisitos. Esse artefato sintetiza o problema, o objetivo da solução, os limites do protótipo e as premissas adotadas no estudo de caso. O Quadro 4.6 apresenta essa consolidação.

Quadro 4.6: Documento de visão e escopo preliminar do protótipo *TCCMatch*

Elemento	Definição para o protótipo	Decisão de escopo
Problema central	Alunos, professores e coordenação lidam com informações dispersas sobre temas, vagas, disponibilidade e status das solicitações de orientação.	Delimitar a solução como mediação e acompanhamento do início da orientação, não como gestão completa do TCC.
Objetivo da solução	Apoiar a aproximação entre alunos e orientadores por meio de perfis, busca temática, manifestação de interesse e acompanhamento do status da solicitação.	Priorizar funcionalidades de descoberta, triagem e visibilidade do processo.
Público-alvo	Alunos concluintes, professores orientadores e coordenação do curso.	Restringir o protótipo aos três grupos diretamente envolvidos na formação das duplas de orientação.
Valor esperado	Reduzir assimetria de informação, insegurança no contato inicial, pedidos desalinhados e baixa visibilidade gerencial.	Validar se o fluxo conceitual melhora clareza, previsibilidade e rastreabilidade.
Escopo incluído	Cadastro de perfis, linhas de pesquisa, disponibilidade, busca por orientadores, solicitação de orientação e painel de acompanhamento.	Representar esses itens em requisitos, fluxos, wireframes e protótipo navegável.
Escopo excluído	Implementação técnica, autenticação real, integração com sistemas acadêmicos, gestão de documentos, avaliação de bancas e acompanhamento completo do TCC.	Manter o protótipo concentrada na validação conceitual das fases iniciais.
Premissas	Professores mantêm disponibilidade atualizada, alunos informam tema ou área de interesse e a coordenação acompanha exceções do processo.	Registrar dependências operacionais que precisam ser avaliadas em evolução futura.

O documento de visão permitiu transformar os achados da imersão em uma fronteira de solução mais objetiva. Em seguida, os atores do sistema foram definidos para indicar quais perfis de acesso deveriam ser contemplados na prototipação e quais responsabilidades ficariam fora do recorte inicial. O Quadro 4.7 apresenta essa definição.

Quadro 4.7: Atores do sistema e perfis de acesso considerados no protótipo

Ator	Necessidade de interação	Responsabilidades no sistema	Visão prototipada
Aluno concluinte	Encontrar orientadores compatíveis, compreender disponibilidade e acompanhar retorno da solicitação.	Informar área ou tema de interesse, consultar perfis docentes e registrar manifestação de interesse.	Busca, detalhes da oportunidade e acompanhamento da solicitação.
Professor orientador	Divulgar linhas de pesquisa, controlar vagas e avaliar solicitações recebidas.	Manter perfil docente, indicar disponibilidade e aceitar, recusar ou solicitar ajuste da proposta.	Triagem conceitual de solicitações e atualização de disponibilidade.

Ator	Necessidade de interação	Responsabilidades no sistema	Visão prototipada
Coordenação do curso	Observar o andamento da formação das duplas e identificar gargalos.	Monitorar vagas, alunos sem orientação, solicitações pendentes e necessidade de intervenção.	Painel gerencial de acompanhamento.
Administrador acadêmico	Apoiar manutenção cadastral e parametrizações institucionais mínimas.	Gerenciar cadastros básicos, períodos de solicitação e permissões de acesso.	Fora do protótipo navegável; previsto apenas como dependência futura.

O mapa de stakeholders foi gerado para relacionar atores, interesses e contribuições esperadas dentro da solução. Como artefato de DT, ele permitiu visualizar os grupos afetados pelo problema; como artefato do SDLC, ele apoiou a identificação de atores do sistema e a definição inicial de permissões e visões de interface. O Quadro 4.8 apresenta sua consolidação textual.

Quadro 4.8: Mapa textual de stakeholders do sistema *TCCMatch*

Stakeholder	Papel no processo	Interesses centrais	Contribuição para o sistema
Alunos concluintes	Buscam tema, orientador e encaminhamento do TCC	Encontrar professor aderente, conhecer vagas e obter retorno rápido	Alimentam o sistema com tema, área de interesse e manifestação de interesse
Professores orientadores	Avaliam solicitações e orientam os trabalhos	Receber propostas compatíveis, controlar vagas e divulgar linhas de pesquisa	Cadastram disponibilidade, linhas temáticas e respondem às solicitações
Coordenação do curso	Acompanha e regula o fluxo inicial das orientações	Visualizar pares formados, vagas remanescentes e alunos sem orientação	Monitora o processo, acompanha indicadores e intervém quando necessário

A Figura 4.2 apresenta a versão visual do mapa de stakeholders, posicionando alunos, professores e coordenação em relação ao processo central de formação das orientações.

A etapa também gerou um conjunto inicial de necessidades e restrições de domínio. Esses elementos funcionam como ponte entre a investigação qualitativa e a especificação do sistema, pois indicam o que precisa ser considerado antes de definir requisitos formais. O Quadro 4.9 apresenta essa consolidação.



Figura 4.2: Mapa visual de stakeholders do protótipo *TCCMatch*. **Fonte:** Elaborado pelo autor com apoio do ChatGPT (2026).

Quadro 4.9: Necessidades e restrições iniciais derivadas da imersão

Elemento identificado	Descrição	Impacto no SDLC
Assimetria de informação	Alunos não conhecem claramente áreas, vagas e temas aceitos por docentes	Exige cadastro docente estruturado e busca por critérios temáticos
Comunicação dispersa	Contatos ocorrem por canais externos e sem histórico centralizado	Exige fluxo de manifestação de interesse e registro de status
Capacidade limitada de orientação	Professores possuem número restrito de vagas e disponibilidade variável	Exige controle de vagas e atualização de disponibilidade
Governança acadêmica	Coordenação precisa acompanhar alunos sem retorno ou sem orientador	Exige painel de acompanhamento e indicadores mínimos
Qualificação da solicitação	Professores precisam avaliar tema, perfil e maturidade da proposta	Exige mini perfil do aluno e resumo da intenção de pesquisa

Com base nas necessidades e restrições identificadas, a imersão também produziu um primeiro conjunto de requisitos iniciais, tratados como candidatos à especificação. Esses requisitos

não encerram a análise, mas tornam explícita a passagem entre achados qualitativos e requisitos formais, permitindo que a etapa seguinte organize, refine, agrupe e priorize tais hipóteses. O Quadro 4.10 apresenta essa saída da Imersão.

Quadro 4.10: Requisitos iniciais/candidatos derivados da imersão

Código	Requisito candidato	Origem na imersão	Refino
RI01	O sistema deve permitir cadastro de perfis de aluno, professor e coordenação.	Mapa de stakeholders, atores do sistema e necessidade de diferenciar responsabilidades.	Refinar como RF01
RI02	O sistema deve permitir que professores divulguem linhas de pesquisa, temas de interesse e disponibilidade de vagas.	Assimetria de informação e capacidade limitada de orientação.	Refinar como RF02 e RF03
RI03	O sistema deve permitir busca de orientadores por área, palavra-chave e disponibilidade.	Insegurança dos alunos e dificuldade de identificar docentes compatíveis.	Refinar como RF03
RI04	O sistema deve permitir manifestação de interesse com resumo do tema ou intenção de pesquisa do aluno.	Comunicação dispersa e necessidade de qualificar a solicitação.	Refinar como RF04
RI05	O sistema deve tornar visível o status da solicitação e a resposta do professor.	Dificuldade de acompanhar retorno e demanda por transparência do processo.	Refinar como RF04 e RNF02
RI06	O sistema deve oferecer à coordenação uma visão consolidada de vagas, solicitações pendentes e alunos sem orientação.	Baixa governança acadêmica e ausência de acompanhamento centralizado.	Refinar como RF05
RI07	O sistema deve apresentar informações de forma clara, objetiva e rastreável para reduzir ambiguidade no primeiro contato.	Critérios de transparência, insegurança dos alunos e qualificação da decisão docente.	Refinar como RNF01, RNF02 e RNF03

Como resultado da imersão, foram gerados os seguintes artefatos do SDLC: problema central delimitado, atores do sistema, lista inicial de necessidades, restrições operacionais, escopo preliminar e requisitos iniciais/candidatos. Esses produtos alimentaram diretamente a etapa seguinte, na qual os achados deixaram de ser apenas registros qualitativos e passaram a ser organizados como insumos de especificação.

Análise

A segunda etapa teve como finalidade transformar dados dispersos em interpretações estruturadas. No modelo convergente, essa etapa é decisiva porque cria rastreabilidade entre observações de campo, necessidades dos stakeholders, requisitos de software e decisões de projeto.

Os cartões de insight foram utilizados para registrar recorrências observadas na imersão. Cada cartão representou uma evidência relevante, como insegurança dos alunos, desalinhamento dos pedidos recebidos por professores, ausência de visibilidade para a coordenação e dificuldade de acompanhar respostas. Em seguida, o diagrama de afinidades agrupou esses cartões em quatro eixos interpretativos: *assimetria de informação*, *comunicação dispersa*, *desalinhamento entre perfil e tema* e *baixa governança do processo*.

A Figura 4.3 apresenta o diagrama de afinidades produzido na etapa de análise, mostrando os cartões de insight agrupados nos quatro eixos interpretativos utilizados para derivar requisitos.

Esse agrupamento também serviu para evitar um comportamento colateral comum em projetos iniciados diretamente pela solução: transformar sintomas isolados em funcionalidades independentes. Ao reunir os achados por afinidade, o protótipo deixou de tratar cada reclamação como uma demanda autônoma e passou a identificar padrões de causa. Dessa forma, pedidos como “ver professores”, “mandar mensagem” ou “acompanhar resposta” foram reinterpretados como partes de um problema maior de visibilidade, mediação e rastreabilidade.



Figura 4.3: Diagrama de afinidades do protótipo TCCMatch. **Fonte:** Elaborado pelo autor com apoio do ChatGPT (2026).

A partir desses agrupamentos, foi produzido um mapa conceitual do processo, relacionando atores, informações, solicitações, vagas e decisões. Esse artefato não substitui a especificação, mas ajuda a explicitar dependências do domínio antes da modelagem do sistema. No caso do *TCCMatch*, o mapa conceitual mostrou que a solicitação de orientação depende simultaneamente de tema, área de interesse, disponibilidade docente, qualidade da proposta e acompanhamento institucional.

A Figura 4.4 apresenta o mapa conceitual do processo de orientação, conectando aluno, professor, coordenação, tema, vaga, solicitação e status.

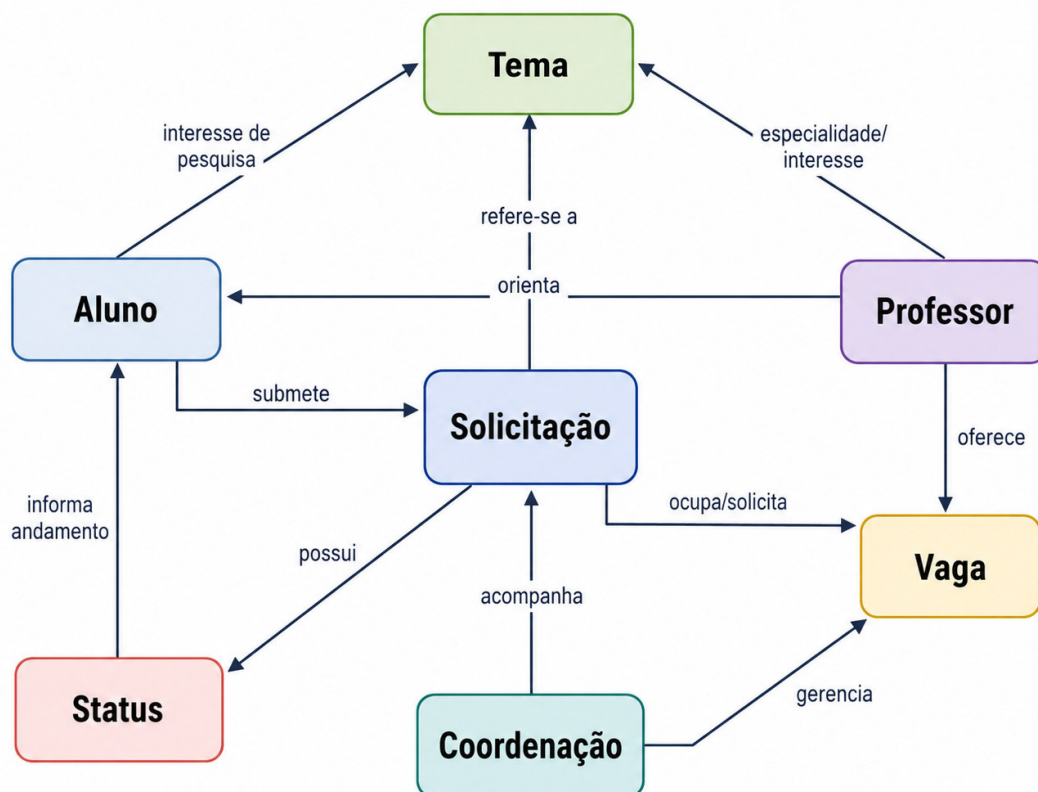


Figura 4.4: Mapa conceitual do processo de orientação do protótipo *TCCMatch*. **Fonte:** Elaborado pelo autor com apoio do ChatGPT (2026).

As personas foram geradas para representar perfis recorrentes de uso. Como artefato de DT, elas sintetizam objetivos, dores e necessidades; como artefato do SDLC, ajudam a orientar requisitos, prioridades e decisões de interface. O Quadro 4.11 apresenta as personas analíticas do protótipo.

Quadro 4.11: Personas analíticas elaboradas para o sistema *TCCMatch*

Persona	Objetivo principal	Dores predominantes	Necessidades para o sistema
Aluno concluinte indeciso	Encontrar tema e orientador aderentes	Falta de informação, insegurança na abordagem e demora de resposta	Filtros temáticos, clareza sobre vagas e fluxo de manifestação de interesse
Professor com linha definida	Receber propostas coerentes com sua área	Solicitações desalinhadas, excesso de mensagens e pouca triagem	Perfil do aluno, resumo do tema e controle de disponibilidade
Coordenação gestora do fluxo	Acompanhar o processo de formação das orientações	Falta de visibilidade, informações dispersas e dificuldade de monitoramento	Painel consolidado, status das solicitações e visão geral de vagas

A Figura 4.5 apresenta a versão visual das personas do protótipo, organizada em cartões para aluno, professor e coordenação.

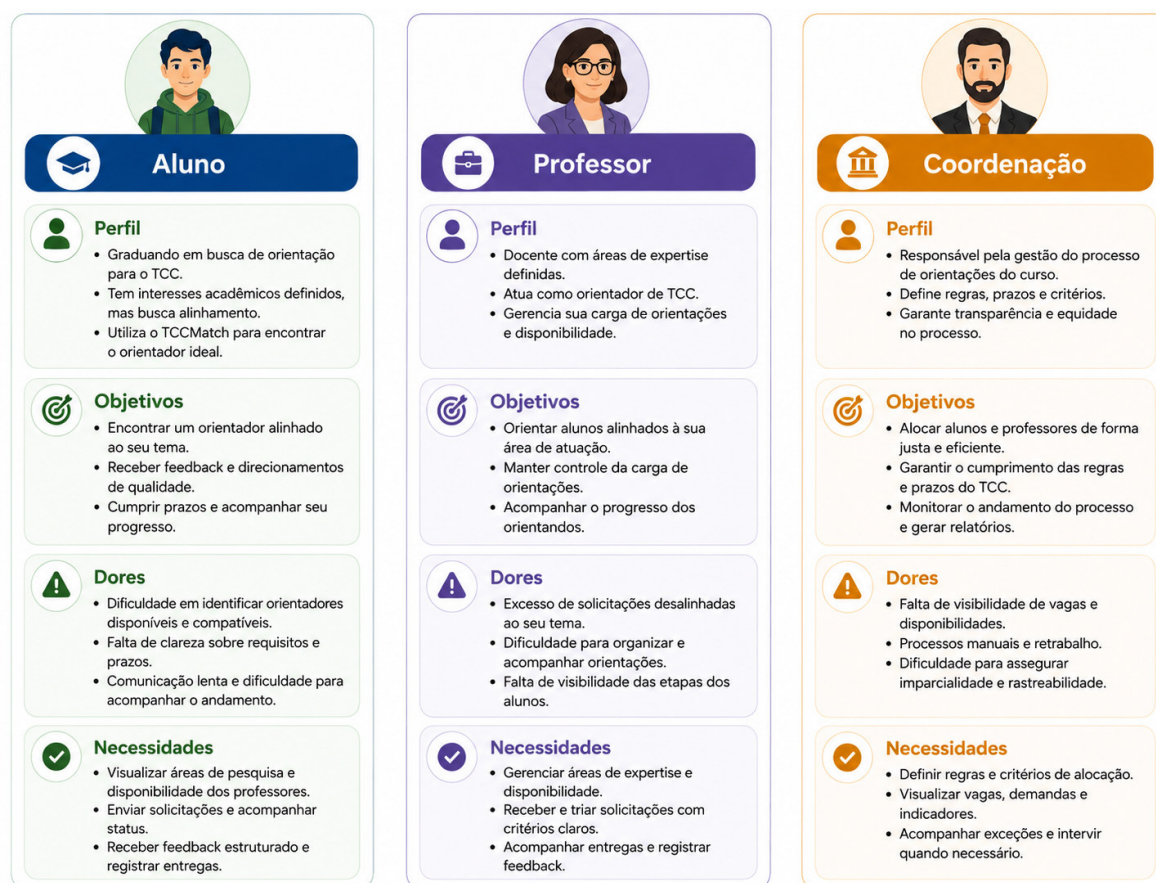


Figura 4.5: Esquema visual das personas do protótipo *TCCMatch*. **Fonte:** Elaborado pelo autor com apoio do ChatGPT (2026).

Também foi projetado o mapa de empatia da persona principal, o aluno concluinte indeciso. Esse artefato permitiu interpretar aspectos subjetivos do problema, como insegurança

ao abordar professores, receio de receber recusa, dificuldade de comparar linhas de pesquisa e expectativa por respostas claras. O resultado obtido foi a definição de critérios de interface voltados à redução de ambiguidade: linguagem objetiva, indicação de disponibilidade, status visível da solicitação e organização das opções por área ou palavra-chave.

A Figura 4.6 apresenta o mapa de empatia elaborado para representar percepções, dores e ganhos esperados dessa persona.



Figura 4.6: Mapa de empatia do aluno concluinte indeciso. **Fonte:** Elaborado pelo autor com apoio do ChatGPT (2026).

A jornada atual do aluno foi estruturada para transformar uma experiência originalmente informal em etapas observáveis do processo de busca por orientação. A organização do percurso permitiu explicitar dores recorrentes, como a incerteza na escolha do tema, a dificuldade de identificar professores compatíveis, o constrangimento no contato inicial e a ausência de previsibilidade durante a espera por resposta. Do ponto de vista do SDLC, esse artefato funcionou como uma ponte entre a compreensão do problema e a definição de requisitos, pois indicou

quais fluxos deveriam existir na aplicação, quais informações precisariam estar visíveis e quais mecanismos de acompanhamento seriam necessários. Assim, a jornada contribuiu para que o protótipo não se limitasse a um catálogo de docentes, mas incorporasse recursos voltados à mediação do processo de orientação. O Quadro 4.12 resume o percurso atual e as oportunidades de projeto.

Quadro 4.12: Jornada atual do aluno no processo de busca por orientação

Etapa	Ação do aluno	Ponto de dor	Oportunidade de projeto
Definição inicial	Escolhe ou esboça um tema de interesse	Incerteza sobre aderência acadêmica do tema	Apoio por áreas e palavras-chave
Busca por orientador	Procura professores por indicação ou tentativa direta	Falta de informação sobre linhas e disponibilidade	Perfil docente com linhas e vagas
Contato inicial	Envia mensagem ou aborda o professor	Constrangimento e ausência de retorno padronizado	Fluxo mediado de manifestação de interesse
Espera por resposta	Aguarda aceite, recusa ou silêncio	Falta de previsibilidade do processo	Status visível da solicitação
Redefinição	Procura nova alternativa após recusa	Retrabalho e perda de tempo	Sugestão de orientadores compatíveis

Como complemento visual à jornada descrita no Quadro 4.12, a Figura 4.7 destaca a concentração dos principais pontos de dor ao longo do percurso do aluno, evidenciando os momentos em que a incerteza, a falta de informação e a ausência de retorno tendem a comprometer a continuidade da busca por orientação.



Figura 4.7: Linha de dor da jornada do aluno no processo de busca por orientação. **Fonte:** Elaborado pelo autor com apoio do ChatGPT (2026).

A matriz de rastreabilidade foi então consolidada para ligar insight, requisito e artefato de software. Esse quadro mostra como o modelo evita que funcionalidades surjam sem relação com evidências do problema. O Quadro 4.13 apresenta essa relação.

Quadro 4.13: Rastreabilidade entre insights qualitativos e requisitos do sistema

Insight qualitativo	Requisito derivado	Artefato associado
Alunos não sabem quem orienta determinados temas	O sistema deve exibir linhas de pesquisa, áreas e disponibilidade do professor	Perfil docente, feed de linhas e filtros de busca
Professores recebem pedidos pouco aderentes ao seu foco	O sistema deve permitir manifestação de interesse a partir de palavras-chave e afinidade temática	Mecanismo de busca e formulário resumido de proposta
Coordenação não acompanha o status das orientações em tempo hábil	O sistema deve consolidar solicitações, aceite, recusa e vagas disponíveis	Painel gerencial da coordenação
Alunos sentem insegurança ao abordar professores diretamente	O sistema deve estruturar o primeiro contato de forma mediada e rastreável	Fluxo de solicitação de orientação
Professores querem conhecer minimamente o aluno antes do aceite	O sistema deve apresentar mini perfil acadêmico e tema pretendido	Cartão de estudante e tela de decisão do professor

A conversão dos insights em requisitos foi consolidada no Quadro 4.14. Nesse quadro, os requisitos funcionais representam capacidades esperadas do sistema, enquanto os requisitos não funcionais indicam critérios de qualidade e experiência que devem orientar a prototipação.

Quadro 4.14: Requisitos consolidados do *TCCMatch* após a organização dos achados

Tipo	Descrição	Origem principal
RF01	Permitir cadastro de perfis por tipo de usuário (aluno, professor e coordenação)	Mapeamento de stakeholders
RF02	Permitir publicação de ideias de TCC e linhas de pesquisa	Assimetria de informação
RF03	Permitir busca por área, palavra-chave e disponibilidade	Dificuldade de identificação de orientador
RF04	Permitir manifestação de interesse e resposta do professor	Comunicação dispersa
RF05	Disponibilizar painel de acompanhamento para a coordenação	Baixa governança do processo
RNF01	Exibir informações de forma clara e navegável	Critérios de usabilidade
RNF02	Tornar visível o status de vagas e solicitações	Transparência do processo
RNF03	Reduzir ambiguidade no primeiro contato entre aluno e professor	Qualificação da decisão

Além da lista de requisitos, foi elaborado um modelo de domínio para representar os con-

ceitos centrais envolvidos no processo de formação das orientações. Esse artefato é importante porque evita que a solução seja descrita apenas como telas ou funcionalidades, explicitando entidades do contexto acadêmico, relações entre elas e informações necessárias para sustentar requisitos, casos de uso e protótipos. O Quadro 4.15 apresenta o modelo de domínio textual do protótipo.

Na geração desse artefato, optou-se por manter o nível conceitual do domínio, sem antecipar tabelas físicas, chaves ou decisões de persistência. Essa escolha evitou que o protótipo se deslocasse prematuramente para uma discussão de banco de dados, preservando o foco do capítulo na compreensão do problema e na validação conceitual da solução. Assim, conceitos como solicitação, vaga, oportunidade e acompanhamento foram tratados primeiro como elementos do processo acadêmico, e não como estruturas técnicas de implementação.

Quadro 4.15: Modelo de domínio textual do protótipo *TCCMatch*

Conceito do domínio	Descrição	Relações principais	Requisitos associados
Aluno concluinte	Estudante em fase de definição de tema e busca por orientação.	Submete solicitação; acompanha status; informa área de interesse e tema pretendido.	RF01, RF03, RF04, RNF02
Professor orientador	Docente que divulga linhas, disponibilidade e avalia pedidos de orientação.	Publica oportunidade; recebe solicitação; aceita, recusa ou solicita ajuste.	RF01, RF02, RF04, RNF03
Coordenação	Ator institucional responsável por acompanhar vagas, pendências e formação das duplas.	Visualiza solicitações; monitora vagas; identifica alunos sem orientação.	RF05, RNF02
Linha de pesquisa	Área ou tema de atuação do professor que orienta compatibilidade entre aluno e docente.	Classifica oportunidades; alimenta filtros; orienta manifestação de interesse.	RF02, RF03
Oportunidade de orientação	Oferta ou ideia de TCC vinculada a professor, área, palavras-chave e vagas.	É consultada pelo aluno; pode originar solicitação; depende de disponibilidade.	RF02, RF03, RF04
Solicitação de orientação	Registro do interesse do aluno em ser orientado por determinado professor.	Relaciona aluno, professor, tema, status e decisão docente.	RF04, RNF02, RNF03
Vaga de orientação	Capacidade disponível do professor para receber novos orientandos.	Condiciona disponibilidade; aparece na busca e no painel da coordenação.	RF03, RF05, RNF02

A Figura 4.8 complementa o modelo textual ao representar visualmente os conceitos do

domínio e suas relações principais. A imagem evidencia Aluno, Professor, Coordenação, Linha de pesquisa, Oportunidade de orientação, Solicitação de orientação e Vaga de orientação, permitindo observar como esses elementos sustentam os requisitos associados.

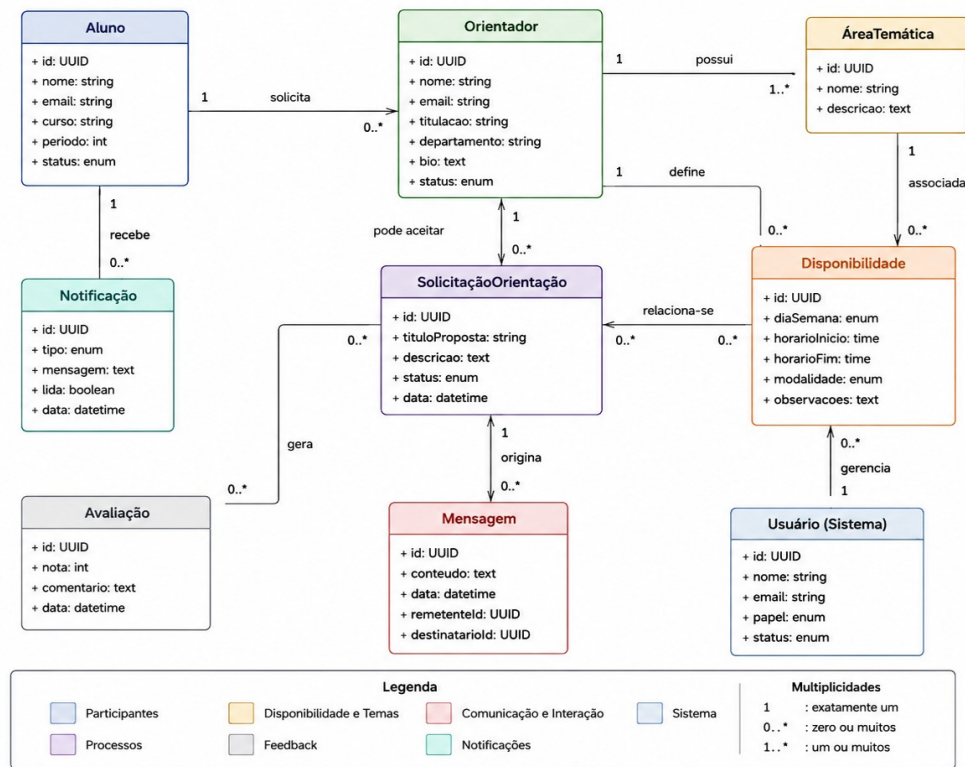


Figura 4.8: Modelo de domínio visual do protótipo *TCCMatch*. **Fonte:** Elaborado pelo autor com apoio do ChatGPT (2026).

Além dos requisitos, foram definidos critérios norteadores e regras preliminares de negócio. Esses elementos ajudam a impedir que a solução avance apenas como protótipo visual, pois orientam decisões de comportamento do sistema e limites de funcionamento. O Quadro 4.16 apresenta essa consolidação.

Ao final da etapa, o resultado obtido foi um conjunto organizado de insumos para ideação: eixos de problema, personas, jornada, critérios norteadores, regras preliminares, requisitos funcionais, requisitos não funcionais e matriz de rastreabilidade. Esses artefatos reduziram a distância entre a compreensão qualitativa do problema e a definição de alternativas de solução.

Quadro 4.16: Critérios norteadores e regras preliminares para o protótipo

Elemento	Aplicação no protótipo	Artefato do SDLC associado
Transparência	Exibir áreas, disponibilidade e status de solicitações	RNF02, telas de busca e perfil
Rastreabilidade	Registrar origem de cada requisito e decisão de design	Matriz de rastreabilidade
Autonomia assistida	Permitir busca direta sem eliminar acompanhamento da coordenação	Fluxo do aluno e painel gerencial
Qualificação do contato	Exigir resumo de tema e área antes da manifestação de interesse	Formulário de solicitação
Controle de capacidade	Considerar vagas disponíveis antes de novas solicitações	Regra de negócio de disponibilidade

Ideação

A terceira etapa teve como função gerar alternativas de solução sem romper a rastreabilidade com os requisitos. Por isso, a ideação não foi tratada como atividade livre e desconectada, mas como momento de explorar soluções possíveis a partir dos achados já organizados.

O brainstorming orientado partiu dos quatro eixos de problema identificados na análise: assimetria de informação, comunicação dispersa, desalinhamento entre perfil e tema e baixa governança do processo. O workshop de cocriação foi projetado para confrontar as perspectivas de alunos, professores e coordenação, evitando que a solução favorecesse apenas um grupo. Como resultado, as ideias geradas foram agrupadas em funcionalidades candidatas para o MVP.

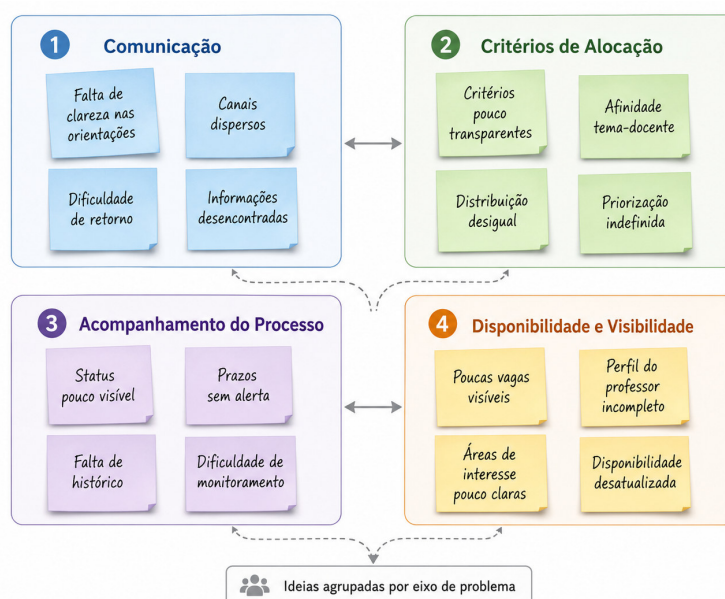


Figura 4.9: Quadro de brainstorming do protótipo *TCCMatch*. **Fonte:** Elaborado pelo autor com apoio do ChatGPT (2026).

O Quadro 4.17 apresenta o cardápio de ideias consolidado. Esse artefato de DT foi usado como ponte para o SDLC porque transformou ideias em candidatas a funcionalidades, indicando quais deveriam avançar para backlog e prototipação.

Quadro 4.17: Cardápio de ideias gerado na etapa de Ideação

Ideia	Problema que atende	Encaminhamento
Feed de ideias e linhas de pesquisa	Falta de visibilidade sobre temas e docentes disponíveis	Incorporar ao MVP como área de descoberta
Busca temática por filtros	Dificuldade de encontrar professor aderente	Incorporar ao MVP como fluxo principal do aluno
Manifestação de interesse mediada	Comunicação dispersa e insegurança no primeiro contato	Incorporar ao MVP como mecanismo de solicitação
Painel da coordenação	Baixa governança do processo de orientação	Incorporar ao MVP como visão de acompanhamento
Chat interno completo	Troca detalhada de mensagens entre aluno e professor	Postergar para iteração futura
Centralização total pela coordenação	Controle administrativo das solicitações	Não priorizar, pois reduz autonomia dos usuários

A matriz de posicionamento foi usada para priorizar funcionalidades segundo impacto, esforço e aderência aos requisitos. O resultado obtido foi a definição de um MVP composto por cadastro de perfis, publicação de linhas ou ideias de TCC, busca temática, manifestação de interesse e painel de acompanhamento. O Quadro 4.18 apresenta essa priorização.

Quadro 4.18: Matriz de posicionamento das funcionalidades propostas

Funcionalidade	Impacto	Esforço	Decisão
Cadastro de perfis e linhas de pesquisa	Alto	Médio	Incluir no MVP
Busca por área e palavra-chave	Alto	Médio	Incluir no MVP
Manifestação de interesse com status	Alto	Médio	Incluir no MVP
Painel consolidado da coordenação	Alto	Médio	Incluir no MVP
Chat interno entre aluno e professor	Médio	Alto	Postergar
Sugestão automática avançada por similaridade	Médio	Alto	Postergar

As funcionalidades priorizadas foram traduzidas em histórias de usuário, que funcionam como artefatos do SDLC derivados da ideação. O Quadro 4.19 apresenta as histórias iniciais do backlog.

A partir das histórias de usuário, também foi derivado o diagrama de casos de uso em

Quadro 4.19: Histórias de usuário priorizadas no backlog inicial do *TCCMatch*

Código	História de usuário	Prioridade
HU01	Como aluno, quero filtrar professores por área e palavra-chave para identificar orientadores compatíveis com meu tema	Alta
HU02	Como aluno, quero manifestar interesse por orientação sem recorrer a canais externos	Alta
HU03	Como professor, quero visualizar resumo do aluno e do tema antes de aceitar uma solicitação	Alta
HU04	Como professor, quero informar linhas de pesquisa e disponibilidade de vagas	Alta
HU05	Como coordenação, quero acompanhar em painel único o status das orientações	Alta
HU06	Como aluno, quero acompanhar o status do meu pedido de orientação	Média

formato textual. Como o objetivo do protótipo não é produzir documentação UML completa, o artefato foi registrado como quadro estruturado de atores, casos de uso e relações principais, suficiente para demonstrar quais interações precisam existir no sistema e como elas se conectam aos requisitos. O Quadro 4.20 apresenta essa visão.

Quadro 4.20: Diagrama textual de casos de uso do protótipo *TCCMatch*

Ator	Caso de uso	Relações	Requisitos
Aluno concluinte	Consultar oportunidades e professores por área, palavra-chave e disponibilidade.	Inclui visualizar detalhes da oportunidade.	RF02, RF03, RNF01
Aluno concluinte	Manifestar interesse em orientação.	Inclui preencher resumo do tema; depende de oportunidade disponível.	RF04, RNF03
Aluno concluinte	Acompanhar status da solicitação.	Estende manifestação de interesse após envio do pedido.	RF04, RNF02
Professor orientador	Manter perfil, linhas de pesquisa e disponibilidade.	Alimenta busca, feed e controle de vagas.	RF01, RF02, RNF02
Professor orientador	Avaliar solicitação recebida.	Inclui visualizar mini perfil do aluno; pode aceitar, recusar ou solicitar ajuste.	RF04, RNF03
Coordenação	Monitorar vagas, solicitações e alunos sem orientação.	Consolida informações de alunos, professores e solicitações.	RF05, RNF02

A Figura 4.10 complementa o quadro textual ao sintetizar graficamente os atores e as interações centrais do sistema. Essa visualização facilita a leitura das dependências entre Aluno concluinte, Professor orientador, Coordenação e os casos de uso principais UC01 a UC04.

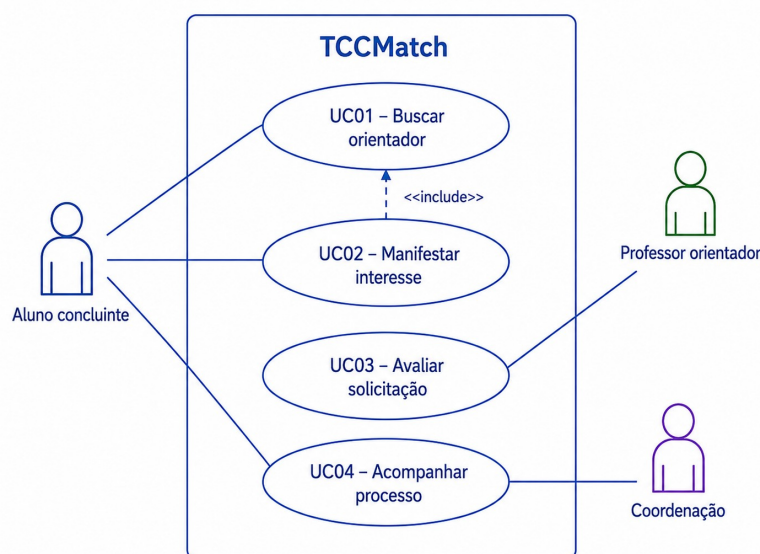


Figura 4.10: Diagrama visual de casos de uso do protótipo *TCCMatch*. **Fonte:** Elaborado pelo autor com apoio do ChatGPT (2026).

Os casos de uso principais foram detalhados em nível conceitual para explicitar objetivo, pré-condição, fluxo básico e resultado esperado. Essa especificação complementa as histórias de usuário porque descreve a interação do sistema de forma mais operacional, servindo como ponte entre requisitos, arquitetura funcional, mapa de navegação e prototipação. O Quadro 4.21 apresenta essa especificação.

Quadro 4.21: Especificação conceitual dos casos de uso principais do *TCCMatch*

Caso de uso	Objetivo	Fluxo básico	Resultado esperado
UC01 – Buscar orientador	Permitir que o aluno encontre professores compatíveis com tema ou área.	Aluno acessa busca, informa área ou palavra-chave, aplica filtros e visualiza resultados com disponibilidade.	Lista de professores ou oportunidades aderentes ao interesse informado.
UC02 – Manifestar interesse	Registrar pedido de orientação de forma mediada e rastreável.	Aluno abre detalhe da oportunidade, informa resumo do tema e envia solicitação ao professor.	Solicitação registrada com status inicial visível ao aluno.
UC03 – Avaliar solicitação	Apoiar o professor na decisão sobre pedidos recebidos.	Professor acessa solicitações, consulta mini perfil e tema, e registra aceite, recusa ou pedido de ajuste.	Solicitação atualizada e resposta disponibilizada ao aluno e à coordenação.
UC04 – Acompanhar processo	Dar visibilidade ao andamento da formação das orientações.	Coordenação consulta painel, verifica vagas, pendências e alunos sem orientação.	Situação consolidada para acompanhamento e intervenção acadêmica.

Para explicitar o backlog inicial como artefato próprio do SDLC, as histórias priorizadas

foram agrupadas em itens de trabalho conceituais. Esse backlog não representa um plano de implementação completo, mas uma lista inicial de funcionalidades suficiente para orientar módulos, fluxos e telas do protótipo. O Quadro 4.22 apresenta essa organização.

Quadro 4.22: Backlog inicial priorizado para o MVP do *TCCMatch*

Item	Funcionalidade	Histórias associadas	Justificativa de prioridade
BI01	Cadastro e manutenção de perfis de aluno e professor	HU03, HU04	Base para qualificar solicitações e divulgar disponibilidade
BI02	Feed de linhas, ideias e oportunidades de orientação	HU01, HU04	Reduz assimetria de informação sobre temas e docentes
BI03	Busca por área, palavra-chave e disponibilidade	HU01	Sustenta o fluxo principal de descoberta do aluno
BI04	Manifestação de interesse com acompanhamento de status	HU02, HU03, HU06	Centraliza comunicação e torna o retorno rastreável
BI05	Painel de acompanhamento da coordenação	HU05	Amplia visibilidade sobre vagas, pendências e alunos sem orientação

A partir das histórias de usuário, foi projetada a arquitetura funcional inicial do protótipo. Esse artefato pertence ao SDLC porque organiza módulos, responsabilidades e relações entre funcionalidades antes da prototipação. O Quadro 4.23 apresenta os módulos definidos.

Quadro 4.23: Arquitetura funcional projetada para o MVP do *TCCMatch*

Módulo	Responsabilidade	Histórias associadas
Perfis de usuário	Registrar dados básicos de aluno, professor e coordenação	HU03, HU04
Descoberta temática	Exibir linhas, ideias de TCC e professores por área	HU01, HU04
Busca e filtros	Permitir filtragem por área, palavra-chave e disponibilidade	HU01
Solicitação de orientação	Mediar manifestação de interesse, aceite, recusa e status	HU02, HU03, HU06
Painel de coordenação	Consolidar vagas, solicitações e estudantes sem orientação	HU05

A definição desses módulos constitui a arquitetura funcional inicial do protótipo. Enquanto as histórias descrevem necessidades sob a perspectiva dos atores, os módulos agrupam responsabilidades do sistema, casos de uso atendidos e informações manipuladas, ajudando a delimitar quais partes da solução devem ser priorizadas na prototipação. Essa passagem é importante

porque evita que o protótipo avance diretamente de requisitos para telas isoladas, sem antes explicitar a lógica de funcionamento que sustenta a plataforma.

A Figura 4.11 complementa a arquitetura funcional ao mostrar como os módulos do MVP se relacionam com os atores e com os casos de uso priorizados. Essa representação visual reforça a passagem entre requisitos, backlog e organização modular da solução.

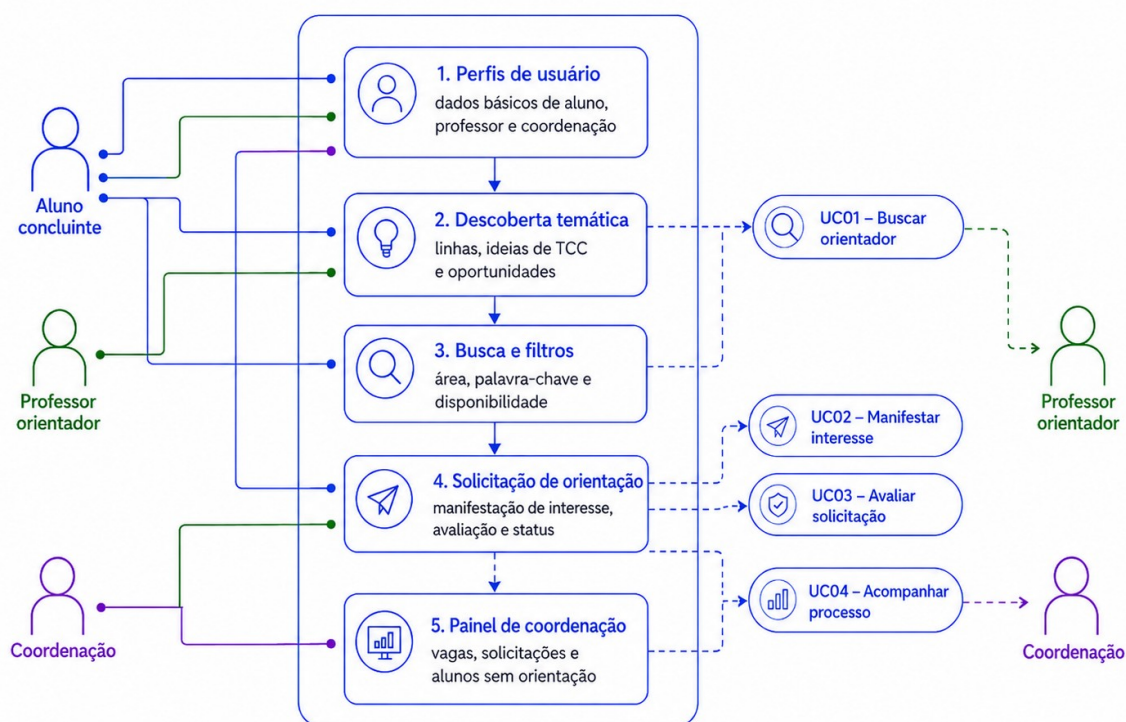


Figura 4.11: Arquitetura funcional visual do protótipo *TCCMatch*. **Fonte:** Elaborado pelo autor com apoio do ChatGPT (2026).

No caso do *TCCMatch*, a organização modular também evidenciou a relação entre experiência do usuário e controle do processo acadêmico. Os módulos de descoberta temática, busca e solicitação de orientação respondem principalmente às dores dos alunos, relacionadas à dificuldade de encontrar professores compatíveis e acompanhar o retorno. Já os módulos de perfis de usuário e painel de coordenação atendem à necessidade de manter dados atualizados, disponibilidade visível e acompanhamento institucional do fluxo de orientação.

Essa estrutura reforça o papel do MVP como uma versão conceitual mínima, mas coerente,

da solução. O objetivo não é contemplar todas as funcionalidades possíveis, e sim reunir os recursos essenciais para testar se a mediação proposta reduz assimetria de informação, melhora a comunicação entre atores e oferece maior rastreabilidade ao processo. A Figura 4.12 apresenta a organização visual do MVP, relacionando os módulos de perfil, descoberta, busca, solicitação e acompanhamento.

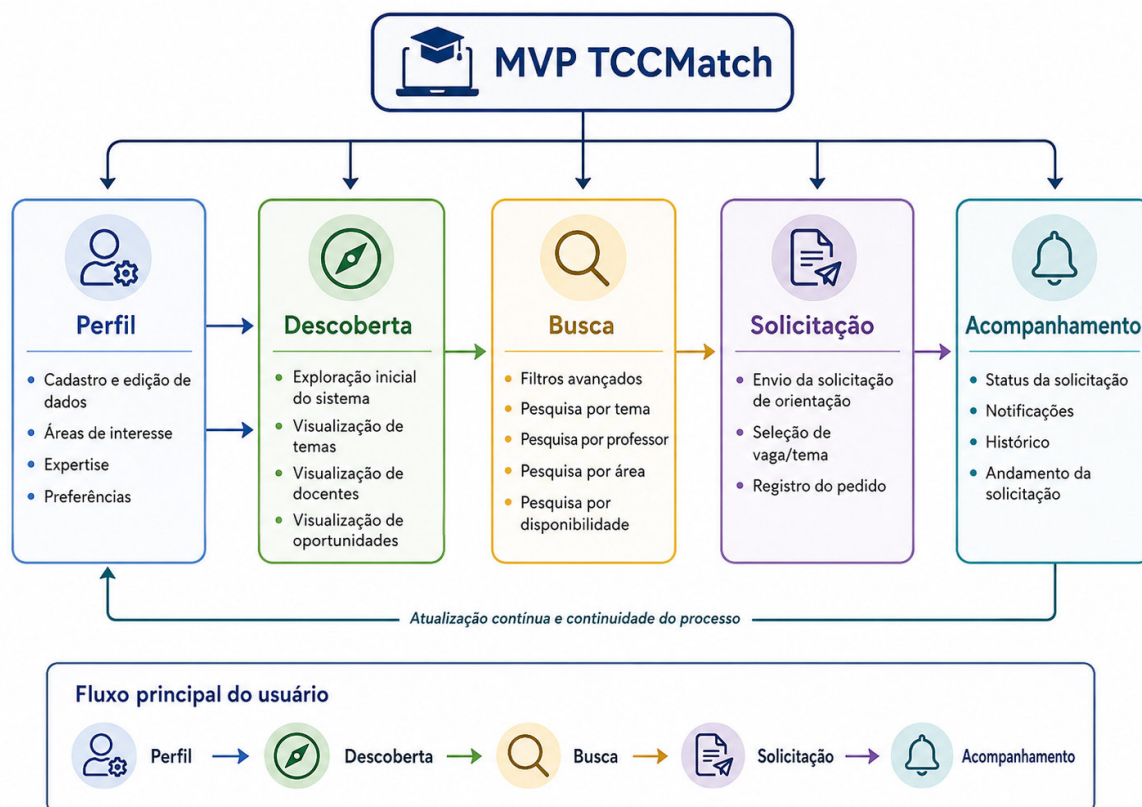


Figura 4.12: Organização funcional do MVP do protótipo *TCCMatch*. **Fonte:** Elaborado pelo autor com apoio do ChatGPT (2026).

O fluxo principal de navegação também foi projetado nesta etapa. Para o aluno, o fluxo inicia no cadastro ou atualização de perfil, passa pela busca temática, visualização de oportunidade, manifestação de interesse e acompanhamento de status. Para o professor, o fluxo envolve cadastro de linhas e disponibilidade, recebimento de solicitações, avaliação do mini perfil e resposta. Para a coordenação, o fluxo concentra acompanhamento de vagas, solicitações pendentes e alunos sem orientação.

Para explicitar o mapa de navegação como artefato do SDLC, os percursos foram organizados por perfil de acesso, ponto de entrada, telas principais e saídas esperadas. O Quadro 4.24

apresenta essa consolidação, que orientou a construção dos wireframes e do protótipo navegável.

Quadro 4.24: Mapa de navegação conceitual do protótipo *TCCMatch*

Perfil	Percurso principal	Telas ou áreas envolvidas	Saída esperada
Aluno	Entrar no sistema, atualizar perfil, buscar oportunidade, visualizar detalhe, manifestar interesse e acompanhar retorno.	Perfil do aluno; busca; resultados; detalhe da oportunidade; solicitação; acompanhamento.	Solicitação registrada e status compreensível.
Professor	Atualizar perfil docente, cadastrar linhas e vagas, receber solicitações, avaliar mini perfil e responder.	Perfil docente; disponibilidade; solicitações recebidas; detalhe do aluno; resposta.	Solicitação decidida e vagas atualizadas.
Coordenação	Acessar painel, filtrar situação geral, identificar pendências, acompanhar vagas e alunos sem orientação.	Painel gerencial; lista de solicitações; indicadores de vagas; alertas de pendência.	Visão consolidada do processo e pontos de intervenção.

A Figura 4.13 apresenta o fluxograma de navegação principal do protótipo, estruturado a partir dos percursos de aluno, professor e coordenação.

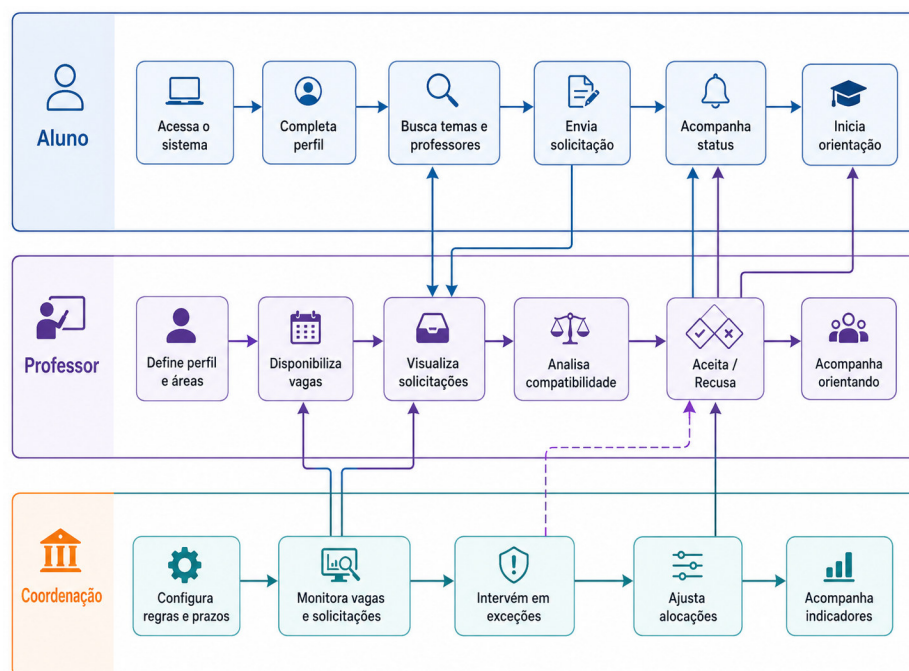


Figura 4.13: Fluxograma de navegação principal do protótipo *TCCMatch*. **Fonte:** Elaborado pelo autor com apoio do ChatGPT (2026).

Além do mapa de navegação, o diagrama de atividades foi descrito em formato textual para representar a dinâmica do processo principal, desde a busca do aluno até a decisão do professor e o acompanhamento pela coordenação. O Quadro 4.25 apresenta essa sequência.

Quadro 4.25: Diagrama textual de atividades do fluxo principal do *TCCMatch*

Ordem	Atividade	Responsável	Condição ou saída
1	Atualizar perfil e tema de interesse.	Aluno concluinte	Perfil mínimo disponível para busca e solicitação.
2	Consultar linhas, oportunidades e professores disponíveis.	Aluno concluinte	Lista filtrada por área, palavra-chave ou disponibilidade.
3	Visualizar detalhe da oportunidade e decidir envio.	Aluno concluinte	Caso haja aderência, segue para manifestação de interesse.
4	Registrar manifestação de interesse com resumo do tema.	Sistema e aluno	Solicitação criada com status pendente.
5	Avaliar solicitação, perfil do aluno e aderência temática.	Professor orientador	Decisão de aceitar, recusar ou solicitar ajuste.
6	Atualizar status da solicitação e disponibilidade.	Sistema	Status visível para aluno e coordenação.
7	Acompanhar pendências, vagas e alunos sem orientação.	Coordenação	Possível intervenção acadêmica quando houver gargalo.

A Figura 4.14 complementa o diagrama textual ao distribuir o fluxo principal em raias de responsabilidade. Essa visualização facilita a compreensão de quando a ação pertence ao aluno, ao sistema, ao professor ou à coordenação, evidenciando a transição entre busca, solicitação, decisão e acompanhamento.

A construção em raias foi adotada para tornar visíveis dependências que poderiam permanecer implícitas em uma descrição puramente textual. No fluxo analisado, o aluno inicia a busca e registra interesse, mas a continuidade depende do registro do sistema, da avaliação do professor e da capacidade da coordenação de acompanhar gargalos. Essa separação ajuda a evitar que o sistema seja desenhado como uma sequência linear controlada apenas pelo aluno, quando, na prática, a experiência depende de respostas e atualizações distribuídas entre diferentes atores.

Outro efeito relevante desse artefato foi explicitar pontos de espera. A passagem da manifestação de interesse para a decisão docente é um momento sensível, pois pode gerar ansiedade, retrabalho ou novas tentativas de contato fora da plataforma. Ao representar essa transição no diagrama, o protótipo reforçou a necessidade de status visível, notificação e histórico da solicitação, evitando que o protótipo reproduzisse o mesmo silêncio informacional observado no processo atual.

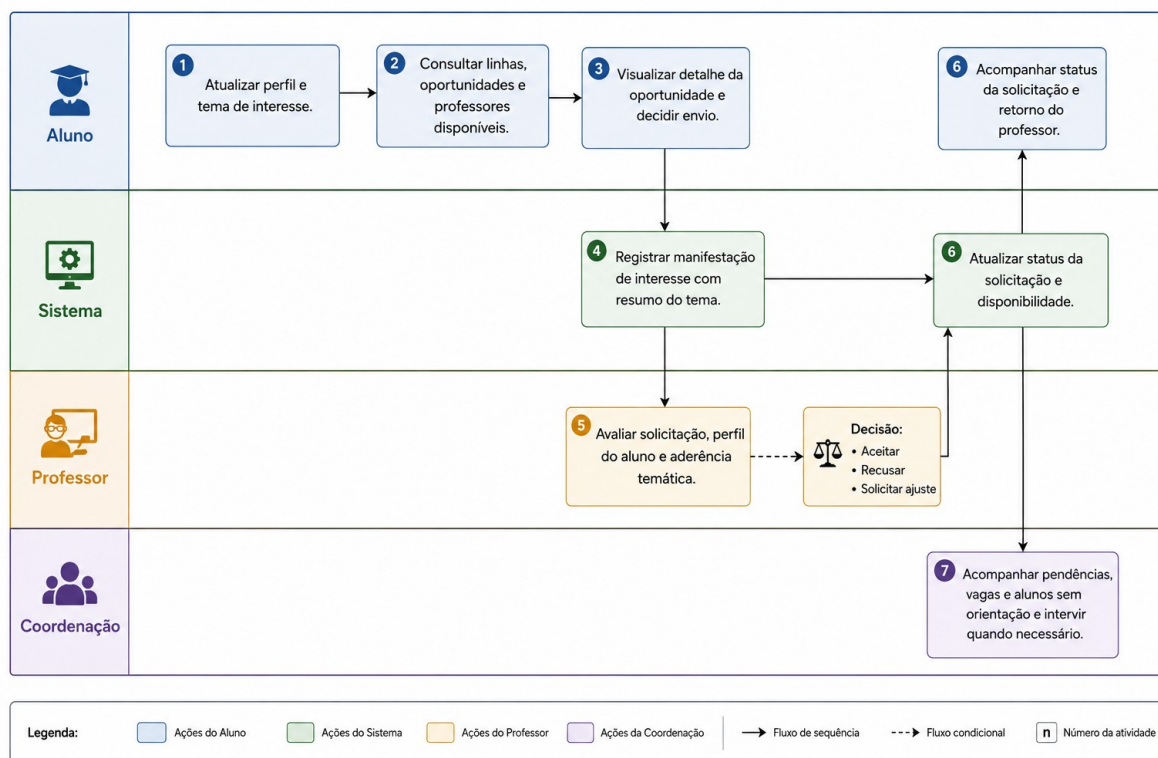


Figura 4.14: Diagrama visual de atividades do protótipo *TCCMatch*. **Fonte:** Elaborado pelo autor com apoio do ChatGPT (2026).

Como resultado da ideação, o protótipo passou a possuir alternativas avaliadas, MVP definido, histórias de usuário, backlog inicial, diagrama e especificação de casos de uso, arquitetura funcional, mapa de navegação e diagrama de atividades. Esses artefatos forneceram a base necessária para construir protótipos coerentes com o problema e com os requisitos organizados na etapa anterior.

Prototipação

A quarta etapa teve como função materializar a solução escolhida em representações visuais capazes de sustentar avaliação antes da implementação técnica. No modelo convergente, a prototipação não é apenas uma ilustração da ideia; ela é um mecanismo de validação conceitual dos requisitos, fluxos e decisões de interface.

Os storyboards foram projetados para representar cenários de uso recorrentes: aluno com tema definido, mas sem orientador; professor que recebe solicitação compatível; aluno que acompanha resposta; e coordenação que identifica estudantes ainda sem encaminhamento. Esse

artefato ajudou a verificar se a solução cobria situações reais do processo e não apenas um fluxo idealizado.

A Figura 4.15 apresenta o storyboard do fluxo do aluno, desde a busca por orientador até a manifestação de interesse e o acompanhamento do retorno.



Figura 4.15: Storyboard do fluxo do aluno no protótipo *TCCMatch*. **Fonte:** Elaborado pelo autor com apoio do ChatGPT (2026).

A prototipação em papel foi planejada como primeira forma de materialização das telas, permitindo testar rapidamente organização de informações, termos utilizados e sequência de ações. Em seguida, os wireframes estruturaram as telas prioritárias: cadastro e perfil, feed de ideias e linhas, busca e filtros, detalhe da oportunidade, manifestação de interesse, painel do professor e painel da coordenação.

Durante a geração desses artefatos, uma das principais dificuldades foi transformar necessidades qualitativas, como insegurança do aluno, falta de clareza sobre disponibilidade docente e ausência de acompanhamento institucional, em elementos visuais objetivos. Nem todos os

achados da imersão podiam ser convertidos diretamente em telas, pois alguns representavam problemas de processo, comunicação ou governança. Nesses casos, o modelo mostrou-se eficiente ao exigir a passagem pelos artefatos intermediários de análise e ideação antes da prototipação. A matriz de rastreabilidade, o backlog inicial e os fluxos de navegação funcionaram como filtros para evitar que o protótipo incorporasse funcionalidades sem relação clara com o problema investigado.

Também foram observados acertos importantes na aplicação do modelo. A separação entre storyboard, wireframe e protótipo navegável permitiu amadurecer a solução progressivamente, sem antecipar decisões visuais antes da definição do fluxo. O storyboard ajudou a verificar se a solução fazia sentido na experiência do aluno; os wireframes permitiram organizar a hierarquia das informações; e o protótipo navegável consolidou a lógica de interação entre aluno, professor e coordenação. Assim, a implementação metodológica do Modelo Convergente DT-SDLC contribuiu para reduzir retrabalho conceitual, pois cada artefato produzido retomava decisões anteriores e preparava a validação seguinte com maior coerência.

Na passagem para os wireframes, a principal decisão foi transformar requisitos em zonas de informação verificáveis. A busca precisava evidenciar filtros e resultados; o detalhe da oportunidade precisava reunir compatibilidade, vagas, critérios e ação de manifestação de interesse; e o painel da coordenação precisava sintetizar indicadores sem assumir a função de sistema acadêmico completo. Essa delimitação evitou dois excessos: uma interface centrada apenas no aluno, que deixaria professores e coordenação sem suporte, e uma interface administrativa pesada demais para um protótipo conceitual.

Também se buscou evitar que a prototipação introduzisse funcionalidades não rastreadas. Por isso, cada bloco visual foi vinculado a requisitos, histórias de usuário ou pontos de dor já registrados. Quando uma informação não possuía relação clara com o problema reformulado, ela foi deixada fora do wireframe ou tratada como evolução futura. Esse critério manteve a simplicidade do MVP e reduziu o risco de confundir validação conceitual com desenho completo do produto final.

As Figuras 4.16, 4.17 e 4.18 apresentam os wireframes de baixa fidelidade das telas de busca,

detalhe da oportunidade e painel da coordenação.



Figura 4.16: Wireframe da tela de busca do protótipo *TCCMatch*. **Fonte:** Elaborado pelo autor com apoio do ChatGPT (2026).

O wireframe de busca foi gerado a partir da necessidade de reduzir tentativas exploratórias pouco direcionadas. A presença de filtros por tema, área, professor e disponibilidade evita que o aluno dependa apenas de indicação informal ou de listas genéricas de docentes. Esse desenho

também previne um efeito colateral indesejado: transformar a plataforma em um catálogo extenso, mas sem critérios visíveis de escolha.

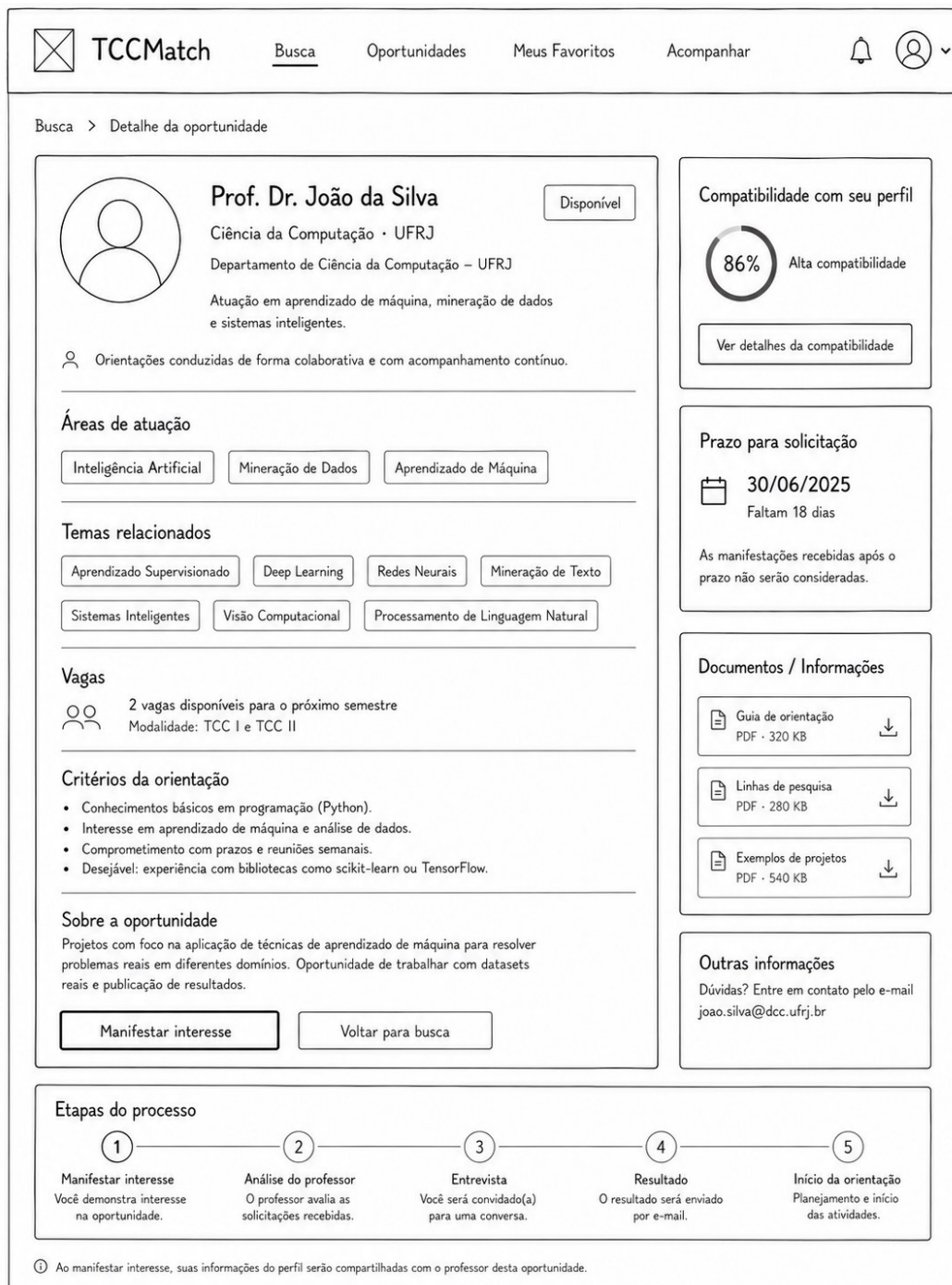


Figura 4.17: Wireframe da tela de detalhe da oportunidade do protótipo *TCCMatch*. **Fonte:** Elaborado pelo autor com apoio do ChatGPT (2026).

No wireframe de detalhe, a organização das informações foi pensada para qualificar a decisão antes do envio da solicitação. Compatibilidade, prazo, vagas, critérios e documentos aparecem

próximos da ação de manifestação de interesse para evitar contatos incompletos ou desalinhados. Assim, o artefato traduz a preocupação metodológica de transformar insegurança do aluno em informações verificáveis no fluxo de uso.

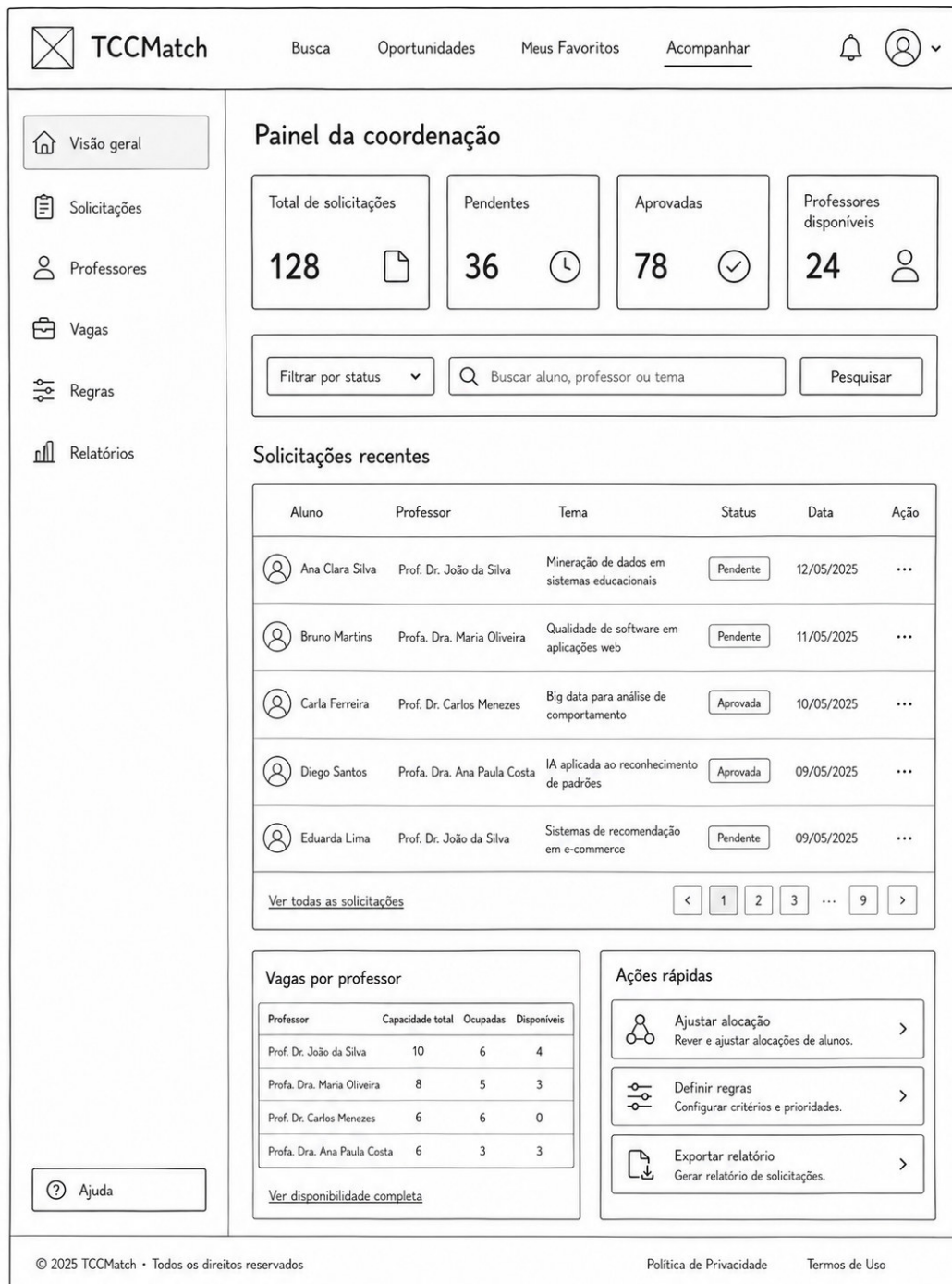


Figura 4.18: Wireframe do painel da coordenação do protótipo *TCCMatch*. **Fonte:** Elaborado pelo autor com apoio do ChatGPT (2026).

O painel da coordenação foi mantido como visão sintética, e não como sistema administrativo

completo. Essa delimitação evita ampliar indevidamente o escopo do protótipo, mas preserva a capacidade de identificar solicitações pendentes, vagas por professor e pontos de intervenção. Dessa forma, o wireframe responde à governança do processo sem deslocar o projeto para funcionalidades de gestão acadêmica fora do recorte conceitual.

A especificação preliminar das telas foi organizada para vincular cada interface ao objetivo de uso e aos requisitos atendidos. O Quadro 4.26 apresenta essa relação.

Quadro 4.26: Especificação preliminar das telas prototipadas no *TCCMatch*

Tela	Objetivo principal	Requisitos associados
Cadastro e perfil	Registrar tipo de usuário, área de interesse e informações iniciais	RF01
Feed de ideias e linhas	Exibir temas, linhas de pesquisa e oportunidades de orientação	RF02, RF03
Busca e filtros	Restringir resultados por área, palavra-chave e disponibilidade	RF03
Detalhe da oportunidade	Exibir resumo do professor ou do aluno e permitir decisão informada	RF04, RNF03
Painel do professor	Visualizar solicitações pendentes e responder aos pedidos	RF04
Painel da coordenação	Monitorar vagas, pares formados e estudantes sem orientação	RF05

Como complemento da especificação preliminar de telas, foi elaborado um modelo conceitual de dados em nível exploratório. Esse artefato não define tabelas físicas, chaves, tipos de dados ou tecnologia de armazenamento; sua função é apenas identificar as principais entidades de informação exigidas pelos requisitos e pelas telas prototipadas. O Quadro 4.27 apresenta essa síntese.

Quadro 4.27: Modelo conceitual de dados preliminar do protótipo *TCCMatch*

Entidade conceitual	Informações principais	Relação com requisitos e telas
Aluno	Nome, curso, período, área de interesse, tema pretendido e resumo da proposta.	Alimenta mini perfil, manifestação de interesse e acompanhamento da solicitação.
Professor	Nome, área de atuação, linhas de pesquisa, disponibilidade, vagas e preferências de orientação.	Sustenta perfil docente, feed de oportunidades, filtros de busca e decisão do aluno.
Linha ou oportunidade de TCC	Título, área, palavras-chave, descrição, professor responsável e vagas relacionadas.	Permite descoberta temática, detalhe da oportunidade e compatibilidade inicial.

Entidade conceitual	Informações principais	Relação com requisitos e telas
Solicitação de orientação	Aluno solicitante, professor destinatário, tema, mensagem resumida, status e histórico mínimo de decisão.	Materializa manifestação de interesse, triagem docente e acompanhamento pelo aluno.
Vaga de orientação	Professor, quantidade disponível, ocupação atual e situação da oferta.	Apoia transparência de disponibilidade e painel da coordenação.
Acompanhamento acadêmico	Alunos sem orientação, solicitações pendentes, pares formados e alertas de gargalo.	Sustenta visão gerencial da coordenação e identificação de exceções no processo.

Para aproximar a prototipação de critérios verificáveis do SDLC, foram definidos critérios de aceitação conceitual. Eles não equivalem a testes funcionais de software implementado, mas orientam a avaliação do protótipo quanto à clareza, completude e aderência aos requisitos. O Quadro 4.28 apresenta esses critérios.

Quadro 4.28: Critérios de aceitação conceitual para avaliação do protótipo

Fluxo avaliado	Critério de aceitação conceitual	Requisito relacionado
Busca de orientador	O aluno deve compreender como filtrar professores por área, palavra-chave e disponibilidade	RF03, RNF01
Manifestação de interesse	O aluno deve identificar claramente quando a solicitação foi enviada e qual seu status	RF04, RNF02
Avaliação pelo professor	O professor deve visualizar tema, área e mini perfil antes de aceitar ou recusar	RF04, RNF03
Painel da coordenação	A coordenação deve identificar alunos sem orientação e vagas disponíveis por docente	RF05, RNF02

A validação conceitual foi conduzida por meio de tarefas orientadas por cenários de uso. O objetivo foi observar se os stakeholders compreenderiam os fluxos principais, se as informações exibidas eram suficientes para tomada de decisão e se o protótipo reduzia as incertezas identificadas nas etapas anteriores. O Quadro 4.29 apresenta o roteiro utilizado.

Quadro 4.29: Roteiro de validação qualitativa aplicado ao protótipo do *TCCMatch*

Tarefa proposta	Aspecto observado	Critério de análise
Encontrar um orientador compatível	Facilidade de localizar filtros e interpretar resultados	Clareza do fluxo de busca
Manifestar interesse por orientação	Compreensão do caminho até a solicitação	Intuitividade da ação principal
Avaliar uma solicitação recebida	Leitura do mini perfil e segurança da decisão	Qualidade da informação exibida
Acompanhar status do processo	Identificação de vagas, pendências e solicitações respondidas	Transparência do estado da interação

Os resultados indicaram que a solução proposta reduziu a incerteza inicial dos alunos, especialmente por tornar explícita a relação entre tema, professor e disponibilidade. A maior contribuição percebida pelos professores foi a possibilidade de triagem mais qualificada das solicitações, enquanto a coordenação destacou o ganho de visibilidade sobre o processo. O Quadro 4.30 apresenta os principais resultados qualitativos observados.

Quadro 4.30: Resumo qualitativo da validação conceitual do protótipo

Aspecto avaliado	Evidência observada	Conclusão para o projeto
Clareza na identificação de orientadores	Alunos localizaram com mais facilidade opções aderentes quando havia filtro por tema e visualização de disponibilidade	A transparência informacional é requisito central do sistema
Qualidade da decisão do professor	Docentes sentiram-se mais seguros quando visualizaram mini perfil do aluno e tema resumido	O aceite deve ser apoiado por contexto mínimo qualificado
Controle do processo pela coordenação	O painel consolidado foi percebido como instrumento de governança acadêmica	O sistema precisa incluir visão gerencial desde o MVP
Compreensão do fluxo de interesse	Houve melhora após explicitar vagas e status da solicitação	Feedback visual e estados do processo devem ser priorizados na interface

As Figuras 4.19, 4.20, 4.21 e 4.22 apresentam as telas de média fidelidade do protótipo navegável, contemplando a busca por orientadores, a visualização de detalhes da oportunidade, o acompanhamento da solicitação pelo aluno e o painel de acompanhamento da coordenação. Essas telas materializam, em nível visual, os requisitos e fluxos definidos nas etapas anteriores do modelo, permitindo observar como as necessidades identificadas na imersão foram convertidas em decisões de interface. Além disso, funcionam como base para a validação conceitual com stakeholders, pois tornam possível discutir navegação, clareza das informações, pontos de decisão e adequação do fluxo antes de qualquer implementação técnica.

Os aprendizados gerados na validação conceitual foram reincorporados ao próprio ciclo de prototipação. Em vez de tratar a validação como encerramento do processo ou a retroalimentação como etapa autônoma, o estudo de caso utilizou seus achados para revisar requisitos, complementar fluxos e fortalecer a coerência entre os artefatos produzidos. Esse retorno iterativo resultou em refinamentos no backlog, apresentados no Quadro 4.31.

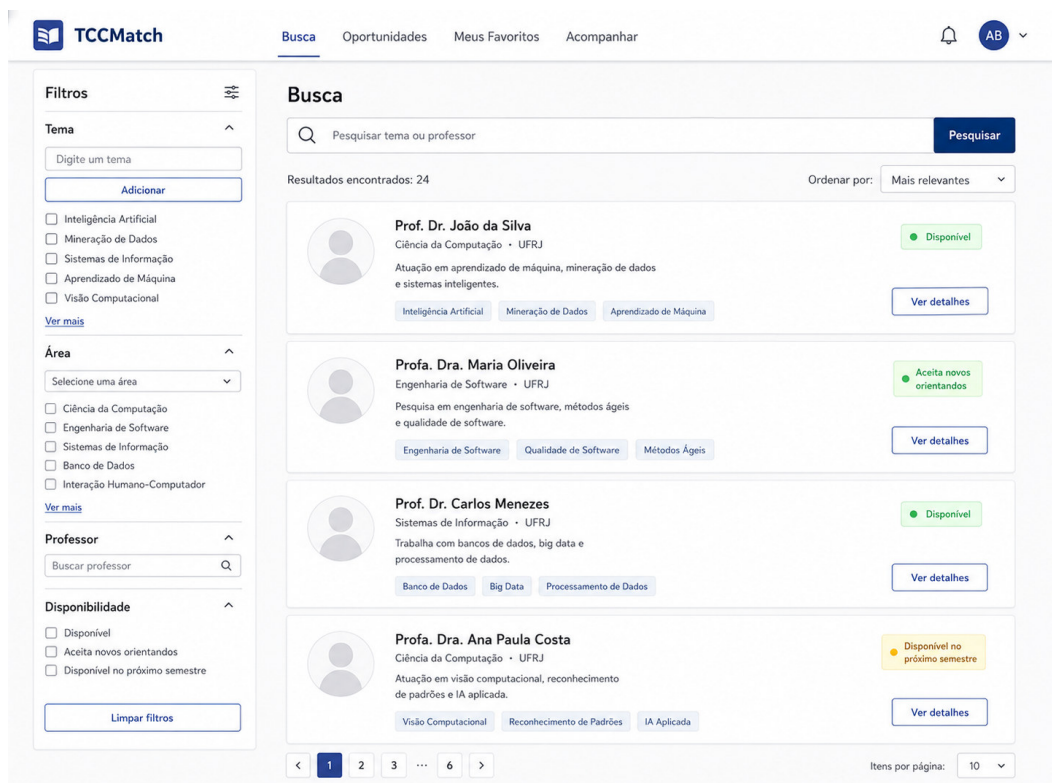


Figura 4.19: Protótipo: tela de busca de orientadores do *TCCMatch*. **Fonte:** Elaborado pelo autor com apoio do ChatGPT (2026).

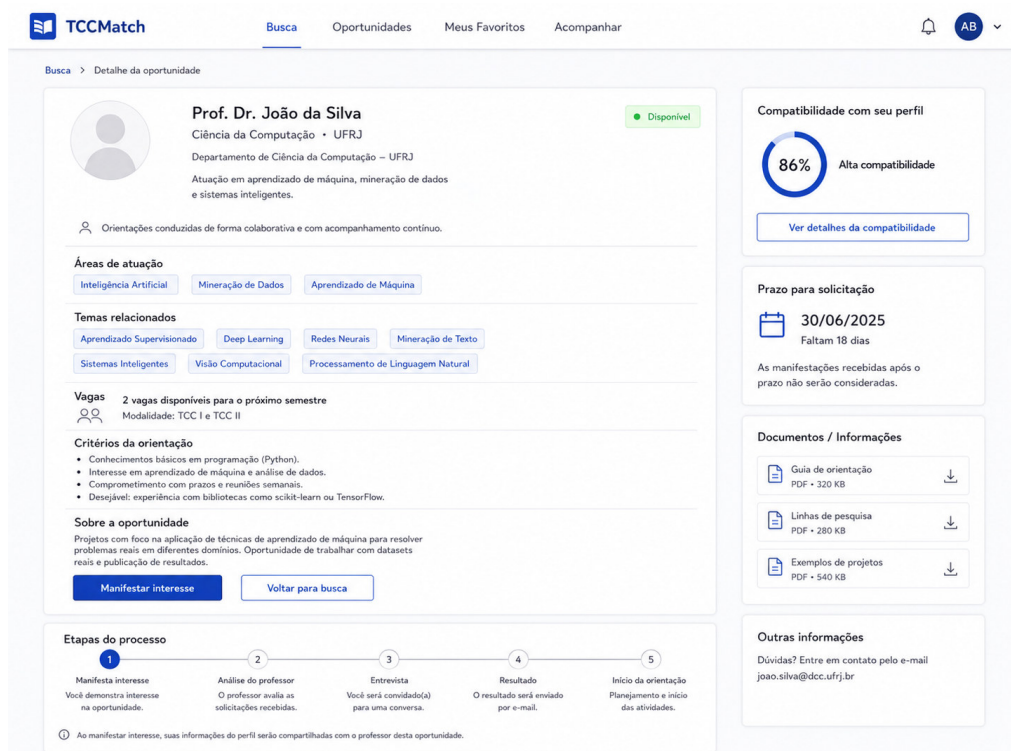


Figura 4.20: Protótipo: tela de detalhes da oportunidade do *TCCMatch*. **Fonte:** Elaborado pelo autor com apoio do ChatGPT (2026).

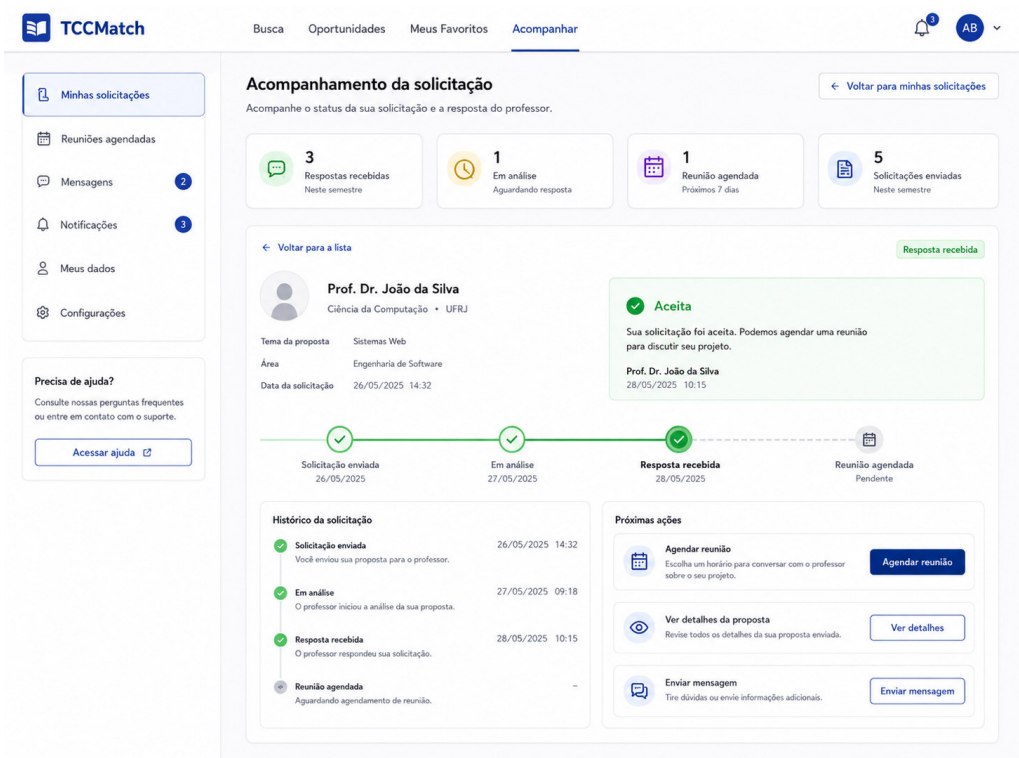


Figura 4.21: Protótipo: tela de acompanhamento da solicitação do TCCMatch. Fonte: Elaborado pelo autor com apoio do ChatGPT (2026).

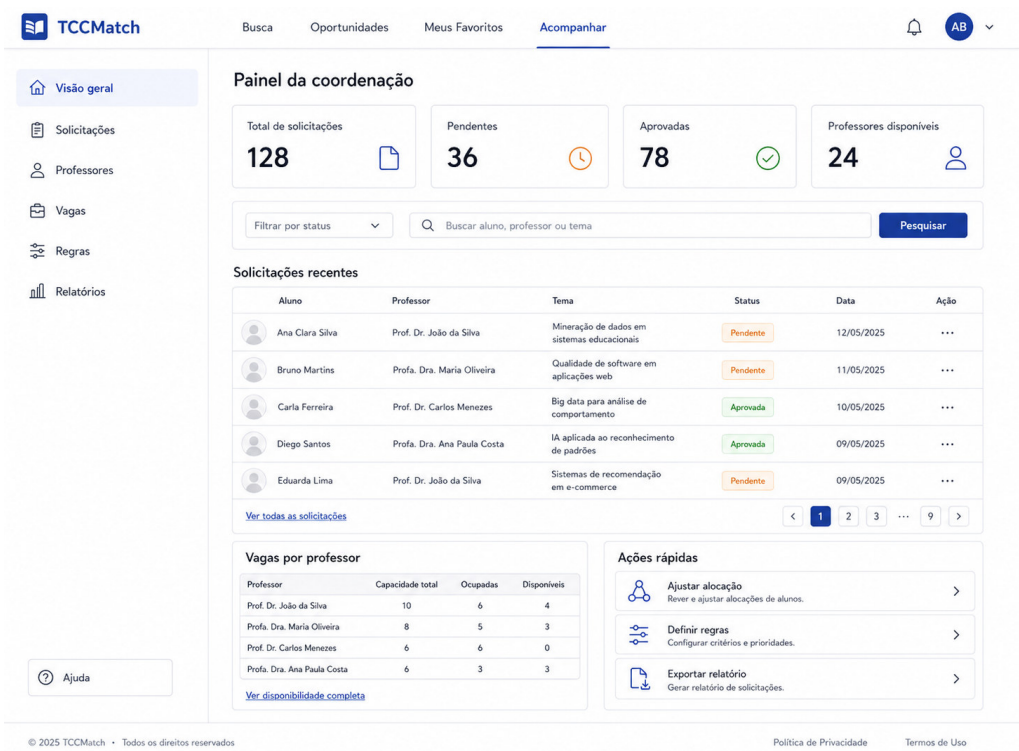


Figura 4.22: Protótipo: painel da coordenação do TCCMatch. Fonte: Elaborado pelo autor com apoio do ChatGPT (2026).

Quadro 4.31: Backlog refinado após a validação conceitual do protótipo

Item de melhoria	Prioridade	Origem do refinamento
Exibir vagas totais e vagas disponíveis do professor	Alta	Dúvidas recorrentes na validação
Adicionar filtros por subárea e palavra-chave	Alta	Necessidade de maior precisão na busca
Permitir publicação de temas sugeridos por professores	Média	Solicitação recorrente dos docentes
Expandir painel da coordenação com marcadores de risco	Alta	Necessidade de monitoramento gerencial
Inserir histórico de solicitações do aluno	Média	Acompanhamento do fluxo de contato
Exibir critérios de compatibilidade entre tema, linha de pesquisa e disponibilidade	Alta	Dificuldade dos alunos em justificar a escolha do orientador
Ordenar resultados por aderência temática, disponibilidade e atualização do perfil	Alta	Necessidade de reduzir buscas dispersas e tentativas pouco alinhadas
Adicionar status intermediários para solicitações em análise, ajuste solicitado, aceita e recusada	Alta	Falta de previsibilidade no acompanhamento do processo
Enviar notificações ao aluno quando houver mudança de status da solicitação	Alta	Ansiedade gerada pela espera sem retorno explícito
Permitir ao professor solicitar ajuste no resumo do tema antes da decisão final	Média	Necessidade docente de esclarecer propostas incompletas
Registrar justificativa padronizada para recusas ou pedidos de ajuste	Média	Demanda por transparência e aprendizagem após retorno negativo
Limitar solicitações simultâneas por aluno para evitar sobrecarga docente	Alta	Regra de negócio identificada a partir do risco de envio indiscriminado
Exibir fila de solicitações recebidas por professor com filtros por status e data	Média	Organização do trabalho docente durante períodos de alta demanda
Criar indicador de alunos sem orientação definida no painel da coordenação	Alta	Necessidade de intervenção acadêmica preventiva
Disponibilizar relatório sintético de vagas, solicitações e orientações confirmadas	Média	Demanda da coordenação por consolidação administrativa
Incluir mensagens orientadoras antes do envio da manifestação de interesse	Média	Redução de dúvidas sobre o conteúdo mínimo da solicitação
Permitir atualização rápida de disponibilidade e áreas de interesse pelo professor	Alta	Risco de informações desatualizadas comprometerem a busca
Manter registro cronológico das interações relevantes da solicitação	Média	Necessidade de rastreabilidade do fluxo entre aluno, professor e coordenação

A ampliação do backlog não teve como objetivo transformar o protótipo em um plano completo de implementação, mas registrar ajustes conceituais descobertos durante a validação. Alguns itens reforçam requisitos já existentes, como transparência de vagas e status; outros

detalham comportamentos necessários para reduzir incerteza, como notificações, histórico de solicitações e justificativas padronizadas. Com isso, o backlog refinado funciona como evidência de aprendizagem do protótipo, e não apenas como lista de novas funcionalidades.

Esse refinamento também indica comportamentos colaterais que o projeto procurou evitar. Sem limites para solicitações simultâneas, por exemplo, a plataforma poderia aumentar a sobrecarga dos professores em vez de reduzi-la. Sem critérios visíveis de compatibilidade, a busca poderia parecer arbitrária ou induzir escolhas pouco justificadas. Sem indicadores para a coordenação, o fluxo poderia resolver parcialmente a experiência individual do aluno, mas manter invisíveis os gargalos institucionais do processo de orientação.

Outro ponto observado foi que parte das melhorias corresponde a regras de mediação, não apenas a recursos de interface. A atualização rápida de disponibilidade, o registro cronológico das interações e os status intermediários ajudam a preservar a rastreabilidade entre intenção do aluno, decisão docente e acompanhamento institucional. Dessa forma, o backlog refinado mantém a coerência com o problema reformulado na imersão: reduzir assimetria de informação, insegurança e falta de previsibilidade no processo de formação das orientações.

Como resultado da prototipação, o protótipo passou a contar com storyboards projetados, wireframes, especificação preliminar de telas, critérios de aceitação conceitual, roteiro de validação, registro de feedback e backlog refinado. Esses artefatos demonstraram a retroalimentação prevista no modelo: os achados da validação retornaram para requisitos, fluxos e decisões de interface, mantendo a rastreabilidade entre problema, solução e protótipo. A ausência de artefatos técnicos mais completos não representa lacuna metodológica nesta etapa, mas consequência do recorte adotado: o protótipo buscou validar a coerência conceitual da solução, e não demonstrar sua implementação ou operação em ambiente real.

4.4 Resultados e discussão

A aplicação do *Modelo Convergente DT-SDLC* ao caso do *TCCMatch* demonstrou que a integração entre práticas de *Design Thinking* e SDLC é particularmente útil quando o problema envolve relações humanas, expectativas divergentes e alto grau de subjetividade, como ocorre

no processo de formação de orientações de TCC. O principal ganho observado não foi apenas a produção de uma solução funcional em nível conceitual, mas a melhoria da qualidade do entendimento do problema ao longo do processo.

Do ponto de vista qualitativo, o estudo evidenciou que as dores dos stakeholders não estavam restritas à ausência de uma ferramenta digital, mas à ausência de visibilidade, previsibilidade e alinhamento no processo atual. Ao permitir que essas dimensões fossem explicitadas na imersão e reinterpretadas na etapa seguinte, o modelo tornou possível construir requisitos menos abstratos e mais conectados à experiência real de uso. Esse aspecto reforçou a contribuição do DT nas fases iniciais do SDLC, sobretudo quando o desenvolvimento depende de decisões sensíveis ao contexto humano.

A comparação aqui proposta não tem caráter experimental, pois não houve execução paralela do mesmo projeto por uma equipe usando apenas Engenharia de Requisitos tradicional. Trata-se de uma discussão analítica sobre os tipos de artefatos e de informações que tenderiam a emergir em cada abordagem. Em um processo tradicional de Engenharia de Requisitos, provavelmente seriam produzidos artefatos como levantamento de requisitos, especificação funcional, casos de uso, modelo de domínio e backlog inicial. Esses produtos são relevantes e também aparecem no estudo, mas tendem a privilegiar a formalização do que o sistema deve fazer, deixando menos explícitos os aspectos emocionais, contextuais e relacionais que antecedem a definição da funcionalidade.

No caso do *TCCMatch*, o acréscimo do Design Thinking permitiu capturar dimensões que dificilmente seriam evidenciadas apenas por entrevistas tradicionais ou por uma lista inicial de requisitos. O mapa de empatia explicitou inseguranças, receios e expectativas dos alunos ao abordar professores; a jornada do usuário evidenciou momentos de espera, incerteza e ruptura no processo de busca por orientação; os cartões de insight e o diagrama de afinidades organizaram recorrências qualitativas que não estavam formuladas como requisitos; e os artefatos de síntese, como personas e quadro CSD previsto no modelo, ajudaram a separar perfis, certezas, suposições e dúvidas antes da consolidação da solução. Assim, o ganho do modelo não esteve apenas em gerar mais documentos, mas em qualificar a origem dos requisitos e reduzir o risco de

transformar uma compreensão superficial do problema em funcionalidades prematuras.

O Quadro 4.32 sintetiza essa comparação analítica entre uma condução tradicional de requisitos e a condução apoiada pelo *Modelo Convergente DT-SDLC*.

Quadro 4.32: Comparação analítica entre Engenharia de Requisitos tradicional e Modelo Convergente DT-SDLC

Condução tradicional de requisitos	Condução com DT-SDLC	Ganho analítico observado
Levantamento de requisitos por entrevistas, documentos e reuniões com stakeholders	Imersão com reenquadramento, mapa de stakeholders, registros de observação e síntese de entrevistas	Ampliação da compreensão do problema antes da definição da solução
Requisitos funcionais, requisitos não funcionais e regras de negócio	Cartões de insight, diagrama de afinidades, mapa de empatia, personas e jornada do usuário conectados aos requisitos	Captura de dores, expectativas, inseguranças e oportunidades de melhoria da experiência
Casos de uso, modelo de domínio e backlog inicial	Histórias de usuário, priorização orientada por impacto, fluxos, storyboards, wireframes e protótipos	Conversão de requisitos em cenários de uso e decisões de interface avaliáveis
Validação centrada na consistência dos requisitos e na revisão dos artefatos especificados	Validação conceitual com roteiro de tarefas, feedback e refinamento do backlog	Antecipação de ajustes antes da implementação e redução de ambiguidades no fluxo proposto

Essa comparação permite observar que uma abordagem tradicional provavelmente identificaria a necessidade de cadastrar professores, registrar áreas de interesse, consultar disponibilidade e acompanhar solicitações. Entretanto, o uso do DT ajudou a revelar por que essas funcionalidades eram necessárias e como deveriam ser organizadas para responder às dores reais dos usuários. Por exemplo, a funcionalidade de status da solicitação não surgiu apenas como requisito administrativo, mas como resposta à ansiedade do aluno durante a espera; a exibição de disponibilidade docente não foi apenas um dado cadastral, mas um mecanismo para reduzir tentativas frustradas de contato; e a priorização por aderência temática respondeu à dificuldade dos estudantes em avaliar compatibilidade entre tema e orientador.

O Quadro 4.33 sintetiza os principais resultados obtidos a partir da geração articulada e seletiva de artefatos de DT e do SDLC.

A análise confirmou a premissa construída na definição do *Modelo Convergente DT-SDLC*. No estudo de caso, a Imersão revelou atores, dores e necessidades implícitas; a Análise transfor-

Quadro 4.33: Síntese dos resultados obtidos com os artefatos do modelo convergente

Etapa	Resultado obtido	Contribuição para o protótipo
Imersão	Problema reformulado, stakeholders mapeados e necessidades iniciais delimitadas	Evitou que a solução fosse tratada apenas como cadastro de orientadores
Análise	Insights convertidos em requisitos, critérios e rastreabilidade	Tornou explícita a passagem entre evidência qualitativa e decisão de software
Ideação	Alternativas comparadas, MVP definido e backlog inicial produzido	Direcionou a solução para funcionalidades de maior impacto e menor dispersão
Prototipação	Telas, critérios de aceitação, roteiro de validação e backlog refinado	Antecipou ajustes antes de implementação técnica e fortaleceu a coerência da solução

mou dados qualitativos em requisitos, critérios e rastreabilidade; a Ideação apoiou a comparação de alternativas e a seleção do MVP; e a Prototipação permitiu testar fluxos e interfaces antes de qualquer implementação técnica.

Também se observou que a robustez do modelo residiu na sua capacidade de articular técnicas diferentes sem perder coerência metodológica. Pesquisa desk, entrevistas, cartões de insight, diagrama de afinidades, personas, jornada do usuário, brainstorming, cardápio de ideias, matriz de posicionamento, storyboards e protótipos não foram empregados como atividades isoladas, mas como partes de um fluxo com artefatos rastreáveis. Dessa forma, o estudo de caso fortaleceu a tese central deste trabalho: o DT não substitui o SDLC, mas amplia sua efetividade nas fases em que compreender, interpretar, decidir e validar conceitualmente são tão importantes quanto documentar e especificar. Ao mesmo tempo, a aplicação mostrou que o SDLC não exige a produção indiscriminada de artefatos; exige a produção dos artefatos adequados ao objetivo, ao risco e ao estágio do projeto.

Por fim, o caso aplicado mostrou que o modelo proposto é adequado ao desenvolvimento conceitual de plataformas web, especialmente quando a solução precisa equilibrar experiência do usuário, clareza de requisitos e viabilidade de evolução futura. A aplicação permitiu evidenciar como um processo orientado por dados qualitativos, registros do contexto acadêmico e recorrências observadas no processo de TCC pôde gerar conclusões concretas sobre problema, requisitos, fluxo de uso e critérios de validação, oferecendo sustentação metodológica à proposição desenvolvida neste trabalho.

Considerações Finais

Este trabalho teve como objetivo investigar como o *Design Thinking* (DT) pode ser incorporado às fases iniciais do SDLC e aplicar essa integração no desenvolvimento conceitual do *TCCMatch*, uma plataforma web de mediação de orientações de TCC. A partir da revisão de literatura, da análise comparativa entre abordagens de DT e do estudo de suas convergências com práticas do SDLC, foi construída uma proposta metodológica denominada *Modelo Convergente DT-SDLC*, orientada à compreensão do problema, à organização de requisitos, à concepção de soluções e à avaliação conceitual por meio de protótipos.

No percurso teórico do trabalho, observou-se que as abordagens de Brown (2009), Lindberg et al. (2011) e Vianna et al. (2012) oferecem contribuições distintas para o uso do DT em projetos de software. Essa análise foi conduzida com base em critérios comparativos explicitados na definição do modelo, como forma de estruturação do processo, presença de desdobramentos operacionais nomeados e quantidade de recursos metodológicos mapeados na sistematização adotada. Brown contribui sobretudo com uma visão ampla de inovação centrada no ser humano; Lindberg et al. destacam a relação entre espaço do problema, espaço da solução, interdisciplinaridade e alinhamento iterativo no contexto de TI; e Vianna et al. apresentam uma base mais operacional, com etapas, subetapas, técnicas e artefatos mais diretamente aplicáveis ao contexto deste trabalho. A principal conclusão dessa etapa foi que a integração entre DT e SDLC não deve ocorrer por simples justaposição de métodos, mas pela articulação coerente entre técnicas exploratórias e artefatos próprios do desenvolvimento de software.

Com base nessa análise, o modelo proposto foi estruturado a partir das quatro etapas de Vianna et al. (2012): *Imersão, Análise e Organização, Ideação e Prototipação*. No contexto da proposta, essas etapas foram reinterpretadas em diálogo com atividades das fases iniciais do

SDLC, resultando em um fluxo voltado à Engenharia de Requisitos, à organização de achados qualitativos, à ideação orientada por requisitos e à prototipação com avaliação conceitual. A validação foi tratada como avaliação de entendimento, coerência e aderência dos protótipos, e não como teste técnico ou homologação de um sistema implementado. Já a retroalimentação foi incorporada como princípio transversal para revisar artefatos sempre que novas necessidades, inconsistências ou oportunidades fossem identificadas.

A aplicação do modelo no estudo de caso conceitual do *TCCMatch* indicou que a integração entre DT e SDLC pode apoiar a estruturação de soluções em contextos marcados por fatores humanos, comunicação entre stakeholders e necessidade de clareza na definição do problema. A análise do caso permitiu organizar dores recorrentes no processo de formação de orientações de TCC, como assimetria de informação, comunicação dispersa, desalinhamento entre perfil e tema e baixa visibilidade gerencial. A partir desses achados, foi possível derivar requisitos, critérios de usabilidade, histórias de usuário, fluxos de navegação e protótipos conceituais mais coerentes com o problema investigado.

Outro resultado relevante foi a consolidação de uma sequência de artefatos que materializam a aplicação do modelo. Ao longo do estudo de caso, foram produzidos mapa de stakeholders, quadro de achados qualitativos, personas analíticas, jornada do usuário, requisitos funcionais e não funcionais, regras de negócio, matriz de rastreabilidade, histórias de usuário, priorização impacto-esforço, especificação preliminar de telas, roteiro de avaliação conceitual e backlog refinado para iterações futuras. Essa produção evidenciou que a integração entre DT e SDLC pode favorecer um processo mais organizado, no qual empatia, interpretação e criatividade são conectadas a rastreabilidade, formalização e tomada de decisão orientada por artefatos.

A análise também sugeriu que o uso do modelo contribui para reduzir ambiguidades na definição da solução e antecipar decisões importantes antes da implementação. Isso ocorreu porque a abordagem desloca parte do esforço inicial para a compreensão qualificada do problema e para a materialização progressiva da solução em artefatos avaliáveis, em vez de concentrar a robustez apenas na documentação técnica final. No caso do *TCCMatch*, esse percurso ajudou a evitar que a solução fosse tratada apenas como um cadastro de orientadores, ampliando sua

formulação para uma proposta de mediação, transparência e acompanhamento do processo de orientação.

Dessa forma, conclui-se que o *Modelo Convergente DT-SDLC* atendeu ao objetivo geral deste trabalho ao oferecer uma proposta aplicável de integração entre *Design Thinking* e SDLC nas fases iniciais do desenvolvimento. Sua principal contribuição está em mostrar, em nível conceitual e exploratório, como práticas de investigação qualitativa, organização de achados, ideação guiada e prototipação podem fortalecer a passagem entre compreensão do problema, especificação preliminar e concepção da solução. Ao mesmo tempo, o trabalho indicou que o DT ganha maior robustez quando suas práticas são conectadas a artefatos, critérios e decisões próprios da engenharia.

Por fim, este trabalho permite afirmar que a convergência entre DT e SDLC é uma alternativa metodologicamente promissora para projetos de software centrados no usuário, especialmente quando o desafio envolve incerteza sobre o problema, múltiplos stakeholders e necessidade de validação conceitual antes da implementação. Como limitação, destaca-se que a aplicação realizada se restringiu a um protótipo para validação do modelo e não incluiu desenvolvimento funcional, testes técnicos, implantação ou avaliação empírica em ambiente real de uso. Assim, a proposta aqui construída deve ser compreendida como uma contribuição teórico-metodológica e exploratória, que oferece base para aplicações futuras mais amplas e validações em projetos de software implementados.

4.5 Trabalhos Futuros

Os resultados alcançados neste trabalho indicam possibilidades de continuidade tanto para o aprofundamento acadêmico do modelo quanto para a evolução prática da solução proposta. Entre os desdobramentos mais relevantes, destacam-se:

- **Aplicação do modelo em novos projetos:** utilizar o *Modelo Convergente DT-SDLC* em outros projetos de software, especialmente em contextos educacionais, para observar sua contribuição na compreensão do problema, na clareza dos requisitos, na organização dos

artefatos e na avaliação conceitual de soluções.

- **Formalização e padronização do método:** transformar o *Modelo Convergente DT-SDLC* em um *framework metodológico* documentado, contendo diretrizes, templates, critérios de seleção de técnicas e modelos de rastreabilidade para uso do DT dentro do SDLC em contextos educacionais e corporativos.
- **Evolução do protótipo *TCCMatch*:** aprimorar o protótipo desenvolvido e avançar para uma versão funcional que contemple recursos como controle de vagas, painel da coordenação, acompanhamento das solicitações, histórico de interações e métricas de apoio à gestão das orientações.

Essas perspectivas indicam que o presente trabalho constitui um ponto de partida para novas aplicações do modelo e para o amadurecimento do protótipo desenvolvido. A continuidade da pesquisa poderá contribuir para consolidar uma abordagem de desenvolvimento de software que equilibre empatia, rigor técnico, organização metodológica e validação progressiva das decisões de projeto.

Referências Bibliográficas

- BOEHM, B. W. A spiral model of software development and enhancement. *IEEE Computer*, v. 21, n. 5, p. 61–72, 1988.
- BOEHM, B. W. A view of 20th and 21st century software engineering. In: *Proceedings of the 28th International Conference on Software Engineering (ICSE)*. [S.l.: s.n.], 2006.
- BROWN, T. Design thinking. *Harvard Business Review*, v. 86, n. 6, p. 84–92, 2008.
- BROWN, T. *Change by Design: How Design Thinking Transforms Organizations and Inspires Innovation*. New York: Harper Business, 2009. ISBN 9780061766084.
- CANEDO, E. D.; COSTA, R. P. da. The use of design thinking in agile software requirements survey: A case study. In: *Design, User Experience, and Usability: Theory and Practice*. [S.l.]: Springer, 2018. (Lecture Notes in Computer Science, v. 10918), p. 642–657.
- CANEDO, E. D. et al. Design thinking use in agile software projects: Software developers' perception. In: *Proceedings of the 22nd International Conference on Enterprise Information Systems (ICEIS)*. [S.l.: s.n.], 2020. v. 2, p. 217–224.
- GIL, A. C. *Como Elaborar Projetos de Pesquisa*. 4. ed. São Paulo: Atlas, 2002.
- HEHN, J.; MENDEZ, D. Combining design thinking and software requirements engineering to create human-centered software-intensive systems. *arXiv preprint*, arXiv:2112.05549, 2021. Disponível em: <<https://arxiv.org/abs/2112.05549>>.
- HEHN, J. et al. On integrating design thinking for human-centered requirements engineering. *IEEE Software*, v. 37, n. 2, p. 25–31, 2020.
- JUNIOR, J. C.; BARROS, M. de F. S.; ROCHA, A. R. C. Applying design thinking to requirements engineering. In: *Proceedings of the International Conference on Software Engineering and Knowledge Engineering (SEKE)*. Boston, USA: [s.n.], 2013. p. 118–123.
- KAHAN, E.; GENERO, M.; OLIVEROS, A. Refining a design thinking-based requirements elicitation process: Insights from a focus group. *Science of Computer Programming*, v. 237, p. 103137, 2024.
- LINDBERG, T. J. et al. Design thinking: A fruitful concept for it development? In: *Design Thinking: Understand – Improve – Apply*. [S.l.]: Springer, 2011. p. 3–18.
- MARTINS, H. F. et al. Design thinking: Challenges for software requirements elicitation. *Information*, v. 10, n. 12, p. 371, 2019.

- NIELSEN, J. *Usability Engineering*. San Diego: Academic Press, 1994.
- NORMAN, D. A. *The Design of Everyday Things: Revised and Expanded Edition*. New York: Basic Books, 2013.
- PALMA, J. G.; ARAÚJO, R. T. de; SOUZA, J. A. Uma abordagem de design thinking no desenvolvimento de software. *Conjecturas*, v. 22, n. 5, p. 1–15, 2022.
- PARIZI, R. et al. How has design thinking being used and integrated into software development activities? a systematic mapping. *Journal of Systems and Software*, v. 187, p. 111217, 2022.
- PEREIRA, J. C.; RUSSO, R. F. S. M. Design thinking integrated in agile software development: A systematic literature review. *Procedia Computer Science*, v. 138, p. 775–782, 2018.
- PINTO, F. A. de C.; SIQUEIRA, F. L. Problemas do design thinking para a engenharia de requisitos: uma revisão sistemática da literatura. In: *Anais do Workshop em Engenharia de Requisitos – Proceedings of the 23rd Workshop on Requirements Engineering (WER2020)*. [S.l.: s.n.], 2020.
- POHL, K. *Requirements Engineering: Fundamentals, Principles, and Techniques*. Berlin: Springer, 2010.
- PRESSMAN, R. S. *Engenharia de Software: uma abordagem profissional*. 8. ed. Porto Alegre: AMGH, 2016.
- PRESSMAN, R. S.; MAXIM, B. R. *Engenharia de Software: uma abordagem profissional*. 8. ed. Porto Alegre: AMGH Editora, 2019. ISBN 9788580555251.
- SOMMERVILLE, I. *Engenharia de Software*. 10. ed. São Paulo: Pearson, 2019.
- SOUZA, A. F. B. de; FERREIRA, B. M.; CONTE, T. Aplicando design thinking em engenharia de software: um mapeamento sistemático. In: *Ibero-American Conference on Software Engineering: Experimental Software Engineering Latin America Workshop (CibSE-ESELAW)*. [S.l.: s.n.], 2017. p. 719–732.
- VETTERLI, C. et al. From palaces to yurts: Why requirements engineering needs design thinking. *IEEE Internet Computing*, IEEE, v. 17, n. 2, p. 91–94, 2013.
- VIANNA, M. et al. *Design Thinking: Inovação em Negócios*. Rio de Janeiro: MJV Press, 2012. ISBN 9788564086008.
- WIEGERS, K.; BEATTY, J. *Software Requirements*. 3. ed. Redmond: Microsoft Press, 2013.
- YIN, R. K. *Estudo de Caso: Planejamento e Métodos*. 5. ed. Porto Alegre: Bookman, 2015.