UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS DE RUSSAS

CURSO DE ENGENHARIA DE SOFTWARE

ISABELA CARVALHO LOPES SILVA

ANÁLISE COMPARATIVA ENTRE A ABORDAGEM ÁGIL E TRADICIONAL NA ENGENHARIA DE REQUISITOS: UM ESTUDO DE CASO EM PROJETOS DE DESENVOLVIMENTO DE SOFTWARE

RUSSAS

2022

ISABELA CARVALHO LOPES SILVA

ANÁLISE COMPARATIVA ENTRE A ABORDAGEM ÁGIL E TRADICIONAL NA ENGENHARIA DE REQUISITOS: UM ESTUDO DE CASO EM PROJETOS DE DESENVOLVIMENTO DE SOFTWARE

Trabalho de Conclusão de Curso apresentado ao Curso de Engenharia de Software do Campus Russas da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Engenharia de Software.

Orientadora: Profa. Dra Patrícia Freitas Campos de Vasconcelos.

RUSSAS

2022

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Sistema de Bibliotecas
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

C323a Carvalho Lopes Silva, Isabela.
Análise Comparativa Entre A Abordagem Ágil E Tradicional Na Engenharia De Requisitos : Um Estudo De Caso Em Projetos De Desenvolvimento De Software / Isabela Carvalho Lopes Silva. – 2022.
73 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Russas, Curso de Engenharia de Software, Russas, 2022.
Orientação: Profa. Dra. Patricia Freitas Campos de Vasconcelos..

1. Elicitação de Documentação de Requisitos. 2. Análise Comparativa. 3. Metodologias Tradicionais. 4. Métodos Ágeis. 5. SCRUM. I. Titulo.

CDD 005.1

ISABELA CARVALHO LOPES SILVA

ANÁLISE COMPARATIVA ENTRE A ABORDAGEM ÁGIL E TRADICIONAL NA ENGENHARIA DE REQUISITOS: UM ESTUDO DE CASO EM PROJETOS DE DESENVOLVIMENTO DE SOFTWARE

Trabalho de Conclusão de Curso apresentado ao Curso de Engenharia de Software do Campus Russas da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Engenharia de Software.

Aprovada em: 11/07/2022.

BANCA EXAMINADORA

_____

Profª. Dra. Patrícia Freitas Campos de Vasconcelos (Orientador)
Universidade Federal do Ceará (UFC)

_____

Profª. Dra. Marília Soares Mendes
Universidade Federal do Ceará (UFC)

_____

Prof. Ms. Pitágoras Graça Martins
Universidade Federal do Ceará (UFC)

A Deus.

A todos aqueles que participaram desta jornada acadêmica, apoiando-me e fazendo-me acreditar que esta conquista seria possível.

AGRADECIMENTOS

Primeiramente, agradecer a Deus pela minha vida, e por me permitir ultrapassar todos os obstáculos encontrados ao longo da jornada universitária.

Dedico este projeto de pesquisa ao meu irmão Manoel Lopes *(in memoriam)*, meu maior parceiro e incentivador que sempre celebrou minhas conquistas bem mais do que eu desde o início.

À minha mãe, Isabel, agradeço por ser minha maior motivação e por nunca me permitir desistir, por nunca me limitar e por sempre acreditar no meu melhor.

Aos meus familiares e irmãos, Moacir, Maria Luiza, Markelly, Moacir Júnior, Renato, Rochellon, Cesarina, Luiza e Salomé, que me apoiaram desde o meu ingresso na educação, incentivaram nos momentos difíceis e agradeço por serem meu combustível para almejar mais do que acho que consigo.

Ao meu namorado e companheiro, Ícaro Azrael, por fazer parte da minha vida, compartilhando felicidades e angústias, além de ser meu psicólogo particular e suportar os surtos diários.

Aos amigos, Letícia Freitas, Marina Rocha, Maria Victoria, Gesly Maia, Joyce Simões, Gabriela Leal, Lucas Alves e Tiago Moreira, que sempre estiveram ao meu lado, pela amizade incondicional e pelo apoio demonstrado ao longo de todo o período acadêmico.

À professora orientadora, Patrícia Vasconcelos por ter desempenhado com excelência tal função com dedicação, paciência, amor e amizade.

Aos professores, Marília Mendes, Osvaldo Mesquita e Pitágoras Martins, pelas correções e ensinamentos que me permitiram apresentar um melhor desempenho no meu processo de formação profissional ao longo do curso.

A todos aqueles que contribuíram, de alguma forma, para a realização deste trabalho.

A todos que participaram, direta ou indiretamente do desenvolvimento deste trabalho de pesquisa, enriquecendo o meu processo de aprendizado.

Às pessoas com quem convivi ao longo desses anos de curso, que me incentivaram e que certamente tiveram impacto na minha formação acadêmica.

“A persistência é o caminho do êxito”

(Charles Chaplin)

RESUMO

Os métodos ágeis estão cada vez mais populares nas organizações, uma vez que eles oferecem maior produtividade, um custo menor, melhor qualidade e satisfação do cliente. Como estes são objetivos comuns a muitas organizações, muitas empresas têm adotado estes métodos. Sobre outra perspectiva, existem ainda as metodologias tradicionais, conhecidas também como orientadas a planejamentos, que devem ser aplicadas apenas em situações em que os requisitos do sistema são estáveis e requisitos futuros são previsíveis. À vista disso, o objetivo geral desta pesquisa é apresentar uma análise comparativa de execução de um processo de elicitação de requisitos entre uma abordagem prescritiva (utilizada por um projeto de extensão de uma instituição de ensino) e uma abordagem ágil com SCRUM (utilizada por uma empresa de desenvolvimento de software). Além disso, pretende-se investigar como as práticas, princípios e valores dos Métodos Ágeis são utilizados para as atividades de elicitação e documentação de requisitos.

Palavras-chave: SCRUM; análise comparativa; metodologias tradicionais; elicitação e documentação de requisitos.

ABSTRACT

Agile methods are increasingly popular in organizations as they offer higher productivity, lower cost, better quality and customer satisfaction. As these are common goals for many organizations, many companies have adopted these methods. From another perspective, there are still traditional methodologies, also known as planning-oriented, which should be applied only in situations where system requirements are stable and future requirements are predictable. In view of this, the general objective of this research is to present a comparative analysis of the execution of a requirements elicitation process between a prescriptive approach (used in a university extension project) and an agile approach with SCRUM (used in a software development). Furthermore, it is intended to investigate how the practices, principles and values of Agile Methods are used for requirements elicitation and documentation activities.

Keywords: SCRUM; comparative analysis; traditional methodologies; requirements elicitation and documentation.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

ARAnalista de Requisitos

EREngenharia de Requisitos

FDDFeature Driven Development

IAInteligência Artificial

PBProduct Backlog

POProduct Owner

XPExtreme Programming

SUMÁRIO

1 INTRODUÇÃO

Os métodos ágeis estão cada vez mais populares nas organizações, uma vez que eles oferecem maior produtividade, um custo menor, melhor qualidade e satisfação do cliente. Como estes são objetivos comuns a muitas organizações, muitas empresas têm adotado estes métodos. (Barbosa; Fraga, 2017, p.1) Entretanto, segundo Barbosa e Fraga (2017), apesar de trazerem benefícios, a mudança do desenvolvimento tradicional para o ágil não é simples e nem rápida. Quando ocorre a transformação de um processo de desenvolvimento tradicional para o ágil, mudanças técnicas, gerenciais, pessoais e culturais são necessárias, trazendo várias barreiras a serem vencidas. (Barbosa; Fraga, 2017, p.1)

Barbosa e Fraga (2017), destacam que a metodologia ágil vem ganhando destaque na indústria de software, principalmente por promover uma intensa produtividade do time, reduzindo custos e entregando valor ao cliente. Tendo isso em vista, muitas empresas que atuam na área de desenvolvimento de software estão sentindo a necessidade de iniciar a substituição de processos tradicionais para realizar a implantação de métodos ágeis.

A abordagem ágil tem sido apontada como uma alternativa às abordagens tradicionais para o desenvolvimento de software. A abordagem tradicional, também conhecida como pesada ou orientada ao planejamento, deve ser aplicada apenas em situações em que os requisitos do sistema são estáveis e previsíveis ao longo do tempo. Entretanto, em projetos em que há muitas mudanças, em que os requisitos são passíveis de alterações, onde refazer partes do código não é uma atividade que apresenta alto custo, as equipes são pequenas, as datas de entrega do software são curtas e o desenvolvimento rápido é fundamental, não pode haver requisitos estáticos, necessitando, então, de uma abordagem ágil de desenvolvimento. (Soares, 2004, p.1)

Além disso, o ambiente das organizações é dinâmico, não permitindo que os requisitos sejam estáticos. Soares (2004) afirma que os métodos tradicionais devem ser utilizados quando se há requisitos estáveis, ou seja, que não são alterados ou modificados com frequência, sua alteração é algo excepcional. Em paralelo a isso tem-se a abordagem ágil que trata os requisitos como passíveis de mudança, que são os mais comuns quando se fala de desenvolvimento de software.

Processos orientados a documentação para o desenvolvimento de software são, de certa forma, fatores limitadores aos desenvolvedores e muitas organizações não possuem recursos ou inclinação para processos pesados de produção de software. Por esta razão, as organizações pequenas acabam por não usar nenhum processo. Isto pode levar a efeitos desastrosos na qualidade do produto final, além de dificultar a entrega do software nos prazos

e custos pré-definidos. O déficit que existe na documentação tradicional e a inexistência desta pode resultar em um possível retrabalho enorme impactando diretamente a entrega final (Soares, 2004).

O manifesto de desenvolvimento ágil de software é apontado como sendo o guia para todos os métodos de desenvolvimento considerados ágeis. (Calazans *et al*., 2018) Esse manifesto foi lançado em 2001 com o intuito de discutir ideias e procurar alternativas aos processos burocráticos e às práticas adotadas nas abordagens tradicionais de Engenharia de Software e Gerência de Projetos. O manifesto define os seguintes valores a serem utilizados por todos os Métodos Ágeis:

- Indivíduos e interações são mais importantes que processos e ferramentas;
- Software funcionando é mais importante que documentação completa e detalhada;
- Colaboração com o cliente é mais importante que negociação de contratos;
- Adaptação às mudanças é mais importante que seguir um plano. (Calazans et al., 2018)

Considerando o contexto abordado por Calazans et al. (2018), surge a necessidade de perscrutar esses estudos de modo a identificar quais técnicas citadas pela literatura possuem maior aplicabilidade na indústria e quais as formas utilizadas pela indústria para documentar os requisitos nos modelos ágeis e tradicionais. O foco deste estudo está nas atividades de elicitação e na documentação do desenvolvimento de requisitos, realizando assim, uma comparação da aplicação e eficácia dos métodos tradicionais versus métodos ágeis.

Especificamente, será feita uma análise comparativa entre as abordagens de elicitação e documentação de requisitos adotadas por uma empresa que utiliza o modelo Cascata, como representante dos métodos tradicionais, e um projeto que adota o framework ágil SCRUM. A empresa que segue o modelo Cascata trabalha com um processo sequencial e bem estruturado, enquanto o projeto em análise, inserido em um ambiente de desenvolvimento ágil, adota práticas mais dinâmicas, adaptativas e colaborativas. A comparação visa avaliar a eficácia e a aplicabilidade de cada abordagem na documentação de requisitos.

Conforme os autores Sutherland e Schwaber (2013) pontuam, o SCRUM aplica uma abordagem iterativa e incremental para otimizar a previsibilidade e controlar o risco. Esse método permite o engajamento de grupos de pessoas que coletivamente têm todas as habilidades e as experiências necessárias para executar o trabalho e para compartilhar ou adquirir essas habilidades conforme o necessário. A pesquisa em questão prioriza o

detalhamento de diversos modelos de desenvolvimento, como o clássico modelo Cascata (detalhado no capítulo 3.1.2), e metodologias ágeis, incluindo o SCRUM, que serão abordados no capítulo 3.2.1, devido à relevância de cada um para os projetos inseridos no estudo de caso.

Por conseguinte, o trabalho está organizado da seguinte forma: Capítulo 2 expõe o objetivo geral e os objetivos específicos. Capítulo 3 apresenta a fundamentação teórica e os conceitos abordados na pesquisa. O Capítulo 4 aborda os trabalhos relacionados com esta pesquisa. O Capítulo 5 define o procedimento metodológico a ser adotado nesta pesquisa científica, explicando cada etapa. O capítulo 6 refere-se à aplicação do processo metodológico. O capítulo 7 ressalta os resultados obtidos durante a pesquisa. O capítulo 8 lista as referências utilizadas neste estudo. O capítulo 9 pontua as conclusões do estudo e possíveis trabalhos futuros. E, por fim, o capítulo 10 exibe o apêndice que contém o checklist aplicado neste estudo.

2 OBJETIVOS

2.1 Objetivo geral

O objetivo geral é apresentar uma análise comparativa de execução de um processo de elicitação de requisitos entre uma abordagem prescritiva (utilizada em um projeto de uma instituição de ensino) e uma abordagem ágil com SCRUM (utilizada em uma empresa de desenvolvimento de software).

2.2 Objetivos específicos

São objetivos específicos deste trabalho:

- Identificar técnicas e práticas de elicitação e de especificação/documentação de requisitos utilizados na Abordagem Tradicional;
- Identificar as técnicas e práticas de elicitação de especificação/documentação de requisitos utilizados pelos Abordagem Ágil;
- Identificar as técnicas e práticas de elicitação e de especificação mais utilizadas no ambiente de uma empresa que utiliza o SCRUM;
- Comparação de resultados obtidos no projeto que utiliza a metodologia Cascata com o projeto que segue a abordagem SCRUM ao fim do ciclo dos projetos.

3 FUNDAMENTAÇÃO TEÓRICA

Nesta seção serão apresentados os conceitos utilizados como base nesta pesquisa e necessários para o entendimento dos termos usados ao decorrer deste trabalho.

3.1 Métodos Tradicionais

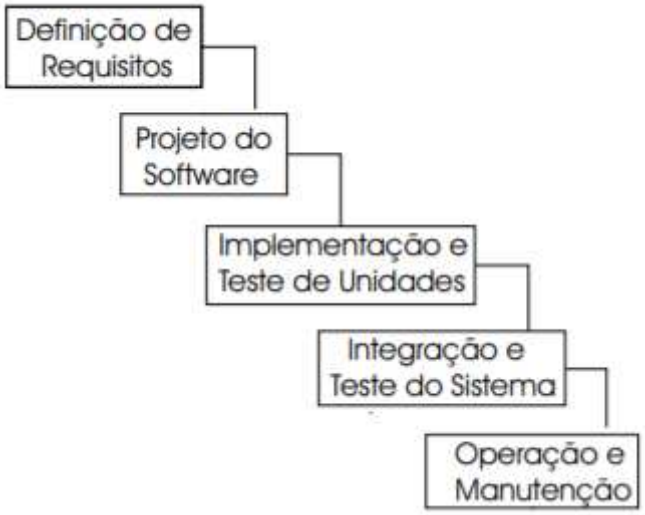
As metodologias tradicionais são também chamadas de pesadas ou orientadas a documentação. Essas metodologias surgiram em um contexto de desenvolvimento de software muito diferente do atual, baseado apenas em um mainframe e terminais burros. (Royce, 1970). Na época, o custo de fazer alterações e correções era muito alto, uma vez que o acesso aos computadores eram limitados e não existiam modernas ferramentas de apoio ao desenvolvimento do software, como depuradores e analisadores de código. Por isso o software era todo planejado e documentado antes de ser implementado. A principal metodologia tradicional e ainda utilizada até hoje é o modelo Clássico. (SOARES, 2004).

Neste trabalho será abordado os modelos prescritivos comumente utilizados, como: Modelo Clássico, Cascata com Subprojetos e Modelo V, que serão melhor detalhados nas seções 3.1.1, 3.1.2 e 3.1.3.

3.1.1 Modelo Clássico

Conforme aborda Pressman (2001), o modelo Clássico ou Sequencial (ou ainda cascata) foi o primeiro processo publicado de desenvolvimento de software. Desde sua introdução tem sido muito utilizado. É um modelo em que existe uma sequência a ser seguida de uma etapa a outra. Cada etapa tem associada ao seu término uma documentação padrão que deve ser aprovada para que se inicie a etapa imediatamente posterior. De uma forma geral fazem parte do modelo Clássico as etapas de definição de requisitos, projeto do software, implementação e teste unitário, integração e teste do sistema, operação e manutenção. O problema do modelo em Cascata é sua inflexível divisão do projeto em fases distintas, o que dificulta possíveis alterações que são comuns no desenvolvimento de um projeto. É um modelo que deve ser usado somente quando os requisitos forem bem compreendidos. A Figura 2 ilustra graficamente o modelo Clássico.

Figura 2: Modelo Clássico

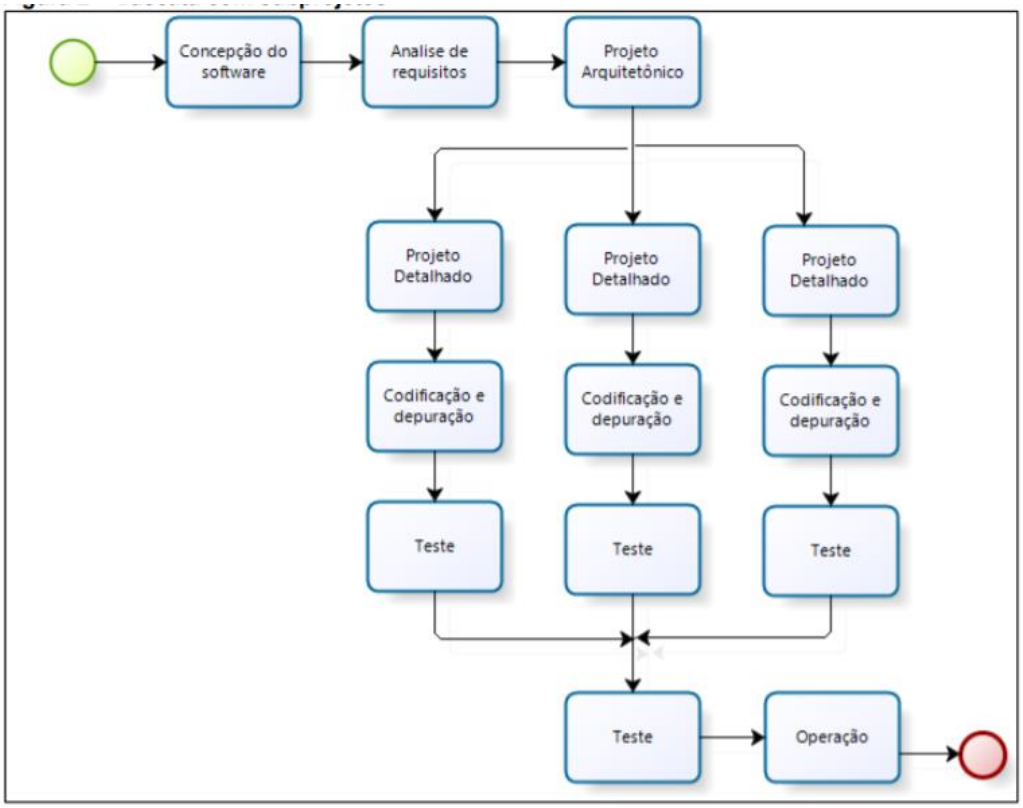


Fonte: Soares (2014).

3.1.2 *Cascata com subprojetos*

Neste modelo é permitido que na fase de projetos exista a divisão de tarefas para que as mesmas sejam executadas paralelamente. Deste modo, ocorre a modularização do sistema para que sejam entregues à medida que são desenvolvidas. Visto que o projeto é dividido em subprojetos, o mesmo propicia um gerenciamento mais fácil, desenvolvimentos rápidos, além de permitir ao usuário visualizar o progresso mais facilmente, pois partes funcionais do sistema são entregues de acordo com que são desenvolvidas (WAZLAWICK, 2013). Os problemas do modelo cascata com subprojetos ocorrem quando há interdependências imprevistas entre os subsistemas, quando há inconsistências que prejudiquem o projeto em uma versão final ou na integração final de todos os subsistemas. Segundo Wazlawick (2013), os principais problemas deste modelo derivam de uma arquitetura falha seguida de um gerenciamento inadequado. A Figura 3 apresenta o modelo cascata com subprojetos.

Figura 3: Modelo cascata com subprojetos



Fonte: :Wazlawick (2013).

Baseado também, em desenvolvimento top-down, a única diferença deste modelo para o modelo cascata padrão é a segmentação da fase de projetos em uma quantidade de etapas significativas.

A fase de projetos pode ser segmentada em várias partes, as quais tendem a ser executadas em paralelo. Cada segmentação se transforma em um micro projeto. Após a finalização de cada miniprojeto, existe a fase para unir todos os módulos que foram implementados de forma separada. Em outras palavras, é a adaptação de todos os segmentos. (STANKIEWICZ, 2017).

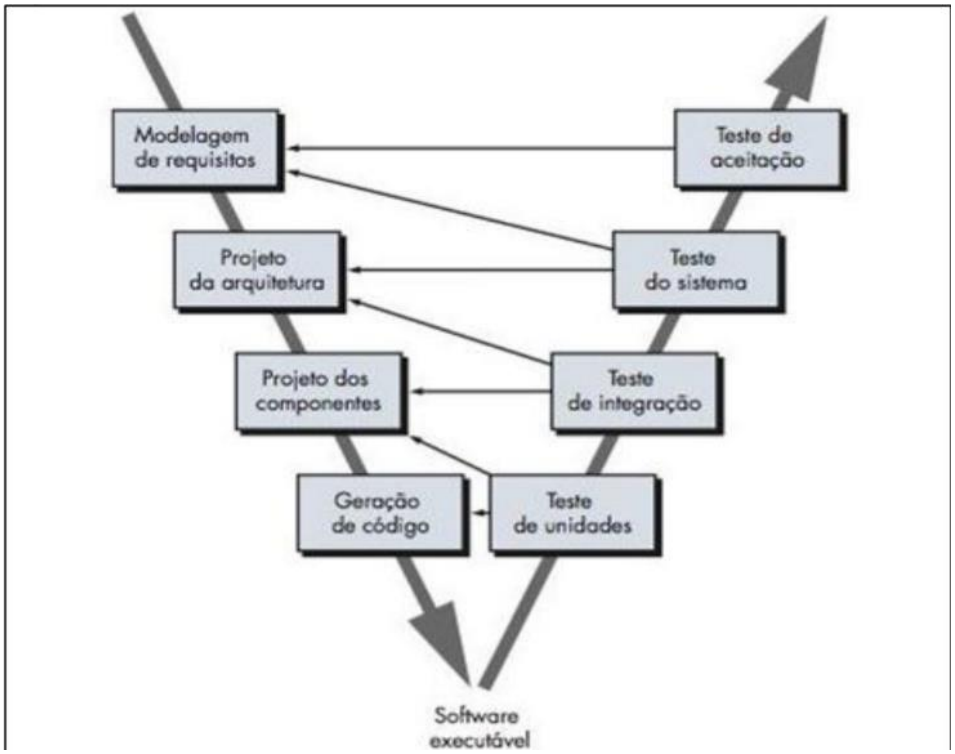
3.1.3 *Modelo V*

O Modelo V é uma adaptação do modelo cascata, porém ele determina uma fase de verificação e validação para cada etapa de construção. Pelo fato deste modelo ser sequencial,

como no modelo cascata, há a necessidade dos requisitos serem estáveis e o domínio reconhecido (LENZ; MOELLER, 2004).

A Figura 4 ilustra o Modelo V, este é formado pelas fases: modelagem de requisitos, projeto da arquitetura, projeto dos componentes, geração de código, teste de unidade, teste de integração, teste do sistema e teste de aceitação (PRESSMAN, 2011).

Figura 4: Modelo V



Fonte: Pressman (2010)

Segundo Lenz e Moeller (2004), o desenvolvimento utilizando o Modelo V, inicia na fase de modelagem de requisitos. Nesta são determinados os requisitos funcionais da aplicação. A próxima fase é o projeto da arquitetura em que é escolhida a arquitetura do software que será implementada. Esta arquitetura considera os requisitos coletados na primeira etapa. A terceira e quarta fases projetam os componentes do sistema e se inicia a implementação das funcionalidades. Ao fim da quarta etapa, iniciam-se as fases de testes.

O primeiro teste a ser realizado é o de unidade, que visa encontrar problemas por módulo do sistema. Caso isso ocorra, pode-se voltar às fases de projeto dos componentes ou a geração

de código. Seguindo ao próximo teste, teste de integração, são realizadas avaliações das interfaces entre os componentes. Uma vez encontrado algum problema, pode-se retornar às fases de projeto dos componentes e projeto da arquitetura. O teste do sistema verifica o comportamento do software em funcionamento. É focado nos requisitos funcionais e não funcionais da aplicação. Neste teste, caso identificado algum problema, pode-se retornar às fases de projeto da arquitetura e modelagem de requisitos. O último teste, teste de aceitação, é realizado pelo cliente. Nesta etapa foca-se na satisfação das necessidades do usuário e, caso não satisfaça, deve-se retornar à fase de modelagem de requisitos. (STANKIEWICZ, 2017).

3.2 Métodos Ágeis

De acordo com Calazans et al. (2018), para a criação de um produto de software é necessário a interação dos vários fatores, como processos e pessoas. A dinâmica de mercado atual impõe agilidade na entrega de produtos e atendimento, cada vez mais satisfatório, ao que foi demandado pelo cliente. Os Métodos Ágeis representam uma evolução diante das metodologias tradicionais porque valorizam a interação entre pessoas, a comunicação com o cliente e priorizam as atividades que efetivamente agregam valor. Encontram-se na literatura, várias propostas de métodos ágeis de desenvolvimento de sistemas, alguns descritos sucintamente, a seguir.

Como aborda Calazans et al. (2018), o propósito da Programação Extrema (Extreme Programming - XP) é garantir o sucesso no desenvolvimento de softwares, especialmente para equipes pequenas e médias envolvidas com projetos, nos quais os requisitos são desconhecidos e voláteis. O XP foi formalizado por meio de quatro princípios que são: Comunicação, Simplicidade, Feedback e Coragem. É composto de doze práticas: planejamento das iterações, incrementos curtos e pequenos, uso da metáfora para facilitar a comunicação, projeto simples ou incremental, desenvolvimento orientado a testes, reestruturação constante do código, programação em pares, propriedade coletiva, integração contínua, ritmo sustentável, cliente no local e padrão de codificação.

Já o Scrum como retrata Calazans et al. (2018), se baseia em algumas características: flexibilidade dos resultados e dos prazos, times pequenos, revisões frequentes e colaboração. Sua equipe é composta por três principais papéis: Product Owner (PO), Scrum Master e Time Scrum. Outra característica da metodologia Scrum é a existência de três principais artefatos produzidos e acessíveis por toda a equipe: Product Backlog (uma lista, ordenada por prioridades, com todas as necessidades e funcionalidades a serem desenvolvidas, custos

estimados e as datas de entrega); Sprint Backlog (subconjunto do Product Backlog, o qual divide as atividades por períodos de desenvolvimento chamados Sprint's; Gráfico Sprint Burndown (representação gráfica do trabalho já realizado e o que resta a ser realizado.

O método Scrum tem as seguintes fases: reunião de planejamento (seleção do Backlog e definição das estimativas), planejamento da sprint (backlog da sprint com atividades ou tarefas), sprint de desenvolvimento (análise, projeto, evolução teste, entrega) e reunião de revisão do sprint e retrospectiva.

A literatura apresenta outros métodos, tais como: Crystal, ASD (Desenvolvimento ágil de software), DSDM (Metodologia de Desenvolvimento de Sistemas Dinâmicos), FDD (Desenvolvimento Guiado por Funcionalidades) etc. (Calazans et al., 2018).

Embora exista uma ampla variedade de metodologias ágeis como já mencionadas: XP, Scrum, ASD, DSDM e FDD, a pesquisa em questão terá enfoque na metodologia Scrum devido ao fato de que os projetos inseridos no estudo de caso utilizam essa abordagem.

3.2.1 SCRUM

O Scrum é fundamentado em estudos que afirmam que o conhecimento provém da experiência e da tomada de decisões baseadas no que é conhecido, sustentando-se, portanto, em três pilares: transparência, inspeção e adaptação.

No framework Scrum, pequenos times possuem papéis e responsabilidades e, juntos, realizam eventos de duração fixa (ciclos iterativos). Para realização desses eventos utilizam artefatos específicos e aplicam regras que integram os papéis e artefatos, administrando assim, as relações e interações entre eles (CRUZ, 2013).

No início do projeto é feita uma reunião de planejamento com desenvolvedores, clientes, parceiros e demais envolvidos, para que seja definido o Backlog do Produto, principal artefato do Scrum. Ele lista todas as características, funções, requisitos e funcionalidades do produto a ser entregue, ordenados de acordo com sua prioridade, que provavelmente serão desenvolvidos no projeto. Vale ressaltar, que esta listagem deve ser dinâmica e atualizada sempre que houver mudanças nos requisitos de negócio, condições de mercado ou tecnologia, tornando assim o produto mais apropriado e competitivo (SUTHERLAND e SCHWABER, 2013).

O responsável pelo gerenciamento Backlog do Produto, incluindo seu conteúdo, disponibilidade e ordenação é Product Owner ou Dono do Produto. Ele trabalha diretamente com os clientes para que as especificações sejam claras o suficiente e deve garantir que o Time de Desenvolvimento entenda todos os requisitos, além de assegurar que ele esteja visível a

todos. Ou seja, este membro tem um amplo conhecimento acerca do negócio do produto e deve garantir a entrega de um produto que agregue valor ao cliente (CRUZ, 2013).

A transformação do Backlog do Produto em incrementos de funcionalidades é de responsabilidade do time de desenvolvedores, que deve possuir habilidades necessárias, enquanto equipe, para transformar os requisitos em um produto utilizável. Os times de desenvolvimento organizam e gerenciam seu próprio trabalho, para atingir a meta estabelecida pelo Product Owner.

Um dos desenvolvedores é eleito Scrum Master, pessoa responsável por garantir que a metodologia seja entendida e aplicada e que a equipe esteja aderindo aos valores, práticas e regras do Scrum. Ele deve atuar na remoção de obstáculos que inviabilizam o trabalho dos demais membros da equipe, além de ajudar o time a entender e usar o autogerenciamento e a interdisciplinaridade (RISING e JANOFF, 2000).

O Scrum tem seu progresso baseado em uma série de iterações bem definidas, chamadas Sprints, no qual são implementados os itens definidos no Backlog do Produto, resultando em um produto que gere valor tangível para o cliente. Normalmente, os Sprints têm duração de um mês ou menos e não sofrem alterações que possam colocar em perigo o objetivo do Sprint (SUTHERLAND e SCHWABER, 2013).

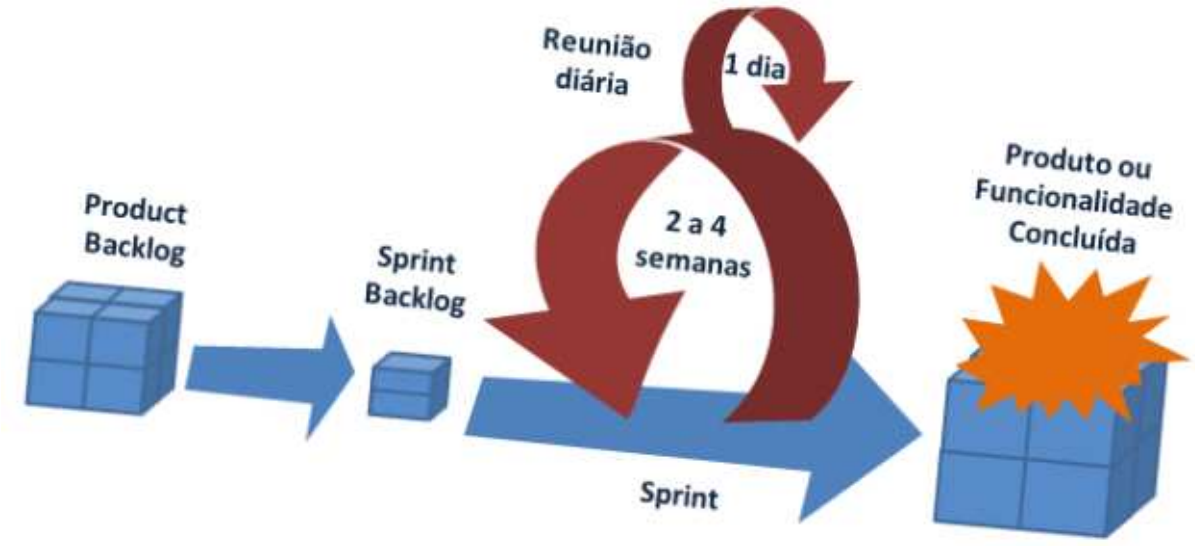
Na reunião de planejamento do Sprint, são avaliados quais pontos do Backlog de Produto serão atendidos, bem como capacidade da equipe e tarefa a ser desenvolvida por cada membro. Esta reunião resulta no chamado Sprint Backlog, uma lista de tarefas específicas a serem executadas no Sprint (CRUZ, 2013). Dentro de cada Sprint ocorrem reuniões diárias (Daily Scrum), de quinze minutos a meia hora, que visa analisar os resultados das tarefas do dia anterior e definir quais tarefas serão executadas no decorrer do dia.

Cabe aos Scrum Master levantar, priorizar e tratar os obstáculos, caso eles existam. Segundo, Sutherland e Schwaber, a reunião diária é a chave para inspeção e adaptação, uma vez que melhoram a comunicação, eliminam outras reuniões, identificam e removem impedimentos para o desenvolvimento, destacam e promovem rápidas tomadas de decisão, e melhoram o nível de conhecimento do time de desenvolvimento (SUTHERLAND e SCHWABER, 2013).

Ao final de um Sprint, são realizadas duas reuniões: a revisão do Sprint (Sprint Review) e a retrospectiva do Sprint (Sprint Retrospective). Na revisão do Sprint, o Product Owner juntamente com o time, valida todos os itens entregues e avalia se o objetivo foi atingido. Já na reunião de retrospectiva do Sprint são discutidos os erros, acertos, fazendo os devidos ajustes para os Sprints seguintes, proporcionando um ambiente de melhoria contínua nos projetos e nos

times (BROD, 2013). Vale ressaltar que, os papéis, artefatos e regras do Scrum são imutáveis e que, portanto, devem ser aplicados em sua totalidade para que o resultado final seja a abordagem ágil Scrum. A Figura 1 ilustra o ciclo de desenvolvimento do Scrum de forma simplificada.

Figura 1: Ciclo de desenvolvimento SCRUM



Fonte: Barboza et al. (2016)

3.2.1.1 Histórias de Usuário

A história de usuário é definida por uma ou duas frases do cotidiano ou do negócio do usuário final do sistema, a fim de capturar a essência do que deve ser desenvolvido. Os métodos ágeis utilizam as histórias de usuário para representar a necessidade do cliente. As histórias devem ser simples, objetivas e conter o benefício que será entregue. As histórias de usuário têm o objetivo de ancorar uma discussão futura. Os requisitos são discutidos de forma detalhada antes e/ou durante a implementação. Entretanto, não necessariamente e, às vezes, raramente esses levantamentos são documentados. A comunicação face a face, em alguns projetos,

substitui a necessidade de requisitos definidos e documentados formalmente, salvo se houver uma obrigação para tal. (BARBOSA E FRAGA, 2017)

3.2.2 Extreme Programming - XP

Sommerville (2007) afirma que o XP, é talvez o mais conhecido e mais amplamente usado dos métodos ágeis. Segundo ele, o nome foi criado por Kent Beck, seu criador, porque a abordagem foi desenvolvida pelo avanço de reconhecida boa prática, tal como o desenvolvimento iterativo e o envolvimento do cliente em níveis "extremos". Segundo Teles (2006), o XP, é um processo de desenvolvimento de software voltado para projetos cujos requisitos são vagos e mudam com freqüência, desenvolvimento de sistemas orientados a objeto, equipes pequenas, preferencialmente até 12 desenvolvedores, e desenvolvimento incremental (ou iterativo), onde o sistema começa a ser implementado logo no início do projeto e vai ganhando novas funcionalidades ao longo do tempo. Teles (2006), ainda destaca que o XP assim como os processos ágeis na sua totalidade, compartilham a premissa de que o cliente aprende sobre suas necessidades, na medida em que é capaz de manipular o sistema que está sendo produzido.
Para isso, o XP conta com quatro valores fundamentais sendo eles o Feedback(Quando o cliente aprende com o sistema que utiliza e re-avalia as suas necessidades, ele gera feedback para a equipe de desenvolvimento), a Comunicação (ocorrida entre o cliente e a equipe permitindo que todos os detalhes do projeto sejam tratados com a atenção e a agilidade que merecem), a Simplicidade (Se aplica ao aprender a implementar apenas aquilo que é suficiente para atender a cada necessidade do cliente), e a Coragem (A equipe precisa ser corajosa e acreditar que, utilizando as práticas e valores do XP, será capaz de fazer o software evoluir com segurança e agilidade) como cita Teles(2006).
Estes valores são importantes para manter o foco das equipes que usam o XP em um processo diferente do convencional, com maior dinamismo e capacidade de resposta.
Teles (2006) traz em sua obra 13 (treze) importantes práticas que compõem o XP. Estas são apresentadas no Quadro 1 a seguir.

Quadro 1: Práticas do XP

Prática do XP	**Descrição**
Cliente Presente	O XP trabalha com a premissa de que o cliente deve conduzir o desenvolvimento a partir do feedback que recebe do sistema.
Jogo do Planejamento	No início de cada iteração ocorre o jogo do planejamento. Trata-se de uma

	reunião onde o cliente avalia as funcionalidades que serão implementadas.
Stand Up Meeting	A equipe se reúne a cada manhã para avaliar o trabalho que foi executado no dia anterior e priorizar aquilo que será implementado no dia que se inicia.
Programação em Par	Os desenvolvedores implementam as funcionalidades em pares, ou seja, diante de cada computador, existem sempre dois desenvolvedores que trabalham juntos para produzir o mesmo código.
Desenvolvimento Guiado p/ Testes	Testes são escritos para cada funcionalidade antes de codificá-las. Fazendo isso, eles aprofundam o entendimento das necessidades do cliente.
Refactoring	O refactoring é o ato de alterar um código sem afetar a funcionalidade que ele implementa. O objetivo é tornar o software mais simples de ser mantido.
Código Coletivo	Os desenvolvedores têm acesso a todas as partes do código e podem alterar aquilo que julgarem importante sem pedir autorização de outra pessoa.
Código Padronizado	Para facilitar a manutenção no código por parte de toda a equipe, padrões de codificação são definidos tornando o sistema mais homogêneo e permitir que qualquer membro da equipe tenha condições de dar manutenção no sistema.
Design Simples	Para que o cliente possa obter feedback logo, a equipe precisa ser ágil no desenvolvimento, o que a leva a optar pela simplicidade do design.
Metáfora	Para facilitar a criação de um design simples, a equipe de desenvolvimento utiliza metáforas, já que elas têm o poder de transmitir ideias complexas de forma simples.
Ritmo Sustentável	Para garantir que a equipe tenha sempre o máximo de rendimento e produza software com melhor qualidade possível, o XP recomenda que os desenvolvedores trabalhem apenas oito horas por dia e evitem fazer horas-extras, visto que é essencial estar descansado a cada manhã, de modo a utilizar a mente na sua plenitude.
Integração Contínua	Prática utilizada com o objetivo de checar/testar toda a aplicação, sempre que uma nova funcionalidade é implementada, seja de forma manual, ou automática, forma esta, que se utiliza de ferramentas especializadas para tal.
Releases Curtos	O XP tem como objetivo gerar um fluxo contínuo de valor para o cliente. Sendo assim, ele trabalha com releases curtos, ou seja, a equipe produz um conjunto reduzido de funcionalidades e coloca em produção rapidamente.

Fonte: TELES (2006)

Nunes (2017) pontua que uma particularidade que pode ser observada entre XP e Scrum é que ambos contemplam que a cada iteração uma parte do sistema deve ser entregue funcionando. Onde cada iteração deve ser curta e com prazo definido, além disso, com a participação constante do usuário, para que forneça seu feedback à equipe de desenvolvimento.

3.2.3 Feature Driven Development - FDD

Silva et al. (2011) aponta que Feature-Driven Development (FDD) é uma metodologia de desenvolvimento de software que inclui alguns benefícios de processos rigorosos, como

modelagem, planejamento prévio e controle do projeto, assim como contém características de processos ágeis, como foco na programação, interação constante com o cliente e entrega freqüente de versão do produto. Prevê práticas apenas para o desenvolvimento de software em si não se preocupando com outros fatores como a escolha de tecnologias e ferramentas, a definição de procedimentos de aquisição, dentre outros.

Embora não seja tão orientada à documentação quanto o RUP, em FDD relatórios que controlam o estado e o progresso das atividades são previstos. Silva et el. (2011) também relata que os artefatos principais são o plano de projeto, a lista de funcionalidades e o diagrama de sequência. O plano de projeto é o documento principal de saída a ser aprovado pelo cliente, nele está definido o escopo, a lista de funcionalidades, os riscos, as métricas para controle do projeto, os critérios de aceitação, dentre outras informações pertencentes ao domínio da aplicação. A lista de funcionalidades é usada para planejar, dirigir, rastrear e reportar o progresso do projeto e está organizada hierarquicamente com requisitos funcionais. (SILVA et al., 2011)

São cinco os processos da metodologia ágil FDD: Desenvolver um Modelo Abrangente, Construir uma Lista de Funcionalidades, Planejar Através de Funcionalidades, Projetar Através de Funcionalidades e Construir Através de Funcionalidades.

O processo Desenvolver um Modelo Abrangente é responsável pelo estudo detalhado sobre o domínio do negócio e pela definição do escopo do projeto. Segue-se o Construir uma Lista de Funcionalidades, onde todas as funcionalidades necessárias ao cumprimento das necessidades do cliente são levantadas. Os itens desta lista são ordenados por prioridade de desenvolvimento no processo Planejar através de Funcionalidades, considerando inclusive se a funcionalidade é funcional ou não.

Ao final deste processo é gerada uma lista das classes e estas são associadas aos desenvolvedores responsáveis. Um plano de projeto é elaborado pelo arquiteto chefe e aprovado pelo cliente. Inicia-se então várias iterações que compreende os dois processos finais. (SILVA et al., 2011)

Além disso, durante o processo Projetar através de Funcionalidades, para cada funcionalidade da lista é definida uma atividade a ser realizada. Neste processo, segundo Silva et al. (2011), o modelo da interface do usuário é esboçado e os diagramas de sequência e de classe são gerados. Já no processo Construir através de Funcionalidades o código

é gerado, produzindo-se a cada iteração, para cada funcionalidade definida, uma função que agregue valor ao cliente, este chamado de dono do produto.

3.3 Engenharia de Requisitos

Conforme Barbosa e Fraga (2017) abordam, a Engenharia de Requisitos (ER) é um processo para estabelecer os serviços requeridos pelo cliente a respeito de um sistema. Por meio dela, são identificadas as necessidades e as restrições de desenvolvimento. A ER está preocupada com a identificação, modelagem, comunicação e documentação dos requisitos de acordo com o contexto que o sistema será utilizado.
A comunidade de desenvolvimento ágil sabe pouco sobre os papéis, processos e práticas da ER tradicional, assim ela adota uma abordagem mais flexível e dinâmica de trabalho. O principal objetivo do processo de ER tradicional é criar uma documentação para compartilhar o conhecimento, enquanto no desenvolvimento ágil o foco está em uma comunicação face a face entre o cliente e os times ágeis para atingir o mesmo objetivo. (BARBOSA E FRAGA, 2017).
Apesar da importância da ER no sucesso do desenvolvimento do software e na minimização dos riscos de projeto, essa atividade é vista, algumas vezes, nos métodos ágeis como burocrática, podendo tornar o processo menos ágil. A principal justificativa ocorre devido ao fato de que os requisitos mudam tão rapidamente que um documento de requisitos fica desatualizado tão logo seja redigido, desperdiçando todo esforço empreendido na documentação. As atividades de ER (elicitação, documentação, validação e gerenciamento) não são atividades claramente definidas na ER ágil.

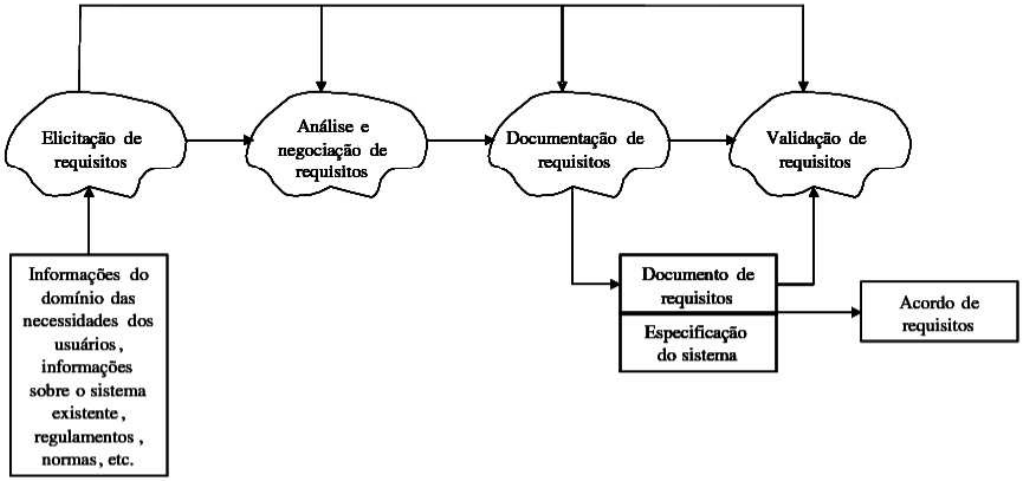
3.3.1 Processo da Engenharia de Requisitos

Um processo de engenharia de requisitos é um conjunto estruturado de atividades que são seguidas para derivar, validar e manter um documento de requisitos de um sistema. Uma descrição completa de um processo deve incluir as atividades que devem ser conduzidas, sua estrutura ou agenda destas atividades, as entradas e saídas de cada atividade e as ferramentas utilizadas para suportar a engenharia de requisitos. (LOPES, 2005)

De acordo com Lopes (2005), não existe um processo único que possa ser utilizado por todas as organizações. Cada organização deve desenvolver seu próprio processo para adaptar-se aos tipos de sistema em desenvolvimento, cultura organizacional, experiência dos envolvidos, entre outros. De forma abstrata, a maioria dos processos de engenharia de requisitos

podem ser descritos em um modelo de atividades de alta granularidade, como apresentado na Figura 5.

Figura 5: Processo da Engenharia de Requisitos



Fonte: Lopes (2005)

3.4 Relação de técnicas de elicitação e documentação de requisitos nos métodos ágeis

O levantamento feito no estudo de Calazans et al. (2018) permitiu identificar as técnicas de elicitação e sua respectiva descrição que são apresentadas no Quadro 2. Da mesma forma, as técnicas de documentação e algumas referências foram identificadas e relacionadas no Quadro 3.

Quadro 2: Relação de técnicas de elicitação de requisitos nos métodos ágeis

Técnica/Referências	**Descrição**
Análise de domínio	O processo de identificação, coleta, organização e representação das informações relevantes é baseado em um domínio. Por exemplo, baseado no estudo de sistemas existentes ou em suas histórias de desenvolvimento.
Análise documental	Estudo e reutilização de documentação de diferentes naturezas (leis, manuais de usuário, relatório de pesquisas de mercado, glossário de termos de negócio) para a identificação de requisitos a serem implementados no sistema.
Brainstorming	Técnica usada para gerar novas ideias e encontrar a solução para uma questão específica, além de promover o pensamento criativo
Brainwriting	É a técnica onde um grupo de pessoas registram por escrito possíveis formas de como resolver um problema, desenvolver um projeto ou melhorar uma situação existente. Também conhecida como Método 6-3- 5.
Cenários (Scenarios)	Trata-se de uma abordagem informal, prática e aplicável a qualquer tipo de sistema. Scenarios são exemplos reais de como um sistema pode ser usado, o

	que geralmente proporciona uma melhor compreensão dos stakeholders.
Entrevistas	Colóquio entre pessoas em local combinado, para obtenção de esclarecimentos, avaliações ou opiniões.
Features	São declarações de alto nível, uma vez que não possuem detalhes necessários para a implementação, derivadas das necessidades dos stakeholders. Para cada feature identificada, temos um ou mais requisitos de software.
Grupo focal	São debates ou entrevistas com um pequeno grupo selecionado de indivíduos, que discute um tópico específico, pré-definido e limitado, sob a orientação de um facilitador ou moderador.
Mapa mental	É uma técnica usada para acelerar a aquisição de conhecimentos, baseado em imagens e textos e seus relacionamentos.
Observações	É uma técnica que acompanha e relata as coisas, os seres, os eventos a serem analisados.
Personas	É uma técnica que consiste na criação de perfis e personificação de grupos de usuários. Representa uma caracterização de um personagem que, embora seja fictício, expõe as características importantes da população de usuários.
Prototipação	Representa o produto real tanto no sentido funcional quanto no sentido gráfico. Trata-se de uma versão inicial do sistema, baseada em requisitos ainda pouco definidos
Questionários	São utilizados quando há a necessidade de coletar as mesmas informações de muitos usuários e ao mesmo tempo.
Refining meetings	Prática que permite o refinamento dos requisitos ou user stories em uma reunião estruturada.
Reuniões	Trata-se de uma técnica de elicitação em grupo usada em uma reunião estruturada. Devem fazer parte do grupo uma equipe de analistas e os stakeholders.
Sessões de JAD	Entrevista de grupo com líder imparcial. É uma técnica que visa reunir autoridades representativas e gerenciais em um workshop organizado para a elicitação de requisitos ou para promover decisões.
UI Mockups	É uma representação estática de média a alta fidelidade de um design. Muitas vezes um mockup é um rascunho bem próximo do design final do produto, ou até o próprio design visual do produto final
User Story	É um requisito de sistema de software formulado como uma ou duas frases na linguagem cotidiana do usuário.
Workshops	Técnica de elicitação em grupo realizada em uma reunião estruturada com apresentação por parte de uma ou mais equipes.

Fonte: Calazans et al. (2018).

Quadro 3: Relação de técnicas de documentação de requisitos nos métodos ágeis

Técnica	**Descrição**
Cenários (Scenarios)	Trata-se de uma abordagem informal, prática e aplicável a qualquer tipo de sistema. Scenarios são exemplos reais de como um sistema pode ser usado, o que geralmente proporciona uma melhor compreensão dos stakeholders.
Data models	É um subconjunto do modelo de implementação que descreve a representação lógica e física dos dados persistentes no sistema. Pode se valer da Linguagem Natural (técnica extremamente flexível e expressiva, não adota nenhuma estrutura e semântica pré-definida), de uma Notação Semiformal (possui alguma estrutura e semântica) ou de uma Notação Formal (que utiliza semântica precisa e detalhamento completo de modelos).
Features	São declarações de alto nível, uma vez que não possuem detalhes necessários para a implementação, derivadas das necessidades dos stakeholders. Para cada feature identificada, temos um ou mais requisitos de software.
Goal Preference Model	Técnica onde os stakeholders atribuem um valor inteiro para cada meta descrita em uma lista, denominada de lista inicial de metas. Esse valor representa o grau de preferência de prioridade de cada meta.
Modelos de Casos de Uso	É uma técnica que especifica uma sequência de interação entre um sistema e um ator externo incluindo variantes e extensões que o sistema pode executar.
Ontologias	Ontologias permitem uma representação de tais abstrações de forma fidedigna e não ambígua. Dessa forma, podem atuar na complementação a modelos, facilitando o entendimento do domínio.
Personas	É uma técnica de usabilidade, que consiste na criação de perfis e personificação de grupo de usuários, ou seja, representa uma caracterização de um personagem que, embora seja fictício, expõe as características importantes da população de usuários para a qual se destina o produto ou projeto.
Protótipos	Representa o produto real tanto no sentido funcional, quanto no sentido gráfico. Trata-se de uma versão inicial do sistema, baseada em requisitos ainda pouco definidos.
Story card	Tem o objetivo de capturar os requisitos macros, e se concentram em “quem, o quê e porquê de um recurso, e não como”. Story Card deve ser escrito a partir de uma perspectiva do usuário e deve ser utilizada a linguagem do negócio, para que seja compreendido por todos.
Structured requirements list	É a declaração oficial do que é exigido para os desenvolvedores do sistema. Ela deve incluir uma definição das necessidades dos utilizadores e uma especificação dos requisitos do sistema.
Tasks	São unidades menores das histórias de usuário. Durante a o sprint planning, os itens do backlog que serão desenvolvidos na sprint são escolhidos. Em seguida, esses itens são quebrados em unidades menores, o que é denominado de tasks

User stories	É um requisito de sistema de software formulado como uma ou duas frases na linguagem cotidiana do usuário.
Wall	A ideia básica da parede de cartões é de ter o fluxo de trabalho visível na própria parede.
XSBD	O objetivo principal do processo XSBD é que o engenheiro de usabilidade possa desenvolver eficientemente um sistema de software que atenda a usabilidade do projeto e metas de desenvolvimento
XXM (Extreme XMachine)	São utilizados para mostrar um modelo de alto nível da totalidade do sistema. Uma Extreme X-Machine permite modelar as estruturas de controle de uma história e, finalmente, de todo o sistema.

Fonte: Calazans et al. (2018)

O estudo de Lucia e Qusef (2010) realizou uma comparação das atividades da ER tradicional implementadas no método ágil Scrum, resumidas no Quadro 4. As atividades da ER tradicional são de responsabilidade do Analista de Requisitos enquanto no Scrum, os requisitos são definidos e priorizados pelo PO. Os métodos apresentam diferenças na execução de cada atividade, conforme mostrado no Quadro 4. Uma destas diferenças é o fato de os métodos ágeis trabalharem com requisitos descritos e gerenciados por meio de histórias de usuário.

Quadro 4: Reimplementação da Engenharia de Requisitos no Scrum

Atividades da Engenharia de Requisitos (ER)	**ER Tradicional**	**Metodologia SCRUM**
Elicitação de Requisitos	• Analista de Requisitos (AR) levanta os requisitos como cliente.	• Product Owner (PO) cria o Product Backlog (PB); • Alguns stakeholders podem participar da criação do Product Backlog.
Análise de Requisitos	• AR retira as dúvidas com o cliente e analisa a viabilidade dos requisitos.	• Reunião para o refinamento do Backlog; • PO prioriza o PB; • PO analisa a viabilidade dos requisitos.
Documentação de Requisitos	• O AR documenta os requisitos através de casos de uso e diagramas	• Comunicação face a face;
Validação dos Requisitos	• O AR solicita que o	• Reunião de Review é a cerimônia em

	cliente valide o entendimento dos requisitos.	que o projeto é avaliado em relação aos objetivos planejados no início da *sprint.* Nessa reunião inclui o Product Owner, Gerente de Projetos, time de desenvolvedores e cliente que será responsável pela validação.
Gerência dos Requisitos	• Acompanhar desenvolvimento dos requisitos. • Alterações nos requisitos em função de controle rígido de mudanças.	• Reunião de Planejamento da Sprint; • Acompanhar itens do PB; • Mudanças de requisitos são incluídas ou excluídas do PB.

Fonte: Barbosa e Fraga (2017)

Com base nesse contexto, será realizada uma análise comparativa da execução de um processo de elicitação de requisitos entre métodos tradicionais e ágeis, utilizando o modelo Cascata como referência para a abordagem prescritiva e o framework Scrum como referência para a abordagem ágil.

4 TRABALHOS RELACIONADOS

Pesquisas de comparação de métodos ágeis com métodos tradicionais são comuns para analisar a aplicação de técnicas de cada método por meio de um estudo de caso. Nesta seção serão apresentados trabalhos relacionados a este trabalho. Essas pesquisas foram obtidas através de buscas no Google Scholar, utilizando na busca as palavras-chaves: Engenharia de Requisitos, Metodologias Ágeis, Modelo Cascata, SCRUM, Comparação entre Metodologias Ágeis e Tradicionais e Técnicas de Elicitação de Requisitos.

4.1 Pesquisas que abordam as técnicas de elicitação e documentação nos Métodos Ágeis

No trabalho de Calazans et al. (2018), o objetivo do estudo foi identificar as técnicas de elicitação e documentação de requisitos para os métodos ágeis. A partir da revisão da literatura, foi realizado um estudo de caso no setor de desenvolvimento de software de uma Instituição de Ensino Superior (IES) com uso do método *Delphi* com *Q-sort.* O caso analisado foi o do setor de desenvolvimento de uma instituição de ensino, onde trabalham 20 indivíduos com experiência em métodos ágeis. Para análise dos dados utilizou-se o software Minitab e o coeficiente *Kendall's W*. Como grau de concordância dos investigados, foi obtido 0,579 para as técnicas de elicitação e 0,437 para a documentação. As técnicas de elicitação de maior aplicabilidade identificadas foram entrevistas, brainstorming e user stories. Com relação à documentação, destacaram-se as técnicas *tasks, user stories* e protótipos. Os resultados servem de referência para pesquisadores e para equipes que buscam maior produtividade.

No trabalho de Longo e Da Silva (2014), analisa se a utilização de histórias de usuário é suficiente durante a fase de levantamento de requisitos, abordando as diferentes maneiras de descrever, compreender e utilizá-las atualmente. Para ilustrar as fases de desenvolvimento da história de usuário é apresentado um estudo de caso que visou aplicar técnicas deste método comparadas às técnicas de levantamento de requisitos dirigidas por casos de uso.

4.2 Pesquisas sobre a Engenharia de Requisitos em Métodos Ágeis

O trabalho de Barbosa e Fraga (2017), ressalta que por não possuírem características prescritivas, não há, nos métodos ágeis, uma definição formal sobre como as atividades de Engenharia de Requisitos (ER) devem ser executadas. Isso faz com que haja uma variedade de formas de se definir como serão levantados, priorizados, especificados e validados os requisitos. Assim, torna-se necessário compreender como as pesquisas em ER ágil têm caracterizado estas

atividades. O objetivo deste trabalho foi, por meio de uma revisão sistemática da literatura, identificar as práticas e técnicas utilizadas para cada processo da ER em projetos ágeis. O trabalho identificou ainda importantes desafios e lições aprendidas que devem direcionar as evoluções nesta área.

O trabalho de Alves (2015), tem como objetivo investigar como a engenharia de requisitos e as metodologias ágeis vêm sendo utilizadas conjuntamente na prática em projetos de desenvolvimento de software aplicados na indústria. Para isso, foi realizado um mapeamento sistemático da literatura que encontrou 24 estudos primários relevantes, cujos dados foram extraídos e sintetizados. Esse mapeamento identificou as técnicas (Meetings e Brainstorming) e processos (User Stories, Protótipos, Features e Story Card) de engenharia de requisitos que estão sendo mais utilizados no contexto de desenvolvimento ágil e quais os principais problemas e limitações encontradas. Após a execução do mapeamento, verificou-se que a falta de envolvimento do usuário associada às características das atuais técnicas utilizadas para especificar requisitos e suas constantes mudanças são os principais desafios a serem superados.

4.3 Pesquisas que propõe uma análise comparativa entre as abordagens ágeis e tradicionais

No trabalho de Soares (2004), é feita uma comparação entre as metodologias tradicionais para desenvolvimento de software e as metodologias ágeis. Em particular é feita a comparação da Extreme Programming (XP), uma metodologia ágil muito usada, e o modelo Clássico ou Sequencial. As práticas da XP são apresentadas, enfatizando que suas características são ideais para projetos que devem ter um desenvolvimento rápido e que podem ter requisitos alterados constantemente pelos clientes.

No trabalho de Barboza et al. (2016), tem por objetivo realizar a verificação prática dos benefícios, limitações e impactos decorrentes da implantação do método ágil Scrum quando comparado ao gerenciamento tradicional de projetos, por meio de um estudo de caso em uma empresa do setor industrial. Com o intuito de captar e explorar a perspectiva dos envolvidos nos projetos, bem como desenvolver novas compreensões sobre a magnitude dos aspectos observados, desenvolveu-se uma pesquisa qualitativa e descritiva, cuja finalidade é observar, registrar e analisar os fenômenos ocorridos. Os resultados alcançados indicam que todas as abordagens de gestão podem ser eficientes. O desenvolvimento ágil oferece benefícios importantes, no entanto, não é indicado para todos os projetos, produtos, equipes e situações, assim como métodos robustos e tradicionais também não são. Fundamental é que, para cada

projeto, os stakeholders possam decidir qual a melhor forma de conduzi-lo. O contexto, a natureza e o segmento são os três axiomas básicos que devem ser levados em consideração para chegar à definição da metodologia a ser implantada.

4.4 Semelhanças e diferenças entre os trabalhos relacionados

Nos trabalhos citados acima, pode-se perceber a diversidade de metodologias e contextos abordados pelos autores em suas pesquisas. Os trabalhos de Calazans et al. (2020) e Longo e Da Silva (2014), são voltados a técnicas de elicitação e documentação nos Métodos Ágeis. Já os trabalhos de Barbosa e Fraga (2017) e Alves (2015) são direcionados a apresentar sobre a Engenharia de Requisitos e seu processo em um contexto ágil, ressaltando os principais desafios nesse ramo, bem como as lições aprendidas.

Este trabalho assemelha-se ao de Soares (2004) e Barboza et al. (2016), ao propor uma análise comparativa entre a abordagem ágil e tradicional, realizando um estudo de caso em projetos de desenvolvimento de software, com o objetivo de obter resultados relacionados ao proposto no início do projeto e ao que foi entregue ao final do ciclo. Essa pesquisa diferencia-se de Soares (2004) e Barboza et. al (2016) quando foca exclusivamente no estudo de caso das metodologias ágeis e tradicionais nos projetos.

No Quadro 5 são apresentadas as principais semelhanças e diferenças entre os trabalhos relacionados e este estudo.

Quadro 5 - Comparativo entre os trabalhos relacionados e este trabalho

Trabalho	Aborda conceitos de técnicas de elicitação	Realiza uma análise comparativa de métodos	Analisa a Engenharia de Requisitos	Estudo de caso envolvendo projetos de desenvolvimento de software	Pontua desafios encontrados para obter resultados
Calazans et al. (2020)	**x**				
Longo e Da Silva (2014)	**x**				
Barbosa e Fraga (2017)			**x**		**x**

Alves (2015)	**x**		**x**		**x**
Soares (2004)		**x**			
Barboza et al. (2016)		**x**		**x**	
Este Trabalho	**x**	**x**	**x**	**x**	**x**

Fonte: Autora (2022)

5 PROCEDIMENTOS METODOLÓGICOS

Este capítulo apresenta as atividades que foram necessárias para realizar essa pesquisa e atingir os objetivos propostos. Com o objetivo de detalhar melhor as atividades foram divididas em cinco subseções: (1) Revisão da literatura, (2) Identificação do processo adotado na Engenharia de Requisitos dos projetos de desenvolvimento de software, (3) Entender sobre o escopo previsto dos projetos, (4) Comparar entregas previstas com as entregas realizadas dos projetos ao final de cada ciclo e (5) Identificação de desafios, lições aprendidas e oportunidades de melhoria na Engenharia de Requisitos dos projetos, conforme é mostrado na Figura 6..

Figura 6 - Processo metodológico

Fonte: Autora (2022)

5.1 Revisão da literatura

Inicialmente, foi realizada uma busca pelos trabalhos relacionados e fundamentação teórica que tem como objetivo guiar a compreensão sobre termos e assuntos necessários para basear essa pesquisa e prover uma visão geral do trabalho.

As buscas pelos trabalhos foram feitas através do Google Acadêmico. Para realizar a busca pelos trabalhos nesta área foram adotadas palavras-chaves como: Engenharia de Requisitos, Metodologias Ágeis, Modelo Cascata, SCRUM, Comparação entre Metodologias Ágeis e Tradicionais e Técnicas de Elicitação de Requisitos. Por meio dos objetivos da pesquisa foi tomado como base os trabalhos identificados com o intuito de coletar mais informações sobre as técnicas de elicitação utilizadas nos Métodos Ágeis e verificar a análise de um estudo de caso que aborda a Engenharia de Requisitos no método tradicional e ágil.

Mediante isso, foram selecionados trabalhos que abordam as técnicas de elicitação e documentação nos Métodos Ágeis, Engenharia de Requisitos ágil e análise comparativa entre as abordagens ágeis e tradicionais, dessa forma, por meio da revisão da literatura tem-se um

embasamento teórico maior para tratar a abordagem ágil e clássica inserida em diferentes contextos. Resultados desta fase foram apresentados no capítulo 4.

5.2 Identificação do processo adotado na Engenharia de Requisitos dos projetos

O presente trabalho busca identificar os benefícios, limitações e impactos decorrentes da adoção da abordagem ágil Scrum quando comparado ao gerenciamento tradicional de projetos. Com o intuito de captar e explorar a perspectiva dos envolvidos, bem como desenvolver novas compreensões sobre a magnitude dos aspectos observados, adotou-se um tipo de análise que tem por base conhecimentos teóricos empíricos. (Barros e Lehfeld, 2007). Como Barros e Lehfeld (2007) menciona, a pesquisa tem por objetivo captar e desbravar a perspectiva dos envolvidos, dessa forma, torna-se necessário que durante essa etapa seja realizada a identificação do processo de Engenharia de Requisitos adotado na metodologia tradicional utilizada em uma instituição de ensino e na metodologia ágil usada em uma empresa de desenvolvimento de software.

5.3 Entender sobre o escopo previsto dos projetos

Tendo em vista o objetivo de comparar metodologias ágeis e tradicionais inseridas em um estudo de caso em contexto diferentes, torna-se de extrema importância conhecer sobre o escopo dos projetos, envolvendo tempo de duração, quantidade de pessoas no time de desenvolvimento e suas respectivas responsabilidades.

Nesta pesquisa, serão analisados os escopos planejados para os projetos de metodologia ágil e tradicional, de modo que a autora compreenda as necessidades dos stakeholders e entenda os entregáveis dos projetos, alinhando as expectativas com as entregas para compreensão total do escopo.

5.4 Comparar entregas previstas com as entregas realizadas dos projetos ao final de cada ciclo

Mediante as atividades planejadas para cada ciclo dos projetos de acordo com o backlog, estimativa das atividades e impedimentos que surgem no desenvolvimento das atividades, esta pesquisa busca analisar os entregáveis planejados no começo do ciclo, para posteriormente compará-los com os entregáveis realizados pelo time de desenvolvimento.

Logo, busca-se analisar os fatores que impedem a entrega completa das atividades e o impacto que ocasiona no projeto. Esses fatores serão listados para servir de apoio e lição

aprendida para o time de desenvolvimento dos projetos envolvidos nessa pesquisa. Também busca-se analisar diferenças nas entregas de projetos tradicionais e ágeis. A análise será feita por meio da observação dos backlogs dos projetos.

5.5 Identificação de desafios, lições aprendidas e oportunidades de melhoria na Engenharia de Requisitos dos projetos

Com a execução e análise do processo proposto pela autora é possível identificar os pontos positivos e negativos da execução. Partindo disso, foi desenvolvido um checklist com a finalidade de coletar as técnicas de elicitação e especificação de requisitos para analisar a abordagem adotada na Engenharia de Requisitos dos processos utilizados no contexto ágil e tradicional. O checklist foi composto por itens elaborados pela autora da pesquisa juntamente com a orientadora e baseado em técnicas de elicitação de requisitos utilizados na abordagem ágil e tradicional. O Apêndice A apresenta os Quadros 6 e 7 em que aponta a versão final do checklist que será utilizado na pesquisa e o Apêndice B apresenta os Quadros 8 e 9 que lista as questões e respostas dos questionários aplicados com membros do Projeto A e Projeto B localizado no capítulo 11.

Logo, uma lista de lições aprendidas foi desenvolvida para relatar a adoção do processo proposto pela autora desta pesquisa. Posteriormente, foram identificadas oportunidades de melhorias nos projetos, propondo possíveis soluções.

6 APRESENTAÇÃO DOS PROJETOS

Este estudo visa comparar dois projetos distintos, em que um foi desenvolvido por um projeto de extensão de uma instituição de ensino aplicando abordagem tradicional de desenvolvimento (Projeto A) e o outro por uma empresa de desenvolvimento de software utilizando a abordagem ágil (Projeto B) . Os projetos A e B serão descritos nas seções 6.1 e 6.2.

6.1 Projeto A - Abordagem Tradicional

O projeto A trata-se do desenvolvimento de uma aplicação web com o intuito de facilitar a gestão de atendimentos psicológicos. Através desta, os psicólogos podem acompanhar atendimentos que foram agendados, remarcados ou cancelados, além de ter acesso ao histórico dos pacientes e, assim, conseguir acompanhar a rotina de atendimentos.

Deu-se início ao projeto no período de março de 2020 e o primeiro ciclo durou 365 dias, sendo conduzido por um time composto de gerente de projeto, analistas de requisitos, analistas de negócio, designer, desenvolvedores e coordenador. O gerente do projeto era responsável por liderar e direcionar a equipe, visando o cumprimento dos prazos e requisitos exigidos pelo cliente, administrando o custo e monitorando os riscos, assim como prevê o guia de melhores práticas em gerenciamento de projetos (PMI, 2013).

No contexto do ano de 2020 dava-se início aos tempos pandêmicos (COVID -19), em que, foi aplicado um questionário para levantar quais necessidades a população desejava suprir. Obteve-se aproximadamente 120 respostas, na qual, a maioria dos resultados obtidos estavam associados a atendimentos psicológicos *online.* A partir disso, o time se reuniu para definir os próximos passos que foram: reuniões de refinamento, elicitação e especificação de requisitos e desenvolvimento da aplicação.

As técnicas de elicitação predominantemente utilizadas foram entrevistas, questionários e *brainstorming*, já na especificação foi utilizado modelagem baseada em casos de uso, *tasks,* matriz de rastreabilidade entre requisitos e casos de uso.

Este projeto teve resultados em diferentes frentes de atuação junto às estratégias para melhorias e desenvolvimento das funcionalidades no sistema de gerenciamento dos atendimentos e armazenamento de informações dos pacientes.

Dentre estes resultados, podemos destacar que, o sistema foi desenvolvido em PHP Usando *framework Laravel* com *Livewire* sendo usado para controle de estado no front-end e a aplicação foi implementada com arquitetura Cliente/Servidor, usando serviços oferecidos pela *Amazon Web Services* (AWS). A princípio, foi utilizado o padrão *Model-View-Controller*

(MVC), usando conceito de design *pattern Repository* com camada de serviço. As entregas principais giraram em torno do cadastro e solicitação de agendamento via *chatbot* e a gestão dos atendimentos agendados via sistema web.

Tendo em vista os resultados obtidos, pôde ser observado que na visão do coordenador e gerente do projeto, mesmo com algumas oportunidades de melhoria e processos, o projeto foi entregue com êxito.

6.2 Projeto B - Abordagem Ágil

O projeto B trata-se do desenvolvimento de uma aplicação web com o intuito de desenvolver uma nova interface para o sistema responsável pela configuração e gerenciamento das partes que compõem as plataformas da empresa. Uma parte são todos os componentes (peças) que compõem as diferentes plataformas, como Notebooks, Desktops, Servidores e demais periféricos.

O objetivo geral deste projeto foi pesquisar e desenvolver uma solução tecnológica para implementar uma nova versão da aplicação. Para alcançar o objetivo geral, os seguintes objetivos específicos foram definidos: (i) redesenhar e desenvolver novas interfaces de usuário para a aplicação, a fim de aprimorar a usabilidade e experiência do usuário utilizando componentes modernos e reativos; (ii) pesquisar e desenvolver recursos de auditoria para as modificações feitas nas relações de compatibilidade entre os produtos e peças e nos guias de resolução de problemas; (iii) pesquisar soluções de IA para o desenvolvimento de um assistente inteligente que auxilia os usuários na resolução de problemas.

Deu-se início ao projeto no período março de 2021 e o primeiro ciclo durou 11 meses, sendo conduzido por um time composto de gerente de projeto, gerente de produto, analistas de teste, designers, desenvolvedores, pesquisadores e coordenador. O gerente do produto era responsável por liderar e direcionar a equipe, visando o cumprimento dos prazos e requisitos solicitados e acordados com o cliente.

No período de março a maio de 2021 ocorreram diversas reuniões técnicas com o time, nas quais foram apresentadas as funcionalidades e problemas da aplicação atual e ocorreram várias demonstrações do uso da aplicação para o entendimento geral da solução. Além disso, foram realizadas entrevistas com a equipe de TI[1] e negócios para identificar os papéis, responsabilidades dos usuários dentro da aplicação, problemas e necessidades.

[1] Tecnologia da informação

As técnicas de elicitação predominantemente utilizadas foram entrevistas, questionário, *brainstorming*, personas, análise de domínio, *brainwriting*, prototipação de baixa fidelidade e *refining meetings*. Já durante a fase de documentação foram implementadas as técnicas de especificação de requisitos: *user stories, features, tasks*, cenários e protótipos.

Com relação à comunicação, reuniões foram realizadas no início e final de cada sprint (a cada 15 dias em média) entre cliente e executante, a fim de alinhar o desenvolvimento do projeto. Além disso, o cliente atuou de forma que houvesse maior envolvimento dos usuários, visando um levantamento de requisitos mais acurado e, consequentemente, a elaboração de um sistema que se adequasse às necessidades de quem realmente iria utilizá-lo. Dessa forma, este projeto obteve resultados em diferentes frentes de atuação junto às estratégias para melhorias e desenvolvimento de novas funcionalidades no sistema de gerenciamento e armazenamento de informações de partes e plataformas da empresa. Dentre estes resultados, podemos destacar:

- Nova arquitetura da aplicação

A nova arquitetura foi projetada para melhorar a manutenção, estabilidade e disponibilidade dos serviços que mantêm o sistema funcionando. A arquitetura da aplicação anterior, orientada a serviço foi modificada para a modelagem Domain Driven Design (DDD). O framework utilizado passou do .NET framework versão 4.7 para o .NET versão 5. A integração da nova arquitetura é feita utilizando RESTful e a documentação feita no Swagger.

- Módulo de auditoria

Para obter o módulo de auditoria, algumas ações foram realizadas, dentre elas:

- Entendimento da base de dados referente ao sistema de auditoria, como esses dados estavam relacionados e quais os requisitos de desempenho associados às consultas nesta base;
- Fase de UX/UI para criar um conjunto de interfaces de usuário para melhorar como as trilhas de auditoria são apresentadas;
- Propor e desenvolver a interface do usuário da aplicação contendo trilhas de auditoria;
- Endpoints relevantes foram identificados e desenvolvidos para fornecer dados a serem apresentados pela UI.
- Novas interfaces de usuário

A nova interface foi implementada de acordo com os critérios de cores e design system da empresa, respeitando as diretrizes de usabilidade e acessibilidade. A partir de estudos e

análises da aplicação atual, foram desenvolvidos protótipos de alta fidelidade com a nova interface de usuário do sistema, com novos fluxos funcionais e melhorias de usabilidade, com o objetivo de facilitar a navegação dos usuários no sistema. Foram implementados funções de acessibilidade para auxiliar a navegação de pessoas com baixa visão ou cegas através de navegação por teclado, alto contraste e aumento e diminuição de fontes. A nova interface Web utiliza o framework Vue 3.0.0, uma versão mais moderna e atualizada do que é usado na interface antiga, Vue 2.6.11. Além disso, a utilização do Webforms foi substituída pelo Vue 3.0.0, padronizando as ferramentas utilizadas e simplificando a manutenção do código.

- Modelos de IA[2] para auxílio na resolução de feedbacks:

Foram desenvolvidos modelos de inteligência artificial para auxiliar a resolução de feedbacks para o time de negócios e time de suporte técnico da empresa. Os modelos estão descritos abaixo e foram implementados utilizando Python 3.8 e as bibliotecas de código aberto NLTK 3.5, Spacy 2.3.2 e Scikit-Learn 0.23.2.

- Classificação de Issue Summary: Este modelo consiste em classificar os Comentários de um feedback do Suporte Técnico em um dos 11 issue summaries existentes. Tem como objetivo recomendar a classificação mais adequada e evitar o envio de resumos de problemas incorretos. Este modelo utilizou o TF-IDF para a codificação de texto e o classificador Extreme Gradient Boosting (XGBoost). O modelo obteve uma acurácia de 84% de assertividade.
- Similaridade de Feedback: Este modelo consiste em encontrar feedbacks semelhantes já resolvidos para simplificar questões recorrentes e repetitivas. Permite verificar se o novo feedback requer informações adicionais de feedbacks semelhantes anteriores, evitando descrições incompletas. Este modelo utilizou o TF-IDF para a codificação de texto e a similaridade por cosseno.
- Templates de Feedbacks: Esse modelo classifica o tipo de feedback recebido e aplica regras de negócios para fornecer um modelo de resolução de feedback padrão para auxiliar a equipe de negócios. Foram desenvolvidos 11 templates, em diferentes categorias. Este modelo utilizou o TF-IDF para a codificação de

[2] Inteligência Artificial

texto e o classificador Random Forest. O modelo obteve uma acurácia de 72% de assertividade.

7 ANÁLISE QUALITATIVA

Nesta seção serão apresentados os dados coletados através do questionário e a análise realizada para avaliar a aplicação do processo. Os questionamentos feitos aos participantes para obter os dados desta análise podem ser encontrados no Apêndice B e estão listados abaixo:

- Q1 - Quais técnicas de elicitação de requisitos foram mais utilizadas durante o projeto?
- Q2 - Quais técnicas de especificação/documentação foram adotadas durante o projeto?
- Q3 - As reuniões internas e com o cliente para a validação do que estava sendo desenvolvido eram feitas com que frequência? Você acredita que quanto mais reuniões implica em maior acompanhamento do projeto?
- Q4 - Você acredita que a utilização de alguma(s) outras técnicas de elicitação e documentação poderiam ter resultado em melhor compreensão das atividades? Se sim, cite.
- Q5 - Quais foram os maiores desafios encontrados durante o desenvolvimento do projeto?
- Q6 - A expectativa dos stakeholders foi alcançada após a entrega do projeto? O que foi pontuado de pontos positivos e negativos na visão deles que você conseguiu interpretar?

A análise qualitativa desta pesquisa foi realizada pela autora deste trabalho e posteriormente validada pela orientadora. O método utilizado para realizar a análise foi o *Grounded Theory* (GT) que é baseado na codificação a partir da identificação de trechos que são usados como códigos e categorias. Para Barbosa (2017) o GT é orientado para coleta e análise de dados não numéricos com o objetivo de alcançar a profundidade da informação.

O processo de codificação pode ser dividido em três fases: codificação aberta, axial e seletiva. Este estudo utilizou a codificação aberta dos dados obtidos através do questionário. Além disso, fez uso do software ATLAS.ti para análise e manipulação dos dados.

Por meio do procedimento de codificação pode-se identificar as seguintes categorias:

(I) Técnicas de elicitação de requisitos no Projeto A e Projeto B, (II) Técnicas de especificação de requisitos no Projeto A e Projeto B, e (III) Frequência de reuniões internas e com clientes, (IV) Outras técnicas de elicitação e especificação que poderiam ser utilizadas, (V) Desafios encontrados no desenvolvimento, (VI) *Feedback* após a entrega do produto. Foram criadas redes de códigos relacionados por categoria.

Foram criadas redes a partir da união dos códigos relacionados por categoria. Os números abaixo de cada código representam o Grau de fundamentação teórica (indica quantas

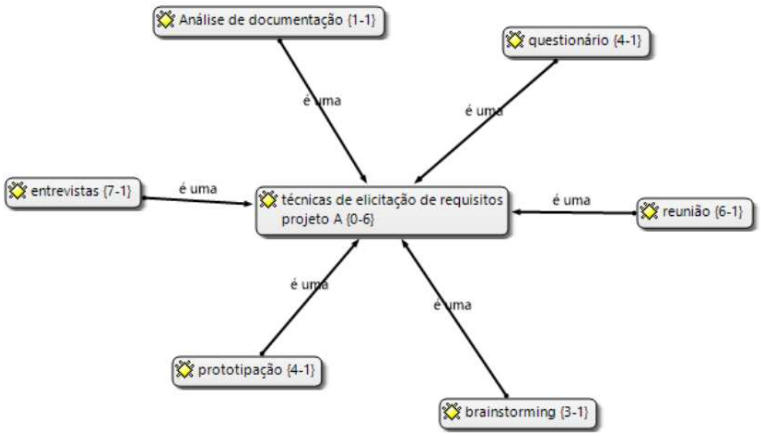
vezes este código foi identificado nos dados analisados) e Grau de densidade teórica (indica a quantidade de relacionamentos deste código com outros códigos ou categorias (ROCHA, 2021).

Os participantes foram identificados de P1 a P3 no Projeto A e P1 a P3 no Projeto B como melhor forma de identificação e confidencialidade. As respostas para cada questionamento podem ser encontradas no Apêndice B deste trabalho e os resultados serão apresentados nas subseções seguintes para cada categoria identificada com o uso de redes e apresentação de citações para os códigos identificados.

7.1 Técnicas de elicitação de requisitos no Projeto A e Projeto B

A *network* criada teve como objetivo selecionar as técnicas de elicitação de requisitos utilizadas no Projeto A e Projeto B, como mostra a Figura 7 e Figura 8. Na *network* apresentada na Figura 7 destacam-se quatro códigos principais: (i) reunião, (ii) questionário, (iii) entrevistas e (iv) prototipação.

Figura 7: Técnicas de elicitação de requisitos no Projeto A



Fonte: Autora (2022)

Reunião: apresenta os relatos dos participantes que apontaram a reunião como técnica de elicitação utilizada no projeto A, como no relato de P1, P2 e P3

- **P1:** "*Reuniões internas, entrevistas, brainstorming e questionários"*
- **P2:** *"Entrevista, prototipação, reunião com o time"*
- **P3:** *"Análise de documentação, entrevista, reuniões com a equipe, questionário, prototipação de baixa"*

Questionário: apresenta o relato dos participantes que apresentaram as técnicas de elicitação que foram utilizadas no projeto A, como no relato de P1 e P3.

- **P1:** "*Reuniões internas, entrevistas, brainstorming e questionários"*
- **P3:** *"Análise de documentação, entrevista, reuniões com a equipe, questionário, prototipação de baixa"*

Entrevistas: apresenta os relatos dos participantes que apontaram a entrevista como técnica de elicitação utilizada no projeto A, como no relato de P1, P2 e P3.

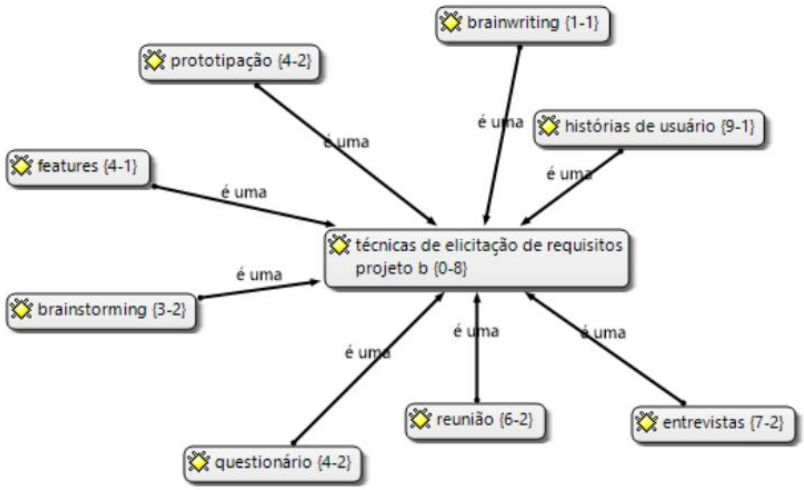
- **P1:** "*Reuniões internas, entrevistas, brainstorming e questionários"*
- **P2:** *"Entrevista, prototipação, reunião com o time"*
- **P3:** *"Análise de documentação, entrevista, reuniões com a equipe, questionário, prototipação de baixa"*

Prototipação: apresenta os relatos dos participantes que apontaram a prototipação como técnica de elicitação utilizada no projeto A, como no relato de P2.

- **P2:** *"Entrevista, prototipação, reunião com o time"*

Na *network* apresentada na Figura 8 destacam-se cinco códigos principais: (i) Histórias de Usuário, (ii) Reunião, (iii) entrevistas, (iv) prototipação e (v) features.

Figura 8: Técnicas de elicitação de requisitos no Projeto B



Fonte: Autora (2022)

Histórias de Usuário: apresenta os relatos dos participantes que apontaram a história de usuário como técnica de elicitação utilizada no projeto B, como no relato de P1, P2 e P3:

- **P1:** *"Reuniões de refinamento, entrevistas com cliente, histórias de usuário, brainstorming, design thinking, features"*
- **P2:** *"Questionários, entrevistas, reuniões com o time, prototipação, histórias de usuário, features"*
- **P3:** *"Brainstorming, brainwriting, entrevistas, questionários, reuniões de alinhamento com time e cliente, histórias de usuário, features"*

Reunião: apresenta os relatos dos participantes que apontaram a reunião como técnica de elicitação utilizada no projeto A, como no relato de P1 e P3:

- **P1:** *"Reuniões de refinamento, entrevistas com cliente, histórias de usuário, brainstorming, design thinking, features"*
- **P2:** *"Questionários, entrevistas, reuniões com o time, prototipação, histórias de usuário, features"*
- **P3:** *"Brainstorming, brainwriting, entrevistas, questionários, reuniões de alinhamento com time e cliente, histórias de usuário, features"*

Entrevistas: apresenta os relatos dos participantes que apontaram a entrevista como técnica de elicitação utilizada no projeto B, como no relato de P1, P2 e P3.

- **P1:** *"Reuniões de refinamento, entrevistas com cliente, histórias de usuário, brainstorming, design thinking, features"*
- **P2:** *"Questionários, entrevistas, reuniões com o time, prototipação, histórias de usuário, features"*
- **P3:** *"Brainstorming, brainwriting, entrevistas, questionários, reuniões de alinhamento com time e cliente, histórias de usuário, features"*

Prototipação: apresenta os relatos dos participantes que apontaram a prototipação como técnica de elicitação utilizada no projeto B, como no relato de P2:

- **P2:** *"Questionários, entrevistas, reuniões com o time, prototipação, histórias de usuário, features"*

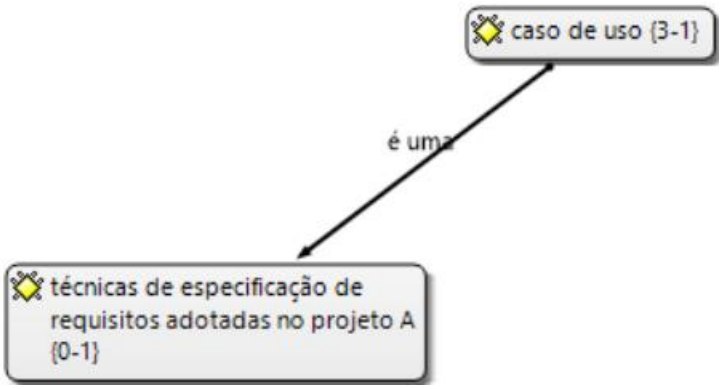
Features: apresenta os relatos dos participantes que apontaram a *feature* como técnica de elicitação utilizada no projeto B, como no relato de P1, P2 e P3.

- **P1:** *"Reuniões de refinamento, entrevistas com cliente, histórias de usuário, brainstorming, design thinking, features"*
- **P2:** *"Questionários, entrevistas, reuniões com o time, prototipação, histórias de usuário, features"*
- **P3:** *"Brainstorming, brainwriting, entrevistas, questionários, reuniões de alinhamento com tim e e cliente, histórias de usuário, features"*

7.2 Técnicas de especificação de requisitos no Projeto A e Projeto B

A *network* criada teve como objetivo selecionar as técnicas de especificação de requisitos utilizadas no Projeto A e Projeto B, como mostra a Figura 9 e Figura 10. Na *network* apresentada na Figura 9 destaca-se um código principal: (i) casos de uso.

Figura 9: Técnicas de especificação de requisitos no Projeto A



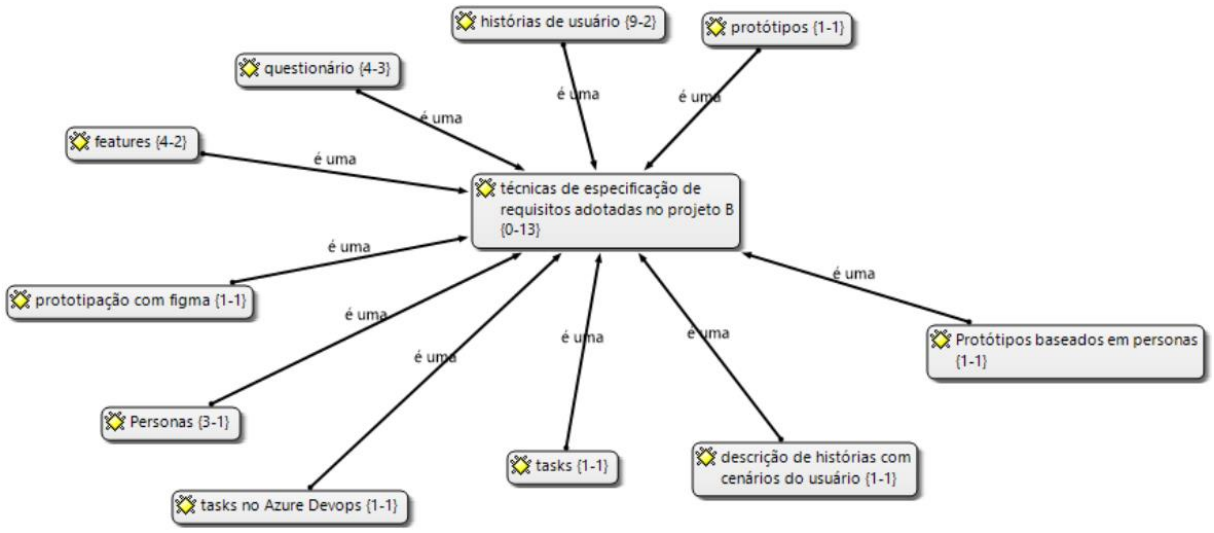
Fonte: Autora (2022)

Casos de Uso: apresenta os relatos dos participantes que apontaram os casos de uso como técnica de especificação utilizada no projeto A, como no relato de P1, P2 e P3:

- **P1:** "*Modelo de casos de uso e Trello"*
- **P2: "***Documentação tradicional baseada em casos de uso e também contou com um protótipo de baixa fidelidade"*
- **P3:** "*Era utilizado o trello para o acompanhamento das atividades do time e a documentação foi feita a partir da especificação de casos de uso, em que, era descrito o caminho feliz, fluxo alternativo e de exceção"*

Na *network* apresentada na Figura 10 destacam-se dois códigos principais: (i) histórias de usuário e *(ii) features.*

Figura 10: Técnicas de especificação de requisitos no Projeto B



Fonte: Autora (2022)

Histórias de Usuário: apresenta os relatos dos participantes que apontaram histórias de usuário como técnica de especificação utilizada no projeto B, como no relato de P1, P2 e P3:

- **P1:** *"Histórias de usuários, descrição BDD baseada em cenários de uso da aplicação, personas, protótipos"*
- **P2:** *"Personas, features, histórias de usuário, tasks, descrição de histórias com cenários do usuário, prototipação com figma, tasks no ambiente Azure Devops"*
- **P3:** *"Protótipos baseados em personas, histórias de usuário, prototipação."*

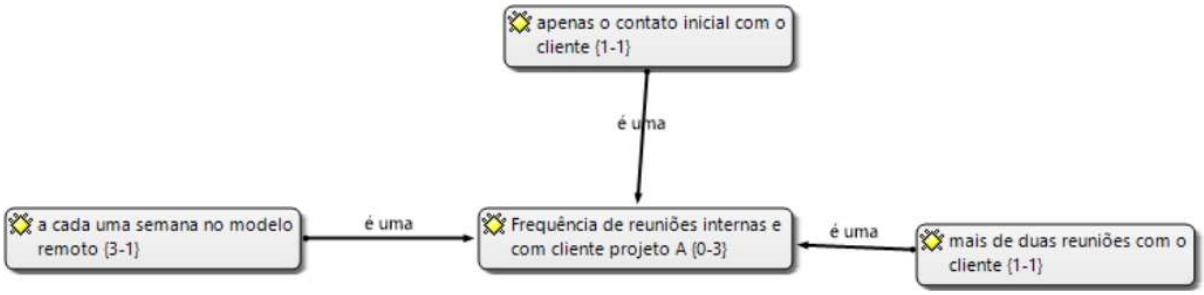
Features: apresenta os relatos dos participantes que apontaram features como técnica de especificação utilizada no projeto B, como no relato de P2.

- **P2:** *"Personas, features, histórias de usuário, tasks, descrição de histórias com cenários do usuário, prototipação com figma, tasks no ambiente Azure Devops"*

7.3 Frequência de reuniões internas e com clientes

A *network* criada teve como objetivo selecionar a frequência de reuniões internas e com clientes no Projeto A e Projeto B, como mostra a Figura 11 e Figura 12. Na *network* apresentada na Figura 11 destaca-se um código principal: (i) a cada uma semana no modelo remoto.

Figura 11: Frequência de reuniões internas e com clientes no projeto A



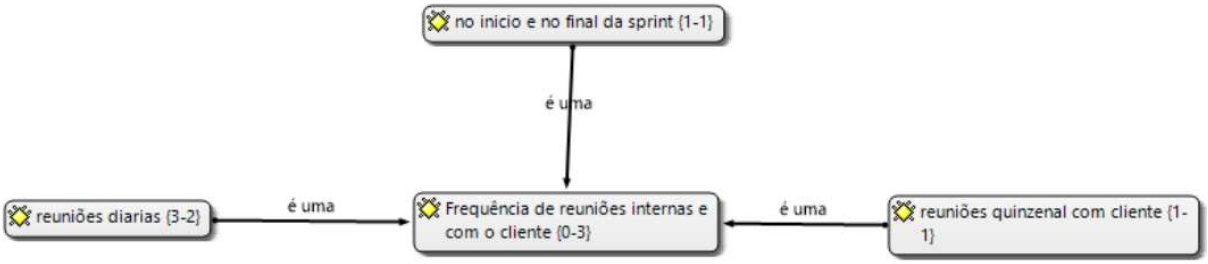
Fonte: Autora (2022)

A cada uma semana no modelo remoto: apresenta os relatos dos participantes que apontaram que a cada uma semana existem reuniões entre o time do projeto A, como no relato de P1, P2 e P3:

- **P1:** *"As reuniões internas eram a cada semana online…"*
- **P2:** *"...Já as internas aconteciam semanalmente com a professora coordenadora.".*
- **P3:** *"As reuniões internas aconteciam 1x por semana online via meet ou a pessoa enviava o status das atividades no grupo do Whatsapp…"*

Na *network* apresentada na Figura 12 destaca-se um código principal: (i) reuniões diárias.

Figura 12: Frequência de reuniões internas e com clientes no projeto B



Fonte: Autora (2022)

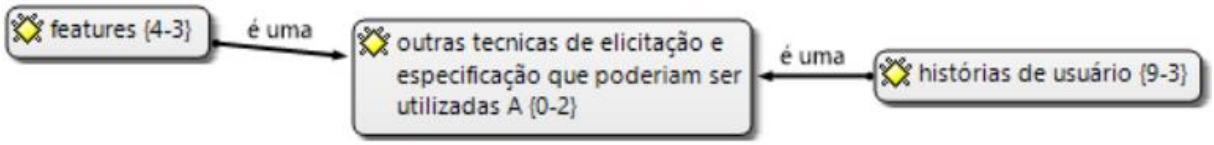
Reuniões Diárias: apresenta os relatos dos participantes que apontaram que existem reuniões diárias com o time do projeto B, como no relato de P1, P2 e P3.

- **P1:** *"Reuniões internas todos os dias "daily"..."*
- **P2:** *"Existiam dailies que ocorrem todos os dias..."*
- **P3:** *"São realizadas diariamente para verificar o andamento das atividades direcionadas ao time..."*

7.4 Outras técnicas de elicitação e especificação que poderiam ser utilizadas

A *network* criada teve como objetivo selecionar outras técnicas de elicitação que poderiam ser utilizadas no Projeto A e Projeto B, como mostra a Figura 13 e Figura 14. Na *network* apresentada na Figura 13 destacam-se dois códigos principais: (i) Features e (ii) Histórias de Usuário.

Figura 13: Outras técnicas de elicitação e especificação que poderiam ser utilizadas no Projeto A



Fonte: Autora (2022)

Features: apresenta os relatos dos participantes que apontaram que existem outras técnicas de elicitação e documentação como Features, que poderiam ter sido utilizadas no Projeto A, como no relato de P3.

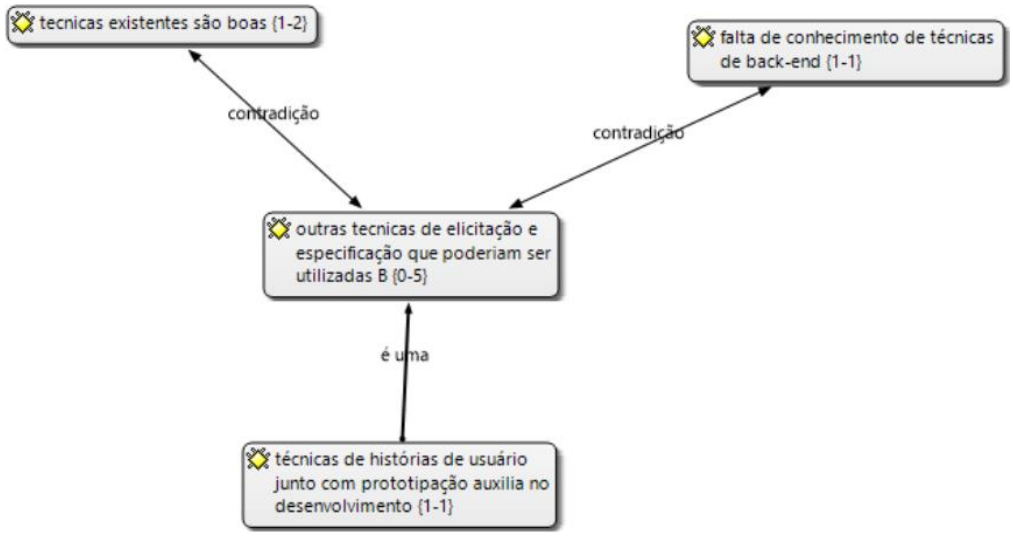
- **P3:** *"Sim, acho que basear-se em features, quebrar o projeto em macro entregas, facilitaria o entendimento se unido a histórias de usuários e tornaria as entregas incrementais sem esperar a conclusão do projeto para ver a entrega de fato. "*

Histórias de Usuário: apresenta os relatos dos participantes que apontaram que existem outras técnicas de elicitação e documentação como Histórias de Usuário, que poderiam ter sido utilizadas no Projeto A, como no relato de P1, P2 e P3.

- **P1:** *"Sim, o uso de histórias de usuário ajudam muito na compreensão das ações do usuário quando ligadas a uma persona do sistema, auxiliando tanto a nível técnico como a nível de negócio"*
- **P2:** *"Sim, poderia encaixar histórias de usuário que são muito utilizadas em projetos ágeis e também exemplificar cenários para cada história de usuário mostrando todos os cenários de ações dentro de uma página da aplicação"*
- **P3:** *"Sim, acho que basear-se em features, quebrar o projeto em macro entregas, facilitaria o entendimento se unido a histórias de usuários e tornaria as entregas incrementais sem esperar a conclusão do projeto para ver a entrega de fato. "*

Na *network* apresentada na Figura 14 não destacam-se códigos principais, pois não foi mencionado pelos participantes do Projeto B nenhuma técnica específica, abaixo pode ser visto o relato de P1, P2, e P3.

Figura 14: Outras técnicas de elicitação e especificação que poderiam ser utilizadas no Projeto B



Fonte: Autora (2022)

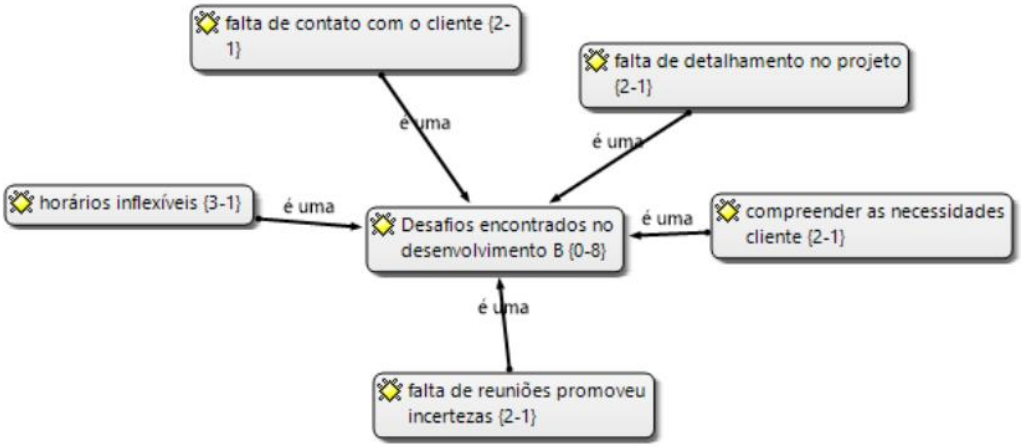
- **P1:** *"Não, acredito que as técnicas de histórias de usuário junto com prototipação das telas auxilia muito no desenvolvimento, Não conheço outra técnica que poderia auxiliar melhor"*
- **P2:** *"Não sei se existe alguma técnica que auxilia em atividades técnicas de back-end, por exemplo. Das técnicas conhecidas, acho que não."*

- **P3:** *"Não, as técnicas utilizadas oferecem uma granularidade muito boa, é possível saber detalhadamente o que é preciso fazer em uma tarefa sem precisar de outros recursos."*

7.5 Desafios encontrados no desenvolvimento

A *network* criada teve como objetivo listar os desafios encontrados em ambos projetos, como mostra a Figura 15 e Figura 16. Na *network* apresentada na Figura 15 destacam-se três códigos principais: (i) Horários inflexíveis, (ii) falta de contato com o cliente, (iii) falta de detalhamento no projeto.

Figura 15: Desafios encontrados no desenvolvimento no Projeto A



Fonte: Autora (2022)

Horários inflexíveis: apresenta os relatos dos participantes que apontaram que os horários inflexíveis dos participantes foi um desafio no desenvolvimento do Projeto A, como no relato de P1, P2 e P3.

- **P1:** *"...horário restrito dos participantes do projeto, desencontro de horários, ..."*
- **P2:** *"... Horário disponível para o time se reunir".*
- **P3:** *"Horários inflexíveis, poucos alinhamentos com o time e cliente..."*

Falta de contato com o cliente: apresenta os relatos dos participantes que apontaram que a falta de contato com o cliente foi um desafio no desenvolvimento do Projeto A, como no relato de P1 e P3.

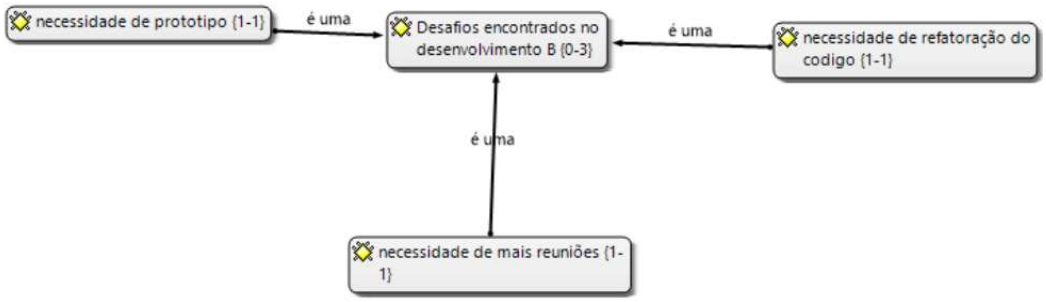
- **P1:** *"Pouco contato com cliente..."*
- **P3:** *"...poucos alinhamentos com o time e cliente"*

Falta de detalhamento no projeto: apresenta os relatos dos participantes que apontaram que a falta de detalhamento do projeto foi um desafio no desenvolvimento do Projeto A como no relato de P2 e P3.

- **P2:** *"...entender e imaginar cenários em uma tela que estava sendo desenvolvida sem ter isso especificado na documentação. Horário disponível para o time se reunir."*
- **P3:** *"... Pouca descrição de atividades, descritas de forma geral sem muitos detalhes."*

Na *network* apresentada na Figura 16 destacam-se dois códigos principais: (i) necessidade de protótipo e (ii) necessidade de mais reuniões.

Figura 16: Desafios encontrados no desenvolvimento no Projeto B



Fonte: Autora (2022)

Necessidade de protótipo: apresenta os relatos dos participantes que apontaram que a necessidade de prototipar novamente uma página validada previamente foi um desafio no desenvolvimento do Projeto B como no relato de P1 e P3

- **P1:** *"Por vezes era validado um protótipo e quando passava para estilizar o cliente sugeria outros pontos não mapeados e aprovados antes."*
- **P3:** *"Solicitação de melhorias dos clientes em páginas já integradas e aprovadas, "pensei nisso aqui agora, será que ficaria bom se a gente fizesse?"*

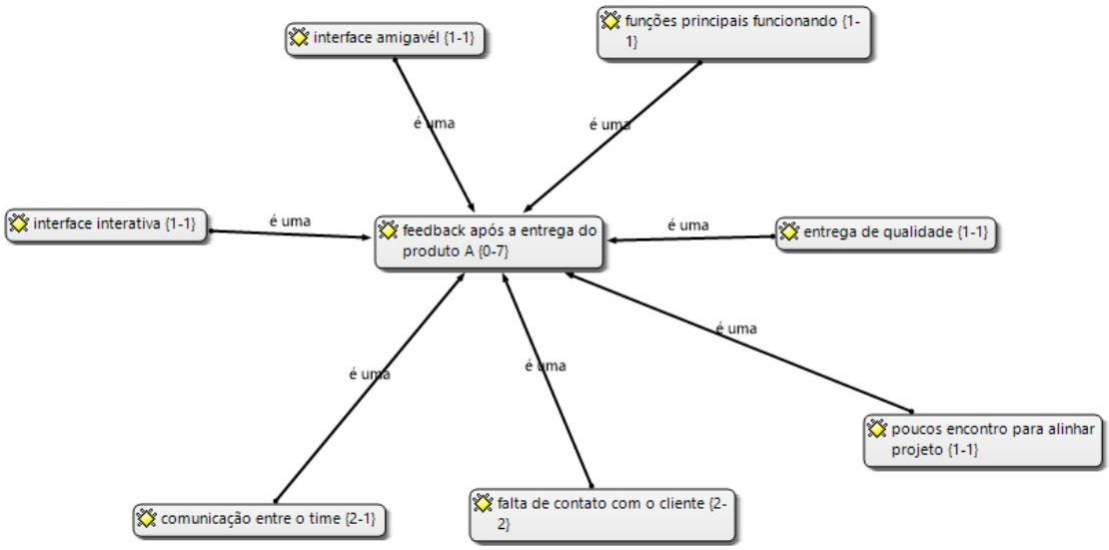
Necessidade de mais reuniões: apresenta os relatos dos participantes que apontaram que a necessidade de mais reuniões fora do planejamento foi um desafio no desenvolvimento do Projeto B como no relato de P2.

- **P2:** *"Complexidade de algumas páginas precisaram de algumas reuniões fora do planejamento."*

7.6 *Feedback* após a entrega do produto.

A *network* criada teve como objetivo selecionar os códigos de feedbacks do produto no Projeto A e Projeto B, como mostra a Figura 17 e Figura 18. Na *network* apresentada na Figura 17 destacam-se dois códigos principais: (i) comunicação entre o time e (ii) falta de contato com o cliente.

Figura 17: Feedback após a entrega do produto no projeto Projeto A



Fonte: Autora (2022)

Comunicação entre o time: apresenta os relatos dos participantes que apontaram como feedback que a comunicação entre o time e com o cliente poderia ter sido melhor no Projeto A como no relato de P1, P2 e P3.

- **P1:** *"...Negativos: pouco contato com o time desenvolvedor para o alinhamento das expectativas."*
- **P2:** *"...Negativos: poucas reuniões de alinhamento com time e cliente"*
- **P3:** *"...Negativos: comunicação entre time, revisão de código e testes, poucos encontros para alinhar andamento do projeto, objetivo e expectativa."*

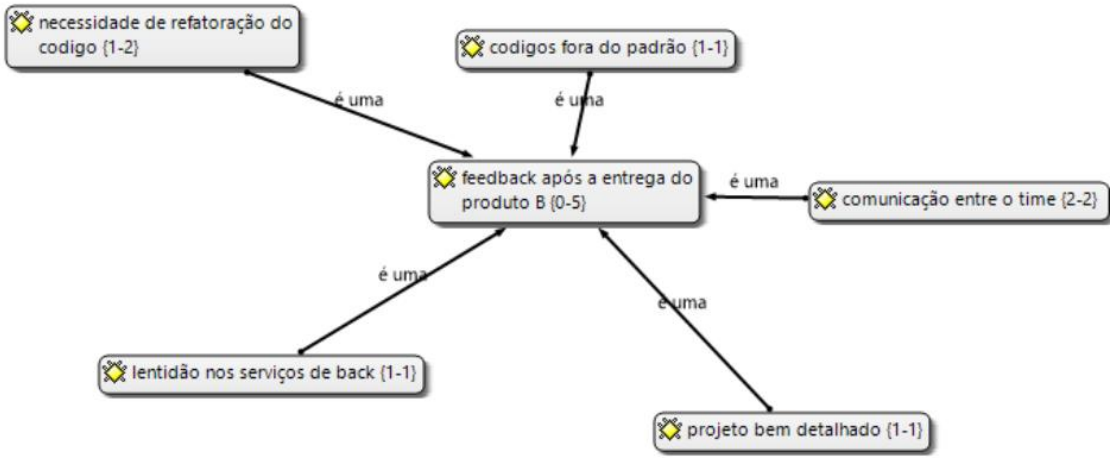
Falta de contato com o cliente: apresenta os relatos dos participantes que apontaram como feedback que o cliente poderia ter sido mais presente no Projeto A como no relato de P1, P2 e P3.

- **P1:** *"...Negativos: pouco contato com o time desenvolvedor para o alinhamento das expectativas."*

- **P2:** *"...Negativos: poucas reuniões de alinhamento com time e cliente"*
- **P3:** *"...Negativos: comunicação entre time, revisão de código e testes, poucos encontros para alinhar andamento do projeto, objetivo e expectativa."*

Na *network* apresentada na Figura 18 destacam-se um código principal: (i) comunicação entre o time

Figura 18: Feedback após a entrega do produto no projeto Projeto B



Fonte: Autora (2022)

Comunicação entre o time: apresenta os relatos dos participantes que apontaram como feedback que a comunicação entre o time foi bastante eficaz e positiva no Projeto B, como no relato de P2 e P3.

- **P2:** *"...Positivos: time super comprometido e cliente sempre presente validando tudo a cada sprint. Negativos: lentidão em alguns serviços do back e não foi resolvido a tempo."*
- **P3:** *"...Positivos: time muito comunicativo, produtivo e proativo junto com cliente engajado, projeto bem documentado, impedimentos resolvidos sem delongas."*

8 RESULTADOS

Em síntese, tem-se que o objetivo geral desta pesquisa foi identificar e analisar as técnicas de elicitação e documentação voltadas para os Métodos Ágeis e Métodos Tradicionais, além de contextualizar sua atuação em cada contexto de desenvolvimento de projetos que adotam metodologias distintas, sendo SCRUM e Cascata.

Dessa forma, para atingir os objetivos específicos identificaram-se as técnicas de elicitação e documentação mais utilizadas no contexto de desenvolvimento ágil e tradicional, para isso realizou-se um estudo de caso em uma organização que utiliza o SCRUM para o desenvolvimento de software (Projeto B) e uma instituição de ensino que aplica o desenvolvimento Cascata (Projeto A).

O estudo mostrou que o Projeto A aplicou as técnicas de elicitação: *Brainstorming,* Entrevistas, Prototipação, Questionários, e Reuniões, já o Projeto B adotou as técnicas: *Brainstorming, Scenarios, Refining Meetings, User Stories, Features,* Reuniões, Questionários, Entrevistas, Personas e Prototipação.

Nesse sentido, quanto às técnicas de especificações abordadas nos projetos, o Projeto A utilizou: Modelo de Casos de Uso, Protótipo, *Tasks* e *Wall*, já o Projeto B destacou-se por usufruir das técnicas: Cenários*, Features, Personas,* Protótipos*, Story card, Structured requirements list, Tasks, User stories, Wall* e XXM *(Extreme XMachine).*

Os resultados mostram que, no contexto do estudo de caso, as técnicas de elicitação mais utilizadas nos projetos do estudo de caso são: Entrevistas, *Brainstorming,* Grupo Focal, Prototipação, Questionários, *Refining Meetings* e Reuniões, foram as técnicas de elicitação consideradas de maior aplicabilidade. Com relação a documentação, os resultados mostram que as técnicas: *Tasks, Wall* e Protótipos foram as técnicas de documentação consideradas de maior aplicabilidade.

Desse modo, tem-se que ao utilizar as técnicas *User Stories, Scenarios* e Personas, o Projeto B apresentou resultados satisfatórios referente ao entendimento por parte do time durante a execução das atividades propostas, o que não ocorreu com o Projeto A, pois, ao optar por uma documentação tradicional gerou ambiguidades e incertezas durante a implementação das funcionalidades, ocasionando dúvidas no time de desenvolvimento.

Seguindo nesse contexto, o Projeto B utilizou a técnica de *features* para proporcionar uma entrega mais concisa e de forma incremental, permitindo assim entregas menores e agregando valor ao usuário. Dessa maneira, há uma melhor visualização do sistema e permite ao gerente de produto juntamente com o time de desenvolvimento detalhar melhor as entregas de cada

marco do projeto. Em contrapartida, o Projeto A utilizando documentação tradicional acabou gerando conflitos nas entregas das atividades, visto que há uma exibição macro das atividades.

9 CONCLUSÃO E TRABALHOS FUTUROS

Pode-se afirmar que a contribuição principal deste trabalho é indicar que as técnicas de elicitação e documentação ágil são eficazes também no desenvolvimento tradicional de software. O desenvolvimento ágil oferece benefícios importantes, no entanto, não é indicado para todos os projetos, produtos, equipes e situações, assim como métodos robustos e tradicionais também não são. (Barboza et al., 2016) Contudo, é de extrema relevância que para cada projeto, os *stakeholders* percebam quais as melhores técnicas podem ser utilizadas no seu cenário.

Com a execução desse trabalho pôde-se perceber em observação que projeto B em relação aos métodos e metodologias e métodos utilizados se mostrou mais eficiente do que em relação ao projeto A. Percebe-se também que a utilização das metodologias ágeis deixa evidente as funcionalidades dos projetos e suas entregas facilitando assim a compreensão dos requisitos por parte do time de desenvolvimento. Logo, a partir da execução desta pesquisa foi possível perceber a contribuição com a utilização de metodologias ágeis por projetos de empresas de software.

Portanto, a partir do que foi estudado, chega-se à conclusão de que, se as melhores práticas de cada método forem utilizadas de acordo com o contexto e que o projeto está inserido, pode-se atingir níveis elevados de produtividade e performance, com ganhos aos clientes e à equipe executante.

Desse modo, espera-se em trabalhos futuros analisar a utilização de metodologias ágeis pelo projeto com o intuito de posteriormente comparar as principais diferenças em resultados entre o uso das duas metodologias. Essa pesquisa pode ser realizada com analistas e desenvolvedores que atuam em empresas que adotam metodologias ágeis, e ao final confrontar dados do mapeamento com a pesquisa para verificar os pontos de convergência e de divergência.

REFERÊNCIAS

ALVES, Daniela de Castro Pereira. Engenharia de requisitos em projetos ágeis: um mapeamento sistemático baseado em evidências da indústria. **Dissertações de Mestrado - Ciência da Computação - UFPE,** 2015.

BARBOSA, Marcelo Werneck e FRAGA, Bárbara Silveira. A Engenharia de Requisitos nos métodos ágeis: uma revisão sistemática da literatura. **XIII Brazilian Symposium on Information Systems**, Lavras, Minas Gerais, June 5-8, 2017.

BARBOSA, M. W. 2017. Uma análise do uso de grounded theory em engenharia de software. **Revista Produção Online, 17(1), 26-48.**

BARBOZA, Lívia Fernandes, VAZ, Augusto Cezar Florentino, ANTUNES, Thiago Guimaraes Pereira e SALUME, Paula Karina. Análise comparativa entre as abordagens ágil e tradicional de gestão de projetos: Um estudo de caso no setor industrial. **Anais do Simpósio,** 2016.

BROD, Cesar. Scrum: guia prático para projetos ágeis. São Paulo: **Novatec**, 2013.

CALAZANS, Angélica Toffano Seidel, PALDÊS, Roberto Avila, GUIMARÃES, Fernando de Albuquerque, MASSON, Eloisa Toffano Seidel. As Técnicas de Elicitação e de Documentação de Requisitos nos Métodos Ágeis. **Conferência: Congresso Ibero-Americano em Engenharia de Software,** 2018.

CRUZ, Fábio. Scrum e PMBOK unidos no gerenciamento de Projetos. 1 ed. Rio de Janeiro: **Editora Brasport**, 2013.

DE LUCIA, Andréa; QUSEF, Abdallah. Engenharia de requisitos no desenvolvimento ágil de software. Revista de tecnologias emergentes em inteligência web , v. 2, n. 3, pág. 212-220, 2010.

LEHFELD, Neide Aparecida de Souza; BARROS, Aidil Jesus Paes. **Fundamentos da metodologia científica**. 3 ed. MAKRON, 2007.

LOPES, Leandro Teixeira. Um modelo de processo de engenharia de requisitos para ambientes de desenvolvimento distribuído de software. 2005. 142 f. Dissertação (Mestrado em Ciência da Computação) - **Pontifícia Universidade Católica do Rio Grande do Sul, Porto Alegre, 2005.**

LENZ, G., MOELLER, T. **Net:** A Complete Development Cycle. Pearson Education Inc., 2004.

LONGO, Hugo Estevam Romeu e DA SILVA, Madalena Pereira. A Utilização de Histórias de Usuários no Levantamento de Requisitos Ágeis. Int. J. Knowl. Eng. Manag., ISSN 2316-6517, Florianópolis, v.3, n.6, p. 1-30, jul/nov, 2014.

NUNES, R. D. A Implantação das metodologias ágeis de desenvolvimento de software scrum e extreme programming(XP): uma alternativa para pequenas empresas do setor de tecnologia da informação. ForScience, v. 4, n. 2, 26 fev. 2017.

PMI, Project Management Institute (Editor). **Um Guia do Conhecimento em Gerenciamento de Projetos.** 5. ed. Tradução Oficial para o português do PMBOK® (Project Management Body of Knowledge Guide). PMI,2013.

PRESSMAN, Roger S. **Engenharia de Software.** McGraw-Hill, 2001.

PRESSMAN, R. S. **Engenharia de software: uma abordagem profissional.** 7ª Edição. Ed: McGraw Hill, 2011.

RISING, L.; JANOFF, N. S. The Scrum software development process for small teams. **IEEE Software**, v. 17, n. 4, 2000.

ROCHA, Marina da S. et al. Uma análise sobre a importância de um projeto com ações direcionadas ao acolhimento de ingressantes de cursos de Computação: Um estudo qualitativo. In: Anais do XV **Women in Information Technology. SBC, 2021. p. 210-219.**

ROYCE, Winston W. Managing the development of large software systems: concepts and techniques. **Proc. IEEE Westcon**, Los Angeles, CA, 1970.

SILVA, F. G.; HOENTSCH, S. C. P.; SILVA, L. **Uma análise das Metodologias Ágeis FDD e Scrum sob a Perspectiva do Modelo de Qualidade MPS.BR**. Scientia Plena, [S. l.], v. 5, n. 12, 2011. Disponível em: https://www.scientiaplena.org.br/sp/article/view/678. Acesso em: 14 jun. 2022.

SOARES, Michel dos Santos.Comparação entre Metodologias Ágeis e Tradicionais para o Desenvolvimento de Software. **INFOCOMP Journal of Computer Science**, 2004.

STANKIEWICZ, Alessandro. Modelo de interação ágil: uma adaptação do modelo cascata à organização de pequenas e médias empresas. 2017. 45 f. **Trabalho de Conclusão de Curso (Tecnologia em Análise e Desenvolvimento de Sistemas)** - Universidade Tecnológica Federal do Paraná, Ponta Grossa, 2017.

SUTHERLAND, Jeff; SCHWABER**, Ken. Guia do Scrum**, 2013.

TELES, V. M. **Extreme programming**. São Paulo: Novatec Editora, 2006.

WAZLAWICK, R. **Engenharia de software**: conceitos e práticas. Elsevier Brasil, 2013.

APÊNDICE A - COMPARAÇÃO DE TÉCNICAS DE ELICITAÇÃO PROJETOS UTILIZADAS NA ABORDAGEM TRADICIONAL E ÁGIL

Quadro 6: Percepção entre técnicas de elicitação utilizadas na abordagem tradicional e ágil

Tópicos	**Projeto A**	**Projeto B**
Análise de domínio		x
Análise documental	x	
Brainstorming	x	x
Brainwriting		x
Cenários (Scenarios)		x
Entrevistas	x	x
Features		x
Grupo focal	x	x
Mapa mental		
Observações		
Personas		x
Prototipação	x	x
Questionários	x	x
Refining Meetings	x	x
Reuniões	x	x
Sessões de JAD		
UI Mockups		
User Stories		**x**
Workshops		

Fonte: Autora (2022)

APENDICE B - COMPARAÇÃO DE TÉCNICAS DE ESPECIFICAÇÃO DE PROJETOS UTILIZADAS NA ABORDAGEM TRADICIONAL E ÁGIL

Quadro 7: Percepção entre técnicas de especificação utilizadas na abordagem tradicional e ágil

Tópicos	**Projeto A**	**Projeto B**
Cenários (Scenarios)		x
Data models		
Features		x
Goal Preference Model		
Modelos de Casos de Uso	x	
Ontologias		
Personas		x
Protótipos	x	x
Story card		x
Structured requirements list		x
Tasks	x	x
User stories		x
Wall	x	x
XSBD		
XXM (Extreme XMachine)		x

Fonte: Autora (2022)

APENDICE C - QUESTÕES E RESPOSTAS COM MEMBROS DO PROJETO A

Quadro 8: Questões e respostas com membros do Projeto A

Questões	**Respostas**		
	Pessoa 1	**Pessoa 2**	**Pessoa 3**
Q1 - Quais técnicas de elicitação de requisitos foram mais utilizadas durante o projeto?	Reuniões internas, entrevistas, brainstorming e questionários	Entrevista, prototipação, reunião com o time	Análise de documentação, entrevista, reuniões com a equipe, questionário, prototipação de baixa fidelidade.
Q2 - Quais técnicas de especificação/documentação foram adotadas durante o projeto?	Modelo de casos de uso e *Trello*	Documentação tradicional baseada em casos de uso e também contou com um protótipo de baixa fidelidade	Era utilizado o trello para o acompanhamento das atividades do time e a documentação foi feita a partir da especificação de casos de uso, em que, era descrito o caminho feliz, fluxo alternativo e de exceção.
Q3 - As reuniões internas e com o cliente para a validação do que estava sendo desenvolvido eram feitas com que frequência? Você acredita que quanto mais reuniões implica em maior acompanhamento do projeto?	As reuniões internas eram a cada semana online. Com o cliente só houve umas 2 ou 3 se não me engano. Sim, quando nos falamos regularmente é possível verificar o que foi desenvolvido, pendências e impedimentos do time.	Com o cliente foi um contato inicial no começo do projeto, ele não acompanhou de perto a evolução do produto. Já as internas aconteciam semanalmente com a professora coordenadora. Sim, devido ter poucas reuniões surgiam incertezas e dúvidas sobre pontos e algumas coisas eram feitas “às cegas”.	As reuniões internas aconteciam 1x por semana online via meet ou a pessoa enviava o status das atividades no grupo do Whatsapp. O contato com cliente foi um ponto falho durante o projeto, quase nunca ocorria. Sim, poucas reuniões ocasionam ociosidade, desinteresse, dúvidas e até ambiguidade de informações.
Q4 - Você acredita que a utilização de alguma(s) outras técnicas de elicitação e documentação poderiam ter resultado em melhor compreensão das atividades? Se sim, cite.	Sim, o uso de histórias de usuário ajudam muito na compreensão das ações do usuário quando ligadas a uma persona do sistema,	Sim, poderia encaixar histórias de usuário que são muito utilizadas em projetos ágeis e também exemplificar cenários para cada	Sim, acho que basear-se em features, quebrar o projeto em macro entregas, facilitaria o entendimento se unido a histórias de

	auxiliando tanto a nível técnico como a nível de negócio	história de usuário mostrando todos os cenários de ações dentro de uma página da aplicação	usuários e tornaria as entregas incrementais sem esperar a conclusão do projeto para ver a entrega de fato.
Q5 - Quais foram os maiores desafios encontrados durante o desenvolvimento do projeto?	Pouco contato com cliente, horário restrito dos participantes do projeto, desencontro de horários, escrita de requisitos ambíguas	Saber se o cliente está satisfeito com o que foi desenvolvido, entender e imaginar cenários em uma tela que estava sendo desenvolvida sem ter isso especificado na documentação. Horário disponível para o time se reunir.	Horários inflexíveis, poucos alinhamentos com o time e cliente. Pouca descrição de atividades, descritas de forma geral sem muitos detalhes.
Q6 - A expectativa dos stakeholders foi alcançada após a entrega do projeto? O que foi pontuado de pontos positivos e negativos na visão deles que você conseguiu interpretar?	Em partes, o projeto foi concluído, mas o cliente viu vários pontos de melhoria. Positivos: interface interativa, funções não pensadas agregaram valor. Negativos: pouco contato com o time desenvolvedor para o alinhamento das expectativas.	Sim, por parte da coordenadora do projeto e gerente foi, devido aos poucos momentos de alinhamento entregar o que foi feito, foi um bom trabalho dado o que tínhamos em mãos. Positivos: boa entrega mesmo sem revisão das funcionalidades. Negativos: poucas reuniões de alinhamento com time e cliente	Sim, embora tenham sido levantados muitos pontos de melhorias, foi uma entrega super válida. Positivos: interface amigável, funções principais funcionando, proposta de novas funcionalidades e melhorias foram levantadas. Negativos: comunicação entre time, revisão de código e testes, poucos encontros para alinhar andamento do projeto, objetivo e expectativa.

Fonte: Autora (2022)

APENDICE D - Questões e respostas com membros do Projeto B

Quadro 9: Questões e respostas com membros do Projeto B

Questões	Respostas		
	Pessoa 1	**Pessoa 2**	**Pessoa 3**
Q1 - Quais técnicas de elicitação de requisitos foram mais utilizadas durante o projeto?	Reuniões de refinamento, entrevistas com cliente, histórias de usuário, brainstorming, design thinking, features	Questionários, entrevistas, reuniões com o time, prototipação, histórias de usuário, features	Brainstorming, brainwriting, entrevistas, questionários, reuniões de alinhamento com tim e e cliente, histórias de usuário, features
Q2 - Quais técnicas de especificação/documentação foram adotadas durante o projeto?	Histórias de usuários, descrição BDD baseada em cenários de uso da aplicação, personas, protótipos	Personas, features, histórias de usuário, tasks, descrição de histórias com cenários do usuário, prototipação com figma, tasks no ambiente Azure Devops	Protótipos baseados em personas, histórias de usuário, prototipação.
Q3 - As reuniões internas e com o cliente para a validação do que estava sendo desenvolvido eram feitas com que frequência? Você acredita que quanto mais reuniões implica em maior acompanhamento do projeto?	Reuniões internas todos os dias “daily”, reuniões de planejamento e revisão a cada 15 dias com a presença do cliente. Sim, é muito importante o engajamento do cliente e partes interessadas no decorrer do projeto para alinhamento de expectativas.	Existiam dailies que ocorrem todos os dias, em que, o time passa as atualizações da sua atividade, além de mencionar os impedimentos. Já com o cliente os encontros acontecem a cada 15 dias nas plannings e reviews e também quando há necessidade o PO junta as dúvidas e impedimentos, levando para o cliente . Sim, é essencial ele estar por dentro do que é produzido, principalmente para o cliente final.	São realizadas diariamente para verificar o andamento das atividades direcionadas ao time, com o cliente ocorre no final e começo de cada sprint que dura 2 semanas que é o momento onde as expectativas são alinhadas e é mostrado o que foi desenvolvido no período estipulado da sprint. Sim, acredito que quanto maior o engajamento melhores serão os resultados do produto que está se desenvolvendo.
Q4 - Você acredita que a utilização de alguma(s) outras técnicas de elicitação e documentação poderiam ter resultado em	Não, acredito que as técnicas de histórias de usuário junto com	Não sei se existe alguma técnica que auxilia em atividades	Não, as técnicas utilizadas oferecem uma granularidade

melhor compreensão das atividades? Se sim, cite.	prototipação das telas auxilia muito no desenvolvimento, Não conheço outra técnica que poderia auxiliar melhor	técnicas de back-end, por exemplo. Das técnicas conhecidas, acho que não.	muito boa, é possível saber detalhadamente o que é preciso fazer em uma tarefa sem precisar de outros recursos.
Q5 - Quais foram os maiores desafios encontrados durante o desenvolvimento do projeto?	Por vezes era validado um protótipo e quando passava para estilizar o cliente sugeria outros pontos não mapeados e aprovados antes.	Complexidade de algumas páginas precisaram de algumas reuniões fora do planejamento.	Solicitação de melhorias dos clientes em páginas já integradas e aprovadas, “pensei nisso aqui agora, será que ficaria bom se a gente fizesse?”
Q6 - A expectativa dos stakeholders foi alcançada após a entrega do projeto? O que foi pontuado de pontos positivos e negativos na visão deles que você conseguiu interpretar?	Sim, com certeza, ficaram super satisfeitos, é tanto que foi renovado. Positivos: descrição de atividades bem detalhadas unindo a entrega incremental facilitou visualização do andamento das tarefas e levantar prioridades. Negativos: não sei	SIM, fomos super elogiados, como tínhamos entregas constantes ficou mais fácil vendo pontos de falha e já ir corrigindo. Positivos: time super comprometido e cliente sempre presente validando tudo a cada sprint. Negativos: lentidão em alguns serviços do back e não foi resolvido a tempo.	Sim, foi um grande sucesso, conseguimos agregar valor em um sistema utilizado constantemente para uma empresa a nível mundial. Positivos: time muito comunicativo, produtivo e proativo junto com cliente engajado, projeto bem documentado, impedimentos resolvidos sem delongas. Negativos: no começo os códigos não eram padronizados, quando foi detectado precisou de refatoração.

Fonte: Autora (2022)