ESCOLA POLITÉCNICA DA UNIVERSIDADE DE SÃO PAULO
BACHARELADO EM ENGENHARIA DE PRODUÇÃO

CHRISTIAN KENITI ASAMURA HUKAI

ANÁLISE E IMPLEMENTAÇÃO DE MELHORIAS DE CICLO DE VIDA DE DESENVOLVIMENTO DE SOFTWARE EM UMA FINTECH

SÃO PAULO – SP
2021

ESCOLA POLITÉCNICA DA UNIVERSIDADE DE SÃO PAULO
BACHARELADO EM ENGENHARIA DE PRODUÇÃO

CHRISTIAN KENITI ASAMURA HUKAI

ANÁLISE E IMPLEMENTAÇÃO DE MELHORIAS DE CICLO DE VIDA DE DESENVOLVIMENTO DE SOFTWARE EM UMA FINTECH

Trabalho de Formatura apresentado à Escola Politécnica da Universidade de São Paulo, como requisito para o recebimento do Bacharelado em Engenharia de Produção

SÃO PAULO – SP
2021

ESCOLA POLITÉCNICA DA UNIVERSIDADE DE SÃO PAULO
BACHARELADO EM ENGENHARIA DE PRODUÇÃO

CHRISTIAN KENITI ASAMURA HUKAI

ANÁLISE E IMPLEMENTAÇÃO DE MELHORIAS DE CICLO DE VIDA DE DESENVOLVIMENTO DE SOFTWARE EM UMA FINTECH

Trabalho de Formatura apresentado à Escola Politécnica da Universidade de São Paulo, como requisito para o recebimento do Bacharelado em Engenharia de Produção

Orientador: Professor Doutor Mauro de Mesquita Spinola

SÃO PAULO – SP
2021

Catálogo-na-publicação

Hukai, Christian Keniti Asamura
Análise e implementação de melhorias de ciclo de vida de desenvolvimento de software em uma fintech / C. K. A. Hukai – São Paulo, 2021.
94 p.

Trabalho de Formatura – Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Produção.

1. Metodologia de projetos 2. Tecnologia 3. Ciclo de Vida de Desenvolvimento de Software I. Universidade de São Paulo. Departamento de Engenharia de Produção II. T.

AGRADECIMENTOS

Inicialmente gostaria de agradecer à empresa do estudo de caso, a *Nexoos Sociedade de Empréstimo Entre Pessoas*, por ter disponibilizado a oportunidade de aplicar a metodologia desenvolvida neste trabalho. Seu ambiente aberto para críticas e opiniões possibilita o aprendizado de todos os seus colaboradores, tornando-se propício para inovação. Sobretudo, seus líderes que me desafiaram e me apoiaram em diversas aventuras: Naufal, Khei, Deco e Nico.

Ao meu orientador, professor doutor Mauro de Mesquita Spinola, que me acompanhou durante a jornada de desenvolvimento deste trabalho, cuja colaboração tornou-se indispensável para o aprendizado e a realização do trabalho.

À minha família: meu pai, minha mãe, meus dois irmãos, e à minha irmã, que estiveram do meu lado durante a minha vida, servindo como pilar na fundação de quem eu sou.

E, sobretudo, aos desenvolvedores da empresa do estudo de caso: Francisco, Marinaldo, Gyordano e Marlon, que estiveram lá como principal equipe atuadora nas mudanças e implementações realizadas.

RESUMO

Com a crescente demanda por sistemas de informação, desde websites ou aplicativos para o usuário, até sistemas de gestão internos de uma empresa, a necessidade pela diversificação das metodologias para um projeto acompanha tal crescimento, visando a obter resultados mais eficazes e rápidos. Este trabalho visa a explorar algumas metodologias emergentes para desenvolvimento de software, em uma startup *Nexoos Sociedade de Empréstimo Entre Pessoas*. A *Nexoos* atua no mercado de crédito para pequenas e médias empresas (PMEs) no modelo *Peer-to-Peer*, cujos investidores são pessoas físicas, que dão capital para pessoas jurídicas, em troca do pagamento de juros, pelo empréstimo. A startup tem cerca de 50 funcionários, com um crescimento grande e contínuo nos últimos anos de operação. A principal dor identificada na empresa, identificada pela alta gestão, é a ineficiência na metodologia de projetos de tecnologia. Para resolvê-la, este trabalho foi desenvolvido, estudando as diversas metodologias de projeto presentes no mercado, utilizá-las internamente e compreender melhor os gargalos e oportunidades que existem no processo atual. Dessa forma, foi implementada na empresa uma metodologia para melhorias do ciclo de vida de desenvolvimento de software, para que fosse possível aumentar a entrega de valor pelas equipes de tecnologia e de produto. Como resultado da implementação, houve aumento do valor das entregas pelas equipes de tecnologia e de produto, mas com uma deterioração da satisfação dos colaboradores dessas duas equipes, restando alguns pontos de melhoria para serem aplicados nos próximos passos. Todas as análises realizadas buscaram compreender o problema, encontrar soluções e priorizá-las, de modo que ao fim de sua implementação fossem avaliados os resultados obtidos e definido o que seria melhorado.

***Palavras-chave**: metodologia de projetos, tecnologia, ciclo de vida de desenvolvimento de software*

ABSTRACT

There is an increasing demand for information systems, ranging from websites to phone applications, or even internal management systems for a company; following this growth, there is also an increase of diversification of projects methodologies, aiming to obtain more efficient and faster results. This paper focuses in applying some of the emerging methodologies for software development in a startup called *Nexoos Sociedade de Empréstimo Entre Pessoas. Nexoos* operates in the credit market for small and medium enterprises (SMEs) with the *Peer-to-Peer* model, in which investors are individuals who fund companies in exchange for interests for the loan. The startup has up to 50 employees, with a large and continuous growth in the last years of operation. The biggest challenge the company deals, identified by the upper management, is the inefficiency of the tech project methodology. To solve it, this work was developed, studying a range of project methodologies in the market, using them internally and better comprehending the process bottleneck and opportunities that exists. Thus, it was implemented improvements in the project methodology, increasing the efficiency of the software development life cycle of the company. This improved the value delivery by the technology and product teams; however, it brought dissatisfaction of the employees from these two teams, leaving some improvements to be done in the following steps. All the analysis were made to comprehend the problem, find the solution and prioritize them, so that by the end of its implementation, the results could be measured and the following improvements could be defined.

***Keywords**: project methodology, technology, software development life cycle*

Lista de Figuras

Sumário

1. Introdução

Este capítulo introduz a empresa onde foi realizado este estudo de caso, a Nexoos Sociedade de Empréstimo Entre Pessoas, junto aos objetivos do trabalho. O autor realizou estágio na startup no início do ano de 2019 até a conclusão do ano de 2020, passando pela área de produto. Esse estudo de caso é um trabalho de formatura da Escola Politécnica da Universidade de São Paulo, cuja finalidade é de explorar os conceitos aprendidos em sala de aula e aplicá-los em um caso real, explorando seus resultados e suas dificuldades.

1.1. O mercado de crédito *Peer to Peer* (P2P)

A empresa do estudo de caso atua como provedora de crédito para Pequenas e Médias Empresas (PMEs) na modalidade *Peer to Peer* que, neste tipo de operação financeira, o capital é fornecido por um investidor pessoa física através de um título creditício (CCB, cédula de crédito bancária). Com isso, o devedor receberá o capital investido pelo credor que, em troca, receberá os juros da operação.

Assim, para que a operação faça sentido para o credor, os juros arrecadados devem ser maiores do que os inadimplência de seus devedores; caso contrário, a operação trará prejuízo e desinteresse pelo lado do investidor. Em suma, o mercado de crédito P2P trata-se de um “aluguel do dinheiro” fornecidos por pessoas físicas, em que a taxa do aluguel, o juro, é proporcional ao risco de o devedor honrar a sua dívida.

Diferentemente de operações comuns de crédito, em que grandes instituições financeiras emprestam o capital para as empresas, a operação P2P faz uso do capital de credores pessoa física. A necessidade de trocar o fornecedor do capital nesse tipo de operação financeira nasceu pela falta de oferta no mercado brasileiro por uma taxa de juros menor.

A taxa de juros alta no Brasil é causada pelo efeito do monopólio no sistema bancário em que cinco bancos comerciais chegavam a deter 84,8% do mercado de crédito em 2018 (Martello, 2019). Este monopólio causava o aumento nas taxas de juros e a diminuição da qualidade das análises de crédito; portanto devedores que na realidade são bons pagadores acabavam sendo classificados no mesmo grupo de risco de devedores que apresentam uma alta probabilidade de inadimplência. Por isso, muitas startups têm desenvolvido serviços no mercado de crédito, com o objetivo de oferecer taxas mais justas aos devedores, junto a um atendimento mais personalizado e próximo ao cliente.

Sobretudo, além das altas taxas de crédito pelos grandes bancos comerciais (Martello, 2019), a taxa SELIC diminuiu desde 2015 (Banco Central do Brasil, 2020), ocasionando a migração do investimento de pessoas físicas da renda fixa para a renda variável, aumentando a oferta de capital para essa modalidade de investimento. Com o aumento da oferta de capital por pessoas físicas e a demanda por capital por PMEs, o tipo de operação financeira P2P é fomentado.

Assim, a startup brasileira *Nexoos* foi fundada em 2016 atuando no mercado de P2P, tornando-se uma instituição financeira regulamentada pelo BACEN em 2019, sendo a primeira SEP (Sociedade de Empréstimo Entre Pessoas) operacional do Brasil.

1.2. Descrição da área de produto

A área de produto da *Nexoos* foi o principal setor a ser explorado por este trabalho de formatura. Portanto neste capítulo é definido o seu escopo, seus objetivos e suas dificuldades.

A área de produto intermedia os *stakeholders* (as áreas de negócio, que têm necessidade de tecnologia) com a área de desenvolvimento (responsável pelo desenvolvimento de novas funcionalidades, melhorias e correções em software). Por exemplo: ao surgir uma necessidade da mesa de crédito, que é um *stakeholder*, para criar uma nova funcionalidade, fica a cargo da área de produto receber e compreender essa demanda, estruturar e elaborar as necessidades de tecnologia, repassar para a área de desenvolvimento e, por fim, acompanhar o seu andamento.

Contudo, conforme vem crescido as demandas dos *stakeholders,* as áreas de produto e de desenvolvimento não estavam sendo capazes de supri-las, pois ambas se sentem ineficientes ao desenvolver. Uma das dores atuais, é a “planilha de demandas”, onde é realizado o contato entre a empresa e a área de produto, acessível por qualquer funcionário da empresa. Dessa planilha, poucas são as demandas que de fato prosseguem no processo de desenvolvimento, já que a maioria das ideias sugeridas são negadas no início, consumindo um tempo desnecessário da área de produto.

Além disso, os processos da área de produto não estão bem padronizados entre todos os colaboradores da área, com uma divergência em métodos para documentar a demanda dos *stakeholders*, como o uso de ferramentas diferentes, uma

documentação dividida em diversos lugares... Isso causa a falta de alinhamento com os *stakeholders*, tanto dos requisitos do projeto, como também as expectativas do que será entregue, já que muitas das informações são perdidas na elaboração do produto.

Os rituais são outra dor identificada, pois alguns são alguns excessivos e outros são insuficientes. Por exemplo, enquanto que a equipe de desenvolvimento se sente sobrecarregada com reuniões que não agregam valor para a equipe, a equipe de *stakeholders* se sente ignorada pela falta de alinhamentos durante o processo de desenvolvimento de novas funcionalidades.

Algumas soluções já foram adotadas, como a contratação de mais funcionários, ou até mesmo a troca de gestores; todavia, o processo ainda se encontra com as mesmas dificuldades.

1.3. Objetivos do trabalho

Com as dificuldades da área de produto em mente, este trabalho visa à melhoria dos processos de desenvolvimento de software da *Nexoos*, enfatizado pela alta gestão como o principal problema, necessitando de uma reavaliação da metodologia atual. Seu objetivo é elaborar e avaliar uma nova metodologia de desenvolvimento para a empresa. O resultado esperado é, com base nessa metodologia, melhorar a eficiência da equipe de produto e de tecnologia.

1.4. Estrutura do Trabalho

Por fim, este trabalho de formatura foi estruturado em 8 capítulos para descrever a jornada desde a revisão da literatura a ser aplicada, o desenvolvimento e a aplicação de uma metodologia elaborada pelo autor, o relato da implementação e os resultados obtidos.

Neste primeiro capítulo foi apresentado o problema de maneira introdutória, situando o leitor do contexto da empresa *Nexoos* e os problemas em questão, junto aos objetivos do trabalho;

Para o segundo capítulo, será detalhada a revisão de literatura construída ao decorrer do trabalho, em que os conhecimentos necessários foram estudados, desenvolvidos e documentados nesta seção;

No terceiro capítulo, será construída a metodologia para a atuação do trabalho de formatura, como um planejamento das ações que serão realizadas para a aplicação do projeto na empresa;

Para o quarto capítulo será iniciada a aplicação da metodologia desenvolvida, com a identificação e diagnóstico dos problemas que a área de produto sofre;

Com os problemas em mãos, o quinto capítulo planejará definir a solução desenvolvida pelo autor, tanto como o planejamento da implementação da melhoria nos processos de desenvolvimento em software;

No sexto capítulo serão detalhados os relatos da implementação elaborados previamente, servindo para descrever como foi realizado e suas dificuldades, tanto como a reação dos colaboradores e os esforços investidos neste processo;

O sétimo capítulo trará os resultados da implementação dos novos processos de desenvolvimento em software, com uma análise de seus pontos positivos e negativos;

E, por fim, o oitavo capítulo trará a conclusão do trabalho de formatura, com os aprendizados e reflexões que foram agregados ao autor.

2. Revisão de Literatura

Para o embasamento do trabalho de formatura, foi necessária uma revisão do conhecimento teórico obtido durante o curso de graduação, além de uma pesquisa adicional para aprofundamento dos temas. Nesta seção, a revisão bibliográfica será apresentada, através de pesquisas em artigos científicos e livros.

2.1. Modelos CVDS (Ciclo de Vida de Desenvolvimento de Software)

Modelos CVDS definem quais os processos e etapas percorridos pelo desenvolvimento da construção de software, variando de metodologia em metodologia para atender às especificidades de cada projeto (Khan, Parveen, & Sadiq, 2014). A escolha do CVDS correto traz maior valor para os *stakeholders*, tanto quanto diminui o custo de desenvolvimento, quanto também traz maior qualidade para manutenções futuras.

Estes modelos podem ser classificados em três categorias: HDM (Metodologias de Desenvolvimento Pesadas); LDM (Metodologias de Desenvolvimento Leve); e HYDM (Metodologias de Desenvolvimento Híbridas).

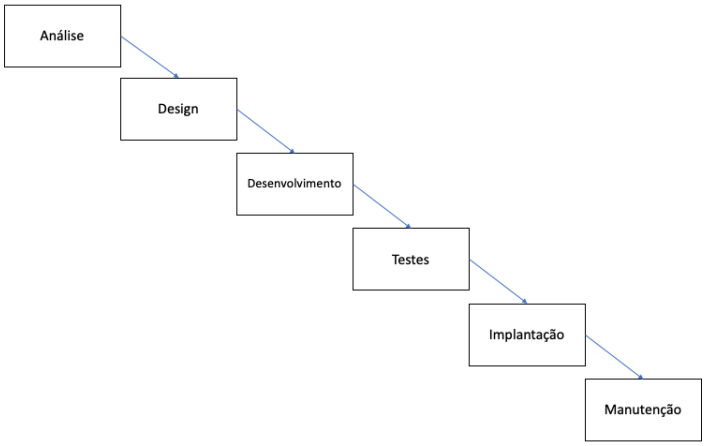
2.1.1. HDM

Os modelos de HDM são constituídos por etapas bem definidas (Khan, Parveen, & Sadiq, 2014), como com a definição dos requisitos, a construção da solução, os testes e a implantação.

De forma geral, KHAN trata os HDMs como estruturas bem planejadas, aplicáveis a projetos cuja complexidade é grande e os objetivos são bem esclarecidos, trazendo os benefícios da menor incerteza e da segurança; contudo, para projetos cujos objetivos não estão bem definidos, esses modelos se tornam inviáveis por apresentarem uma estrutura mais rígida, proporcionando dificuldades em aplicar mudanças nos requisitos ao longo do projeto.

Um dos mais conhecidos modelos HDM é o Cascata, cujas etapas de desenvolvimento estão representadas na figura abaixo. De acordo com Balaji e Murugaiyan, (S.Balaji & Murugaiyan, 2012) este modelo é sequencial, em que cada etapa precisa ser preenchida e concluída para iniciar a próxima, não podendo ter duas etapas sobrepondo.

Figura 1 – Modelo Cascata



Fonte: Metodologias Clássicas, 2013.

Alguns dos benefícios do modelo cascata é a extensa documentação e planejamento do projeto, o que traz um bom alinhamento para o início do desenvolvimento. Em contrapartida, como o uso do produto pelo *stakeholder* foi realizado somente após o planejamento, são geradas muitas alterações que não foram cogitadas no decorrer das fases posteriores. Com uma estrutura rígida, o modelo cascata não permite essas alterações até o início de um novo ciclo, seguindo as seguintes etapas:

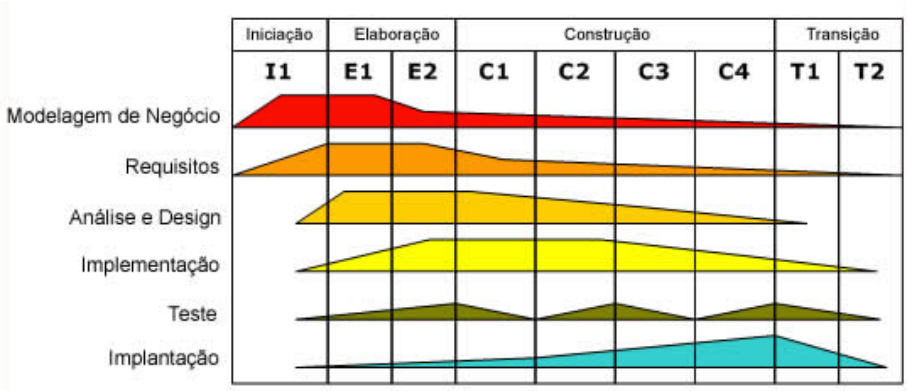
- **Análise**, quando são elaborados os requisitos e as necessidades da funcionalidade, com sua documentação antes do início do desenvolvimento do produto;
- **Design**, quando são elaboradas a experiência do usuário e a interface do sistema;
- **Desenvolvimento**, quando o software é construído;
- **Testes**, quando o usuário faz uso do sistema;
- **Implantação**, quando o sistema fica ao ar, para uso de todos;
- **Manutenção**, a etapa em que conforme surgirem novas funcionalidades ou erros, os desenvolvedores aperfeiçoam o projeto.

Outro exemplo é o Modelo RUP (*Rational Unified Process*) um modelo de desenvolvimento de software criado pela *Rational Software Corporation*, uma divisão da IBM. Esse modelo, novamente, é utilizado para desenvolvimento de sistemas muito

grandes e complexos que envolvem diferentes áreas de negócio, cujos sistemas reutilizam funcionalidades de outros sistemas. Ele segue as seguintes etapas:

- **Iniciação**, quando é elaborada a definição do que construir, seus custos, seus riscos e a arquitetura do projeto;
- **Elaboração**, quando é elaborado como construir, quais são os requerimentos do projeto e qual o tamanho da equipe, junto à validação da arquitetura anteriormente elaborada;
- **Construção**, quando inicia o desenvolvimento do projeto, junto também dos testes e da documentação;
- **Transição**, quando os usuários iniciam o uso do software, para que, no final, o software seja implementado no servidor de produção.

Figura 2 – Modelo de CVDS do RUP



Fonte: MARTINEZ, Marina, 2021

Por fim, outro exemplo estudado de modelos de HDM é o modelo Espiral, eficiente para aplicação de grandes projetos, que, por serem muito custosos, trazem riscos com prejuízos muito grandes. As principais etapas do modelo espiral são (Khan, Qurashi, & Khan, 2011):

- **Decisão dos Objetivos**, quando os objetivos para cada etapa do projeto são definidos;
- **Análise e Redução dos Riscos**, quando os riscos-chave do processo são identificados;

- **Desenvolvimento e Validação**, quando os desenvolvedores iniciam o código;
- **Planejamento**, quando o projeto é revisado e preparado para a próxima etapa do espiral.

2.1.2. LDM

As metodologias leves, também chamadas ágeis são caracterizadas por serem adaptáveis no decorrer do projeto por não terem um planejamento muito extenso e por essa razão são capazes de sofrer ajustes de maneira mais eficiente (Khan, Parveen, & Sadiq, 2014). Sobretudo, são metodologias cooperativas, visando ao alinhamento da equipe de maneira constante, além de serem objetivas e simples, para que sejam de fácil compreensão e que não gerem perca de eficiência devido a compreensão do processo. Esse grupo de modelos seguem o Manifesto Ágil, seguindo seus 12 princípios (Qureshi & Hussain, 2008):

1. A satisfação do cliente deve existir da entrega do software até ao seu uso, trazendo melhorias para que agreguem valor;
2. As mudanças devem ser acolhidas em formato de requisitos novos, mesmo que em últimas etapas de desenvolvimento;
3. Os primeiros incrementos devem surgir rapidamente, em poucas semanas, e também posteriormente à conclusão do software completo;
4. A construção do software deve ser feita junto aos *stakeholders* e desenvolvedores diariamente, para o acompanhamento contínuo;
5. A construção do projeto deve estar de acordo com o *stakeholder*, seguindo seus interesses;
6. Reuniões devem ser realizadas de maneira periódica com todos os *stakeholders*, trazendo alinhamento;
7. Software funcional deve ser a principal medida para o progresso, e não somente o que está no papel ou uma funcionalidade que ainda não esteja completa;
8. Todos do projeto devem manter um ritmo constante para que o projeto seja mais previsível e não tenha problemas de *burnout* (desenvolvimento intenso com muitas entregas; seguido de poucas entregas);

9. Atenção contínua em excelência técnica e design aperfeiçoam a agilidade, de tal maneira que evita retrabalhos durante o projeto;
10. Respeito e confiança devem ser mantidos entre todos os membros do projeto;
11. Melhores arquiteturas, requisitos e designs surgem de times que são autônomos e se auto organizam;
12. Regularmente, os times devem refletir em como ser mais eficientes trazendo melhorias contínuas para os processos.

Um exemplo de metodologia ágil é o ASD (Desenvolvimento de Software Adaptável) cuja estrutura prevê que os projetos não caiam em "caos", propondo soluções para grandes sistemas. As etapas do projeto são separadas em entregas rápidas: comunicação, planejamento, análise, design, desenvolvimento, teste e implantação. O desenvolvimento é um processo interativo, com incrementos.

Já para o processo de construção de software AM (Modelagem Ágil), seu foco é de reduzir a documentação e a elaboração ao que for possível, explorando a comunicação, simplicidade, feedback e coragem dos seus colaboradores, trazendo a necessidade do cliente ao foco.

Existem muitos outros modelos ágeis, dentre os quais o *Kanban*, que será melhor tratado no capítulo 2.2 e o *SCRUM*, que será melhor tratado no capítulo 2.3.

2.1.3. HYDM

Por fim, existem também metodologias híbridas, que combinam aspectos das metodologias leves com as metodologias pesadas. As metodologias híbridas são fundamentadas no princípio que, enquanto as metodologias ágeis trazem maiores flexibilidades ao escopo dos projetos, as metodologias pesadas trazem melhores análises e planejamentos prévios ao desenvolvimento; as metodologias híbridas então combinam ambas essas características.

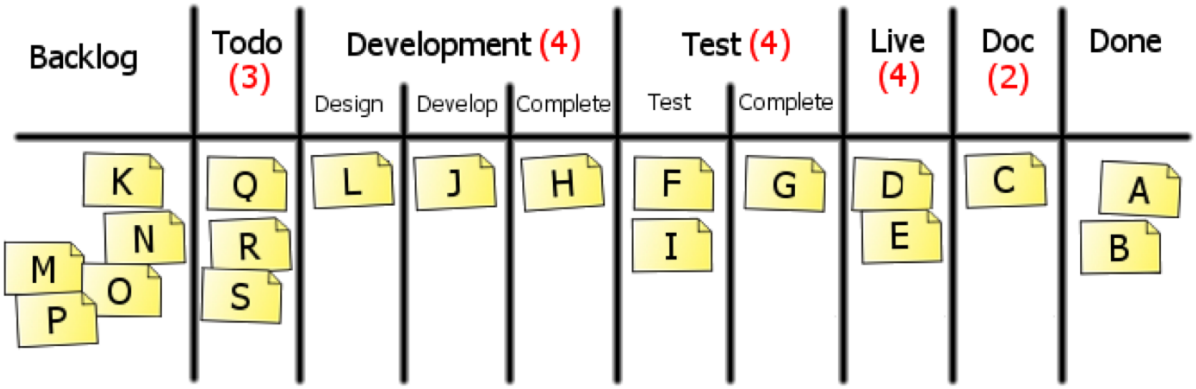
2.2. Modelo ágil *Kanban*

O *Kanban* foi criado na década de 1950, usado pelo sistema de administração JIT (*Just In Time*) para controlar o fluxo em que uma peça teria que percorrer em diferentes estações no processo de manufatura (Ahmad, Markkula, & Ovio, 2013). De forma simplista, o *Kanban* é um cartão que carrega as informações de um produto para ser percorrido pelo chão da fábrica.

Em 2004 a Microsoft passou a utilizar o quadro do *Kanban* para organizar as tarefas de uma equipe. Essas tarefas percorriam por “etapas”, da mesma maneira que um produto em uma fábrica: elas eram representadas por um cartão que caminhava pelo quadro *Kanban* conforme era concluído uma etapa do processo de desenvolvimento. Desde então, o uso do quadro *Kanban* foi aperfeiçoando em modelos de CVDS.

Nesse modelo, conforme chegassem novas demandas de *stakeholders*, estas seriam descritas em cartões que seriam colocados no quadro *Kanban*. O quadro é dividido em colunas, em que cada um representa uma etapa do processo que a tarefa poderá passar prosseguindo até o final, conforme figura abaixo.

Figura 3 – Exemplo de quadro *Kanban*



Fonte: JHA Ankita, 2016

Cada coluna tem um limite de WIP (Trabalho em Progresso) escrito em vermelho na figura acima, que representa o limite de cartões que essa coluna poderá conter. Por fim, após o percorrer do quadro *Kanban* (da esquerda para a direita), a tarefa está concluída.

A revisão sistemática de literatura de Ahmad, Markkula e Ovio identificou alguns dos benefícios da metodologia *Kanban* aplicadas em diversas empresas (Ahmad, Markkula, & Ovio, 2013). Por exemplo, o modelo traz alinhamento interno da equipe do processo inteiro de desenvolvimento, já que o quadro de *Kanban* é visível

para todos da equipe. Foi também identificado a melhoria de qualidade do software produzido, pois como as entregas são mais curtas, elas podem ser avaliadas pelos *stakeholders* de maneira mais rápida, o que traz um atendimento melhor às necessidades e requisitos pelos projetos.

Todavia, uma das principais desvantagens notadas por Ahmad, Markkula e Ovio foi a dependência do modelo *Kanban* com outras metodologias ágeis, atuando como um complemento. Por isso, costumeiramente as empresas combinam a metodologia *Kanban* com outras metodologias ágeis.

Por fim, outra dificuldade agora apontada por Patil e Neve (Patil & Neve, 2018) é a restrição nas tarefas do dia-a-dia de um desenvolvedor. O *Kanban* apresenta objetivos claros que precisam ser realizados, dando menor flexibilidade para um desenvolvedor aprofundar em temas diferentes (como a manutenção e melhoria de códigos não relacionados àquela tarefa) que são rotinas que não entregam valor ao *stakeholder*, mas agregam valor ao negócio.

2.3. Modelo ágil *SCRUM*

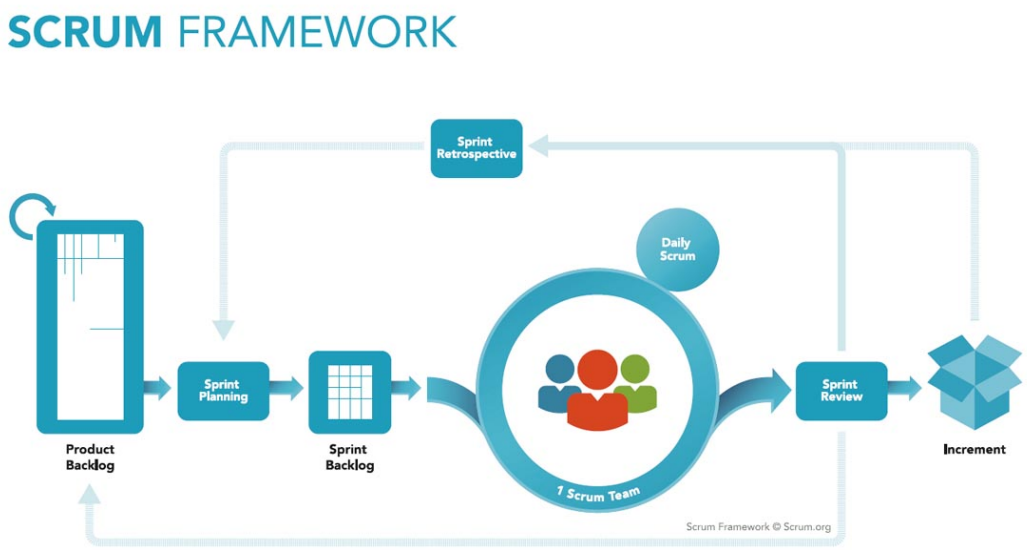
O modelo *SCRUM* sugere uma maneira de trabalhar com flexibilidade nos requisitos a cada iteração, sem a necessidade de processos muito rígidos. Com isso, o modelo visa a aumentar a velocidade de desenvolvimento, trazer um melhor alinhamento individual e organizacional, como também melhorar a qualidade de vida dos colaboradores (Srivastava, Bhardwaj, & Saraswat, 2017).

Na estrutura, existem três papeis: *SCRUM Master*, que é o responsável por eliminar impedimentos; Time *SCRUM*, que consiste numa diversidade de atuações, como desenvolvedores, testadores, designers; e o *Product Owner*, que representa o lado do negócio, definindo quais são as prioridades e requisitos do projeto.

O *Product Owner* é o responsável por acompanhar o *Product Backlog* que é a documentação das novas funcionalidades solicitadas por *stakeholders*. Ele quem planeja a *Sprint*, que é a definição do que precisa ser realizado num prazo de 1 a 3 semanas pela equipe, o *Sprint Backlog*. Durante o desenvolvimento da *Sprint*, não pode ser alterado o que foi definido, com alinhamento diárias do que foi realizado e, no fim de cada *Sprint*, é revisada a entrega, com intuito de visualizar o que pode ser melhorado, o que foi entregue e o que foi bem sucedido. Posteriormente, o processo

repete-se a cada *Sprint*, trazendo um fluxo de melhoria contínua, conforme figura abaixo.

Figura 4 – Processo do *SCRUM*



Fonte: RAMOS Leonardo, 2019

2.4. Principais diferenças entre *Kanban* e *SCRUM*

De acordo com Patil e Neve (Patil & Neve, 2018), o *SCRUM* tende a ter mais artefatos e cerimônias, trazendo uma estrutura do CVDS em *Sprints*, inspecionando e adaptando o projeto através de processos empíricos, focando na entrega de uma *Sprint*, evitando alterações do que foi elaborado semanalmente.

Em paralelo a uma metodologia mais metódica, o modelo do *Kanban* não tem um objetivo planejado em uma *Sprint*; mas sim num fluxo interativo, em que as tarefas entram no quadro do *Kanban*, passam por estágios e são entregues. O foco passa a ser a diminuição do tempo de ciclo de cada tarefa do que em uma entrega (conjunto de tarefas) específica.

Existe também a diferença do tratamento do WIP, em que no *SCRUM* ele é definido nas cerimônias de Planejamento, em paralelo que no *Kanban* ele é limitado pela quantidade de tarefas se encontram naquele estágio.

Em suma, o *SCRUM* traz uma estrutura mais bem elaborada, porém rígida, do que precisa ser realizado, incluindo um planejamento prévio mais metódico, com uma estrutura menos flexível para mudanças de direcionamento; enquanto que o *Kanban* traz uma estrutura menos elaborada, em que somente as demandas essenciais são levadas para o *Backlog*, trazendo maior facilidade para alterar o que precisa ser feito conforme o passar do tempo.

Ambas as metodologias apresentam vantagens e desvantagens, cabendo a necessidade da compreensão do cenário e da disponibilidade da equipe para seguir uma delas.

2.5. Cerimônias de metodologias ágeis

Cerimônias de metodologias ágeis são as reuniões realizadas de maneira periódica pela equipe, caracterizadas de maneira generalista em 5 partes (Khan, Siddiqui, Waheed, Hassan, & Duc, 2020):

- **Informar**: são reuniões de alinhamento, para que todos compreendam como está o status do projeto, quais são seus impeditivos e o que já foi realizado;
- **Planejar**: levando em conta o que deverá ser realizado, definir quais são as prioridades, junto aos benefícios e aos riscos envolvidos das tarefas;
- **Refinamento**: das tarefas planejadas, realizar a definição de seus escopos e os custos da mão de obra;
- **Retrospectiva**: do que foi entregue, definir quais foram os valores das entregas como também quais foram as dificuldades enfrentadas pela equipe;
- **Investigação**: das dificuldades enfrentadas, compreendê-las melhor e evitar futuros riscos (como por exemplo, retrabalho de tarefas realizadas).

Na metodologia *SCRUM* por exemplo, são realizadas reuniões diárias para informar a situação do projeto por cada membro; uma reunião de planejamento para elaborar o que será feito na próxima *Sprint* e aprofundamento de cada tarefa; e outra reunião de retrospectiva para avaliar a entrega realizada, junto à investigação dos problemas enfrentados durante a entrega.

2.6. CVD (Entrega Contínua de Valor)

CVD é uma prática para garantir entregas rápidas para o *stakholder* de maneira frequente e processual (Khan, Siddiqui, Waheed, Hassan, & Duc, 2020) muito presente em várias metodologias ágeis como o *SCRUM*, *Nexus*, *EVO* ou *LeSS*.

A prática do CVD inicia na identificação dos *stakeholders* principais: são aqueles que carregam o conhecimento para trazer funcionalidades com maior valor para o negócio. Um diretor, por exemplo, pode não necessariamente ser um *stakeholder* principal quando comparado a um gerente de campo, que pode conhecer melhor as dores e necessidades do cliente.

Com os *stakeholders* principais, a equipe elabora quais as funcionalidades deverão ser focadas, conectando-as com a estratégia de outras funcionalidades. Logo, essas novas funcionalidades deverão ser capazes de ser reutilizáveis em outras ferramentas para agregar maior valor final e escalabilidade do negócio.

Por fim, a prática do CVD estimula a mensuração do benefício das funcionalidades desenvolvidas pela equipe, tal como os ganhos diretos e indiretos para a empresa.

2.7. Entrevistas semiestruturadas

Entrevista semiestruturada é uma metodologia de entrevista que, ao mesmo tempo em que é planejado quais as informações que deseja se obter do entrevistado, lhe dá também liberdade para poder responder e aprofundar em diferentes temas.

No capítulo 19 do Guia de Avaliações Práticas (Newcomer, Hatry, & Wholey, 2004) são descritas as boas práticas para uma entrevista semiestruturada, iniciando pela escolha do grupo de entrevistados. Por exemplo, para um espaço populacional muito grande, é possível escolher amostras randomizadas; todavia para uma população pequena, é necessário escolher pessoas chave para o processo, que são aquelas que poderão trazer mais informações sobre o processo.

Com a lista de entrevistados, deve-se realizar um convite, explicando a importância da entrevista de uma forma agradável, como por exemplo, um convite por conversa traz uma conexão mais pessoal do que um e-mail. O Guia de Avaliações Práticas relata que um questionamento comum pelos entrevistados é o da duração da entrevista, cuja resposta aconselhável é de ser o mais amplo o possível, como por exemplo “ela não deve durar mais que...”, já que um tempo muito pequeno pode trazer

uma expectativa errada e, da mesma forma, um tempo muito grande pode desencorajar a participação.

É possível então iniciar o roteiro da entrevista que deve ser planejado cuidadosamente. Segue alguns dos pontos destacados pelo Guia de Avaliações Práticas:

- Caso o número de tópicos para seguir seja muito grande, priorizar quais são os problemas essenciais para serem desenvolvidos;
- Para perguntas fechadas (ou seja, que são de sim ou não; ou de melhorou ou piorou) uma boa prática é perguntar o porquê;
- Além de garantir confidencialidade durante as entrevistas, uma boa estratégia é de mostrar os pontos em questão de forma neutra, tal como “Um grupo de pessoas citou que...” acompanhado de “Como você vê essa situação?”;
- O roteiro da entrevista não é definitivo, podendo tomar outros caminhos, porém esses caminhos precisam estar bem elaborados para não desfocar;
- Começar com perguntas e problematizações mais fáceis é uma boa maneira para descontrair o entrevistado, trazendo perguntas que são mais relevantes no meio;
- Em perguntas que são mais delicadas, é interessante questionar os pontos positivos sobre os problemas “Sobre essa questão, quais você acredita serem os pontos fortes da equipe? (...) Agora, o que você acredita que pode ser melhorado?”;
- O entrevistado no fim da entrevista precisa se sentir como um amigo do entrevistador, alguém que ele confia e pode falar sobre os problemas;
- A cada entrevista, o roteiro pode então ser revisado para compreender melhor quais questões podem ser melhor abordadas.

Com o roteiro em mãos, é possível iniciar a entrevista: boas impressões são importantes, logo é uma boa prática trajar uma vestimenta que seja melhor ou do mesmo nível que o entrevistado, como também levar materiais para tomar anotações durante o processo.

Em suma, as entrevistas semiestruturadas trazem uma forma mais amistosa e mais próxima ao entrevistado permitindo compreender melhor o problema. Como é

um tipo de entrevista que dura mais tempo, é ideal que traga novos conhecimentos e percepções para o problema, e não somente uma amostragem, como em questionários de formulário.

2.8. Análise de causa raiz (ACR)

A ACR é utilizada para investigação de causas raiz para problemas, como de eficiência, de qualidade, de segurança, de saúde, ambiental ou de qualidade, contribuindo para encontrar o “o quê”, “como” e “por quê”, prevenindo sua reincidência. De acordo com ROONEY e HEUVEL (Rooney & Heuvel, 2004), a análise não se abstém ao erro ocorrido, mas ela percorre pelos caminhos do por quê de tê-lo ocorrido.

Por exemplo, não é apropriado justificar a baixa produtividade por causa de um clima hostil, pois com essa informação ainda não é possível atuar sobre a problemática para que se possa trazer um aprendizado e uma oportunidade de melhora. As causas raiz são definidas por serem específicas por alguma causa facilmente identificável e controladas para que possam gerar recomendações eficientes para suas melhorias.

Os 4 passos abordados por Rooney e Heuvel (Rooney & Heuvel, 2004) são: coletânea dos dados, diagrama de fatores causais, identificação da causa raiz e por fim recomendações/implementações.

Primeiramente, são coletados os dados para a compreensão do evento; ou seja, conhecer e investigar os fatores causais e causas raiz através de entrevistas com os responsáveis junto ao conhecimento dos problemas em questão.

Posteriormente é produzido um diagrama de fatores causais que provê uma estrutura que organiza e analisa as informações obtidas no passo anterior, trazendo os defeitos do processo que gera o fator causal.

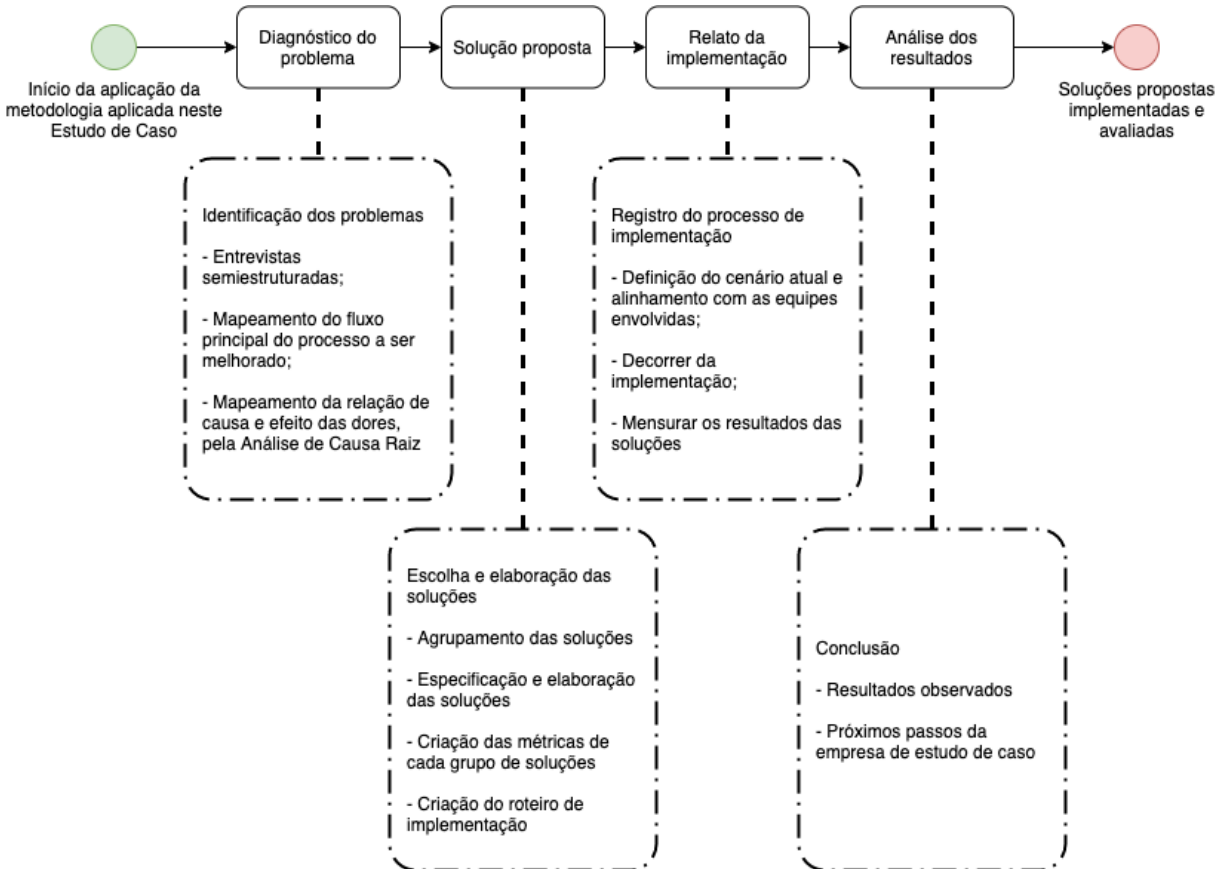
Posteriormente são identificadas as causas raiz, identificando quais foram as causas que contribuíram para o fator causal, para que os problemas ao redor dessas causas possam ser tratados.

E, por fim, é possível gerar as recomendações factíveis a partir da identificação das causas para evitar com que elas ocorram novamente.

3. Metodologia

Com o embasamento desenvolvido na revisão bibliográfica do capítulo anterior junto aos conhecimentos desenvolvidos durante o curso, esse capítulo tem como objetivo detalhar a metodologia que irá ser aplicada nesse estudo de caso. Essa metodologia deverá aprimorar o processo de desenvolvimento de software aplicado no cenário atual da empresa, seguindo conforme o fluxo de processos abaixo, que será descrito nos capítulos a seguir.

Figura 5 – Fluxo do processo de aplicação da metodologia



Fonte: Elaborado pelo autor

3.1. Diagnóstico do problema

O início desta metodologia se dará pelo diagnóstico do problema, em que serão realizadas: as entrevistas semiestruturadas, o mapeamento do fluxo principal do processo a ser melhorado e a Análise de Causa Raiz (ACR). Para as duas primeiras ferramentas, as dores identificadas serão consolidadas em quadro, para que possam ser utilizadas na Análise de Causa Raiz para o mapeamento da relação de causa e

efeito destas. Com essa imersão, será possível compreender quais são as maiores dores que afetam a área estudada, e a relação de causa e efeito entre elas.

3.1.1. Entrevistas semiestruturadas

As entrevistas serão conduzidas seguindo o Guia de Avaliações Práticas (Newcomer, Hatry, & Wholey, 2004) para que seja possível extrair o máximo de informação da maneira mais eficiente dos entrevistados.

Newcomer, Hatry e Wholey propõem a elaboração de um roteiro antes da realização das entrevistas, iniciando com perguntas mais descontraídas, como a explicação do propósito da conversa, para então avançar com perguntas mais decisivas, abrindo espaço para que os entrevistados possam se expressar.

Os entrevistados deverão ser colaboradores chave da empresa, para que seja possível extrair, compreender e estudar suas opiniões, de forma sistemática. Eles deverão representar as áreas problematizadas no trabalho, para trazer a visão de cada área como um todo e não uma expressão de seus problemas pessoais.

Com o roteiro e a escolha dos entrevistados, as entrevistas deverão ser realizadas individualmente, em que cada entrevistado terá entre 30 minutos e 1 hora para se expressar, com a imparcialidade do entrevistador.

Ao seu fim, as dores identificadas pelas entrevistas serão consolidadas em formato de quadro, segmentadas entre os envolvidos; neste caso, entre as áreas de produto, de tecnologia e dos *stakeholders*.

3.1.2. Mapeamento do processo de desenvolvimento de software

Através dos insumos obtidos no capítulo anterior, será possível identificar qual o processo de desenvolvimento de software atual da empresa. Este processo deverá corresponder ao fluxo principal de informação e de tarefas realizadas na empresa para o desenvolvimento de uma nova funcionalidade, desde o início da demanda de um *stakeholder* até a sua entrega.

O processo deverá ser dividido em etapas, equivalentes às etapas de CVDS (Ciclo de Vida de Desenvolvimento de Software) que S. BALAJI, MURUGAIYAN (S.Balaji & Murugaiyan, 2012) descreve nas metodologias de desenvolvimento de software do mercado. Embora que o processo de desenvolvimento de software de uma empresa possa não estar padronizado, é possível identificar qual é o padrão idealizado por todos das equipes.

Com o mapeamento deste processo, serão detalhadas as atividades e comunicação entre as áreas em formato de fluxograma, para que seja possível encontrar os gargalos, os desperdícios e as falhas. Estes problemas deverão alimentar o quadro da consolidação das dores identificadas no capítulo anterior.

3.1.3. Análise de Causa Raiz

Para a conclusão do diagnóstico do problema do estudo de caso, as dores identificadas serão exploradas pela ACR (Análise de Causa Raiz), uma ferramenta para encontrar os principais problemas de um processo e selecionar as mais efetivas. ROONEY e HEUVEL (Rooney & Heuvel, 2004) descreve o funcionamento da ACR da seguinte maneira: através do diagrama de fatores causais será traçado o percurso do fator causal até suas causas raiz, percorrendo pela fonte primária do problema, para depois encontrar as categorias do problema, depois as categorias da causa raiz, e, então, encontrar as causas raiz. Ou seja, o diagrama de ACR irá traçar a trajetória dos problemas para encontrar as causas raiz, para que seja possível propor as soluções dos problemas que realmente estão impactando os processos.

As causas raiz são problemas bem definidos, que são facilmente identificáveis e que apresentam uma solução óbvia para ser identificada. Por exemplo, o problema “cultura da empresa não é organizada” não é uma causa raiz, já que não é facilmente identificável e nem facilmente solucionável; enquanto que o problema “os colaboradores da empresa descentralizam a informação em diversos documentos” é uma causa raiz, já que é facilmente identificável e traz uma solução óbvia: não descentralizar a informação.

Com a definição das causas raiz, será proposta uma solução para cada causa. As soluções deverão colaborar na melhoria do fator causal, que representa a maior dor identificada pela empresa, alcançando o objetivo da ferramenta.

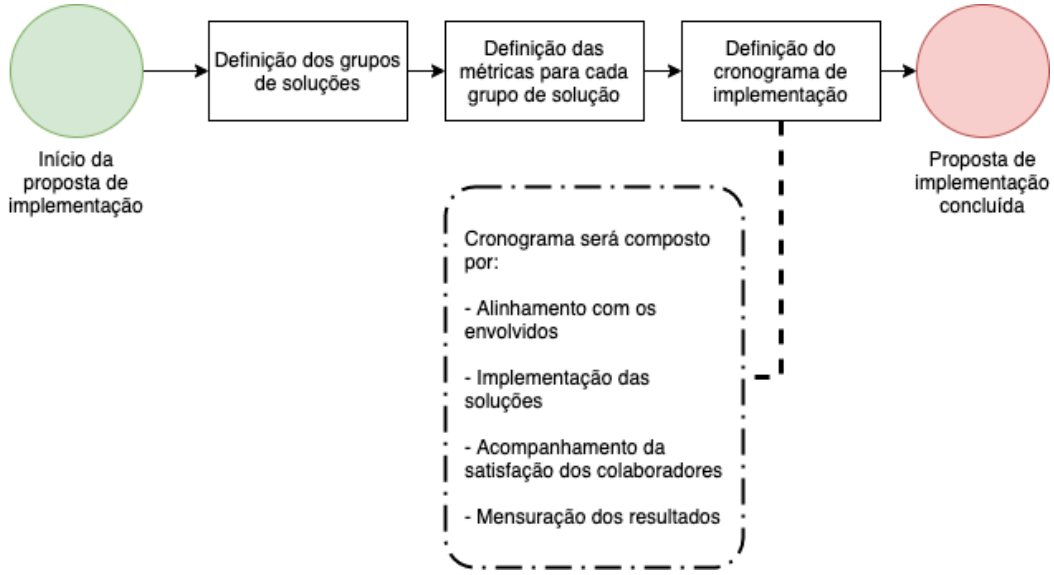
Em conclusão, a ferramenta de ACR é capaz de selecionar quais são os problemas a serem resolvidos, compreender melhor a relação de causa e efeito entre eles e, por fim, selecionar quais serão as soluções abordadas.

3.2. Solução proposta

Com a identificação dos problemas será possível construir as soluções para a melhoria, para que se possa definir o cronograma de implementação, com etapas e prazos a serem cumpridos.

Logo, a metodologia para a solução proposta e sua implementação será embasada em modelos HDM, com etapas e prazos pré-estabelecidos. De acordo com KHAN, QURASHI & KHAN (Khan, Qurashi, & Khan, 2011), as metodologias pesadas têm maior custo para desenvolver o planejamento do projeto ao longo de alguns meses; enquanto que as metodologias leves são interativas, com pequenas entregas de poucas semanas, para que possa ser observado o impacto das soluções e novas ideias sejam propostas. Levando em conta que a definição do escopo desse projeto já foi bem desenvolvida (através das entrevistas, desenho dos processos e a ACR) e que os requisitos do projeto não se enquadram com o manifesto ágil (Qureshi & Hussain, 2008), é mais razoável realizar a proposta de implementação com uma metodologia pesada. Segue o diagrama abaixo da proposta de implementação:

Figura 6 – Diagrama da proposta de implementação



Fonte: Elaborado pelo autor

Primeiramente as diversas soluções serão agrupadas, para que sejam elaborados planos de ação mais assertivos, por um objetivo em comum. Por exemplo, um grupo de solluções “Estruturação” conterá todas as soluções que têm foco semelhante. Em caso de uma solução não ter um grupo em comum, ela será separada e reavaliada se deverá ser implementada; desta forma, ficará mais claro para a empresa o que se deseja melhorar com o projeto.

Em cada grupo de solluções deverá ser especificado o que deverá ser realizado, tanto como quais as ferramentas que serão utilizadas. Essa etapa será fundamental para a definição de contorno para cada grupo de soluções, para que não surjam novas ideias sobrepondo as soluções definidas no diagnóstico do problema, da mesma forma que o CVDS é realizado nas metodologias pesadas.

Posteriormente, serão criadas métricas para cada grupo de solluções, para que o resultado das implementações seja mensurável. As métricas deverão corresponder aos problemas identificados pela ACR, a fim de compreender se as causas do fator causal foram solucionadas.

O método de mensuração das métricas será a percepção do entrevistador durante entrevistas semiestruturadas, compostas por perguntas amplas, com os colaboradores chave de cada área. Elas deverão ser aplicadas individualmente com cada colaborador, com um roteiro de perguntas específico para cada área (área de produto, de tecnologia e dos *stakeholders*), já que a especificidade da pergunta pode ser aplicada somente para um setor. Os roteiros das entrevistas começarão com perguntas mais genéricas sobre os resultados, enquanto que as últimas perguntas serão verificações se a solução foi implementada ou não, atendendo às recomendações do Guia de Avaliações Práticas (Newcomer, Hatry, & Wholey, 2004).

Com as soluções e suas métricas definidas, um cronograma de implementação será elaborado. As implementações mais fáceis e com maior retorno serão as primeiras a serem aplicadas, enquanto que as mais difíceis, com menor retorno, serão as últimas a serem aplicadas, seguindo o método de priorização definido por KHAN, SIDDIQUI, WAHEED, HASSAN E DUC na entrega de valor contínuo (Khan, Siddiqui, Waheed, Hassan, & Duc, 2020). O cronograma então será composto por uma etapa de alinhamento inicial com todos os envolvidos, depois de implementação das soluções, depois o acompanhamento da satisfação dos colaboradores, com posterior mensuração dos resultados.

3.3. Relato da implementação

O relato da implementação acompanhará o roteiro previsto anteriormente. Inicialmente será descrito o cenário atual da empresa, para demonstrar aos colaboradores da empresa o que era previamente à implementação das soluções. Nesse momento, também será necessário alinhar com os setores impactados todos

os pontos levantados e quais serão as soluções propostas, para que eles estejam cientes das transformações que irão por vir.

Com o decorrer da implementação, as propostas de solução deverão ser acompanhadas, registrando como foi o processo para colocar as soluções em prática, suas dificuldades e a reação das equipes. Além disso, é importante nessa etapa verificar com as áreas envolvidas o andamento das soluções, já que é possível ocorrer o descumprimento das soluções, devido à falta de costume ou desatenção pelos colaboradores.

Após a implementação das soluções, haverá um período de estabilização, para acompanhamento das soluções ou pequenos ajustes. Esse período será antes da mensuração dos resultados, para que sirva como consolidação da implementação das soluções.

Ao fim da implementação, serão realizadas as entrevistas semiestruturadas, para mensurar os resultados das implementações. Os colaboradores chave serão entrevistados individualmente, da mesma forma que será realizada no diagnóstico do problema. Com as entrevistas realizadas, as métricas para cada grupo de solluções serão avaliadas pelo entrevistador de forma subjetiva, verificando se o entrevistado identificou melhoria ou piora da métrica definida.

3.4. Análise dos Resultados

Por fim, serão discutidos os impactos das soluções adotadas na empresa desse estudo de caso, para que os resultados de cada grupo de solluções sejam observados. Não abstendo aos resultados medidos através das métricas, aqui servirá também de espaço para as sugestões ou às opiniões que foram obtidas nas entrevistas semiestruturadas durante a implementação do projeto.

Além disso, deverão ser descritos os impactos na cultura da empresa: os aprendizados desenvolvidos e as novas experiências durante esse período de mudança. Isso porque as propostas de melhoria para o processo de desenvolvimento não se aplicam somente ao aperfeiçoamento do processo, mas como também ao desenvolvimento de todos os colaboradores envolvidos.

Por fim, a metodologia para melhoria de processos de desenvolvimento de software apresentada nesse relatório deverá ser reavaliada, com propostas de melhoria para o próprio processo: desde o diagnóstico do problema, proposta de

implementação até a sua implementação, para aperfeiçoar este método a cada vez que aplicada.

4. Diagnóstico do Problema

As três ferramentas da metodologia desse trabalho para o diagnóstico serão aplicadas conforme o especificado: primeiramente serão conduzidas as entrevistas semiestruturadas; posteriormente o principal processo da área será documentado, a fim de identificar pontos de melhoraria; por fim, uma análise de causa raiz será construída para elencar os problemas e identificar como eles estão relacionados.

4.1. Entrevistas semiestruturadas

Os seguintes resultados foram obtidos através das entrevistas semiestruturadas, para uma imersão ao problema. Ao fim, as dores identificadas serão consolidadas em formato de quadro.

4.1.1. Roteiro da Entrevista

O roteiro abaixo foi desenvolvido seguindo as mesmas diretrizes estabelecidas na metodologia de projeto:

Tabela 1 – Roteiro das entrevistas para diagnóstico do problema

#	Pergunta
1	Objetivos da conversa: • Estamos querendo encontrar quais são os pontos forte das áreas de produto e de tecnologia da mesma forma que queremos explorar os pontos a serem melhorados; • Você (entrevistado) é uma pessoa essencial para esse estudo, pois é quem está convivendo com a área e pode trazer insumos para a discussão; • Os pontos fortes das áreas então devem continuar a serem desenvolvidos, da mesma forma que os pontos a melhorar podem trazer novas percepções e ideias;
2	O que você define como área de produto e de tecnologia?
3	Dessa definição, qual é a sua percepção da área?
4	Quais seriam os pontos fortes da área de produto?
5	O que você acredita que pode ser melhorado na área de produto?
6	Quais seriam os pontos fortes da área de tecnologia?
7	O que você acredita que pode ser melhorado na área de tecnologia?
8	Como você avalia a organização da área de produto? Por quê?
9	Como você avalia a organização da área de tecnologia? Por quê?
10	Como você avaliaria a qualidade da entrega de ambas as áreas? Por quê?
11	Como você avaliaria a eficiência da entrega de ambas as áreas? Por quê?
12	Qual a maior dor que você acredita existir em ambas as áreas? (essa pergunta deverá ser feita no final, mas caso não haja tempo durante a conversa, ela deverá ser priorizada).

Fonte: Elaborado pelo autor

4.1.2. Entrevistas com as áreas de *stakeholders*

Para representar as áreas de *stakeholders*, os diretores do comercial e do operacional se disponibilizaram para serem entrevistados, seguindo o roteiro do capítulo anterior.

Ambos os responsáveis pelas áreas iniciaram a conversa apresentando insatisfação com as entregas da área de tecnologia, pois havia diversos projetos que a empresa necessitava que não eram atendidos. Previamente, como solução inicial, foram contratados novos funcionários mais qualificados; todavia, embora aumentasse o número de funcionários, os resultados não correspondiam às suas expectativas.

Tanto a área de comercial como a área de operações reportavam dificuldades na utilização do ERP (sistema de gestão integrado) construído pelas equipes de produto e de tecnologia, como também erros que impediam o usuário final para realizar suas tarefas. Isso repetidamente era reportado num canal de comunicação de problemas e mesmo assim não era solucionado.

Sobretudo, quando era solicitado o andamento de uma funcionalidade nova, a área de produto não era capaz de responder, pois o processo era muito incerto e poderiam surgir imprevistos no caminho. No passar das semanas, algumas funcionalidades que eram mais essenciais não eram entregues, enquanto que outras menos relevantes surgiam.

Ao tardar, as funcionalidades que estavam entregues apresentavam erros, precisando voltar para os setores de produto e de tecnologia, pois não respeitavam os requisitos do projeto. Como a demanda demorava para ser atendida, as áreas de *stakeholder* receavam solicitar alterações, já que a funcionalidade poderia demorar para ser ajustada, então muitos dos colaboradores se adaptavam a ferramentas que não funcionavam conforme o esperado.

4.1.3. Entrevistas com a área de tecnologia

Em seguida, dois desenvolvedores da área de tecnologia foram entrevistados nos mesmos moldes.

O primeiro entrevistado apresentou desmotivação com a área de desenvolvimento, pois sentia que a área estava sendo detratada pela empresa por causa das entregas. Embora existisse esforço pelos desenvolvedores, as demandas eram mal estruturadas, havendo dificuldade na compreensão das tarefas e retrabalho após elas terem sido entregues.

Outro problema reportado foi a existência de muitos débitos técnicos nas plataformas. Estes são melhorias nos códigos que não foram entregues no momento de implementação para que fosse construído mais rapidamente; embora ser uma prática normal no mercado, os débitos técnicos usualmente são consertados num período logo após a entrega, enquanto que na empresa, eles permaneciam na plataforma. Por essa razão, atualmente é demorado construir novas funcionalidades, já que ocorrem muitos erros durante o desenvolvimento de novas funcionalidades, fazendo com que a manutenção da plataforma precisasse de constante atenção.

Em paralelo aos problemas internos da área de tecnologia, a área de produto os prejudicava: havia uma quantidade excessiva de reuniões para alinhamento com todos da área, havendo diversas interrupções que não agregavam valor.

Além das reuniões, os desenvolvedores sentiam que os testes de uma funcionalidade nova feitos pela área de produto não eram bem realizados e que, por isso, havia muitos defeitos poderiam ter sido prevenidos antes da entrega ao *stakeholder*.

Por fim, a especificação da funcionalidade para ser desenvolvida era má descrita, pois não existia padronização do que era solicitado, como também ocorria descentralização das anotações.

4.1.4. Entrevista com a área de produto

A última entrevista foi realizada com a área de produto, com o antigo diretor da área. Devido às reestruturações que a empresa passava, o antigo diretor da área de produto foi substituído, e estava agora atuando na área de investidores. Ele foi escolhido devido ao seu conhecimento da empresa, já que desempenhou o cargo desde a criação da startup.

A área de produto e de desenvolvimento convergiam na hipótese de que o crescimento rápido da empresa causou inúmeros débitos técnicos na plataforma, através de medidas paliativas que prejudicavam a eficiência de produzir uma nova funcionalidade. Essa maneira em que a plataforma foi desenvolvida não é escalável, pois está caminhando para um ambiente insustentável de programação.

Por causa desse crescimento acelerado, a demanda de funcionalidades novas cresceu abruptamente, com prazos que eram inviáveis de entrega. Esses prazos mais curtos acabavam por incentivar entregas paliativas, já que eram mais rápidas, gerando mais débitos técnicos. Continuamente esse fluxo se repetia, e as áreas de produto e

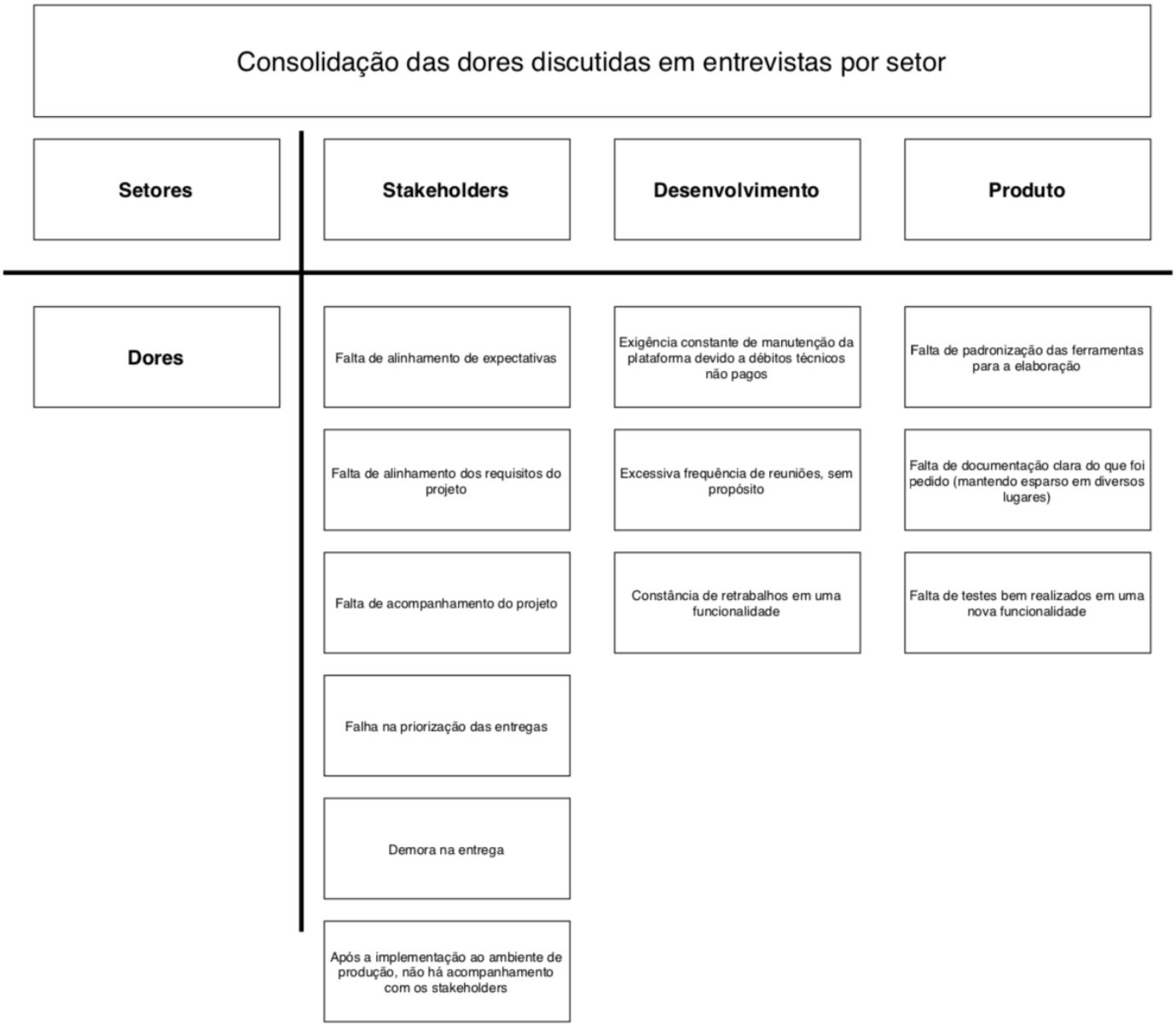
de tecnologia ficavam reféns desse ciclo; em paralelo que o processo de construção de software é muito complexo e as funcionalidades deveriam ser construídas mais estrategicamente.

Contudo, o antigo diretor da área de produto confessou que a área precisava melhorar em diversos pontos, como alinhamento de expectativa e acompanhamento das entregas; embora já tivesse melhorado bastante.

4.1.5. Consolidação das dores

Assim, com as entrevistas realizadas com os representantes das áreas de *stakeholders*, de tecnologia e de produto, as dores foram consolidadas e dispostas no quadro abaixo, para melhor visualização dos problemas relatados pelos entrevistados.

Figura 7 – Consolidação das dores discutidas em entrevistas, por setor



Fonte: Elaborado pelo autor

Para os *stakeholders*, a insatisfação girava em torno da falta de alinhamento: não se sabia em que estágio estava a nova funcionalidade, como também existia uma divergência do que era idealizado no planejamento e do que era entregue.

Em contrapartida, a área de desenvolvimento sentia-se ineficiente, tanto pelo excesso de reuniões, como pela dificuldade de desenvolver uma nova funcionalidade, devido aos débitos técnicos.

Por fim, a área de produto apresentava dificuldade na comunicação com a área de desenvolvimento, pela falta de ferramentas padronizadas ou uma especificação ruim do que era demandado pelos *stakeholders*.

4.2. Processo de desenvolvimento de software da empresa

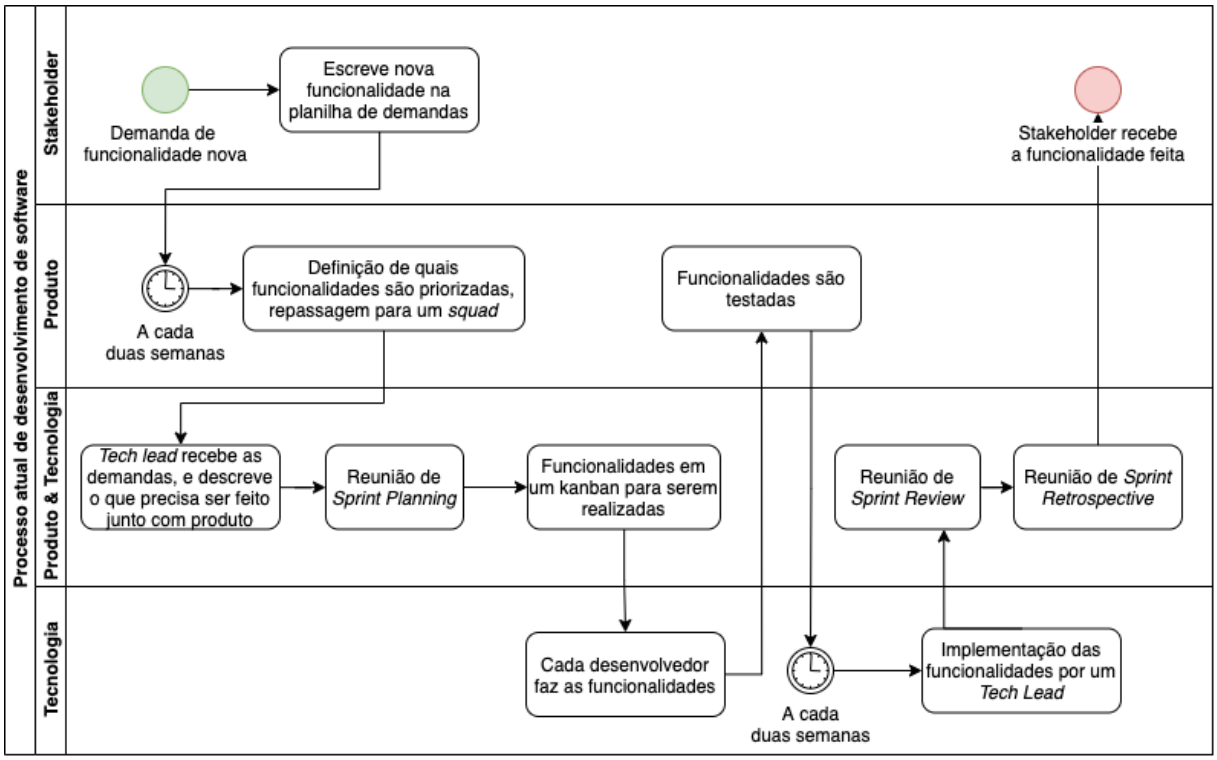
Após a realização das entrevistas como primeira imersão à área, os colaboradores da empresa foram consultados para descrever melhor processo das áreas de produto e de tecnologia.

4.2.1. Mapeamento do processo

Uma das dificuldades encontradas para mapear o processo de desenvolvimento de software atual é que nem sempre esse processo era respeitado. Isso ocorria por causa de imprevistos durante o processo, que gerava muitos retrabalhos e voltas nas etapas de desenvolvimento.

Para contornar esse empecilho, foi coletado o “processo comum” que uma funcionalidade passa, desde a solicitação por um *stakeholder* até a sua saída, com a entrega, conforme figura abaixo.

Figura 8 - Processo de desenvolvimento de software



Fonte: Elaborado pelo autor

O fluxo de desenvolvimento de software se inicia pela demanda de funcionalidades novas por *stakeholders*, como por exemplo: um colaborador da equipe de crédito necessita de uma automatização na ferramenta do seu setor, enquanto que outro colaborador da equipe de comercial necessita de uma funcionalidade nova, que atualmente está impedindo o trabalho da equipe. Ambos descrevem suas necessidades na “planilha de demandas”, onde se concentra todas as novas demandas por funcionalidades da empresa, explicando o porquê e o impacto que a funcionalidade iria trazer.

Essas funcionalidades são priorizadas pela equipe de produto uma vez a cada duas semanas, repassando-as para o *squad* responsável (equipe formada por desenvolvedores e produto). Se uma funcionalidade não foi priorizada, ela deverá ser solicitada novamente pela “planilha de demandas” no próximo ciclo de *Sprint*, em duas semanas.

Então todas as funcionalidades priorizadas irão passar por um líder técnico (*Tech Lead*) que irá analisá-las junto com a área de produto para desenhar seu escopo. Assim, com a equipe inteira, é realizada a reunião de *Sprint Planning*, para

alinhamento do que será entregue para as próximas duas semanas. Todas as funcionalidades novas são repassadas para um quadro *Kanban*, disponível para os desenvolvedores.

Durante esse período de duas semanas, cada desenvolvedor aloca algumas tarefas para si mesmo, para início de seu desenvolvimento. No final, todas as funcionalidades são testadas pela área de produto, para garantir o cumprimento dos requisitos. Após os testes e eventuais correções, um *Tech Lead* é responsável por implementar as novas funcionalidades no ambiente de produção (ambiente em que está disponível para usuários, tanto internos quanto externos) no fim da *Sprint*.

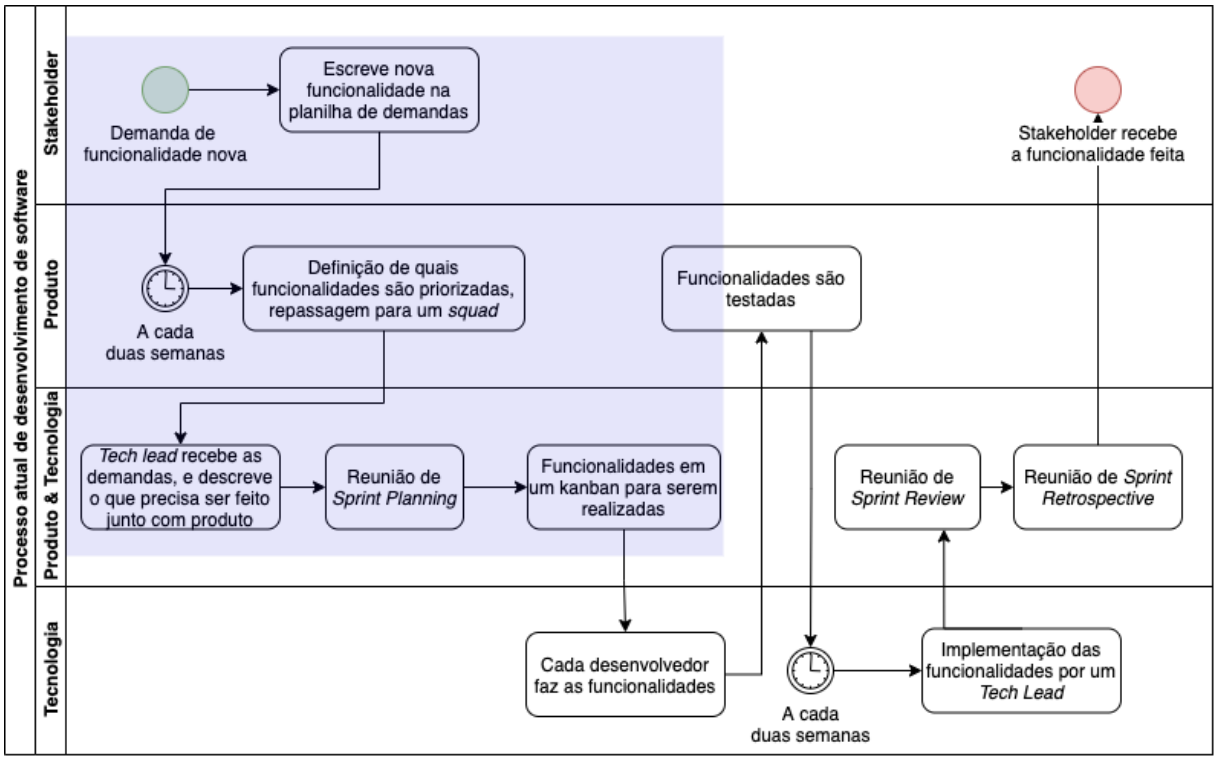
Por fim, para a conclusão desse ciclo são realizados dois rituais: a *Sprint Review* e a *Sprint Retrospective*. Dessa forma, a funcionalidade é entregue ao *stakeholder*, e o ciclo de desenvolvimento é concluído.

Para o detalhamento do processo, ele foi dividido em três etapas, que serão descritas abaixo: planejamento das entregas, desenvolvimento e testes das funcionalidades e conclusão das entregas. Posteriormente, neste capítulo, serão consolidadas as dores observadas nesse fluxo.

4.2.2. Planejamento das entregas

Essa etapa inicia pela demanda de uma nova funcionalidade pelos *stakeholders* até a entrega de todas as funcionalidades no quadro *Kanban*, conforme figura abaixo.

Figura 9 – Planejamento das entregas



Fonte: Elaborado pelo autor

A rotina da área de produto é de ler e compreender as demandas dos *stakeholders*, pela “planilha de demandas”. Caso uma demanda fosse fácil de ser desenvolvida e agregasse bastante valor, algum colaborador da área de produto abordaria o *stakeholder* interessado para melhor compreendê-la. Contudo, um dos problemas reportados foi a quantidade excessiva por demandas de funcionalidades novas que não agregariam valor na planilha, pois esta é aberta para todos da empresa, independentemente da hierarquia.

Seguindo a rotina, a cada duas semanas todas as funcionalidades que foram priorizadas iniciam o detalhamento de seu escopo, junto a um *Tech Lead* do setor de desenvolvimento. Outro problema aqui reportado é a falta de alinhamento com produto, pois usualmente o *Tech Lead* construía o escopo inteiro por causa da indisponibilidade de produto.

Com a definição do escopo da funcionalidade, tanto pelo lado dos *stakeholders* (detalhado por produto) tanto pelo lado técnico (detalhado pelo *Tech Lead*), a reunião

de *Sprint Planning* é realizada, para que todos da equipe estejam cientes do que está sendo demandado. O ritual de *Sprint* Planning da metodologia *SCRUM* traz alinhamento para a equipe compreender o que precisará ser realizado, tanto quanto compreender a dificuldade ou planejamento de cada tarefa.

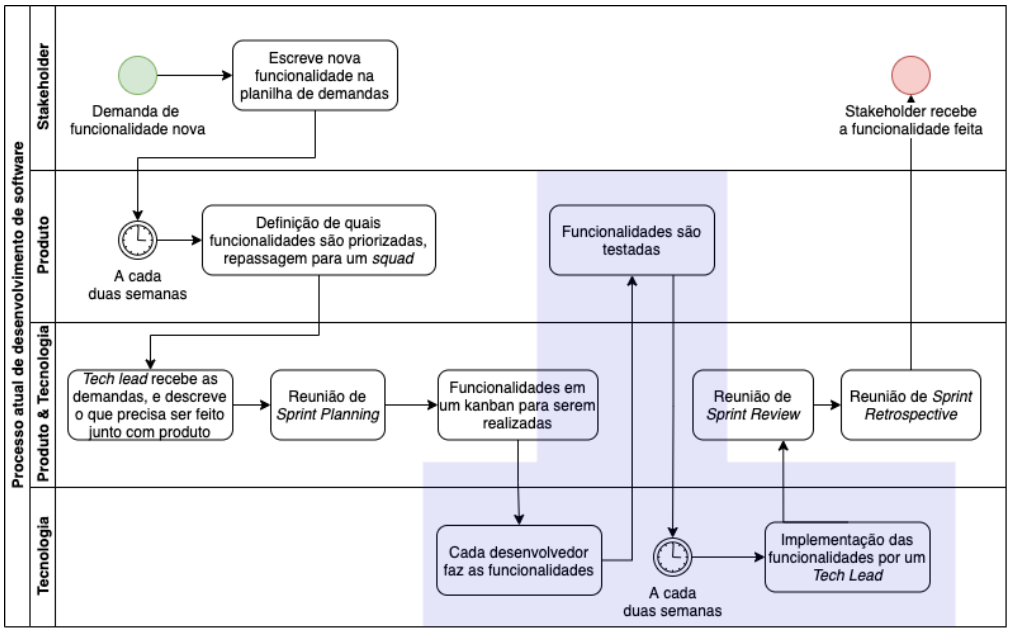
Com a *Sprint Planning* realizada, as tarefas são movidas para o quadro *Kanban*, para que cada desenvolvedor tenha a flexibilidade de escolher quais funcionalidades deseja trabalhar.

4.2.3. Desenvolvimento e teste das funcionalidades

Essa etapa do processo é mais técnica, em que os desenvolvedores utilizam editores de texto específicos para codificação (VS Code), plataformas de versionamento (Github), entre outros aspectos. Seguindo o escopo deste trabalho, essa etapa será descrita de maneira ampla, sem o objetivo de aprofundar no desenvolvimento do código, mas sim em como é realizada a gestão.

Neste momento do processo, os desenvolvedores têm as funcionalidades para desenvolver no quadro *Kanban*, junto às suas especificações. Na conclusão, todas as funcionalidades desenvolvidas são implementadas no ambiente de produção, conforme figura abaixo.

Figura 10 – Desenvolvimento e teste das funcionalidades



Fonte: Elaborado pelo autor

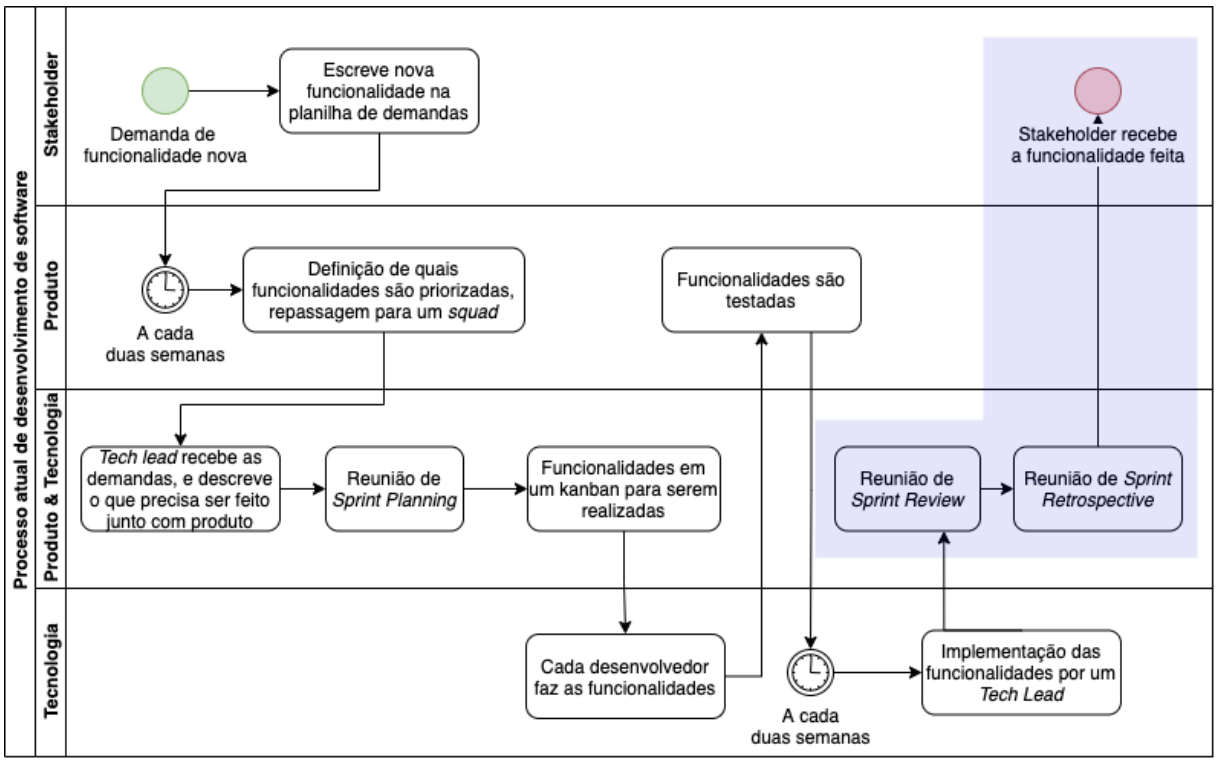
Após o desenvolvimento das funcionalidades pela área de tecnologia, a área de produto tem a responsabilidade de testá-las em um ambiente de testes, analisando se estas atendem às necessidades dos *stakeholders*. Um dos problemas reportados foi a falta de qualidade destes testes, resultando em muitas funcionalidades que não atendiam às necessidades dos *stakeholders* e que não eram percebidas por produto.

Com as funcionalidades analisadas e aprovadas pela área de produto, fica a cargo de um *Tech Lead* implementá-las em ambiente de produção, para que elas possam ser utilizadas. Todas as funcionalidades são implementadas de uma vez, na conclusão da *Sprint* a cada duas semanas.

4.2.4. Conclusão das entregas

Assim, o ciclo da *Sprint* é concluído com duas reuniões: *Sprint Review* e *Sprint Retrospective*, conforme figura abaixo.

Figura 11 – Conclusão das entregas



Fonte: Elaborado pelo autor

Na reunião de *Sprint Review*, tanto a área de produto quanto a área de tecnologia discutem sobre o que foi entregue e o que não foi entregue. Frequentemente a equipe de desenvolvimento não conseguia entregar tudo o que era solicitado, pois havia diversas interrupções que atrapalhava o ciclo de desenvolvimento, por exemplo algum erro na plataforma.

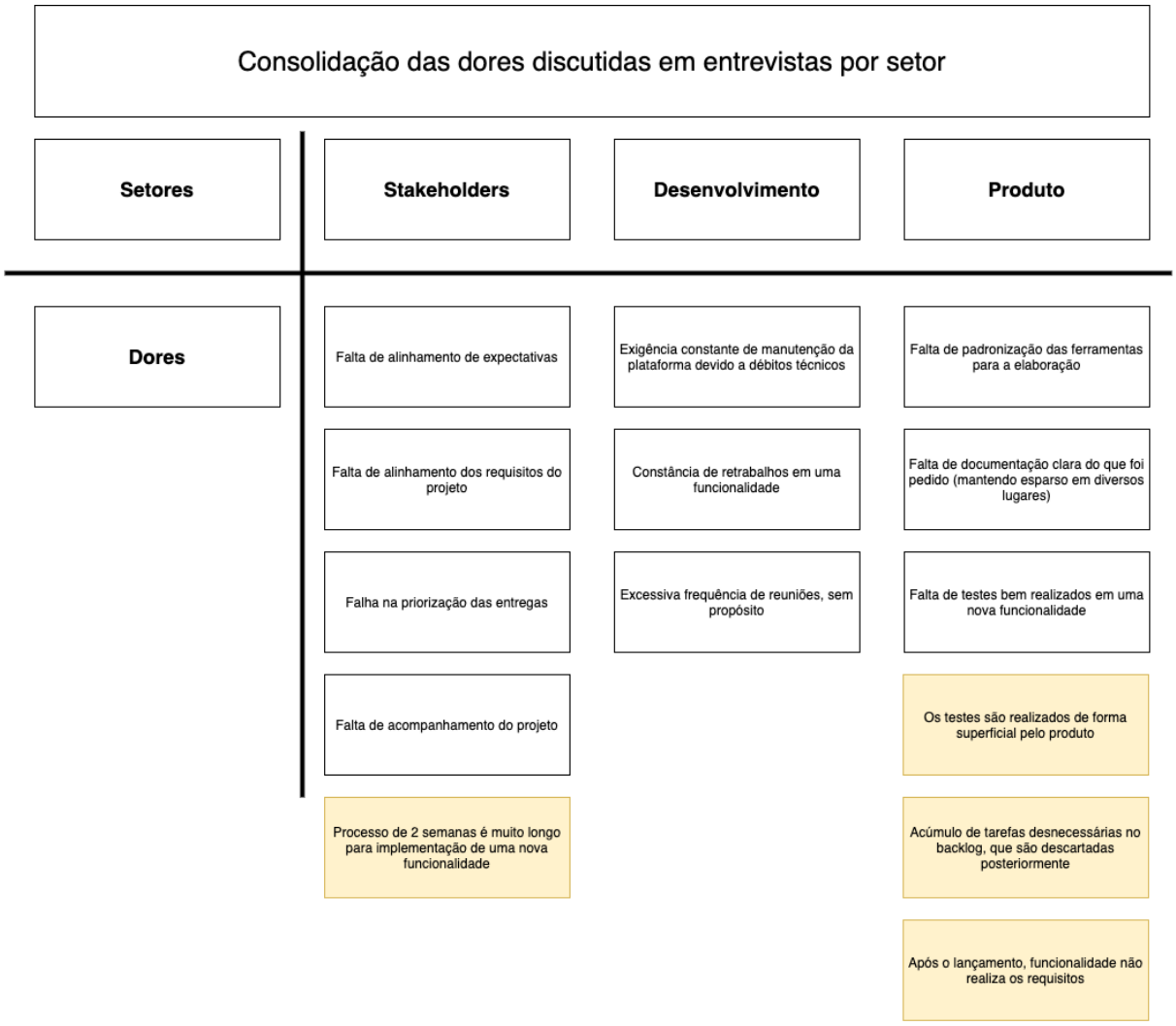
Posteriormente era realizada a *Sprint Retrospective*, para discutir o que foi feito bem, para estimular esses comportamentos (como ajudar outros desenvolvedores, ou aplicação de um conhecimento novo); e discutir o que foi feito mal, para melhorar em próximas entregas (como pressão de *stakeholders,* ou mudança de escopo da funcionalidade durante o período de desenvolvimento).

Assim, a tarefa era entregue para os *stakeholders*, que iniciavam seu uso.

4.2.5. Consolidação das dores

Com o mapeamento do processo de desenvolvimento de software da empresa, o diagrama do capítulo 4.1.5. foi incrementado com as informações obtidas (em amarelo, na figura abaixo), com as dores mais relevantes.

Figura 12 - Consolidação das dores descobertas pelo mapeamento do processo



Fonte: Elaborado pelo autor

Uma das novas dores identificadas foi o período longo de duas semanas para cada entrega. Assim, caso uma demanda fosse solicitada no meio de um ciclo, seria necessário o fechamento deste, para que a entrega pudesse ser realizada em seguida. Contudo, caso a funcionalidade não atendesse aos requisitos definidos pelos *stakeholders*, ela teria que passar por dois ciclos (4 semanas), sendo inviável para o crescimento atual da empresa. Muitas das funcionalidades não atendiam aos requisitos, pela falta de testes por produto, já que eles eram feitos de maneira rasa. Contudo, testes mais profundos custariam muito tempo, algo que a área de produto carecia, pois os colaboradores estavam sobrecarregados pelo acúmulo de tarefas desnecessárias na “planilha de demandas”.

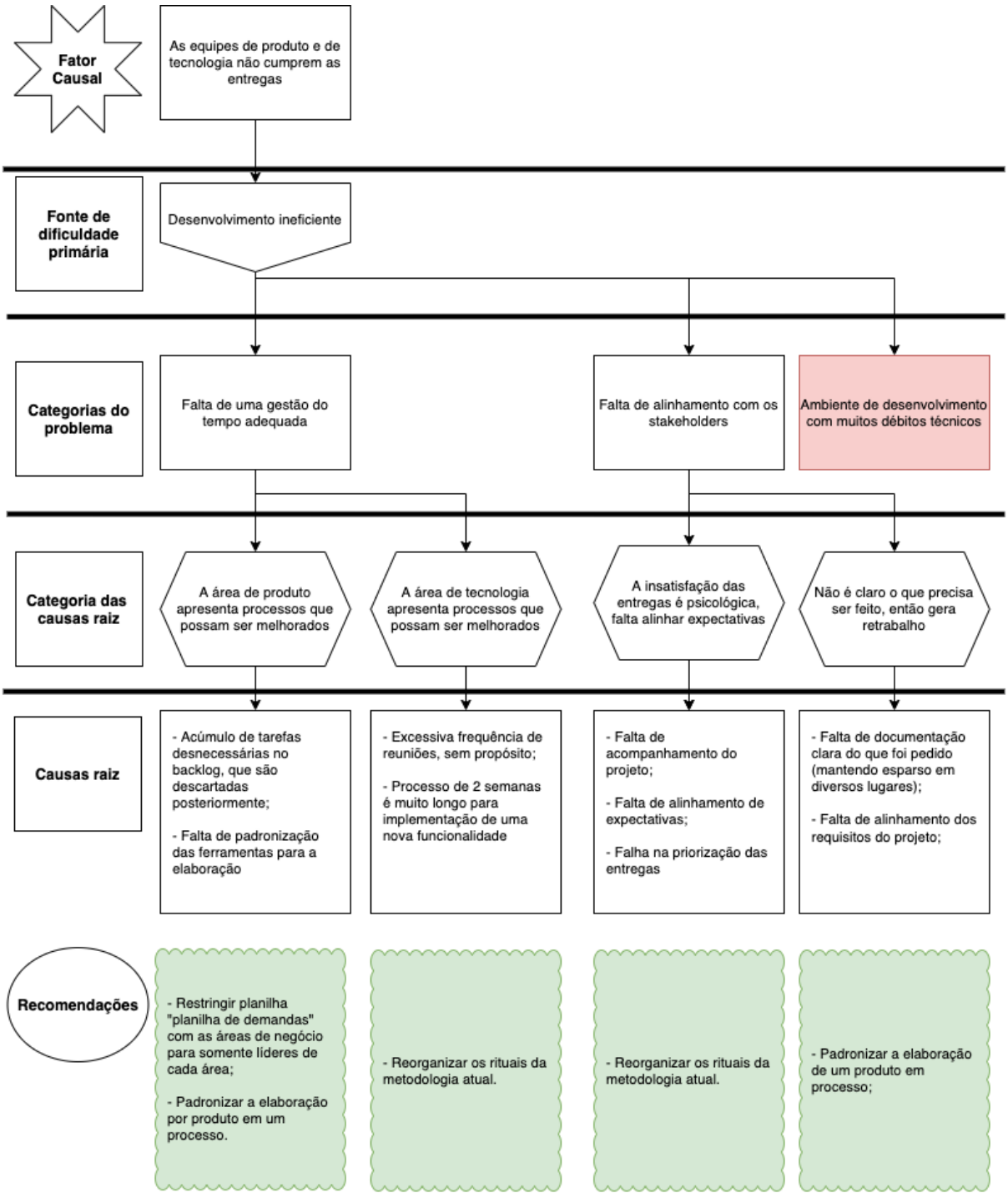
4.3. Análise de causa raiz (ACR)

Para a conclusão do diagnóstico do problema, os resultados obtidos pela ACR são demonstrados a seguir, seguindo as diretrizes da metodologia de melhoria de processos de desenvolvimento elaboradas nesse trabalho.

4.3.1. Apresentação da ACR

A ACR desenvolvida neste estudo de caso está representada na figura abaixo. O fator causal identificado foi “as equipes de produto e de tecnologia não cumprem as entregas”, uma dor identificada pelos próprios *stakeholders* e classificada como a necessidade mais impactada pela empresa; ou seja, o objetivo final do estudo de caso seria o tratamento deste fator causal, que é a consequência de outras problematizações. Observa-se na figura abaixo essa relação de causa-efeito das dores observadas no capítulo 4.2.5.

Figura 13 – Árvore de causa raiz completa



Fonte: Elaborado pelo autor

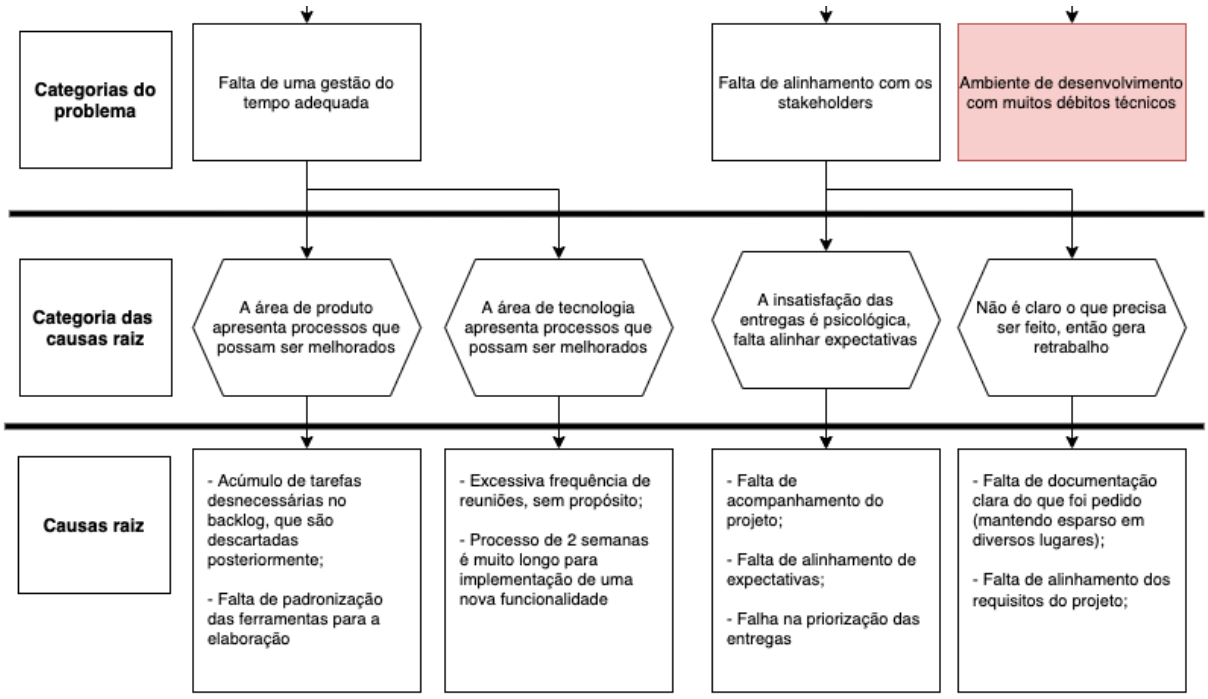
O não cumprimento das entregas é uma das consequências da fonte de dificuldade primária “desenvolvimento ineficiente”, já que atualmente, o desenvolvimento de funcionalidades novas é dificultado, sendo esta uma das principais dores dos desenvolvedores.

Assim, o “desenvolvimento ineficiente” é causado por três principais razões: existe falta de gestão do tempo adequada; existe falta de alinhamento com os *stakeholders*; e existe um ambiente de desenvolvimento com muitos débitos técnicos, o que atrasa a produção de funcionalidades novas.

Esta terceira categoria do problema não será abordada neste estudo de caso, pois foge do escopo. Por exemplo, uma das abordagens propostas pelo líder técnico da empresa foi de mudar a estrutura das plataformas para “micro-serviços”, uma maneira de granularizar os sistemas, tornando-se mais fácil para realizar as demandas; contudo, como o intuito deste trabalho é focado em metodologias de projeto, essa categoria do problema não será desenvolvida aqui.

Assim, as duas categorias do problema serão melhor descritas nos capítulos abaixo. Então para cada categoria da causa raiz serão discutidas suas causas raiz e suas soluções.

Figura 14 – Árvore de causa raiz – categorias do problema



Fonte: Elaborado pelo autor

4.3.2. Categoria do problema: gestão de tempo; Categoria da causa raiz: processos em produto

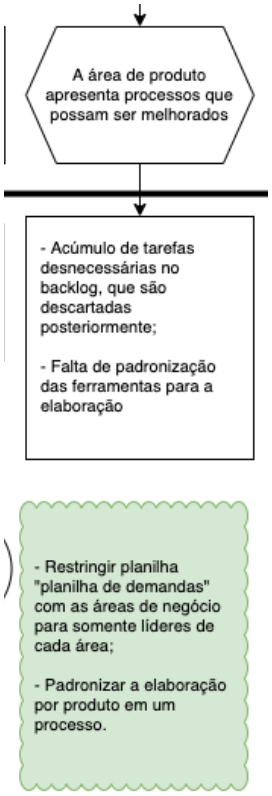
Uma das categorias da causa raiz identificadas para a falha na gestão do tempo são os processos falhos na área de produto. Atualmente cada colaborador desta área realiza a mesma tarefa de maneiras diferentes, o que pode acarretar em problemas.

Isso foi identificado por duas causas raiz, que seriam o “acúmulo de tarefas desnecessárias no backlog, que são descartadas posteriormente” e “Falta de padronização das ferramentas para a elaboração”.

Para solucionar a primeira causa raiz, a "planilha de demandas" será restrita para somente líderes das áreas dos *stakeholders*, precisando haver um alinhamento prévio com os colaboradores da empresa antes dessa tomada de decisão.

Já para solucionar a segunda causa raiz, será necessário padronizar a elaboração por produto em um processo. Ou seja, criar um processo em comum para que todos da área sigam para a elaboração e planejamento das demandas dos *stakeholders*.

Figura 15 – Árvore de causa raiz - gestão de tempo, processos em produto



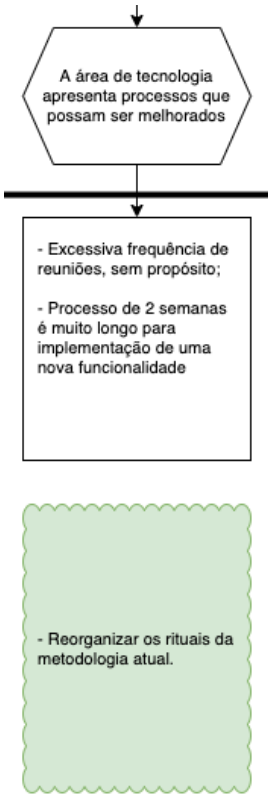
Fonte: Elaborado pelo autor

4.3.3. Categoria do problema: gestão de tempo; Categoria da causa raiz: processos em tecnologia

Além dos processos falhos na área de produto, existem processos que podem ser melhorados na área tecnologia. As duas causas raiz identificadas foram: “excessiva frequência de reuniões, sem propósito” e que o “processo de 2 semanas é muito longo para implementação de uma nova funcionalidade”.

Para ambas as causas raiz a solução a ser adotada é reorganizar os rituais do processo de desenvolvimento de software atual, pois enquanto que alguns rituais são desnecessários, outros podem ser aumentados a frequência com o intuito de realizar entregas semanalmente ao invés de quinzealmente.

Figura 16 – Árvore de causa raiz - gestão de tempo, processos em tecnologia



Fonte: Elaborado pelo autor

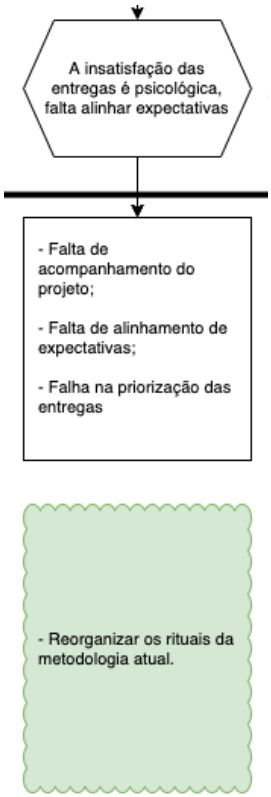
4.3.4. Categoria do problema: falta de alinhamento; Categoria da causa raiz: insatisfação psicológica

Para a categoria do problema de falta de alinhamento, uma das categorias de causa raiz encontrada foi que a insatisfação das entregas é psicológica, que parte do não cumprimento das entregas por produto e tecnologia é causada pelo lado psicológico dos *stakeholders*. A teoria por trás é que, como a expectativa da entrega da demanda é maior do que o cenário real acarreta numa percepção pior. Por exemplo, atualmente não existe um alinhamento bom entre produto e *stakeholder* e, por isso, *stakeholders* assumem que o resultado será 8 de 10 gerando insatisfação quando a entrega é 6 de 10; contudo, caso o resultado alinhado entre produto e *stakeholders* fosse 5 de 10, a entrega de 6 de 10 é superior ao esperado.

As causas raiz dessa categoria são todas relacionadas a comunicação: “falta de acompanhamento das demandas do projeto”, “falta de alinhamento das expectativas” e “falha na priorização das entregas”.

Portanto, a solução dessas causas raiz torna-se a mesma do capítulo 4.3.3., que é a de “reorganizar os rituais da metodologia atual”; isto é, utilizar os rituais do processo de desenvolvimento de software para melhorar a comunicação com os *stakeholders*.

Figura 17 – Árvore de causa raiz - falta de alinhamento, expectativas



Fonte: Elaborado pelo autor

4.3.5. Categoria do problema: falta de alinhamento; Categoria da causa raiz: clareza nas entregas

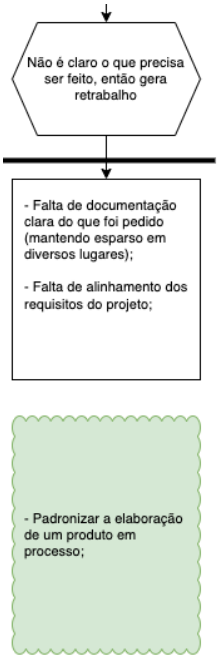
Por fim, a segunda categoria da causa raiz para a falta de alinhamento é que “não é claro o que precisa ser feito, então gera retrabalho”, cujo maior foco é na definição do escopo (elaboração e planejamento) de cada demanda.

Esse escopo não é bem definido atualmente por duas causas raiz: “falta de documentação clara do que foi pedido (mantendo esparso em diversos lugares)” e “falta de alinhamento dos requisitos do projeto”.

A primeira causa raiz é resultado da desorganização de produto, que mantém a documentação descentralizada em vários lugares. Ao descentralizar a documentação da demanda, uma parte da funcionalidade que fosse elaborada em uma reunião poderia se perder, gerando insatisfação do *stakeholder.* Em paralelo que, a outra causa raiz desta categoria é falta de alinhamento dos requisitos da demanda; ou seja, não é definido o que precisa ser realizado para que a entrega esteja completa.

Assim, a solução é a mesma do capítulo 4.3.2., que é a de “padronizar a elaboração de um produto em processo”, de tal forma que melhores métodos e ferramentas para o alinhamento entre o *stakeholder* e o produto seja definido dentro do processo.

Figura 18 – Árvore de causa raiz - falta de alinhamento, clareza nas entregas



Fonte: Elaborado pelo autor

5. Proposta de implementação

A partir do diagnóstico do problema foram identificados dois grupos de soluções que foram implementados neste estudo de caso e que têm o maior potencial de melhoria para da empresa: soluções que realizam melhora na gestão do tempo dos colaboradores das áreas de produto e de tecnologia; e soluções que propõem a padronização do processo para elaboração do produto.

5.1. Grupo de soluções: Gestão do tempo

O grupo de soluções de gestão do tempo visa a reduzir o desperdício de tempo dos colaboradores das áreas de tecnologia e de produto através da melhoria de processos ineficientes ou desnecessários. A primeira solução foi a da restrição da "planilha de demandas", documento utilizado para comunicação entre a área de produto e de *stakeholders*; enquanto que a segunda foi a reorganização dos rituais do processo de desenvolvimento de software atual.

5.1.1. Restrição da "planilha de demandas"

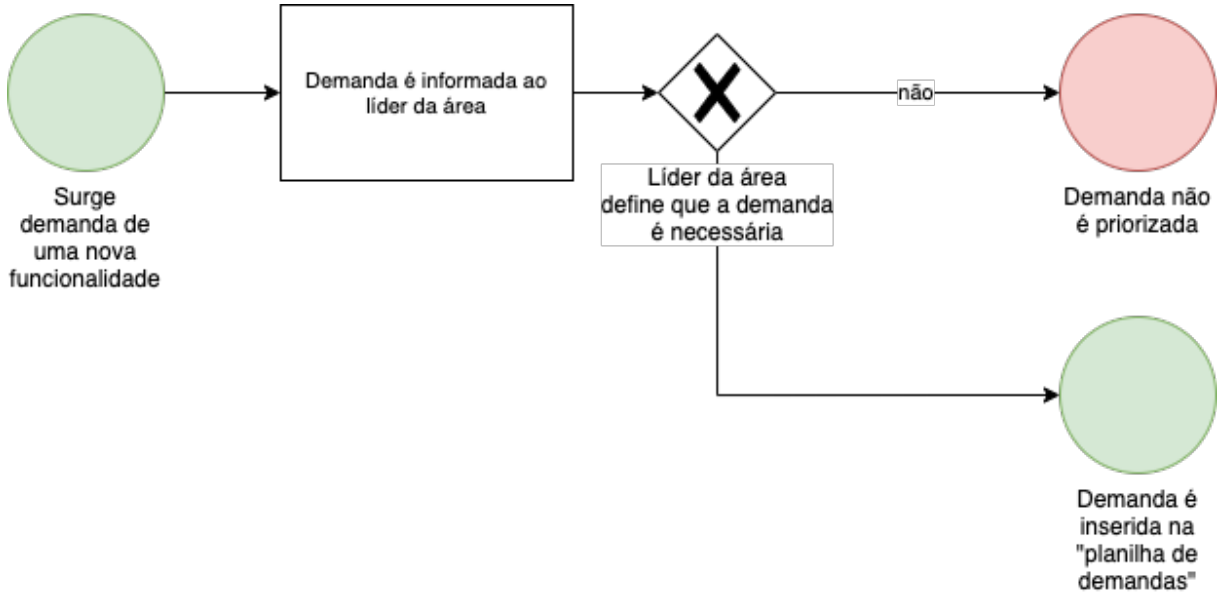
A "planilha de demandas" é o documento utilizado para registrar novas demandas de funcionalidades pelas equipes de negócio, como um botão novo, ou automatização de uma tarefa nova.

Desse modo, a solução proposta foi de restringir esse documento, para que seja acessível somente por líderes de cada área. De acordo com a revisão sistemática de literatura de AHMAD, MARKKULA E OVIO (Ahmad, Markkula, & Ovio, 2013) a comunicação tende a ter um alinhamento mais forte com a alta gestão, e não com todos os colaboradores da empresa, trazendo maior eficiência no processo de elaboração.

Portanto, pretendeu-se resolver a causa raiz: "acúmulo de tarefas desnecessárias no *backlog*, que são descartadas posteriormente", visto que a "planilha de demandas" está disponibilizada para todos da empresa. Parte significativa do trabalho dos analistas da área de produto é ocupado com a leitura de demandas não essenciais do documento, o que leva a desgaste com os colaboradores quando é necessário priorizar o *backlog*.

Conforme a figura abaixo, a solução propôs que a nova funcionalidade seja analisada previamente pelo líder de sua área antes de ser encaminhada para produto. Assim, caso o líder aprove a ideia, a demanda é inserida na "planilha de demandas".

Figura 19 – Proposta de solução: restrição da “planilha de demandas”



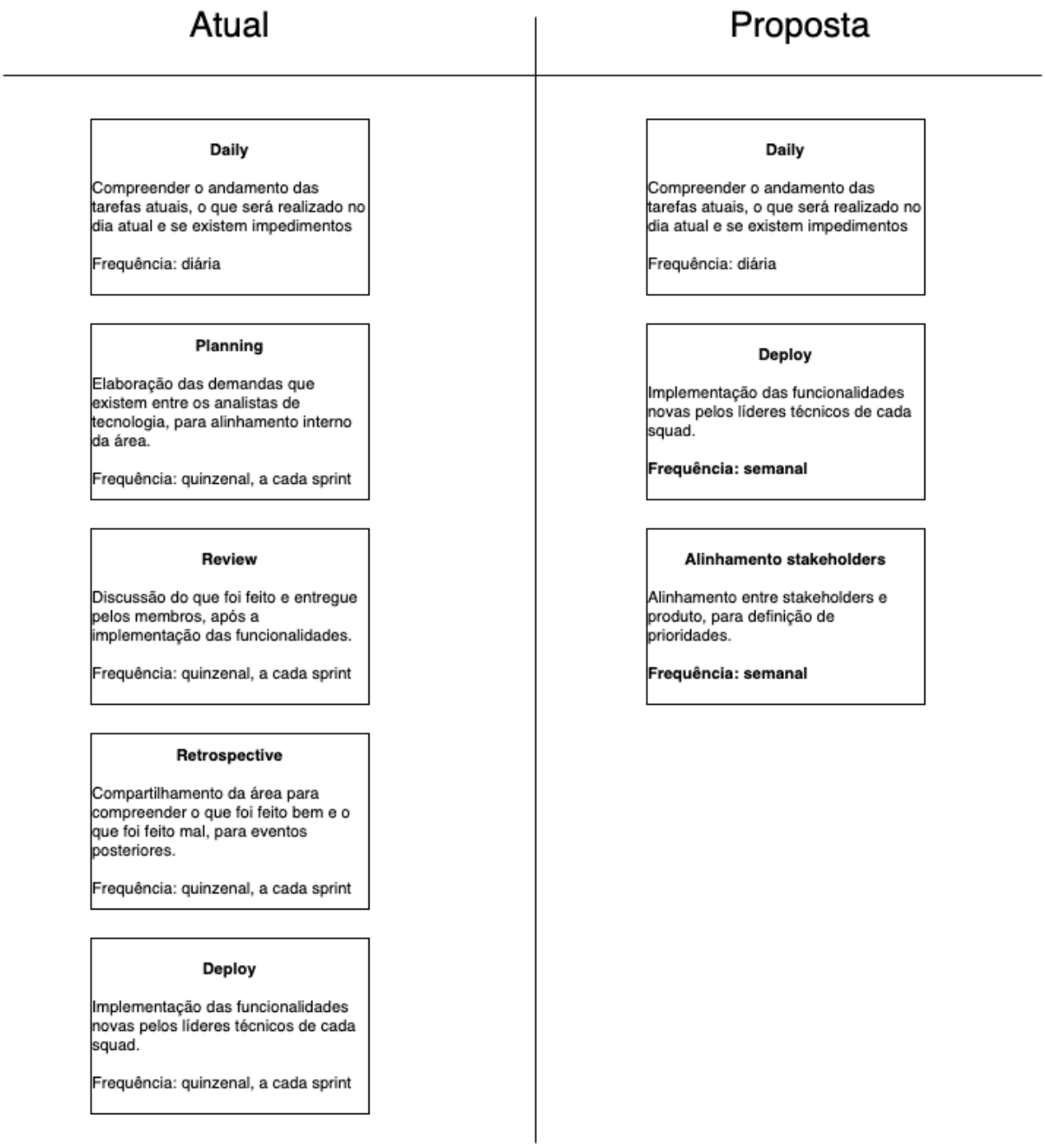
Fonte: Elaborado pelo autor

Em conclusão, a solução propôs a descentralização da tarefa de priorização de *backlog*, uma tarefa recorrente e cansativa da área de produto, para os gestores de cada área. Dessa forma, o processo de desenvolvimento de software se torna mais eficiente, da mesma maneira que traz um melhor alinhamento entre os líderes de cada área e sua equipe sobre as demandas dos sistemas.

5.1.2. Reorganização dos rituais do processo de desenvolvimento

Para melhorar a gestão do tempo no processo de desenvolvimento de software, outra proposta de solução foi a da reorganização dos rituais que atualmente existem: *daily, planning, review, retrospective e deploy*, conforme a figura abaixo:

Figura 20 – Proposta de solução: rituais no processo de desenvolvimento



Fonte: Elaborado pelo autor

Com essa alteração, pretendeu-se resolver a causa raiz de “excessiva frequência de reuniões, sem propósito”, com a remoção dos rituais de *planning*, *review* e *retrospective.* De acordo com SRIVASTAVA, BHARDWAJ, & SARASWAT (Srivastava, Bhardwaj, & Saraswat, 2017), a metodologia *SCRUM* faz uso tradicionalmente dessas reuniões para alinhamento entre os colaboradores; contudo, por conta das reclamações pelos próprios colaboradores, essas reuniões poderiam ser englobadas pelas reuniões de *daily*. KHAN, SIDDIQUI, WAHEED, HASSAN & DUYC (Khan, Siddiqui, Waheed, Hassan, & Duc, 2020) define que os rituais da metodologia ágil têm a função de informar, planejar, refinar, refletir e investigar os problemas; todavia, cabe à empresa moldar os rituais ao seu cenário.

Portanto, a *planning*, poderia ser realizada somente envolvendo um analista de produto e um analista de tecnologia, para não ocupar todos da empresa; enquanto que a *review* e a *retrospective* poderiam ser realizadas durante as dailies, para comentar o que foi realizado. A remoção desses três rituais se distancia da metodologia *SCRUM* para desenvolvimento de software, e se aproxima da metodologia *Kanban*.

De acordo com Patil e Neve (Patil & Neve, 2018), a metodologia *Kanban* quebra o paradigma de *Sprint* da metodologia *SCRUM*, para que o processo de desenvolvimento seja contínuo: o planejamento não é realizado periodicamente, mas constantemente para alimentar o quadro *Kanban*. Assim, as tarefas ficam à disposição dos desenvolvedores e fica a responsabilidade da área de produto alimentar o quadro *Kanban* com novas tarefas constantemente.

A próxima causa raiz que se pretendeu resolver é “processo de 2 semanas é muito longo para implementação de uma nova funcionalidade”, através do aumento da frequência da reunião de *deploy*. Este ritual é realizado quinzenalmente, quando um analista de produto e um *tech lead* se reúnem para implementar as funcionalidades desenvolvidas por toda a equipe. Por isso, uma funcionalidade nova poderia demorar até quatro semanas para a sua entrega: primeiramente é necessário aguardar a conclusão da *Sprint* atual e, em seguida, duas semanas para o próximo *deploy*.

Com o aumento da frequência de *deploy* foi esperado que as funcionalidades sejam entregues semanalmente; com isso os *stakeholders* poderiam testá-las com mais antecedência, tanto como priorizar funcionalidades mais rapidamente entre uma *Sprint* e outra.

Por fim, as últimas causas raiz que essa solução pretendei resolver são: “falta de acompanhamento do projeto”, “falta de alinhamento de expectativas” e “falha na priorização das entregas”, com a criação de um novo ritual: a reunião de alinhamento com *stakeholders*. Não existia um processo formal para acompanhar o projeto: a demanda começava a ser desenvolvida pelas equipes de produto e de tecnologia, mas não existia visibilidade em qual estágio a tarefa está. Isso causava desgaste na relação entre a área de produto e as áreas de *stakeholders*, pois ocorria uma cobrança excessiva sobre o que deveria ser entregue, demandando um acompanhamento com cada área sobre o que poderia ser entregue e o que deveria ser priorizado.

Assim, esse novo ritual pretendeu reunir todos os líderes interessados das áreas de *stakeholders* junto à área de produto para trazer visibilidade do acompanhamento de todas as funcionalidades em processo de desenvolvimento, possibilitando a discussão da priorização das tarefas.

5.1.3. Métricas

Com o grupo de soluções definido acima, as seguintes hipóteses de melhoria foram analisadas após a implementação:

- O tempo das equipes de produto e de tecnologia será utilizado de maneira mais eficiente;
- O alinhamento entre a área de produto e de *stakeholders* melhorará;
- O tempo gasto médio em reuniões reduzirá em 60% na área de tecnologia.

Com essas hipóteses em mente, foram realizadas entrevistas semiestruturadas com as seguintes perguntas, podendo variar para cada área.

Tabela 2 – Roteiro para entrevistas para gestão do tempo

#	Pergunta	Para produto	Para tecnologia	Para stakeholders
1	Você sente que o tempo da sua área está sendo melhor utilizado?	X	X	
2	As expectativas de prazos e priorização, tanto quanto acompanhamento dos projetos sendo cumpridos?	X		X
3	A restrição da "planilha de demandas" otimizou o tempo da sua área? Como isso impactou a qualidade das demandas?	X		X
4	A nova organização dos rituais otimizou o tempo da sua área? Como isso impactou a comunicação?	X	X	
5	A frequência de entregas aumentou. Como isso impactou a eficiência? Como isso impactou na qualidade?	X	X	X
6	Você sente que o tempo gasto em reuniões mudou? Aumentou ou diminuiu? Em quanto porcento?	X	X	
7	Você sentiu uma melhoria nas entregas por produto e por tecnologia?	X	X	X

Fonte: Elaborado pelo autor

A primeira pergunta (“Você sente que o tempo da sua área está sendo melhor utilizado?”) questionou, de maneira objetiva, se o grupo de soluções atingiu o que deveria ser realizado, já que atualmente a frequência excessiva de reuniões, ou processos desnecessários no ciclo de desenvolvimento, consomem um tempo desnecessário de ambas as equipes.

A seguir, a pergunta de número 2 (“As expectativas de prazos e priorização, tanto quanto acompanhamento dos projetos sendo cumpridos?”) avaliou o alinhamento com os *stakeholders*. Com a criação do ritual “alinhamento com stakeholders”, um novo canal de comunicação formal será aberto entre *stakeholders* e produto, sendo esperado que as dificuldades de prazos, priorização e acompanhamento das novas funcionalidades sejam sanadas.

A pergunta de número 3 (“A restrição da "planilha de demandas" otimizou o tempo da sua área? Como isso impactou a qualidade das demandas?”) visou a observar o impacto da restrição da “planilha de demandas”, nas perspectivas de eficiência e de qualidade, pois a solução não poderia ser mantida caso prejudicasse o canal de comunicação de demandas de novas funcionalidades.

Os resultados da nova organização dos rituais foram observados na pergunta de número 4 (“A nova organização dos rituais otimizou o tempo da sua área? Como

isso impactou a comunicação?"), a fim de entender se ocorreu a melhoria e se impactou na comunicação entre as áreas.

Uma das principais mudanças que foi avaliada na nova organização dos rituais foi o aumento da frequência das entregas, para que o *deploy* seja realizado semanalmente. Este tópico foi abordado na pergunta de número 5 ("A frequência de entregas aumentou. Como isso impactou a eficiência? Como isso impactou na qualidade?").

Assim, a penúltima pergunta foi de característica objetiva, numérica, perguntando aos desenvolvedores se o tempo gasto em reuniões foi reduzido. O objetivo seria reduzir o tempo gasto em reuniões em 60%, pois a *Daily* e a reunião de Deploy irão permanecer (2 horas por semana); a *Planning, Review, Retrospective* deixará de existir (3 horas por semana); e, a nova reunião Alinhamento *stakeholders*, não irá afetar tecnologia. Dessa maneira, a proposta é de medir continuamente esse indicador, para que a metodologia proposta seja acompanhada.

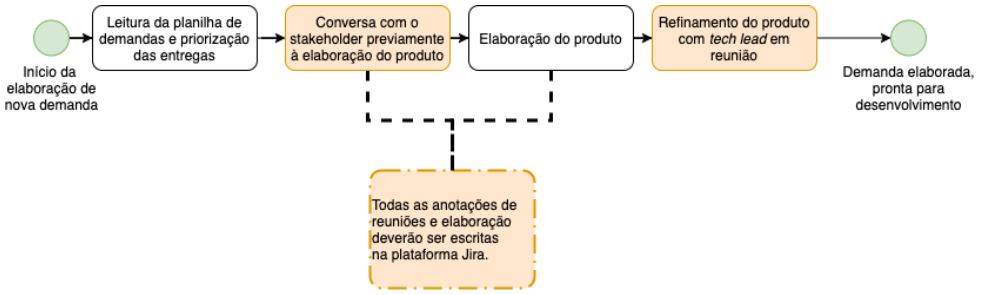
Por fim, foi questionado sobre o fator causal, na pergunta de número 6 ("Você sentiu uma melhoria nas entregas por produto e por tecnologia?") para que pudesse conversar, de forma objetiva, se a maior dor percebida pela empresa foi melhorada.

5.2. Grupo de soluções: Padronização do processo de elaboração do produto

O segundo grupo de soluções visou a padronizar o processo de elaboração do produto, que atualmente apresenta oportunidades significativas de melhoria. A elaboração do produto era realizada de maneira diferente por cada analista de produto, para cada tipo de produto construído; por isso, esse processo se apresentava tanto como ineficaz como ineficiente. A área responsável para estas soluções foi a área de produto, logo, para a implementação deste processo, esta área foi acompanhada durante o período de implementação, para checagem se o processo proposto está sendo seguido, já que corre o risco da não adesão por parte dos colaboradores.

Segue abaixo o fluxograma que engloba esse grupo de soluções, trazendo a obrigatoriedade de três etapas no fluxo, em laranja: conversa com *stakeholder* previamente à elaboração do produto; centralização das informações na plataforma do Jira; e refinamento do produto com *tech lead* em reunião.

Figura 21 – Proposta de solução: processo para elaboração do produto



Fonte: Elaborado pelo autor

5.2.1. Conversa com o *stakeholder* previamente à elaboração do produto

Essa solução pretendeu-se sanar a causa raiz identificada “falta de alinhamento dos requisitos do projeto”, em que requisitos faltantes eram detectados somente após a entrega, pois algumas das funcionalidades eram desenvolvidas sem sequer consultar o *stakeholder* que a solicitou na “planilha de demandas”. A falta de alinhamento causava a perda dos insumos sobre o que era necessário para a funcionalidade sanar as dores do *stakeholder* e, logo, as funcionalidades eram desenvolvidas e implementadas no sistema sem cumprir seu propósito e precisavam voltar para desenvolvimento.

A conversa com *stakeholder* previamente à elaboração do produto definiu que é responsabilidade da área de produto a investigação de qual a dor a nova funcionalidade deveria sanar, para que pudesse estabelecer quais eram seus requisitos e seu escopo através de uma reunião com o *stakeholder*, independentemente de qual é a demanda solicitada.

Assim, nessa etapa do processo foi proposto para o analista de produto a obrigatoriedade de agendar uma reunião com o *stakeholder* solicitante, para que se pudesse compreender e elaborar o produto em conjunto com a área de negócio.

5.2.2. Centralização das informações na plataforma do Jira

Essa próxima solução propôs resolver duas causas raiz: “falta de documentação clara do que foi pedido (mantendo esparso em diversos lugares)” e “falta de padronização das ferramentas para a elaboração”, através da centralização

das informações na plataforma do Jira. Muitas das vezes a informação ficava descentralizada em documentos, apresentações, e-mails ou diagramas; e o fluxo de informação entre os *stakeholders* e produto e entre produto e tecnologia era falho. Essa solução visou a padronizar a documentação em uma só ferramenta, a plataforma do Jira, que já era utilizada pela área de tecnologia para o acompanhamento do quadro *Kanban*.

Ou seja, o quadro *Kanban* passou a ser também uma ferramenta para armazenar as informações de elaboração do produto, para que toda comunicação seja anexada ali, como diagramas, conversas ou e-mails.

5.2.3. Refinamento do produto com *tech lead* em reunião

Por fim, a última solução foi a da obrigatoriedade de uma reunião entre o analista de produto e o *tech lead* para alinhamento do que deveria ser desenvolvido para cada funcionalidade, com o intuito de resolver a causa raiz "falta de alinhamento dos requisitos do projeto", causada também pela dificuldade de comunicação entre produto e tecnologia.

A causa dessa dificuldade foi identificada pela passagem de algumas funcionalidades para a equipe de tecnologia sem uma conversa com produto. Ou seja, a demanda poderia ter sido bem documentada por produto, mas não compreendida por tecnologia por conta dessa falta de alinhamento.

Esse alinhamento obrigatório entre produto e o *tech lead* visou então a resolver essa divergência do que deveria ser realizado, para que as funcionalidades sejam desenvolvidas conforme definido pelos analistas de produto.

5.2.4. Métricas

Da mesma forma do grupo de soluções abordados no capítulo anterior, pretendeu-se mensurar o impacto das soluções através das entrevistas semiestruturadas. Foram testadas no fim das entrevistas as seguintes hipóteses:

- É possível padronizar um processo de elaboração de produto;
- O processo está mais eficiente;
- As funcionalidades estão correspondendo mais ao que foi solicitado;
- O tempo gasto médio em reuniões não aumentará na área de produto;

- O número de novas funcionalidades com defeito, que retornam para desenvolvimento, diminuirá em 50%.

Para que se pudesse identificar esses pontos, foram realizadas as seguintes perguntas aos entrevistados:

Tabela 3 – Métricas para a padronização do processo de elaboração do produto

#	Pergunta	Para produto	Para tecnologia	Para stakeholders
1	Você sente que existe um processo padronizado para a elaboração do produto?	X		
2	O alinhamento com a área de negócio melhorou, piorou, ou não alterou?	X		X
3	A descrição das funcionalidades está mais bem definida?	X	X	
4	A frequência de retrabalhos (funcionalidades que precisam voltar para desenvolvimento) melhorou, piorou, ou não alterou?	X	X	X
5	Quais são as ferramentas utilizadas para descrever o produto?	X	X	
6	Você sente que o tempo gasto em reuniões mudou? Aumentou ou diminuiu? Em quanto porcento?	X		
7	Você sente que o número de novas funcionalidades com defeito, que retornam para desenvolvimento, mudou? Aumentou ou diminuiu? Em quanto porcento?	X	X	X
8	Você sentiu uma melhoria nas entregas por produto e por tecnologia?	X	X	X

Fonte: Elaborado pelo autor

A pergunta de número 1 (“Você sente que existe um processo padronizado para a elaboração do produto?”) visou a compreender se a solução foi realmente implementada, já que pode não ocorrer a adesão de todos os colaboradores.

Nas próximas duas perguntas, de números 2 e 3, visou a analisar o alinhamento entre as áreas. Na pergunta de número 2 (“O alinhamento com a área de negócio melhorou, piorou, ou não alterou?”) pretendeu-se identificar a melhora do alinhamento entre a área de negócio e a área de produto; enquanto que a pergunta de número 3 (“A descrição das funcionalidades está mais bem definida?”) pretendeu identificar a melhora do alinhamento entre a área de produto e a área de tecnologia.

Enquanto que as perguntas de 1 a 3 tenderam a ser mais subjetivas, pois representam a percepção do colaborador perante a esses ajustes, a pergunta de

número 4 (“A frequência de retrabalhos (funcionalidades que precisam voltar para desenvolvimento) melhorou, piorou, ou não alterou?”) foi um indicador objetivo, que é a frequência que funcionalidades precisam retornar para a área de tecnologia, por conta de alguma falha ou desalinhamento do produto entregue.

A pergunta de número 5 (“Quais são as ferramentas utilizadas para descrever o produto?”) verificou se ainda existe uma variedade de ferramentas sendo utilizadas pela equipe de produto; e a pergunta de número 6 (“Aonde estão salvas as novas documentações da elaboração de uma funcionalidade?”) verificou se ainda permanecia a dispersão dos documentos para elaboração das funcionalidades.

A pergunta de números 6 questiona sobre o tempo gasto em reunião de produto de maneira objetiva. Espera-se que o tempo gasto em reuniões não aumente, pois as reuniões de *Planning*, de *Review* e de *Retrospective* não serão mais realizadas, levando a uma redução de 3 horas em reuniões por semana para produto; contudo, a área de produto também terá mais reuniões, como a de Alinhamento *Stakeholders*, ou as reuniões obrigatórias com a parte interessada previamente à elaboração do produto; portanto, espera-se que não aumente o tempo gasto com reuniões para a área.

Em paralelo que a pergunta de número 7 também trata de maneira objetiva sobre o número de funcionalidades com defeito. Com o tempo gasto em reuniões com *stakeholders*, espera-se que o número de funcionalidades com defeito diminua: atualmente a cada cinco entregas realizada por um desenvolvedor, uma apresenta defeito e deve voltar para desenvolvimento; foi estabelecido pela diretoria da empresa, o objetivo de ter uma funcionalidade defeituosa a cada, no mínimo, dez entregas.

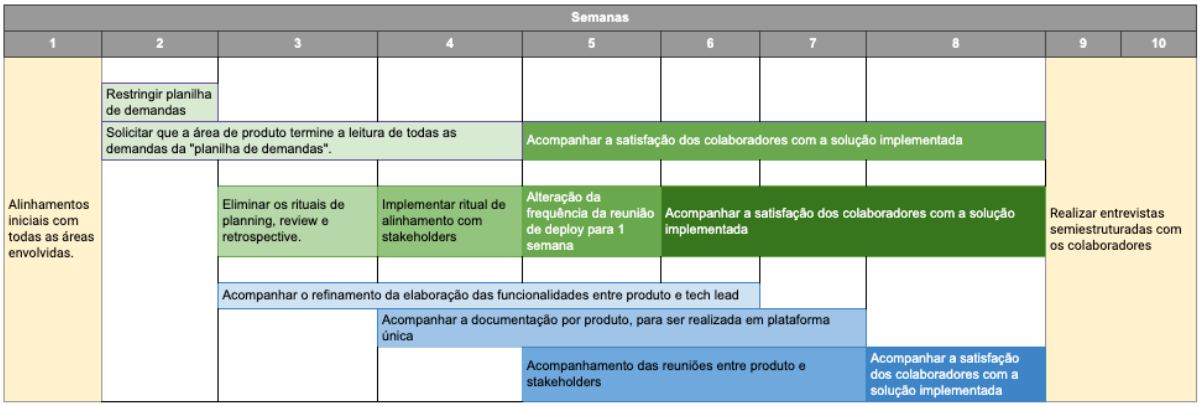
Da mesma forma do tópico 5.1.3., esses indicadores objetivos têm também como função de medir continuamente, para que a metodologia proposta seja acompanhada.

Por fim, a última pergunta dessa entrevista foi a mesma da entrevista anterior, para analisar se os entrevistados sentiram melhoria nas entregas das áreas de produto e de tecnologia.

5.3. Cronograma de implementação das soluções

O seguinte roteiro de implementação foi construído através de um cronograma de implementação para cada etapa. O objetivo deste cronograma não foi de segui-lo à risca, mas foi de acompanhar se o projeto estava caminhando conforme o desejado.

Figura 22 – Roteiro de implementação



Fonte: Elaborado pelo autor

O roteiro de implementação foi dividido em três categorias, conforme a figura acima. As partes de cor amarela são as etapas de planejamento e conclusão da implementação, enquanto que as partes de cor verde é a implementação do grupo de soluções “gestão do tempo” e as partes em cor azul é a implementação do grupo de soluções “processo de elaboração do produto”.

Na primeira semana, pretendeu-se realizar os alinhamentos iniciais com todas as áreas envolvidas, apresentando o cenário atual que a empresa se encontra e os grupos de solução, junto com as métricas criadas para cada solução. A proposta deste alinhamento foi de fazer em reuniões com poucos participantes (até 5 colaboradores) para que todos pudessem estar cientes do projeto. Portanto, como existem muitos envolvidos, a ideia foi de apresentar em várias reuniões durante a semana: uma para a equipe de produto, outra para a equipe de desenvolvimento e outra com os líderes das áreas de *stakeholders*.

A implementação da “restrição da planilha de demandas” foi planejada no início na segunda semana, por conta do seu custo baixo de implementação e maior valor agregado. A cada dia que se passava caso a “planilha de demandas” não fosse restrita, mais demandas poderiam ser adicionadas neste canal, e mais os colaboradores poderiam se frustrar por não serem atendidos; portanto, a implementação teria como início na restrição do canal da “planilha de demandas” para

somente os líderes de cada área; além de realizar o alinhamento entre todos os colaboradores com essa tomada de decisão. Assim, a área de produto focaria em concluir a leitura de todas as demandas até a quarta semana deste cronograma.

Na terceira semana, foi planejado que os rituais de *planning*, *review* e *retrospective* seriam eliminados para que a equipe de desenvolvimento tivesse mais tempo para desenvolvimento de código: tanto o ritual de *review* quanto o ritual de *retrospective* deveriam ser realizados no ritual de *daily*, conforme especificado no capítulo 5.1.2. Já o ritual de *planning* deveria ser substituído por reuniões individuais entre produto e tech lead, através do refinamento das ideias de produto, conforme especificado no capítulo 5.2.3.

Na quarta semana pretendeu-se implementar o ritual de alinhamento com os *stakerholders*, através de reuniões semanais de uma a duas horas, para que a área de produto se disponibilizasse ao fim do dia para que os líderes das áreas de negócio pudessem questionar e entender o que estivesse sendo desenvolvido pela área. Em paralelo ao acompanhamento do processo de elaboração de produto para que este fosse realizado em plataforma única, no quadro *Kanban* do Jira. Sobretudo, nessa semana pretendeu-se terminar a leitura de todas as demandas na "planilha de demandas", que teria dado início na segunda semana de implementação.

Na quinta semana pretendeu-se iniciar o acompanhamento da "restrição da planilha de demandas", que já estaria restrita por três semanas, para verificar qual foi a reação dos colaboradores da empresa, tanto como saber se a decisão precisaria de alguma alteração. Nessa semana também deveria iniciar a frequência semanal do ritual de *deploy*, cuja reunião realizada entre um analista de produto e um tech lead implementa tudo que foi desenvolvido pela equipe. Sobretudo, as reuniões entre produto e *stakeholders* seriam acompanhadas, para verificar se toda a documentação está sendo realizada através da plataforma do Jira.

Na sexta e na sétima semana de implementação, pretendeu-se respectivamente concluir os processos de acompanhamento do refinamento da elaboração das funcionalidades entre o analista de produto e um tech lead, tanto quanto no acompanhamento para que toda a documentação das tarefas fosse padronizada na ferramenta do Jira. Isto é, no fim dessas semanas, foi esperado que as equipes envolvidas já respeitassem o processo, cumprindo com o que foi combinado.

Assim, na oitava semana foi proposto que um acompanhamento da satisfação dos colaboradores com o novo processo de elaboração do produto fosse realizado, verificando se é necessário algum ajuste antes da mensuração dos resultados.

Por fim, nos períodos da nona e da décima semana tiveram como objetivo realizar entrevistas semiestruturadas com os colaboradores envolvidos, com a mensuração das métricas elaboradas nos capítulos 5.1.3. e 5.2.4., compreendendo se os objetivos foram alcançados e quais são as percepções dos envolvidos.

6. Relato da implementação

Nesse capítulo, o trabalho desenvolvido foi apresentado aos colaboradores da empresa do estudo de caso, a *Nexoos Sociedade de Empréstimo Entre Pessoas*, levantando as reflexões obtidas, tanto como as críticas de cada área, apontadas em carácter construtivo. Para isso, o primeiro subcapítulo levantou o cenário atual da empresa, apontando as dificuldades e as dores de cada área, tanto como qual a situação que a empresa enfrenta atualmente.

No subcapítulo seguinte, foi documentado o acompanhamento do cronograma de implementação, definido no capítulo 5.3., através do andamento das etapas definidas pelo cronograma para cada grupo de soluções, descrevendo como as implementações foram recebidas, tanto como quais foram as dificuldades da implementação.

Assim, os insumos obtidos pelas entrevistas semiestruturadas foram apresentados no último subcapítulo, tratando da percepção dos colaboradores das equipes envolvidas, das áreas de produto, de tecnologia e de *stakeholders*.

6.1. Cenário atual da empresa

Como apontado na metodologia desse trabalho, foi essencial realizar uma documentação da situação atual da empresa, para que se pudesse servir de comparação após a implementação das soluções. A documentação do cenário atual da empresa serviu de apresentação para os colaboradores das áreas envolvidas (área de produto, de tecnologia e de *stakeholders*) para que todos estivessem alinhados de como a empresa se encontrava, e o que se pretendia alcançar com as soluções.

A empresa do estudo de caso, a *Nexoos Sociedade de Empréstimo Entre Pessoas*, havia crescido de maneira rápida nos últimos anos, sem que tivesse sido possível criar uma estrutura sólida para os processos de desenvolvimento de novas funcionalidades. Por conta do crescimento desenfreado da empresa, as áreas de negócio começaram a demandar mais funcionalidades, que deviam ser atendidas pelas áreas de produto e de tecnologia. Para tal, as duas áreas haviam crescido sem se preocupar com a estrutura dos processos para o CVDS, carregando uma mentalidade de foco nas entregas. Sem se atentar ao processo de construção de software, ele se tornou ultrapassado, com uma estrutura frágil que dificulta atualmente o desenvolvimento de novas funcionalidades, resultando no crescimento do atrito entre as áreas de *stakeholder* e as áreas de produto e de tecnologia.

Após um profundo diagnóstico realizado através das entrevistas semiestruturadas, do mapeamento do processo de desenvolvimento de software e da análise de causa raiz, foram construídos dois grupos de solução: o primeiro, visou a melhorar a gestão do tempo, aumentando a eficiência do uso do tempo pelas áreas de tecnologia e de produto. Uma das soluções deste grupo de solução foi a da restrição do acesso à planilha de demandas; a outra foi a da reorganização dos rituais da metodologia de desenvolvimento de software atuais. Com isso, pretendeu-se melhorar também o alinhamento com as áreas de negócio, pois muitos dos processos aqui envolviam a comunicação entre a área de produto e de *stakeholders*. As métricas para este grupo de soluções, definidas no capítulo 5.1.3., foram:

- O tempo das equipes de produto e de tecnologia será utilizado de maneira mais eficiente;
- O alinhamento entre a área de produto e de *stakeholders* melhorará;
- O tempo gasto médio em reuniões reduzirá em 60% na área de tecnologia.

O outro grupo de soluções elaborado nesse trabalho foi a da padronização do processo de elaboração do produto, que visou a padronizar o processo de elaboração de produto. Este processo variava para cada analista de produto e para cada projeto, de maneira aleatória; por isso, algumas funcionalidades não ficavam bem definidas e acabavam precisando retornar para desenvolvimento, já que não se enquadravam aos requisitos das áreas de negócio. Assim, algumas alterações no fluxo atual de elaboração do produto visaram a solucionar esses problemas, de foram mensuradas por:

- É possível padronizar um processo de elaboração de produto;
- O processo está mais eficiente;
- As funcionalidades estão correspondendo mais ao que foi solicitado;
- O tempo gasto médio em reuniões não aumentará na área de produto;
- O número de novas funcionalidades com defeito, que retornam para desenvolvimento, diminuirá em 50%.

Foi proposto que a implementação realizada seria no único *squad* de tecnologia, com o único analista de produto restante. Com o início da pandemia do Covid-19 nos últimos meses, a *Nexoos* teve uma queda brusca no volume de

investimentos que recebia em sua plataforma, o que levou à escassez da oferta de crédito para as empresas solicitantes. Ao oferecer menos crédito às empresas, a receita da *Nexoos* diminuiu, levando ao desligamento da maioria dos colaboradores da empresa.

Todavia, o cenário difícil que a empresa esteve favoreceu a implementação das soluções propostas nesse relatório, pois as equipes de tecnologia e de produto se tornaram mais enxutas, facilitando a reestruturação dos processos e da implementação desses grupos de solução na empresa.

6.2. Acompanhamento do cronograma de implementação

Seguindo o cronograma de implementação definido no tópico 5.3., o relato da implementação foi registrado, seguindo como foram implementadas as soluções propostas, de forma a detalhar o desenrolar do que foi proposto.

6.2.1. Implementação do grupo de soluções: Gestão do tempo

Na semana posterior aos alinhamentos iniciais com as equipes envolvidas (áreas de produto, de tecnologia e de *stakeholders*), a "planilha de demandas" foi restringida. Previamente, os gestores de cada área de *stakeholder* já estavam cientes e já haviam repassado para as suas equipes; contudo, por conta dos desligamentos ocorridos devido ao Covid-19, a solicitação por novas demandas já havia diminuído, o que facilitou a restrição do documento.

Além disso, foi solicitado ao analista de produto o início da leitura de todas as demandas do documento, para que este fosse finalizado; contudo, o analista optou por descartar todas as demandas ao invés de lê-las. Como havia um número de demandas muito grande no documento e sua equipe havia diminuído de três membros para somente um, tornou-se inviável essa leitura; sobretudo, mesmo se houvesse uma demanda interessante, ela não seria priorizada nas próximas semanas.

Ao longo das semanas seguintes, a proposta de processo para o uso da "planilha de demandas", representado pela figura 19, foi instaurada; todavia, logo depois o processo deixou de existir. Como os gestores das áreas de *stakeholders* eram os responsáveis por receber as demandas de sua equipe e transcrevê-las na "planilha de demandas", eles optavam por falar diretamente com o analista de produto durante a reunião de alinhamento de *stakeholders*, o ritual que foi implementado posteriormente, devido ao fato de que este é um canal de comunicação mais prático. Sobretudo, os analistas das áreas de *stakeholders* não apresentaram reclamações

com a remoção do documento, pois eles já estavam insatisfeitos com o canal de comunicação antigo. Com isso, a planilha de demandas deixou de existir, sendo substituída pela reunião de alinhamento com *stakeholders*.

Em paralelo à restrição da “planilha de demandas”, os rituais do CVDS foram reorganizados, iniciando pela eliminação dos rituais de *Planning*, de *Review* e de *Retrospctive*. Essa eliminação havia sido solicitada por todos os analistas da área de tecnologia durante o diagnóstico, como acabou também sendo solicitado o retorno desses rituais ao fim da implementação. A *Review* e a *Retrospective* começaram a ser realizadas diariamente durante a reunião de *Daily*, em que o *squad* comentava o que foi feito de bom e o que deve se melhorar; contudo, existia uma falta de sentimento de “conclusão” que esses dois rituais traziam, com o fechamento de um ciclo. Assim, foi possível reparar que os rituais que eram julgados como excessivos e cansativos pela equipe de tecnologia, passaram a ser melhor apreciados com a falta deles.

Além da eliminação dos rituais citados acima, a Planning, que era uma reunião realizada semanalmente com todos do *squad*, foi substituída por uma reunião realizada em dupla, com o analista de produto e o *tech lead*. Inicialmente ela causou em uma alienação e individualidade da equipe de tecnologia, pois existia um laço anteriormente entre os desenvolvedores através do compartilhamento do conhecimento pela reunião de *Planning*, onde o conhecimento técnico era refletido. Em contrapartida, junto a essas reclamações, as equipes começaram a melhorar em suas entregas, já que existia maior foco no que era demandado.

Na semana seguinte, o novo ritual de Alinhamento com os *stakeholders*, uma reunião de acompanhamento dos projetos atuais entre o analista de produto e as áreas de *stakeholder*, foi implementado. Este ritual era realizado no período da noite, um horário vago para todos os gestores das áreas de *stakeholders*, abrindo um canal de comunicação propício para o alinhamento entre as áreas. No início, a reunião apresentava uma aderência alta entre os gestores, que procuravam esclarecimentos das demandas solicitadas por seus analistas; todavia, com o passar do tempo, essa procura diminuiu devido à restrição da planilha de demandas, demonstrando uma melhora no alinhamento e na transparência da equipe de produto. Sobretudo, o analista de produto precisava previamente realizar um alinhamento desgastante, de conversar individualmente com cada líder; enquanto que agora passou a se tornar uma reunião em conjunto, sendo possível descentralizar o trabalho de priorização com as áreas de *stakeholders*.

Por fim, a última solução implementada foi o aumento da frequência da reunião de *deploy*, reunião em que analista de produto e *tech lead* implementavam as funcionalidades desenvolvidas pela equipe de tecnologia no sistema de produção. A frequência, que passou a ser realizada semanalmente, aumentou a eficácia das funcionalidades novas, já que elas eram entregues mais rapidamente, testadas, e corrigidas na semana seguinte. Além disso, os analistas de tecnologia apreciaram essa medida, pois sentiam maior dinamismo ao ver suas funcionalidades implementadas mais rapidamente, da mesma forma que não chegou a aumentar o ritmo de trabalho. Por outro lado, o ritmo de trabalho para produto aumentou, pois essa área era responsável por testar e aprovar as funcionalidades antes de entrarem no sistema. Logo, o trabalho para o analista de produto aumentou consideravelmente, resultando em um gargalo na etapa de validação por produto, especialmente por ser uma área que ele se encontrava só.

6.2.2. Implementação do grupo de soluções: Padronização do processo de elaboração do produto

Na terceira semana de implementação, junto à eliminação dos rituais de *Planning*, *Review* e *Retrospective*, o acompanhamento das reuniões de refinamento entre o analista de produto e o *tech lead* foi iniciado. O intuito desse acompanhamento era de trazer a obrigatoriedade dessa reunião no processo de elaboração do produto, para que o refinamento seja feito em conjunto entre produto (parte de negócio) e *tech lead* (parte técnica). Sobretudo, essa reunião se tornou essencial com a eliminação do ritual da *Planning*, já que ela estaria substituindo este ritual para que fosse feito o planejamento das entregas. Esta reunião trouxe resultados benéficos para a entrega de novas funcionalidades, pois permitiu que os desenvolvedores da equipe de tecnologia pudessem compreender melhor o que precisava ser realizado, tanto do lado técnico como do lado de negócio.

Na semana posterior, as reuniões com as áreas de *stakeholder* foram acompanhadas para que pudesse verificar que tudo estivesse sendo documentado na plataforma do Jira, para centralizar as informações. Inicialmente foi um processo difícil de ser implementado, já que a área de produto não acessava o Jira com muita frequência; contudo, após melhor conhecer a ferramenta e criar o hábito de utilizá-la, a nova ferramenta foi bem aceita. A centralização da documentação da funcionalidade

nova na plataforma do Jira melhorou a produtividade da equipe de produto, pois não seria necessário gastar tempo procurando onde estava toda a informação daquela nova funcionalidade em diversos lugares; tanto como evitou incidências de perda de informação, devido às informações esparsas em diversos lugares.

Por fim, a última implementação foi do acompanhamento das reuniões entre produto e *stakeholders*, para que tornasse essa comunicação durante a elaboração do produto obrigatória. Logo, toda a reunião entre ambas as áreas passou a ser documentada no Jira e encaminhada por e-mail, para que os *stakeholders* realizassem a checagem. Por consequência, as funcionalidades novas passaram a estar mais bem definidas, e os desalinhamentos passaram a ser percebidos durante a checagem do e-mail aos *stakeholders*. Assim, a responsabilidade para o alinhamento da funcionalidade passou a pertencer tanto ao analista de produto como ao *stakeholder*.

6.3. Entrevistas semiestruturadas

Após o período de 7 semanas de implementação, o período de coleta dos resultados se iniciou, através das entrevistas semiestruturadas. Ao realizar as entrevistas dos subcapítulos 5.1.3. e 5.2.4. foram obtidos alguns insumos para a análise dos resultados.

As áreas de produto e de tecnologia sentiram que seu tempo está sendo melhor utilizado, por conta da reorganização dos rituais; todavia, nasceram dois novos problemas. Primeiramente, a área de produto está se sentindo sobrecarregada, pois a quantidade de entregas aumentou substancialmente, precisando validar muitas das funcionalidades entregues; em paralelo que a área de tecnologia acredita que essa alteração individualizou o trabalho, transformando o desenvolvimento de novas funcionalidades em um trabalho focado em entregas, e não na troca de conhecimento entre os colaboradores, o que resultou também na piora na gestão do conhecimento, já que cada desenvolvedor tomaria conta de uma tarefa, sem conhecer o que a equipe inteira fazia.

Em relação às expectativas de prazo, priorização, acompanhamento e alinhamento das novas funcionalidades, as áreas de *stakeholder* e de produto sentiram uma melhoria. O principal ponto levantado foi um aumento na transparência, que não era prioritária antes da implementação das propostas de melhoria deste

trabalho; logo, atividades como de se comunicar e de se alinhar com os *stakeholders* estavam sendo deixadas de lado, e isso gerava grande atrito entre ambas as áreas.

Sobretudo, foi implementada com sucesso a padronização de um processo claro para elaboração de novas demandas, trazendo melhorias na eficiência e na eficácia do processo, já que o alinhamento entre as áreas de *stakeholders* e de produto, e entre produto e tecnologia melhoraram. Com isso, funcionalidades novas não voltavam para desenvolvimento com a frequência que ocorria previamente, levando a um aumento de valor contínuo.

7. Análise dos Resultados

Após a implementação, seus resultados foram analisados levando em conta as métricas elaboradas no capítulo 5, a fim de observar se os objetivos foram atingidos com a implementação dos grupos de solução. Posteriormente, foi analisado o impacto desses resultados no diagrama de ACR (tópico 4.3.1), observando a relação de causa e efeito das dificuldades da empresa.

7.1. Resultados para o grupo de soluções: Gestão do tempo

As quatro hipóteses levantadas com a implementação desse grupo de solução foram: “O tempo das equipes de produto e de tecnologia será utilizado de maneira mais eficiente”, “O alinhamento entre a área de produto e de *stakeholders* melhorará” e “O tempo gasto médio em reuniões reduzirá em 60% na área de tecnologia”.

Para o primeiro ponto, foi observado através das entrevistas semiestruturadas que a eficiência do trabalho de ambas as equipes melhorou. A equipe reduziu a quantidade de processos que não geravam valor para o negócio (os rituais de *Planning*, *Review* e *Retrospective*) sem perder a qualidade das entregas, já que as entregas eram testadas com uma frequência semanal, ao invés de quinzenal, podendo ser validadas com maior frequência. Contudo, houve um efeito de individualização da equipe de tecnologia, causando desmotivação dos membros da equipe: cada colaborador da área de tecnologia trabalhava individualmente, sem o contato com as demais pessoas da equipe. Em média, os desenvolvedores sentiram redução do tempo gasto em reuniões em 70%, podendo chegar a 80% dependendo da semana.

O alinhamento entre a área de produto e de *stakeholders* teve uma melhora significativa, por conta do fator da descentralização de muitos processos da área de produto. Por exemplo, com a remoção da “planilha de demandas”, os próprios interessados, das áreas de *stakeholders*, se tornavam responsáveis por priorizar e se alinhar das demandas necessárias, no novo ritual de “Alinhamento dos *stakeholders*”. Muitas das decisões tomadas pela área de produto passaram a ser tomadas em conjunto, com os *stakeholders*, evitando o atrito que existia na comunicação.

As soluções para melhoria da gestão do tempo trouxeram os resultados que eram almejados, identificados no diagnóstico do problema, mas trouxeram o problema da individualização da equipe de tecnologia. Enquanto que o relacionamento entre as áreas de produto e de *stakeholder* melhoraram, e foram observados melhores

resultados com uma entrega mais eficiente e mais eficaz de novas funcionalidades, será necessário melhorar a motivação da equipe de tecnologia do estudo de caso.

7.2. Resultados para o grupo de soluções: Padronização do processo de elaboração do produto

O próximo grupo de soluções, de padronização de elaboração do produto, pretendia validar cinco hipóteses: primeiramente, validar se é possível padronizar um processo de elaboração do produto; posteriormente, tornar mais eficiente esse processo de elaboração do produto; assim, analisar se as funcionalidades estariam correspondendo melhor ao que foi solicitado; contudo, o tempo gasto em reuniões médio na área de produto não poderia aumentar; resultando em uma diminuição em 50% das funcionalidades que apresentam defeito, e devem retornar para desenvolvimento.

Para o primeiro ponto, foi identificado com todos das áreas de produto e de tecnologia que agora existe um processo bem definido de elaboração de produto. Isso foi muito bem elogiado pelas equipes, pois existiam dúvidas se seria possível padronizar a elaboração de produto, tendo em vista que cada funcionalidade varia de característica, podendo ser diferentes em cada ponto. A maior melhoria identificada é que, com um processo bem definido, foi possível definir tarefas que precisam ser realizadas, seguindo uma rotina bem definida e organizada, para que também pudesse ser difundida para próximos analistas de produto. Outro ponto também é a possibilidade de aplicar melhorias no processo, através da identificação de gargalos, ou proposta de novas ferramentas que auxiliassem produto, no processo de elaboração.

Assim, este processo se tornou mais eficiente, por conta do mesmo fator notado no grupo de soluções anterior: a descentralização das tarefas de produto. Como a área de produto conseguiu ter uma documentação organizada das novas funcionalidades, centralizada na plataforma do Jira, foi possível compartilhar essa documentação com os *stakeholders*. Dessa forma, tornou-se também responsabilidade dos interessados conferir se o que foi solicitado correspondia às suas necessidades, evitando os desalinhamentos que ocorriam. Além disso, trazendo a obrigatoriedade de uma reunião de alinhamento entre *tech lead* e o analista de produto trouxe um embasamento mais técnico do que é necessário para ser desenvolvido, para ser compreendido pelo desenvolvedor. Todavia, esse processo

bem definido de elaboração de produto causou o aumento das tarefas do analista de produto, sobrecarregando sua rotina; a meta de manter o tempo gasto em reuniões não foi bem sucedida, e o analista de produtos sentiu que seu tempo gasto dobrou para alinhamento.

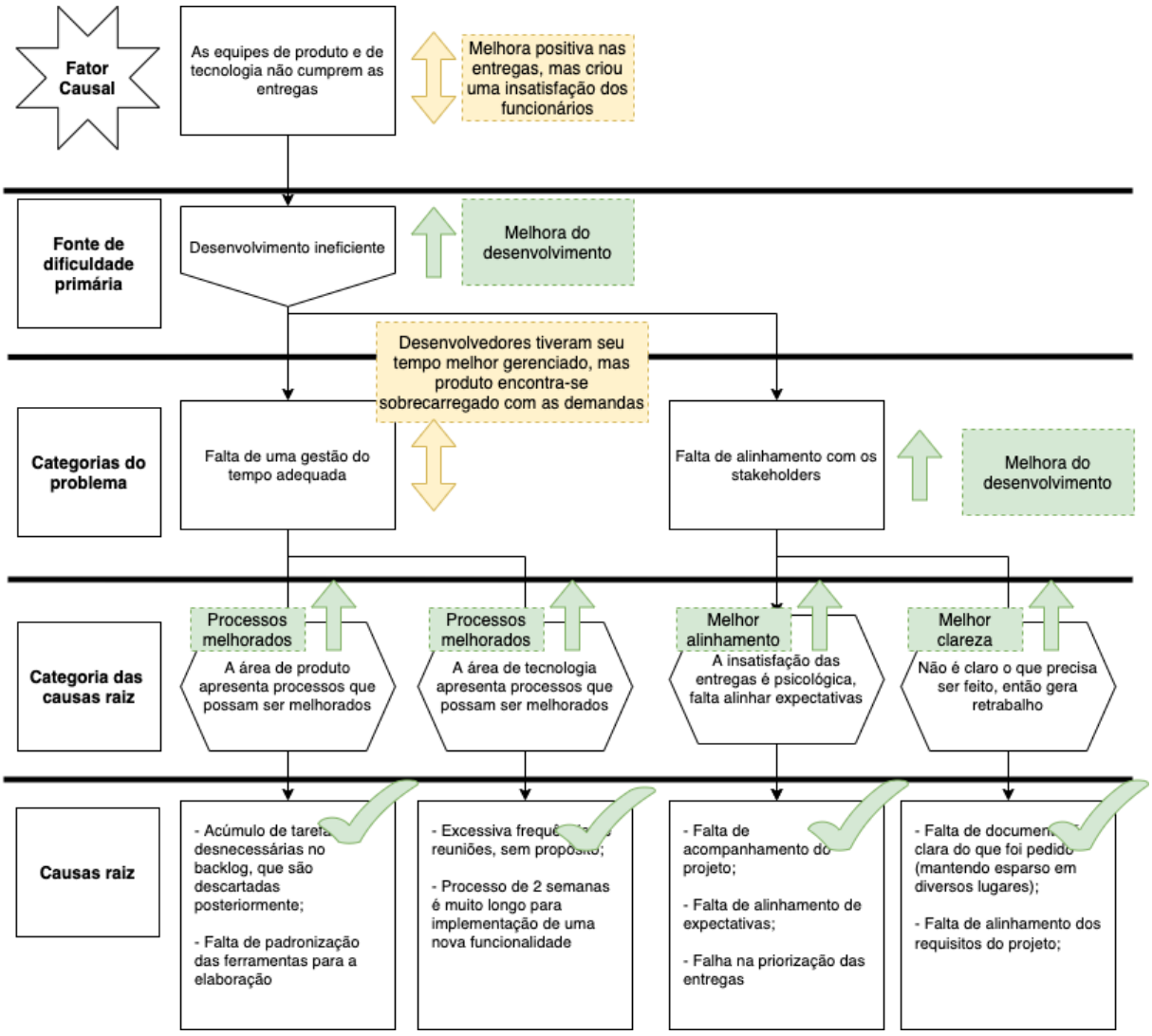
Por fim, a frequência de retrabalhos (funcionalidades que precisam voltar para desenvolvimento) diminuiu, já que correspondiam melhor ao que foi solicitado. Como o alinhamento entre as áreas de *stakeholder* e de produto melhoraram, como também melhorou a comunicação mais técnica entre as áreas de produto e de tecnologia, muitos dos problemas foram sanados, evitando as falhas que existiam na comunicação prévia. Previamente, foi estimado que 1 a cada 5 funcionalidades apresentariam defeitos; enquanto que nas últimas 3 semanas ao fim da implementação foram entregues 12 funcionalidades, e somente 1 apresentava problema.

Com isso, com a implementação deste grupo de soluções, as áreas de *stakeholder*, de produto e de tecnologia foram beneficiadas, trazendo um melhor alinhamento e mais eficiência no processo produtivo. O único ponto de piora foi a sobrecarga na área de produto, resultando em gargalo no processo produtivo na etapa da validação das novas funcionalidades produzidas.

7.3. Impactos na Análise de Causa Raiz

Assim, foram analisados os impactos das implementações dos grupos de solução na ACR, conforme a imagem a seguir, para que fosse possível observar as relações de causa e efeito dos problemas encontrados durante diagnóstico na empresa do estudo de caso.

Figura 23 – Impactos na Análise de Causa Raiz



Fonte: Elaborado pelo autor

A abordagem da análise de causa raiz apresenta um resultado que aparenta ser contraditório: enquanto que todas as categorias das causas raiz foram resolvidas, seu fator causal e uma de suas categorias do problema não foram. Isso se dá por conta da relação distante de causa-efeito entre as causas raiz e os outros elementos: ao solucionar os problemas que estão na edificação do fator causal (da parte mais inferior do diagrama), não necessariamente os problemas mais distantes (partes superiores do diagrama) serão também solucionados, já que podem existir outros fatores que não foram considerados durante a etapa de diagnóstico, tanto como a contenção de novos problemas surgirem.

Dessa forma, pelos resultados obtidos na ACR foi possível notar que as áreas de produto e de tecnologia passaram a ter processos mais eficientes e melhorados, a satisfação dos *stakeholders* melhorou e ficou claro para as equipes o que precisa ser realizado na descrição das novas funcionalidades. Todavia, no nível posterior de categorias do problema, enquanto que o alinhamento com os *stakeholders* foi resolvido, a falta de gestão do tempo adequada não foi resolvida completamente: enquanto que a área de tecnologia teve um melhor aproveitamento de seu tempo, a área de produto se tornou gargalo no processo de produção de novas funcionalidades, por conta de mais atividades obrigatórias que não existiam previamente.

Foi possível notar que, embora a fonte de dificuldade primária dependesse das duas categorias do problema citadas no parágrafo acima, ela foi resolvida, pois houve uma melhora no desenvolvimento de novas funcionalidades. Isso se dá por conta da imprevisibilidade da ACR nos níveis superiores, já que não é garantido que todos os fatores em conta sejam explorados. Ao resolver o problema de desenvolvimento ineficiente, foi gerada outra dificuldade primária: a insatisfação dos colaboradores das áreas de produto e de tecnologia.

Assim, o fator causal foi resolvido, trazendo uma melhora positiva nas entregas, mas ao custo de prejudicar a motivação dos colaboradores das áreas de produto e de tecnologia: a área de tecnologia encontra-se desmotivada, com a individualização de seu trabalho; enquanto que a área de produto encontra-se sobrecarregada, com muitas atividades obrigatórias a serem realizadas.

8. Conclusão

Nesse estudo de caso, o objetivo foi de analisar e implementar melhorias do ciclo de vida de desenvolvimento de software em uma *fintech*, através de uma metodologia de diagnóstico dos problemas, proposta de implementação, relato da implementação e análise dos resultados. Após a realização de todo esse percurso, os resultados implementados foram observados, deixando claros os benefícios que a metodologia trouxe para o ciclo de desenvolvimento de software.

Dessa maneira, mesmo com os contratempos que ocorreram durante a implementação, a metodologia ainda assim conseguiu aumentar substancialmente a produtividade das equipes de tecnologia e de produto; em detrimento da motivação dos colaboradores. Assim, resultou em uma equipe capacitada a entregar valor elevado, mas desmotivada, fator que não seria sustentável por muito tempo; que será discutido nos subcapítulos abaixo.

8.1. Objetivos alcançados

O objetivo principal deste trabalho foi de propor e avaliar a eficiência e a eficácia da metodologia aqui desenvolvida para a resolução do problema identificado como fator causal, através do diagnóstico do problema.

No caso da *Nexoos*, foi identificado como fator causal “As equipes de produto e de tecnologia não cumprem as entregas”, tal qual foi resolvido, conforme analisado no capítulo 7 do estudo. Portanto a metodologia foi útil para identificar as principais dores e resolvê-las através da ACR, pela resolução dos problemas classificados como causas raiz. Tanto a área de produto, quanto a de tecnologia e as de *stakeholders* constataram maior valor de entrega para a empresa, especialmente devido à diminuição da quantidade de funcionalidades com defeito, que inicialmente era uma a cada cinco, passando para uma a cada doze, nas últimas três semanas de implementação. Com o decorrer do projeto, esse indicador passou a ser seguido, sendo consolidado no processo de construção de novas funcionalidades, para que fosse levado em consideração durante o processo de desenvolvimento de software.

Através da identificação e melhoria de processos que eram desnecessários e ineficientes pelo grupo de soluções de gestão do tempo, os colaboradores da empresa sentiram um processo de desenvolvimento de software mais fluido e menos engessado. Desta forma, foi possível notar que, embora a *Nexoos* seguisse os rituais e os processos de uma metodologia ágil, as entregas ocorriam sem respeitar os

princípios do manifesto ágil (Qureshi & Hussain, 2008), como a falta de entregas frequentes e contínuas, junto à desarmonia entre negócio e tecnologia. Junto com a padronização do principal processo da área de produto ("elaboração do produto"), foi possível trazer um melhor alinhamento entre as áreas de *stakeholders* e produto, e de produto e de tecnologia, devido a uma melhor organização deste processo.

8.2. Propostas de melhoria

Contudo, conforme identificado no capítulo 7.3., essa metodologia também trouxe dificuldades para as equipes de produto e de tecnologia, pois enquanto a primeira se encontrava sobrecarregada, sentindo que o tempo gasto em reuniões dobrou, por causa da quantidade de tarefas excessivas que se tornaram obrigatórias, a segunda se encontrava desmotivada, por conta da individualização do trabalho, sentindo que o tempo gasto em reuniões (interação com outros desenvolvedores) chegou a reduzir entre 70% a 80% dependendo da semana da entrega. Assim, foi observado que essa metodologia poderia acarretar efeitos secundários.

Assim, foi observado que, embora os objetivos propostos tivessem sido alcançados, restaria ainda um trabalho contínuo para melhorar o CVDS da *Nexoos*, trazendo uma necessidade de reaplicar a metodologia para que fosse possível identificar seu novo fator causal.

Dessa maneira, a proposta de melhoria levantada durante a aplicação da metodologia desenvolvida foi a de implementá-la como metodologia ágil, e não pesada, para que se pudesse reduzir esses efeitos secundários, através de um planejamento menos enrijecido. Embora o formato de metodologia pesada tenha sido escolhido no início do trabalho para que fosse melhor compreendida pelas áreas negócio, vale a tentativa de implementá-la em modelo ágil, conforme a idealização criada no subcapítulo abaixo.

8.3. Próximos passos

Dessa forma, o próximo passo seria o de reaplicar a metodologia desenvolvida aqui neste trabalho na *Nexoos* encontrando um novo fator causal que pudesse ser resolvido, desenvolver soluções, implementá-las e analisando seus resultados.

Por exemplo, foram identificados na etapa de análise de resultados pontos de melhoria para o CVDS implementado: a individualização do trabalho da área de tecnologia tanto como a sobrecarga de trabalho para produto. Esses novos problemas poderiam então ser estudados, trabalhados e discutidos com todos da equipe

novamente, através das ferramentas apresentadas nesse estudo de caso, no diagnóstico do problema. Logo, o objetivo seria de tornar essa atividade um ciclo contínuo, para que o CVDS da empresa continuamente seria refinado a cada passagem.

8.4. Aprendizados

O estudo de caso em questão capacitou o autor para desenvolver suas habilidades como gestor, atuando como um ponto neutro em três áreas que se encontravam em atrito: as áreas de tecnologia, de produto e de *stakeholders*; além de trazer a oportunidade de aprender a relevância existente nas metodologias de projetos aplicadas através de um caso real, e quais foram seus impactos. Assim, da mesma maneira que se produz um parafuso, uma nova funcionalidade de software passa por processos e consome recursos, tais que necessitam ser gerenciados através de uma análise profunda do CVDS continuamente; pois caso contrário, o processo se torna ultrapassado e pode se tornar ineficiente.

9. Referências bibliográficas

Khan, A. I., Qurashi, R. J., & Khan, U. A. (2011). A Comprehensive Study of Commonly Practiced Heavy and Light Weight Software Methodologies. *IJCSI International Journal of Computer Science Issues, Vol. 8, Issue 4, No 2*.

Ahmad, M. O., Markkula, J., & Ovio, M. (2013). Kanban in software development: A systematic literature review. *Euromicro Conference Series on Software Engineering and Advanced Applications*.

Khan, K., Siddiqui, A., Waheed, U., Hassan, F., & Duc, A. N. (2020). Towards continuous value delivery with review meetings in agile methodologies: A structured review. *Khan et al/International Journal of Advanced and Applied Sciences*.

Khan, M. A., Parveen, A., & Sadiq, M. (2014). A Method for the Selection of Software Development Life Cycle Models using Analytic Hierarchy Process. *IEEE*.

Patil, S. P., & Neve, J. R. (2018). Productivity Improvement of Software Development Process through Scrumban: A Practitioner's Approach. *International Conference On Advances in Communication and Computing Technology (ICACCT)*.

Qureshi, M. R., & Hussain, S. A. (2008). An Adaptive Software Development Process Model. *Advances in Engineering Software, Elsevier*, 654-658.

Duc, A. N., & Abrahamsson, P. (2016). Minimum Viable Product or Multiple Facet Product? The Role of MVP in Software Startups. *Department of Computer and Information Science (IDI)*, 118 - 130.

Srivastava, A., Bhardwaj, S., & Saraswat, S. (2017). SCRUM Model for Agile Methodology. *International Conference on Computing, Communication and Automation*, 864-869.

Khalfallah, M., & Lakhal, L. (2020). The impact of lean manufacturing practices on operational and financial performance: the mediating role of agile manufacturing. *International Journal of Quality & Reliability Management*.

Kootanaee, A. J., Babu, D. K., & Talari, H. F. (2013). Just-in-Time Manufacturing System: From Introduction to Implement. *International Journal of Economics, Business and Finance*, 7-25.

S.Balaji, & Murugaiyan, D. M. (2012). WATEERFALLVs V-MODEL Vs AGILE: A COMPARATIVE STUDY ON SDLC . *International Journal of Information Technology and Business Management*, 26-30.

Fitzgerald, B., Russo, N. L., & O'Kane, T. (2003). SOFTWARE DEVELOPMENT METHOD TAILORING AT MOTOROLA. *Comunications of the acm*, 64-70.

Martello, A. (28 de Maio de 2019). *Cinco maiores bancos comerciais detinham 84,8% do mercado de crédito no fim de 2018, revela BC* . Acesso em Julho de 2020, disponível em Globo - G1: https://g1.globo.com/economia/noticia/2019/05/28/cinco-maiores-bancos-comerciais-detem-848percent-do-mercado-de-credito-no-fim-de-2018-revela-bc.ghtml

Banco Central do Brasil. (17 de Junho de 2020). *Taxas de juros básicas – Histórico*. Acesso em Julho de 2020, disponível em Site oficial do Banco Central do Brasil: https://www.bcb.gov.br/controleinflacao/historicotaxasjuros

Newcomer, K. E., Hatry, H. P., & Wholey, J. S. (2004). *Handbook of practical program evaluation.* Nova Jersey, Estados Unidos: John Wiley & Sons.

Rooney, J. J., & Heuvel, L. N. (2004). Root Cause Analysis For Beginners. *Quality Progress*, 45-53.

Smith, D. (19 de Dezembro de 2017). *The Definition of Done: What does "done" actually mean?* . Fonte: Medium: https://medium.com/@dannysmith/the-definition-of-done-what-does-done-actually-mean-ef1e5520e153