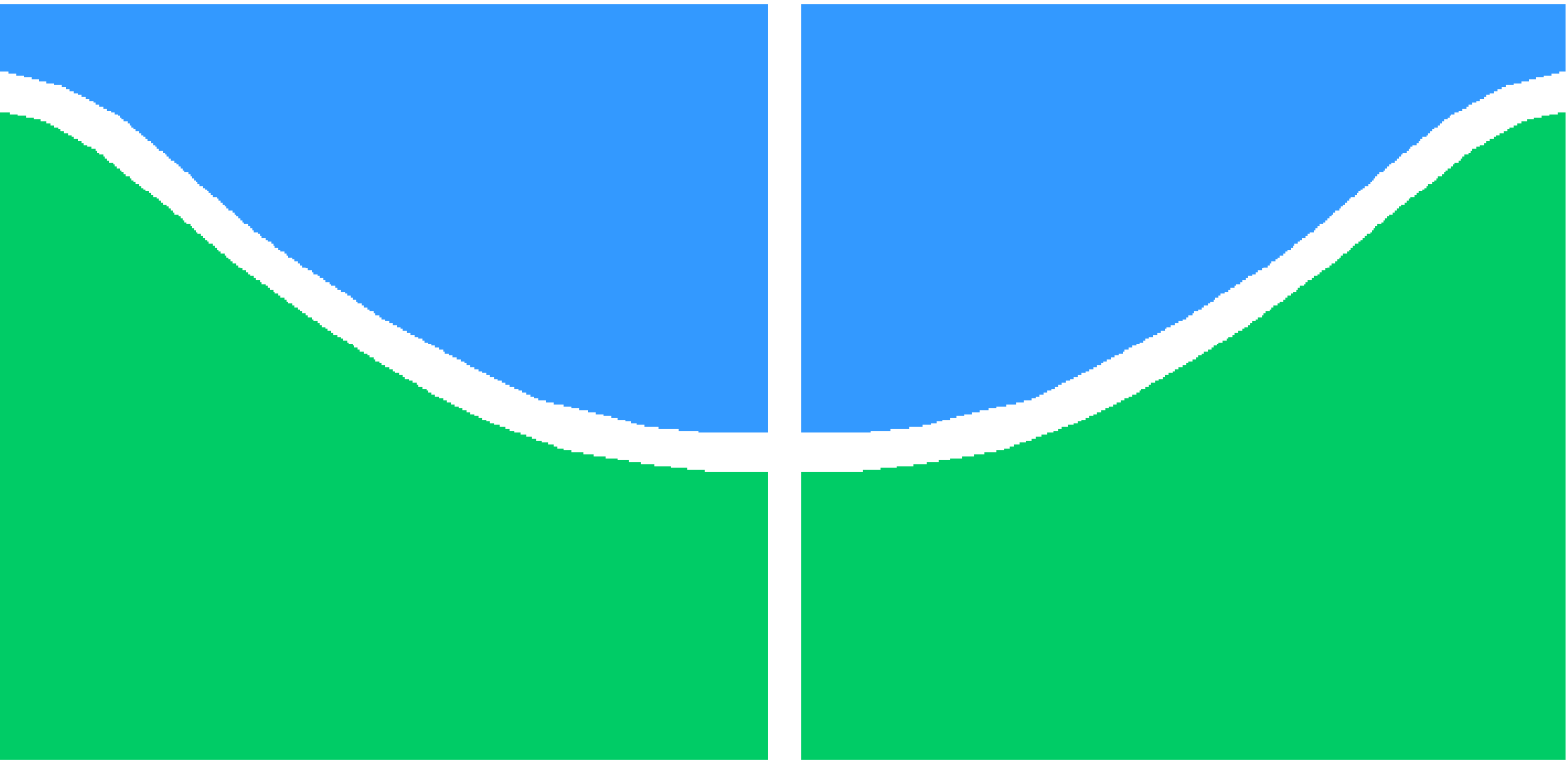

Universidade de Brasília – UnB
Faculdade UnB Gama – FGA
Engenharia de Software

Planejamento de melhoria de processo de software baseado em análise de issues

Autora: Sannya Santana de Arvelos
Orientador: MSc. Critiane Soares Ramos
Coorientador: MSc. Ricardo Ajax Dias Kosloski

Brasília, DF
2020

Sannya Santana de Arvelos

Planejamento de melhoria de processo de software baseado em análise de issues

Monografia submetida ao curso de graduação em Engenharia de Software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software.

Universidade de Brasília – UnB

Faculdade UnB Gama – FGA

Orientador: MSc. Critiane Soares Ramos

Coorientador: MSc. Ricardo Ajax Dias Kosloski

Brasília, DF

2020

Sannya Santana de Arvelos

Planejamento de melhoria de processo de software baseado em análise de issues/ Sannya Santana de Arvelos. – Brasília, DF, 2020-

90 p. : il. (algumas color.) ; 30 cm.

Orientador: MSc. Critiane Soares Ramos

Trabalho de Conclusão de Curso – Universidade de Brasília – UnB
Faculdade UnB Gama – FGA , 2020.

1. Melhoria de Processo. 2. Análise de issues. I. MSc. Critiane Soares Ramos. II. Universidade de Brasília. III. Faculdade UnB Gama. IV. Planejamento de melhoria de processo de software baseado em análise de issues

CDU 02:141:005.6

Sannya Santana de Arvelos

Planejamento de melhoria de processo de software baseado em análise de issues

Monografia submetida ao curso de graduação em Engenharia de Software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software.

Trabalho aprovado. Brasília, DF, 2020:

MSc. Critiane Soares Ramos
Orientador

MSc.Ricardo Ajax Dias Koloski
Convidado 1

Dra. Carla Silva Rocha Aguiar
Convidado 2

Brasília, DF
2020

Dedico este trabalho à toda minha família, à todos os jovens quilombolas e indígenas que nunca pararam de lutar e que acreditam fielmente em seus sonhos.
À minha sobrinha Nicolly que me faz acreditar todos os dias em anjos e à todos meus sobrinhos.

Agradecimentos

Agradeço a Deus pela vida. À Nossa Senhora Aparecida por quem eu sempre roguei nos momentos de aflição.

À minha mãe Augusta Cássia, ao meu pai Luiz Carlos por sempre acreditarem em mim e por não medir esforços para a realização dos meus sonhos e aos meus irmãos Galeno e Amizielle por ter aberto mão de minha companhia para que eu pudesse ter um futuro melhor, por todo o aprendizado que pude obter de todos vocês. Obrigada família por todo esforço que tiveram para que eu tivesse um ensino de qualidade. Gratidão por tudo.

À minha avó Nanci por todos os ensinamentos e por ser esse exemplo de fortaleza, minhas primas Tayná e Monã por ter me ensinado a ser corajosa desde pequena.

Ao tio Reis (in memorian) por ter dado início à minha jornada aqui em Brasília e por ser fonte de inspiração, à tia Eliana por ter sido minha mãe e suporte aqui em Brasília, palavras não pagarão nem metade de tudo o que a senhora fez e faz por mim, gratidão. Obrigada pelos quatro irmãos que herdei e que me acolheram de braços abertos, à Gustavo obrigada pela parceria e pela afilhada Simone por está sempre apostos à ajudar, Gabriel obrigada por alegrar a família, Graciele obrigada por tudo e por nunca medir esforços para me ajudar e Geovanna obrigada pela sua luz e carinho. Obrigada à seus pais Raimundo e Iracy por todo cuidado e carinho e a seus irmão pela acolhimento.

Ao meu cachorro Louis por ser amor sempre e por muitas vezes entender minhas angústias.

À toda minha família, vó Terezinha, avôs, tios em especial Edney, Valéria e Dionizio, primas e primos pela torcida e carinho.

À Telma (in memorian) por todo amor, apoio e entendimento não tenho palavras para dizer o quanto você foi e sempre será importante em minha jornada, à ti meu muito obrigada.

À minha tia avó Alda e toda sua família por ser tão amáveis, pela força e carinho.

À minha tia Jô e a toda sua família pelo acolhimento e carinho. Em especial ao João Victor por ser luz e revolução em minha vida, a Juliany por ser minha confidente leal, a Carol por ser meu amor e por todo carinho e a Dinha por ser tão amável.

Agradeço todos os amigos de infância, em especial Amanda, Adriele e Liliane por terem sempre me apoiado em todas as minhas decisões e por estarem comigo, mesmo eu estando longe. À minha amiga Cris pelo carinho.

Aos meus amigos da academia que cultivei ao longo desses anos Isaque, Daniel, Matteus, Mayana, Bruno, Fabíola, em especial Vanessa, Ana Paula por me alegrar durante minha jornada acadêmica e pelo apoio na vida e à Kairon por sempre me ajudar e acreditar no meu potencial. Gratidão amigos.

À minha amiga Soraya por ter me incentivado desde o início a continuar e que tudo é possível quando se acredita.

Agradeço a todas as mulheres antes de mim que lutaram para eu estivesse aqui onde estou. Agradeço ao governo que fez reformas sociais mais inclusivas e me deram a oportunidade de está cursando uma educação superior pública de qualidade.

Agradeço aos professores escolares que me incentivaram na medida do possível. À todos os meus professores acadêmicos que me ensinaram de formas distintas a grandiosidade da engenharia de software, em especial meus orientadores Cristiane e Ricardo Ajax, por me darem todo suporte e incentivo para a realização deste trabalho.

Agradeço a empresa em que trabalho, por acreditar e confiar em meu potencial, por todo crescimento intelectual e profissional adquirido e por ter feito parte deste Trabalho de Conclusão de Curso.

Resumo

É imprescindível garantir a qualidade do desenvolvimento do produto de software, sendo importante a utilização de metodologias e técnicas que permitam uma melhor organização dos artefatos, fornecendo maior objetividade ao processo na coleta e análise dos dados. O objetivo desse trabalho é apresentar uma abordagem para identificar as oportunidades de melhoria de processo de software a partir de *issues* registradas pelas equipes que trabalham com metodologias ágeis de desenvolvimento. Foi utilizado a abordagem IDEAL, usada para gestão de melhoria contínua de processos de software em conjunto com a ferramenta Jira Software, que é usada pela organização X para realizar o gerenciamento de tarefas dos seus projetos. No contexto da empresa, uma *issues* pode representar uma tarefa do projeto, uma funcionalidade, uma solicitação de ajuste de funcionalidade, falhas do sistema ou de uma funcionalidade. Os achados mostram que aproximadamente 58% dos registros são oriundos de *issues* com nível de severidade 1. Com isso, a partir do estudo causa raiz dos principais problemas identificados, foi proposto um conjunto de oportunidades de melhoria no processo de desenvolvimento de software da organização X, para que ela pudesse então priorizar o que era mais relevante a partir dos seus objetivos de negócio. Conclui-se que há indícios de que as *issues* registradas pelas equipes de desenvolvimento de software durante o processo de desenvolvimento ágil, quando analisadas de forma sistemática, contribuem para a melhoria dos processos de software da organização.

Palavras-chaves: Qualidade de software. Melhoria de processo de software. Desenvolvimento ágil de software. *Issues*.

Abstract

It is essential to guarantee the quality of the development of the software product, being important the use of methodologies and techniques that allow a better organization of the artifacts, providing greater objectivity to the process in the collection and analysis of the data. The objective of this work is to present an approach to identify the opportunities for software process improvement based on issues registered by the teams that work with agile development methodologies. Was used the IDEAL approach, used to manage continuous improvement of software processes in conjunction with the Jira Software tool, which is used by the X company to perform task management for the projects. In the context of the company, an issue can represent a project task, a feature, a feature adjustment request, system or feature failures. The findings show that approximately 58 % of records are from problems with severity level 1. With this, from the root cause study of the main problems identified, it was proposed a set of opportunities for improvement in the software development process of the X company, so that it could then prioritize what was most relevant from its business objectives. Concludes that there are indications that the issues registered by the software development teams during the agile development process, when analyzed in a systematic way, contribute to the improvement of the organization's software processes.

Key-words: *Issues*; Agile Methodologies; Software Quality; Defect Prevention; Root causes.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

PPQA	Process and Product Quality Assurance.
GQM	Goal, Question, Metric.
PDCA	Plan, Do, Check e Action.
SEI	Software Engineering Institute.
IDEAL	Initiating, Diagnosing, Establishing, Acting, Learning.
API	Application Programming Interface.
SCAMPI	Standard CMMI Appraisal Method for Process Improvement.
CMMI	Capability Maturity Model Integration.
MPS-BR	Melhoria de Processos do Software Brasileiro.
GPR	Gerência de Projetos.
REQ	Engenharia de Requisitos.
VER	Verificação.
VAL	Validação.
MED	Medição.
GRH	Gerência de Recursos Humanos.
ORG	Gerência Organizacional.

Sumário

1 Introdução

1.1 Contexto

As mudanças que estão ocorrendo nos ambientes de negócios, têm motivado as empresas a modificar estruturas organizacionais e processos produtivos, saindo da visão tradicional, baseada em áreas funcionais, em direção a redes de processos centrados no cliente. A competitividade depende, cada vez mais, do estabelecimento de conexões nestas redes, criando elos essenciais nas cadeias produtivas. Alcançar competitividade pela qualidade, para as empresas de software, implica tanto na melhoria da qualidade dos produtos de software e serviços correlatos, como dos processos de produção e distribuição de software. (MPS.BR, 2016)

O software vem sendo usado, em uma variedade cada vez maior de áreas de aplicação, e sua operação correta é frequentemente crítica para o sucesso dos negócios. Desenvolver ou selecionar produtos de software de alta qualidade é, portanto, de primordial importância. A especificação e avaliação abrangentes da qualidade do produto de software é um fator essencial para garantir a qualidade adequada. Isso pode ser alcançado através da definição de características de qualidade apropriadas, levando em consideração o objetivo do uso do produto de software. É importante que todas as características relevantes da qualidade do produto de software sejam especificadas e avaliadas, sempre que possível, usando métricas validadas ou amplamente aceitas. (BEHKAMAL; KAHANI; AKBARI, 2009)

Segundo CAMPOS (2004), a qualidade leva a uma melhor produtividade nas empresas, gerando melhores controles de seus recursos e, consequentemente, a uma redução de custos que influenciarão no desempenho organizacional. Pesquisas anteriores indicaram que modelos e padrões de melhoria de processo podem ajudar uma organização no desenvolvimento de processos de alta qualidade, reduzindo custos e tempo de desenvolvimento e aumentando a satisfação do usuário.

Existem essencialmente duas abordagens, que podem ser seguidas para garantir a qualidade do produto, uma sendo a garantia do processo pelo qual o produto é desenvolvido e a outra, a avaliação da qualidade do produto final. (BEHKAMAL; KAHANI; AKBARI, 2009)

A satisfação com o produto está relacionada com seu desempenho e com a ausência de defeitos, erros ou falhas. Portanto, a satisfação com o produto é alcançada quando as necessidades do cliente são supridas, e o produto se comporta como é esperado. Os requisitos representam as necessidades explícitas dos clientes e devem procurar cobrir

a maior parte das necessidades por eles declaradas em relação ao produto. (GUERRA; COLOMBO, 2009)

1.2 Problema

O desenvolvimento de software ágil abrange uma nova abordagem para a tomada de decisões em projetos de software. O desenvolvimento ágil exige que a equipe adote um processo de tomada de decisão colaborativo e mais rápido. Fazendo com que o autogerenciamento de equipes seja vista com bons olhos, quando se trata de eficácia no processo e desenvolvimento de software de sucesso. Tal como, suas técnicas e ferramentas que auxiliam nos projetos de software.

A ferramenta adotada na organização X é o Jira Software, este sistema foi desenvolvido especificamente para equipes de software. O Jira Software combina técnicas de desenvolvimento com recursos ágeis para ajudar as equipes na criação de software. Um uso básico desta ferramenta é rastrear problemas e bugs relacionados ao sistema de software.

No Jira Software, a organização X usa issues para o rastreio de trabalhos/tarefas que devem ser concluídos aos final da sprint pela equipe. Dependendo de como a equipe usa o Jira Software, uma issue pode representar uma tarefa do projeto, um funcionalidade, um solicitação de ajuste de funcionalidade, falhas do sistema ou de uma funcionalidade. No Jira Software, as issues normalmente representam itens como grandes recursos, requisitos do usuário e bugs de software.

Visto que a adoção de programa de melhoria de processo de software deve estar alinhada à realidade da empresa, a questão de pesquisa deste trabalho é: *"Como identificar oportunidades de melhoria de processo de software a partir da análise das Issues cadastradas no sistema de administração de issues da empresa?"*

1.3 Objetivos geral e específicos

O objetivo geral deste trabalho é propor melhorias de processo software a partir da análise das issues cadastradas no sistema de administração de issues da empresa.

A fim de atender a este objetivo geral foram estabelecidos os seguintes objetivos específicos:

- OBJ1 - Identificar e mapear o processo atual de desenvolvimento de software da empresa.
- OBJ2 - Identificar e mapear os problemas que têm ocorrido na execução do processo atual de desenvolvimento de software da empresa.

- OBJ3 - Analisar os problemas registrados como issues.
- OBJ4 - Identificar necessidades de melhorias dos processos de software.
- OBJ5 - Estabelecer uma estratégia para planejamento de programa de melhoria de processo de software.

1.4 Organização do Trabalho

Esta monografia está organizada em 5 capítulos. Neste capítulo de introdução situam-se: o contexto do trabalho, problema, objetivos da pesquisa e a metodologia de pesquisa selecionada.

No capítulo 2, o referencial teórico é descrito. Tal capítulo aborda conceitos imprescindíveis sobre qualidade de software, assim como, processos ágeis e métrica de software.

No capítulo 3, materiais e métodos, apresentam-se os métodos que são os passos e materiais para definição do plano de melhoria para a organização selecionada.

No capítulo 4, será apresentada a estratégia para definição do planejamento do programa de melhoria de processo para a organização X.

Finalmente, as considerações finais são apresentadas no capítulo 5.

2 Revisão de Literatura

2.1 Considerações Iniciais do Capítulo

O desenvolvimento de software englobando aspectos técnicos e não-técnicos, de modo a produzir software de qualidade, de forma eficaz e dentro de custos aceitáveis. Dessa forma, aumenta-se a probabilidade de produzir software de grande porte com qualidade, ou seja, software que satisfaça os requisitos do usuário, bem como as expectativas de tempo e de orçamento.

A qualidade está sempre associada aos conceitos de completeza e correção. A qualidade dos resultados dos sistemas de informação pressupõe a qualidade dos ambientes computacionais e organizacional.

O Capítulo 2 apresenta de forma conceitual ferramentas de qualidade de software, de produto e de processo, juntamente com normas internacionais capazes de auxiliar na qualidade do desenvolvimento de software. É apresentado conceitos de prevenção de defeitos, verificação e validação, processo ágil no desenvolvimento de software, gestão de melhoria de processo e métricas de software.

2.2 Qualidade em Software

Segundo BAZZANA, ANDERSEN e JOKELA (1993), a qualidade de software é definida como o grau em que um cliente ou usuário compreende que o software atende às suas expectativas e necessidades.

Pode-se definir qualidade de produto de software como a conformidade a requisitos funcionais e de desempenho declarados explicitamente, padrões de desenvolvimento claramente documentados e as características implícitas que são esperadas de todo software desenvolvido profissionalmente. As definições de qualidade podem variar em alguns aspectos, porém o aspecto que se refere à satisfação do cliente ou usuário não deve ser esquecido. (GUERRA; COLOMBO, 2009)

Segundo GUERRA e COLOMBO (2009), a norma NBR ISO/IEC 9126-1, futura ISO/IEC 25010 – Quality Model, faz parte desse conjunto de normas ISO. Essa norma propõe características que um software deve possuir, bem como sub-características, para incentivar o uso na prática dessa padronização de qualidade de produto de software. A norma NBR ISO/IEC 9126-1 está preocupada em garantir que as necessidades dos clientes, em relação ao produto de software, sejam providas. Esta norma define quais são as características que um produto de software deve ter e fornece um modelo para ser

utilizado em uma avaliação de verificação da presença de tais características. Isso significa que, a partir de uma entidade de software, disponível a um usuário, é possível utilizar a norma para verificar a qualidade dessa entidade. (GUERRA; COLOMBO, 2009)

A norma NBR ISO/IEC 9126-1 pode ser aplicada nos seguintes momentos: (GUERRA; COLOMBO, 2009)

- Definição dos requisitos de qualidade de um produto de software.
- Avaliação da especificação de software para verificar se ele irá satisfazer aos requisitos de qualidade durante o desenvolvimento.
- Descrição de particularidades e atributos do software implementado, por exemplo, em manuais de usuário.
- Avaliação do software desenvolvido, antes da entrega.
- Avaliação do software desenvolvido, antes da aceitação.

Segundo (GUERRA; COLOMBO, 2009), a engenharia de software tem uma sessão para a qualidade de software, que tem como finalidade garantir que especificações explícitas e necessidades implícitas estejam presentes no produto, por meio da definição e normatização de processos de desenvolvimento.

Resumidamente, "qualidade" é conformidade com requisitos, e estes devem estar definidos para permitir que sejam gerenciados com o uso de medidas, de forma a reduzir o retrabalho e aumentar a produtividade. A melhoria da qualidade deve estar focada nos processos, e não nas pessoas; certamente, é responsabilidade de todos os envolvidos no processo. (GUERRA; COLOMBO, 2009)

Alguns fatores de qualidade passaram a ser exigidos para o produto de software. Entre esses fatores, podem-se citar os externos como usabilidade, portabilidade, manutenibilidade etc. e fatores de qualidade internos como reusabilidade, modularidade, flexibilidade etc. Uma constante melhoria no processo que desenvolve o produto também passou a ter relevância, pois um processo de software de alta qualidade deve, por consequência, gerar um produto de alta qualidade. Pode-se dizer que qualidade é uma das palavras chaves na Engenharia de Software e é medida atualmente de duas formas: (1) qualidade do produto e (2) qualidade do processo.

2.2.1 Qualidade de Produto de Software - Norma SQuaRE

A ISO / IEC 25022, que substitui a ISO / IEC 9126-4, faz parte da série de padrões SQuaRE e foi preparada pelo Comitê Técnico Conjunto ISO / IEC JTC 1, tecnologia da informação, Subcomitê SC 7, Engenharia de Software e Sistemas. (ISO/IEC, 2012)

Uma maneira formal de documentar uma medida é utilizando um módulo de Avaliação. Além disso, ela apresenta uma estrutura para a avaliação da qualidade de produto de software. Essas estruturas são provenientes das normas ISO/IEC 9126-1 e 14598-1, 14598-3, 14598-4, 14598-5 e 14598-6. (GUERRA; COLOMBO, 2009)

As normas anteriores que tratam de qualidade de produto continuam sendo utilizadas até que as da nova série SQuaRE sejam publicadas. As principais diferenças entre as séries ISO/IEC 9126, ISO/IEC 14598 e SQuaRE são: (GUERRA; COLOMBO, 2009)

- Introdução do novo modelo de referência geral;
- Introdução de guias detalhados e direcionados para cada divisão da norma; introdução de elementos de medidas de qualidade dentro da divisão Medição da Qualidade;
- Introdução da divisão Requisitos de Qualidade;
- Incorporação e revisão dos processos de avaliação;
- Introdução de orientações para uso prático em forma de exemplos;
- coordenação e harmonização do conteúdo com a norma NBR ISO/IEC 15939 – Engenharia de Sistemas e de Software – Processo de Medição, publicada em janeiro de 2009.

De acordo com a ISO/IEC (2012), como mostra na Figura 1 a norma SQuare está organizada da seguinte forma:

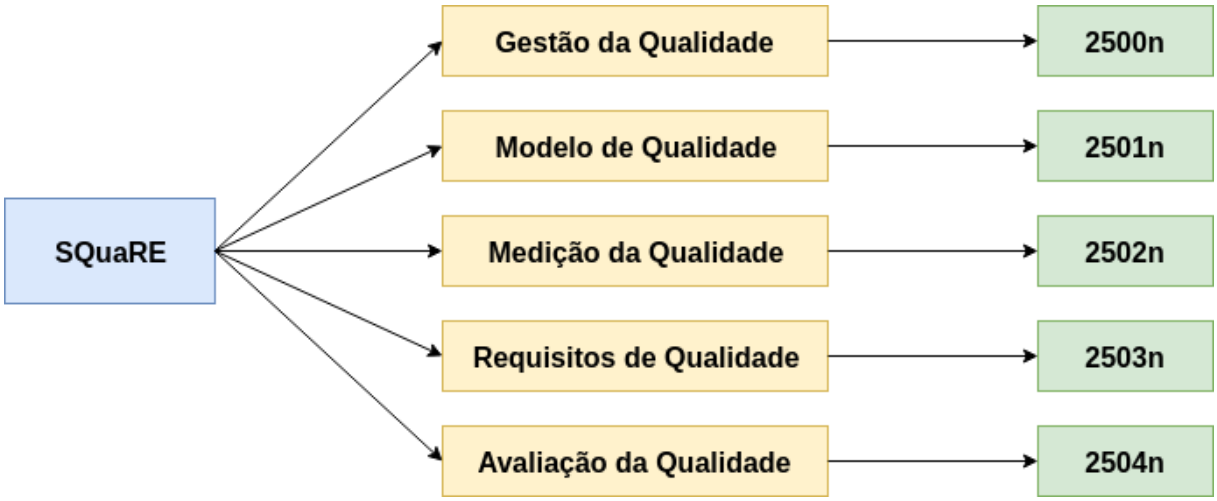

Figura 1 – Divisão da Norma SQuaRE. Fonte: Autora

- **ISO / IEC 2500n - Divisão de Gestão da Qualidade** - Os padrões que formam essa divisão indicam todos os modelos, termos e definições que se estende por outros

padrões da série SQuaRE. A divisão também fornece requisitos e orientações para a função de suporte responsável pelo gerenciamento dos requisitos, especificação e avaliação da qualidade do produto de software.

- **ISO / IEC 2501n - Divisão do Modelo de Qualidade** - Os padrões que formam essa divisão apresentam modelos detalhados de qualidade para sistemas de computadores e produtos de software, qualidade em uso e dados. Também são fornecidas orientações práticas sobre o uso dos modelos de qualidade.

- **ISO / IEC 2502n - Divisão de Medição da Qualidade** - Os padrões que formam essa divisão incluem um modelo de referência, de medição, de qualidade, de produtos, de sistema / software, definições matemáticas de medidas, de qualidade e orientações práticas para sua aplicação. São apresentados exemplos de medidas internas e externas para a qualidade do software e medidas para a qualidade em uso.

- **ISO / IEC 2503n - Divisão de Requisitos de Qualidade** - Os padrões que formam esta divisão ajudam a especificar requisitos de qualidade, com base em modelos e medidas de qualidade. Esses requisitos de qualidade podem ser usados no processo de obtenção de requisitos de qualidade para um produto de software a ser desenvolvido ou como entrada para um processo de avaliação.

- **ISO / IEC 2504n - Divisão de Avaliação da Qualidade** - Os padrões que formam esta divisão fornecem requisitos, recomendações e diretrizes para avaliação de produtos de software, sejam executados por avaliadores, adquirentes ou desenvolvedores. O suporte para documentar uma medida como um módulo de avaliação também é descrito.

A divisão de extensão ISO / IEC 25050 - 25099 SQuaRE atualmente, esses padrões incluem os requisitos de qualidade do software comercial pronto para uso e os formatos comuns da indústria para relatórios de usabilidade.

2.3 Qualidade de Uso

A qualidade em uso pode ser medida por meio da operação do produto final em condições de uso normal ou simuladas, com o objetivo de verificar a existência e o nível das características de qualidade de um produto de software, conforme definidas na norma NBR ISO/IEC 9126-1. (GUERRA; COLOMBO, 2009)

A ISO 9126 faz parte da norma ISO 9000, que é a norma mais importante para garantia de qualidade. Nesse modelo, a totalidade dos atributos de qualidade do produto

de software é classificada em uma estrutura hierárquica em árvore de características e sub-características. O nível mais alto dessa estrutura consiste nas características de qualidade e o nível mais baixo, consiste nos critérios de qualidade do software. O modelo especifica seis características, incluindo Funcionalidade, Confiabilidade, Usabilidade, Eficiência, Manutenibilidade e Portabilidade; que são divididos em 21 sub-características, como mostra na Figura 2. Essas sub características são manifestadas externamente quando o software é usado como parte de um sistema de computador e são o resultado de atributos internos do software. (BEHKAMAL; KAHANI; AKBARI, 2009)

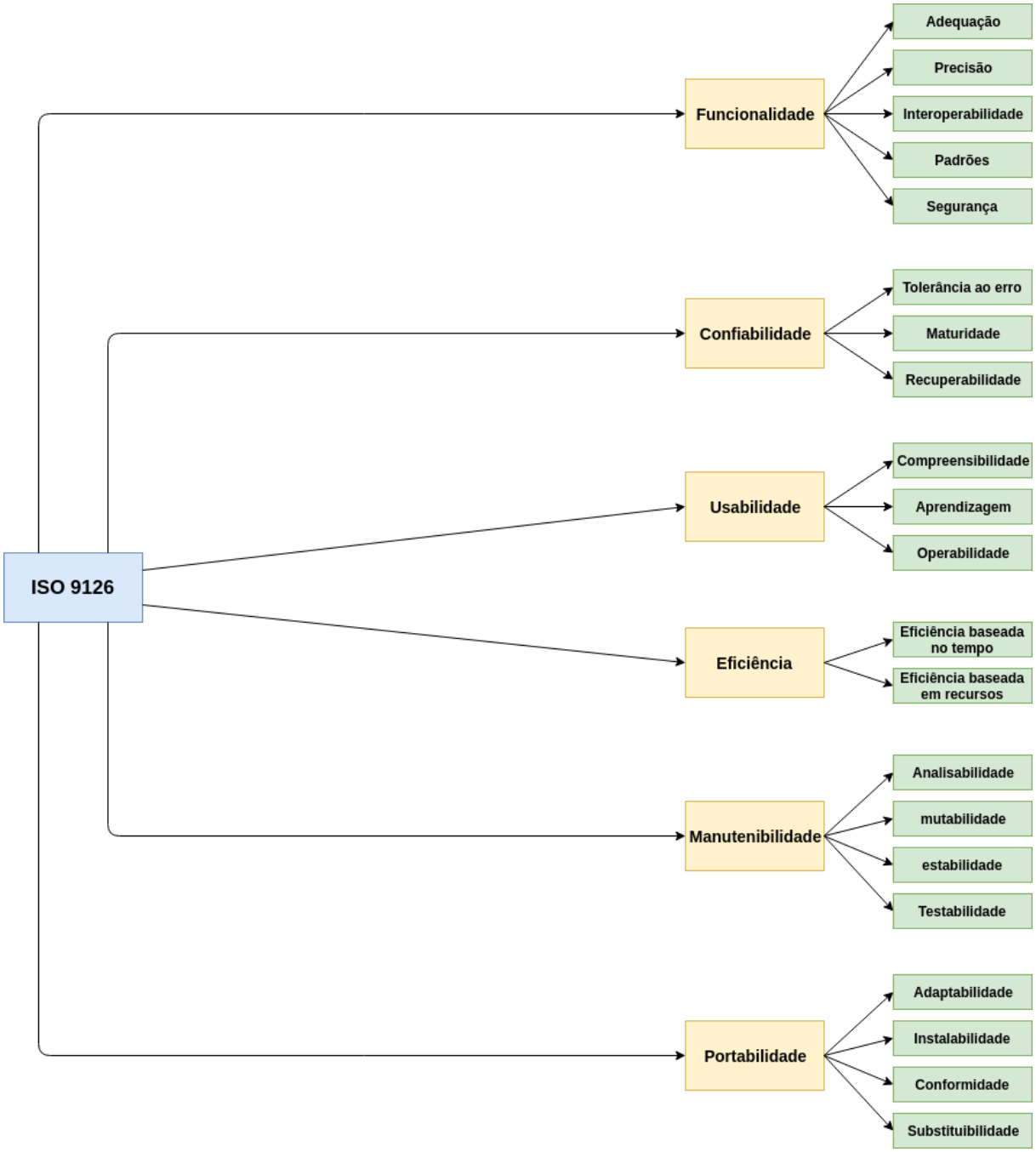

Figura 2 – Divisão da ISO 9126. Fonte: Autora

Qualidade em uso – "satisfazer as necessidades reais de usuários ao utilizar o produto de software, para atingir metas especificadas em contextos de uso especificados", ou seja, o efeito da utilização do produto, medido com relação às necessidades dos usuários. (GUERRA; COLOMBO, 2009)

A qualidade em uso tem como finalidade suprir as necessidades do usuário. Tal qualidade têm quatro características fundamentais: Eficácia, Produtividade, Segurança e Satisfação. Segundo a norma NBR ISO/IEC 9126-1 (2001), estas características são descritas como:

- Eficácia: O software deve permitir que os usuários especificados atinjam metas especificadas com acurácia e completitude, no contexto de uso especificado.
- Produtividade: O software deve permitir que seus usuários diretos e indiretos empreguem quantidade apropriada de recursos em relação à eficácia obtida, no contexto de uso especificado.
- Segurança: O software deve apresentar níveis aceitáveis de riscos de danos a pessoas, negócios, software, propriedades ou ao ambiente, no contexto de uso especificado;
- Satisfação: O software deve satisfazer usuários, no contexto de uso especificado.

Segundo (GUERRA; COLOMBO, 2009), a abordagem da qualidade em uso mede quanto um produto atende às necessidades de usuários especificados, para que estes atinjam metas pré estabelecidas com eficácia, produtividade, segurança e satisfação, em um contexto de uso especificado. A avaliação de qualidade em uso valida a qualidade do produto de software em cenários de uso específicos. Mostram a visão do usuário sobre qualidade, em um ambiente contendo software, e são medidas por meio dos resultados do uso do software em ambientes, e não pelas propriedades do software. (GUERRA; COLOMBO, 2009)

2.3.1 Qualidade do Processo de Software

Segundo a norma NBR ISO 9000 define qualidade como: "A totalidade das características de uma entidade que lhe confere a capacidade de satisfazer às necessidades explícitas e implícitas". O propósito da garantia da qualidade de processo e produto (PPQA) é munir a equipe e a gerência com uma visão clara sobre os processos e seus produtos de trabalho associados. (CMMI-DEV, 2006)

Segundo CMMI-DEV (2006), a área de processo de Garantia da Qualidade de processo e Produto está dividida em:

- Avaliar objetivamente os processos, produtos de trabalho e serviços executados em relação às descrições de processo, padrões e procedimentos aplicáveis;
- Identificar e documentar as não-conformidades;
- Fornecer feedback para a equipe do projeto e gerentes sobre os resultados das atividades de garantia da qualidade;
- Garantir que as não-conformidades sejam tratadas;

A área de processo Garantia da Qualidade de processo e Produto dá suporte à entrega de produtos e serviços com alta qualidade, fornecendo à equipe do projeto e aos gerentes de todos os níveis, a visibilidade apropriada e o feedback sobre os processos e produtos de trabalho associados ao longo do ciclo de vida do projeto. (CMMI-DEV, 2006)

As práticas da área de processo Garantia da Qualidade de processo e Produto garantem que processos planejados sejam implementados, ao passo que as práticas da área de processo Verificação garantem que os requisitos especificados sejam satisfeitos. Estas duas áreas de processos podem, de vez em quando, tratar o mesmo produto de trabalho, mas com perspectivas diferentes. Convém que os projetos tirem vantagem dessa sobreposição a fim de minimizar a duplicação de esforços e, ao mesmo tempo, tomem o cuidado de manter separadas essas perspectivas. (CMMI-DEV, 2006)

2.3.2 Prevenção de Defeitos (Pareto e Ishikawa)

A prevenção de defeitos de software é uma estratégia importante para melhorar a qualidade do software e reduzir os custos de desenvolvimento, impedindo a ocorrência de defeitos.

A prevenção de defeitos pode ser empregada nos estágios iniciais do desenvolvimento de software para reduzir as taxas de introdução de defeitos e, assim, reduzir o esforço de detecção e correção de defeitos. (HUANG; LIU, 2017)

A prevenção de defeitos ressalta a importância da melhoria do processo de software com base no aprendizado de falhas existentes. O processo se inicia quando a equipe especializada escolhe amostras de defeitos de um banco de dados, identifica as causas desses defeitos e, em seguida, produz estratégias de melhoria de processos para o projeto. Segundo HUANG e LIU (2017), este paradigma funciona efetivamente na prevenção de defeitos causados por falhas no processo, por exemplo rastreamento de requisitos insuficiente.

O processo típico da prevenção de defeitos inclui as seguintes etapas:(HUANG; LIU, 2017)

- (1) selecionar amostras de defeitos do banco de dados histórico;
- (2) realizar uma reunião de análise causal para identificar causas de defeitos e desenvolver propostas de ações de prevenção;
- (3) implementar ações preventivas;

Portanto, para implementar o processo de prevenção de defeitos, são necessários três elementos: (HUANG; LIU, 2017)

- (1) um banco de dados e uma ferramenta para seleção de amostras de defeitos e rastreamento de ações;
- (2) um método de classificação de causas para identificar causas-raiz;
- (3) uma equipe de ação com boa experiência em análise de causa raiz;

As causas-raiz são geralmente classificadas em quatro categorias: método, pessoas, ferramenta e requisito; causas detalhadas são analisadas através de brainstorming com diagramas de causa-efeito. (HUANG; LIU, 2017)

Duas técnicas que se mostraram particularmente úteis para apoiar atividades de análise causal de defeitos, são os gráficos de Pareto e os diagramas de causa e efeito (ou diagramas de Ishikawa). Elas apoiam, respectivamente, a identificação das classes de defeitos mais comuns e encontrar as causas para classes específicas de defeitos.

Segundo MARTINS e BASTINI (2012b), o Diagrama de Pareto consiste em um gráfico de barras que dispõe dados de frequências dos sucedidos erros, permitindo assim a priorização dos problemas. Mostra ainda a curva de porcentagens acumuladas. Permitindo uma fácil visualização e identificação das causas ou problemas mais importantes.

Para o Diagrama de Pareto ser aplicado, é importante seguir seis passos básicos: (MARTINS; BASTINI, 2012b)

- Determinar o objetivo do diagrama, ou seja, que tipo de perda será investigada;
- Definir o aspecto do tipo de perda, ou seja, como os dados serão classificados;
- Em uma tabela, ou folha de verificação, organizar os dados com as categorias do aspecto definido;
- Fazer os cálculos de frequência e agrupar as categorias que ocorrem com baixa frequência sob a denominação “outros”;
- Calcular também o total e a porcentagem de cada item sobre o total e o acumulado;

- Traçar o diagrama.

Segundo MARTINS e BASTINI (2012a), o Diagrama de Ishikawa é uma ferramenta que utiliza técnicas para realizar a análise das causas-raízes em avaliações de não conformidades de todos os fatores que envolvem a execução do processo.

Para realizar a análise de causas utilizando o Diagrama de Ishikawa, basta seguir alguns passos: (MARTINS; BASTINI, 2012a)

- Defina o problema (efeito) a ser analisado;
- Desenhe uma seta horizontal apontando para a direita e escreva o problema no interior de um retângulo localizado na ponta da seta;
- Realize um brainstorming para levantar as possíveis causas que possam estar gerando o problema. Para isso, procure responder a seguinte pergunta: "Por que isto está acontecendo?";
- Divida as causas identificadas em categorias, por exemplo: máquina, mão de obra, método e materiais ou da forma que for mais coerente com o problema analisado e o contexto da sua empresa;
- Logo após, defina as sub-causas, ou seja, os fatores que levaram aquela causa a acontecer.

2.4 Verificação e Validação

Segundo CMMI-DEV (2006), a Verificação tem por finalidade assegurar que os produtos de trabalho selecionados atendem aos seus requisitos especificados. Bem como, o propósito da Validação que é comprovar que um produto ou componente de produto se comporte como esperado quando colocado em seu ambiente de produção.

Na verificação uma preparação inicial é necessária para assegurar que os meios para tal, sejam incorporados aos requisitos do produto e de seus componentes, bem como o projeto, o plano de desenvolvimento e o cronograma. A Verificação inclui a seleção, inspeção, teste, análise e demonstração dos produtos de trabalho. Em contra partida, a validação demonstra que o produto fornecido cumprirá seu uso pretendido, enquanto que, a verificação examina se o produto de trabalho reflete adequadamente os requisitos especificados. Em outras palavras, a verificação garante que "você constrói certo o produto"; por outro lado, a validação garante que "você constrói o produto certo". (CMMI-DEV, 2006)

A Verificação são inerentemente um processo incremental porque ocorre ao longo do desenvolvimento do produto e dos produtos de trabalho, começando com a verificação

dos requisitos, passando pela verificação dos produtos de trabalho que vão sendo desenvolvidos e culminando com a verificação do produto acabado. A verificação dos produtos de trabalho aumenta consideravelmente a probabilidade de que os requisitos do cliente, do produto e dos componentes de produtos serão atendidos. São exemplos de métodos de verificação: (CMMI-DEV, 2006)

- Teste de cobertura de caminhos.
- Teste de carga, stress e desempenho.
- Teste baseado em tabela de decisão.
- Teste baseado em decomposição funcional.
- Reuso de caso de teste.
- Testes de aceitação.

Os requisitos e restrições para executar a validação são coletados. Então, os métodos de validação são selecionados com base em suas habilidades de demonstrar que as necessidades do usuário final são satisfeitas. Os métodos de validação não apenas definem a abordagem técnica para a validação do produto, mas também orientam as necessidades de facilidades, equipamentos e ambiente. Requisitos derivados, tais como: requisitos de interface para conjuntos de teste e equipamento de teste, podem ser gerados. Esses requisitos também são passados para o processo de Desenvolvimento de Requisitos para garantir que o produto ou componentes de produto possam ser validados em um ambiente que dá suporte ao método. (CMMI-DEV, 2006)

Os métodos de validação se aplicam ao desenvolvimento, manutenção, suporte e treinamento relacionados ao produto ou componente de produto quando apropriado. São exemplos de métodos de validação: (CMMI-DEV, 2006)

- Discussões com os usuários, talvez no contexto de uma revisão formal.
- Demonstrações de protótipos.
- Demonstrações funcionais (exemplos: demonstração de sistema, de unidades de hardware, de software, de documentação de serviços e de interfaces de usuário).
- Pilotos de materiais de treinamento.
- Testes de produtos e de componentes de produto por usuários finais e por outros stakeholders relevantes.

- Análises de produtos e de componentes de produto (exemplos: simulações, modelagem e análises de usuários).

Segundo SADIKIN (2012), a verificação e validação de software não é apenas para encontrar bugs, mas também pode ajudar a evitar problemas durante o uso ou operação.

Segundo LAWANNA (2013), a manutenção de software como um método de validação que se aplica ao desenvolvimento do produto, é o processo especial no ciclo de vida de desenvolvimento de software. Particularmente, os programadores tentaram reduzir o tamanho dos testes e manter novos softwares enquanto a correção de bugs também é realizada. As grandes quantidades de testes podem levar tempo, especialmente execução e operação. Levando isso em consideração, muitos especialistas propõem as técnicas para seleção de casos de teste, como seleção aleatória e seleção segura, com base no conceito de teste de regressão. No entanto, a capacidade do novo software ainda precisa ser aprimorada. Portanto, o caminho de controle baseado em casos de teste é preferido para aumentar o desempenho do programa criando e selecionando o mínimo caso de teste, assim como a taxa sem falhas é preservada.

Segundo D'OLIVEIRA (2003), os casos de testes requer um responsável pelos testes e que avalie se o produto de software cumpre os requisitos estabelecidos.

Segundo DiMAGGIO (1998), estas questões dependem dos seguintes fatores: (DiMAGGIO (1998) citado em D'OLIVEIRA (2003))

- **a natureza dos erros** - todos os erros menores devem ser revisados pela equipe de projeto para assegurar que o impacto causado seja "menor" para o usuário. Além disso, a freqüência em que o erro se manifesta deve ser pequena;
- **risco de consertá-los** - o impacto negligenciado de um erro pode requerer mudanças complexas nas codificações o que poderá inserir novos erros no código. Nestas situações é melhor documentar o erro e contornar o problema sem consertá-lo;
- **natureza dos erros (contornados)** - um erro pequeno, com um simples contorno no sentido de impedir que o erro se apresente, pode causar dificuldades ao usuário (por exemplo, fazendo que ele tenha que reiniciar a máquina a cada hora) e deve ser consertado; e,
- **acordos contratuais** - nos casos de software desenvolvidos por encomenda, geralmente o número de erros pode estar definido em contrato pelo cliente.

2.5 Processo Ágil em Desenvolvimento de Software

Uma das principais inovações em metodologia de desenvolvimento de software nos últimos anos tem sido a introdução de princípios ágeis. Segundo BERGIN et al. (2004) (citado em VLAANDEREN et al. (2011)) afirma que desde a criação do Manifesto Ágil em 2001, incluindo os anos que levaram à sua criação, vários métodos de desenvolvimento de software ágil entraram em prática. São exemplos de tais métodos o SCRUM, Extreme Programming (XP) e Feature Driven Development (FDD). Na Figura 3 é mostrado algumas técnicas dos três métodos citados acima.

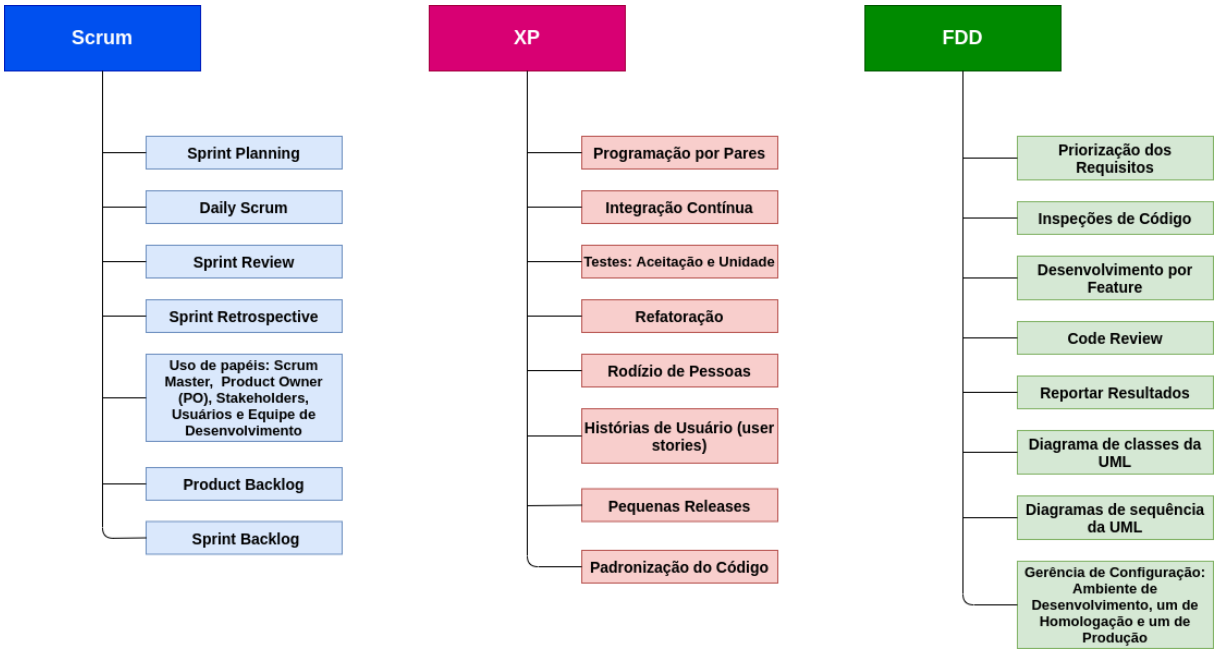

Figura 3 – Práticas das metodologias ágeis. Fonte: Autora

O método de desenvolvimento SCRUM foi proposto em 1995 por Ken Schwaber, no instante em que estava claro para a maioria dos profissionais que o desenvolvimento de software não era algo que poderia ser planejado, estimado e concluído com êxito utilizando os métodos comuns considerados massantes. O método SCRUM adere aos princípios de desenvolvimento de software ágil. Scrum é baseado na teoria do controle empírico do processo ou empirismo. O empirismo assegura que o conhecimento vem da experiência e toma decisões com base no que é conhecido. Esta metodologia usa uma abordagem iterativa e incremental para otimizar a previsibilidade e o controle de risco, oferecendo o próprio projeto ((LOPEZ, 2016)). Scrum framework dissemina princípios que garantem um processo de desenvolvimento de software dinâmico e adaptável. (SOUZA R. T.; ZORZO, 2015)

Em Scrum, a equipe precisa trabalhar em conjunto e cooperar para alcançar um objetivo SOUZA R. T.; ZORZO (2015). Scrum como visto na Figura 3, se concentra na divisão de trabalho completo (Backlog de produtos) em diferentes seções ou blocos que

podem ser abordados em curtos períodos de tempo (1-4 semanas), que são chamados de Sprint ((LOPEZ, 2016)). Durante o período de desempenho das histórias, chamado Sprint Planning, as equipes realizam a reunião diária para um acompanhamento diário de tarefas e dificuldades, denominada Daily. No final da Sprint, o Sprint Review é realizado, considerando a entrega do que foi realizado durante o período e depois disso, a Retrospectiva Sprint visa identificar pontos positivos e negativos e o que pode ser melhorado. ((RISING; JANOFF, 2000), (LEITE; LUCRÉDIO, 2014), (SCHWABER; SUTTHERLAND, 2015) citado em SOUZA R. T.; ZORZO (2015))

A Extreme Programming (XP) é um método amplamente utilizado para desenvolvimento de software. Segundo Putra, Yuliawati e Mursanto (2012), este método é usado para melhorar a qualidade do software. O XP como visto na Figura 3, é uma metodologia que se baseia em uma série de regras e princípios que foram utilizados ao longo da história do desenvolvimento de software, aplicando cada um deles de uma maneira que cria um processo ágil ((LOPEZ, 2016)). No XP, a coleta de requisitos é feita a partir a da escrita das histórias de usuário (user stories) pelo cliente. As user stories são descrições textuais sucintas a respeito das funcionalidades do sistema.(BECK, 2000)

O FDD como visto na Figura 3, sugere que todos os conjuntos de características devem passar pelas etapas de projeto e construção até que o sistema esteja pronto para ser entregue (HIGHSMITH, 2002). No FDD, os requisitos já conhecidos são listados, definidos e documentados através de casos de uso ou users stories. O FDD sugere que seja construído um diagrama de classes da UML para representar a arquitetura do sistema. Para complementar o diagrama de classes também poderão ser gerados diagramas de sequência da UML. (HIGHSMITH, 2002)

Dito isso, uma issue é muito similar a uma história de usuário porém são descritas de forma distintas. Uma issue pode ser descrição de uma funcionalidade, tarefa, melhoria da funcionalidade, falhas do produto, requisitos mal especificados, etc. Uma informação contida na issue pode interferir diretamente no desenvolvimento do produto.

Um issue está relacionada com a tomada ágil de decisões, já que nos oferecem oportunidades mais frequentes de feedback e correção da funcionalidade, tornando-se mais flexível a se adaptar nas práticas ágeis vistas na Figura 3.

2.6 Modelo de Gestão de Melhoria de Processo

O ciclo PDCA (Plan, Do, Check e Action) é um método de gestão e controle que representa o direcionamento a ser seguido para que as metas estabelecidas possam ser atingidas, tornando se uma ferramenta eficiente de implantação de melhorias no processo. ((WERKEMAN, 1995), citado em CHIAMULERA et al. (2017))

As quatro fases estão descritas as seguir: (SILVA; MEDEIROS, 2015)

- Planejar (plan): são estabelecidos os objetivos e metas, visando-se à escolha dos métodos e procedimentos para alcançá-los;
- Fazer (do): etapa de implementação do planejamento;
- Checar (check): verificação da consistência do planejamento, através da comparação entre as metas estabelecidas e os resultados alcançados;
- Agir (action): etapa em que são realizadas correções de eventuais desvios encontrados na fase de monitoramento (check), de modo a melhorar a eficiência e eficácia.

Para entender o papel das ferramentas da qualidade dentro do ciclo do PDCA, devemos novamente destacar que a meta (resultado) é alcançada por meio do método (PDCA). Quanto mais informações (fatos, dados, conhecimentos) forem agregadas ao método, maiores serão as chances de alcance da meta e maior será a necessidade da utilização de ferramentas apropriadas para coletar, processar e dispor estas informações durante o giro do PDCA. (CAMPOS (2004), citado em CHIAMULERA et al. (2017))

É de suma importância destacar que, após passar pelas etapas do ciclo PDCA, como afirma CHIAMULERA et al. (2017), o sistema produtivo da organização passa a um patamar superior de qualidade, em que os surgimentos de novos problemas serão vistos como oportunidades de melhorias. Dessa forma, o modelo IDEAL para KOSLOSKI (2005), foi estabelecido pelo Software Engineering Institute - SEI, da Universidade de Carnegie Mellon, tal modelo funciona como guia para organizações no que tende a implementação e planejamento de um modelo de melhoria de processos de software.

A melhoria é buscada pela introdução de novas políticas, tecnologias, métodos e ferramentas, no intuito de se obter um ambiente de desenvolvimento mais eficiente. O modelo IDEAL é constituído por 5 etapas conforme mostrado na Figura 4 abaixo: (MCFEELEY (1996) citado em KOSLOSKI (2005))

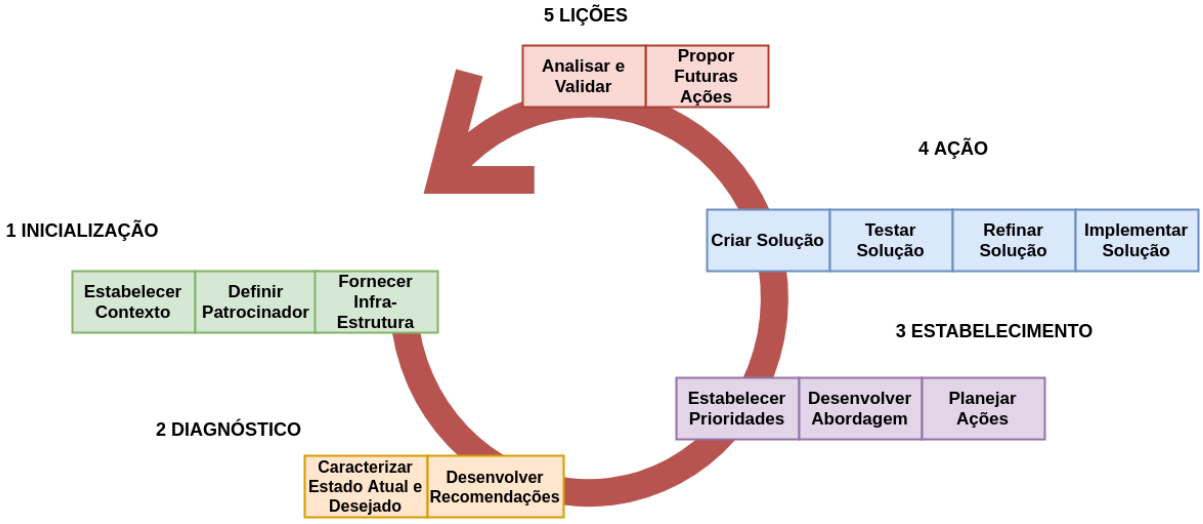

Figura 4 – Etapas do modelo de melhoria de processo IDEAL. Fonte: MCFEELEY (1996)

- **Iniciação** - onde, a partir do conhecimento das necessidades de mudanças das práticas da organização, são estabelecidos os contextos de melhorias pretendidas, definidos os patrocínios e acertadas as necessidades de infraestruturas;
- **Diagnóstico** - onde a preocupação reside em diagnosticar os estados: atual e pretendido, para a melhoria do processo, registrando recomendações sobre meios e procedimentos para as fases posteriores;
- **Estabelecimento** - de prioridades dos esforços de mudança, estratégias e abordagens do trabalho considerando as necessidades e recursos disponíveis além do desenvolvimento de um plano de ações, consistente com a realidade da organização na melhoria pretendida;
- **Ações** - onde efetivamente são criadas, testadas e refinadas as propostas de solução. Os testes envolvem projetos pilotos onde são feitas avaliações quantitativas, que podem realimentar as próprias soluções propostas antes da concepção do plano final de implementação;
- **Lições aprendidas** - onde são feitas avaliações finais, sobre se as necessidades de negócios foram satisfeitas com relação às melhorias estabelecidas. Nessa fase as lições são coletadas, analisadas, resumidas e documentadas como propostas de melhorias para futuras implementações.

2.6.1 Métricas de Software(GQM)

Segundo GENCEL et al. (2013), as organizações de software estão cientes, da importância dos processos de medição na tomada de decisões informadas para gerenciar

melhor os processos, produtos e projetos de software. A própria medição é considerada uma força motriz para a melhoria do processo de software (SCHALKEN; VLIET, 2008). Também facilita a comunicação eficaz entre organizações de software e clientes.

Como em qualquer disciplina de engenharia, o desenvolvimento de software requer um mecanismo de medição para feedback e avaliação. A medição é um mecanismo para criar uma memória corporativa e um auxílio para responder a uma variedade de perguntas associadas à aprovação de qualquer processo de software. Ajuda a apoiar o planejamento do projeto (por exemplo, quanto custará um novo projeto?); permite determinar os pontos fortes e fracos dos processos e produtos atuais (por exemplo, qual é a frequência de certos tipos de erros?); fornece uma justificativa para a adoção / refinação de técnicas (por exemplo, qual é o impacto da técnica XX na produtividade dos projetos?); isso nos permite avaliar a qualidade de processos e produtos específicos (por exemplo, qual é a densidade de defeitos em um sistema específico após a implantação?). A medição também ajuda, durante o curso de um projeto, a avaliar seu progresso, a tomar ações corretivas com base nessa avaliação e a avaliar o impacto de tal ação. (BASILI; CALDEIRA; ROMBACH, 1994)

Várias estruturas, modelos e padrões foram desenvolvidos para apoiar as organizações de software no planejamento de seus programas de medição. (GENCEL et al., 2013)

A abordagem GQM (Goal Question Metric) baseia-se no pressuposto de que, para uma organização medir de maneira proposital, primeiro deve especificar as metas para si e para seus projetos; depois, deve rastrear essas metas para os dados que se destinam a defini-las. Operacionalmente e, finalmente, forneça uma estrutura para interpretar os dados em relação aos objetivos estabelecidos. Portanto, é importante esclarecer, pelo menos em termos gerais, quais necessidades de informações a organização possui, para que essas necessidades de informações possam ser quantificadas sempre que possível, e as informações quantificadas possam ser analisadas para determinar se os objetivos são ou não alcançados

BASILI, CALDEIRA e ROMBACH (1994) afirma que o modelo de medição possui três níveis essenciais:

1. Nível conceitual (GOAL / OBJETIVO): Onde uma meta é definida para um objeto, com relação a vários modelos de qualidade, de vários pontos de vista, em relação a um ambiente específico. Os objetos de medição são:
    - Produtos: artefatos, entregas e documentos produzidos durante o ciclo de vida do sistema. Por exemplo: especificações, projetos, suítes de teste.
    - Processos: atividades relacionadas ao software normalmente associadas ao tempo. Por exemplo: especificar, projetar, testar, entrevistar.

- Recursos: itens usados pelos processos para produzir resultados. Por exemplo: pessoal, hardware, software, espaço de escritório.

2. Nível operacional (QUESTION / QUESTÃO): Um conjunto de perguntas é usado para caracterizar, se caso, a avaliação / realização de um objetivo específico será realizado conforme o modelo de caracterização pré-estabelecido. As perguntas tentam determinar o objeto de medição (produto, processo, recurso) com relação a um problema de qualidade escolhido.

3. Nível quantitativo (METRIC / MÉTRICA): um conjunto de dados que está diretamente associado a todas as perguntas que foram feitas no nível anterior. Os dados podem ser:

   - Objetivo: eles dependem apenas do objeto que está sendo medido e não do ponto de vista de onde foram retirados. Por exemplo: número de versões de um documento, horas do desenvolvedor gastas em uma tarefa, tamanho de um programa.
   - Subjetivo: eles dependem do objeto que está sendo medido e do ponto de vista de onde são retirados. Por exemplo: legibilidade de um texto, nível de satisfação do usuário.

Segundo BASILI, CALDEIRA e ROMBACH (1994), um modelo GQM é uma estrutura hierárquica, começando com um objetivo (especificando o objetivo da medição, o objeto a ser medido, o problema a ser medido e o ponto de vista do qual a medida é tomada). O objetivo é refinado em várias perguntas, que geralmente dividem o problema em seus principais componentes. Cada questão é polida em métricas, algumas objetivas e outras subjetivas. A mesma métrica pode ser usada para responder questões diferentes sob o mesmo objetivo. Vários modelos de GQM também podem ter perguntas e métricas em comum, garantindo que, quando a medida for realmente executada, os diferentes pontos de vista sejam todos levados em consideração.

3 Metodologia de pesquisa

3.1 Considerações Iniciais do Capítulo

O objetivo geral deste trabalho é propor melhorias de processo de software a partir da análise de issues cadastradas no sistema de de administração de issues da organização X.

Neste capítulo apresenta-se a metodologia de pesquisa selecionada neste trabalho. Para isso, inicia-se com a classificação da metodologia de pesquisa, como o tipo, os procedimentos de pesquisas, e as técnicas de coletas de dados adotadas. Em seguida, apresenta-se o plano metodológico empregado neste trabalho.

3.2 Classificação da Metodologia de Pesquisa

A metodologia de pesquisa adotada neste trabalho foi classificada quanto à natureza, à abordagem, à tipologia, aos procedimentos técnicos e as técnicas de coleta de dados.

Quanto à natureza, esta pesquisa é aplicada, pois, tem como objetivo a geração de conhecimentos dirigidos à solução de problemas específicos (GIL, 2008), definindo o planejameto para iniciar um de programa de melhoria contínua de processos de software no contexto de desenvolvimento ágil de software.

Quanto à abordagem, é classificada como qualitativa. A abordagem qualitativa, que não requer o uso de métodos e técnicas estatísticas (GUNTHER, 2006), é empregada para análise textual das issues registradas na ferramenta JIRA, para análise de entrevistas semiestruturadas para análise de causa raiz dos problemas identificados no JIRA.

Em relação ao objetivo, a presente pesquisa caracteriza-se como metodológica, pois refere-se à elaboração de instrumento de captação ou de manipulação da realidade. Está associada a caminhos, formas, maneiras, procedimentos para se atingir determinado fim. (VERGARA, 2000)

Quanto aos procedimentos técnicos de coleta de dados, os meios de investigação adotados foram: pesquisa bibliográfica; documental e observacional.

Quanto às técnicas de coletas de dados, foram elaborados instrumentos como entrevistas, alémn das análises dos testos das próprias issues (análise documental).

Uma representação da seleção metodológica é apresentada na Figura 5.

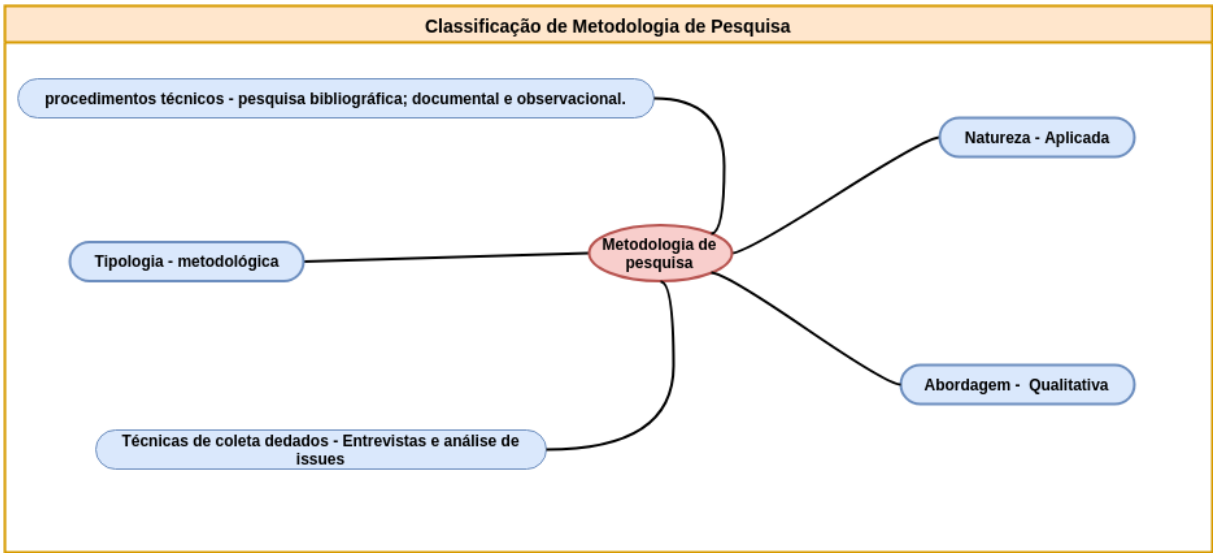

Figura 5 – Classificação metodológica. Fonte: Autora.

3.3 Plano Metodológico Adotado

Segundo Gil (2008), um processo de pesquisa envolve: planejamento do trabalho; coleta de dados com a respectiva análise e interpretação dos mesmos e, em seguida, a redação do resultado. E cada uma dessas grandes fases pode ser subdividida em outras mais específicas, dando origem aos mais diversos esquemas.

Nesta pesquisa, o processo de pesquisa adotado compreende as fases: Planejamento; Coleta de Dados; Análise e Interpretação dos dados, e Redação do Resultado. A partir da metodologia de pesquisa adotada e da determinação das fases que a configuram, foi elaborado um plano metodológico e foram definidas as etapas da pesquisa, nas quais são empregados os procedimentos e as técnicas de coleta de dados, como mostrado na Figura 6.

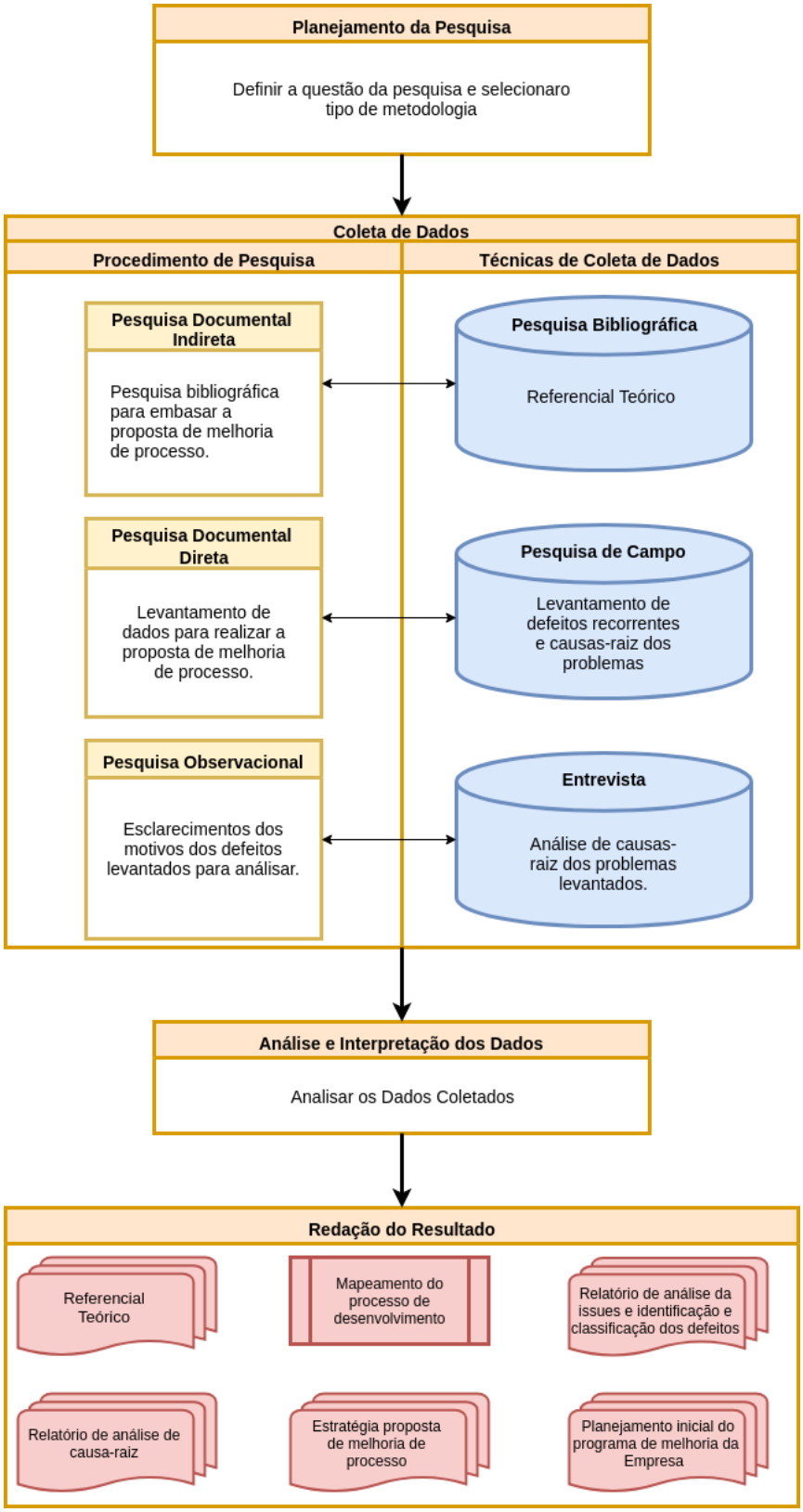

Figura 6 – Classificação metodológica. Fonte: Autora.

3.3.1 Planejamento do trabalho

Esta fase inicial compreendeu o estabelecimento da pergunta de pesquisa, do objetivo a ser atingido, da seleção metodológica, com fases e etapas da pesquisa, e com o planejamento dos procedimentos de pesquisa e das técnicas de coletas de dados a serem executados.

3.3.2 Coleta de dados

As ténicas de coleta de dados aplicadas neste trabalho são: (a) Pesquisa documental indireta; (b) Pesquisa documental direta; e (c)Pesquisa observacional.

Quanto à pequuisa documental, ela é "documental indireta", pois usa a pesquisa bibliográfica para aprofundar o entendimento do problema em relação ao tema de estudo (MARCONI e LAKATOS, 2010. As fontes utilizadas são publicações, tais como livros e artigos técnicos.

Ainda de acordo com Marconi e Lakatos (2010) pesquisa documental direta constitui-se, em geral, no levantamento de dados no próprio local onde os fenômenos ocorrem. Esses dados são obtidos por meio de pesquisa de campo ou de pesquisa de laboratório. Neste trabalho usa-se a pesquisa de campo do tipo exploratória que tem o objetivo: (i) Identificar e classificar os defeitos recorrentes registrados durante a execução do processo de desenvolvimento de software na organização X; e (ii) Determinar junto aos colaboradores da organização a causa raiz dos defeitos selecionados e priorizados para análise.

Para apoiar a pesquisa de campo é usada a pesquisa observacional intensiva por meio da realização de entrevistas não estruturadas (MARCONI e LAKATOS, 2010), de forma que o pesquisador possa explorar e buscar esclarecimentos sobre os motivos dos defeitos selecionados para análise. Sendo assim, a entrevista classifica-se como focalizada.

3.3.3 Análise e interpretação dos dados

Esta é uma fase também muito importante na pesquisa. Nesta fase, os resultados obtidos a partir de cada técnica de coleta de dados são analisados com o objetivo de refinamento e validação. A partir dos resultados obtidos com a análise dos dados, será estabelecido o planejamento do programa de melhoria de processo de software da organização X.

3.3.4 Redação dos resultados

A redação dos resultados desta pesquisa é constituída por: (1) Referencial Teórico deste trabalho, com a execução da pesquisa bibliográfica; (2) Mapeamento do processo de desenvolvimento de software da organização X; (3) Relatório de análise da issues com a

identificação e classificação dos defeitos recorrentes; (4) Relatório de análise de causa raiz dos defeitos recorrentes, incluindo os gráficos de espinha de peixe; (5) Estratégia proposta para estabelecimento de programas de melhoria de processo de software a partir da análise de issues; e (6) Planejamento inicial do programa de melhoria da organização X. Estes produtos serão apresentados nos capítulos e apêndices desta monografia.

3.4 Cronograma e Planejamento do Trabalho

De acordo com o plano metodológico adotado, descrito na seção 3.3, o Trabalho de Conclusão de Curso 1 foi feito o planejamento do trabalho na quinzena 1 e 2, a coleta de dados foi implementado na quinzena 3, análise e interpretação dos dados implementados na quinzena 4 e redação dos resultados na quinzena 5 e 6. O detalhamento dessas implementações podem ser vistas na Figura 7.

Para o Trabalho de Conclusão de Curso 2, será definidas as primeiras métricas, ocorrerá a aplicação do brainstorming para levantar causas-raiz juntamente com a modelagem e execução de diagrama de prevenção de dados, para assim analisar os resultados.

Para esta subseção foi feito um cronograma e uma EAP, estrutura analítica de projeto, vistos na Tabela 1, Tabela 2 e na Figura 7, respectivamente.

O cronograma foi dividido em duas tabela que equivale ao Trabalho de Conclusão de Curso 1 e Trabalho de Conclusão de Curso 2.

Data	**Atividades**
16/08/2019 à 16/09/2019	Decisão de Tema da proposta inicial, exportar e identificar issues da organização X. Determinar o problema, a pergunta de pesquisa, o objetivo geral e os objetivos específicos.
17/09/2019 à 16/10/2019	Estudos preliminares sobre melhoria de processos de software e entender relações entre a qualidade do processo, do produto e em uso. Revisar a literatura.
17/10/2019 à 16/11/2019	Entregar referencial teórico preliminar, introdução, metodologia, proposta de melhoria e fazer ajustes levantados na revisão.
17/11/2019 à 02/12/2019	Revisão e ajustes necessários e entregar Trabalho de Conclusão de Curso 1.

Tabela 1 – Cronograma do Trabalho de Conclusão de Curso 1

Data	**Atividades**
25/03/2020 à 08/04/2020	Revisar e ajustar o Trabalho de Conclusão de Curso 1 de acordo com avaliação feita.
25/03/2020 à 11/09/2020	Classificar issues.
14/09/2020 à 17/09/2020	Realizar brainstorming na organização X para levantar causas-raiz.
18/09/2020 à 02/10/2020	Determinar prevenção de defeitos.
03/10/2020 à 21/10/2020	Modelar plano de ações.
22/10/2020 à 05/11/2020	Validar e ajustar plano de ações com os orientadores.
06/11/2020 à 19/11/2020	Analisar resultados obtidos.
20/11/2020 à 04/12/2020	Definir métricas.
20/11/2020 à 04/12/2020	Demonstrar a conclusão obtida e entrega da primeira versão completa.
07/12/2020 à 09/12/2020	Revisão e ajustes necessários e entregar Trabalho de Conclusão de Curso 2.

Tabela 2 – Cronograma do Trabalho de Conclusão de Curso 2

Para a EPA, o projeto foi dividido em duas grandes entregas que nomeei de "TCC 1"e "TCC 2", onde a primeira entrega é referente ao Trabalho de Conclusão de Curso 1 e a entrega 2 se refere ao Trabalho de Conclusão de Curso 2. O tempo de entrega foi definido em "Quinzenas", na qual foi feita uma listagem do número de quinzenas trabalhadas.

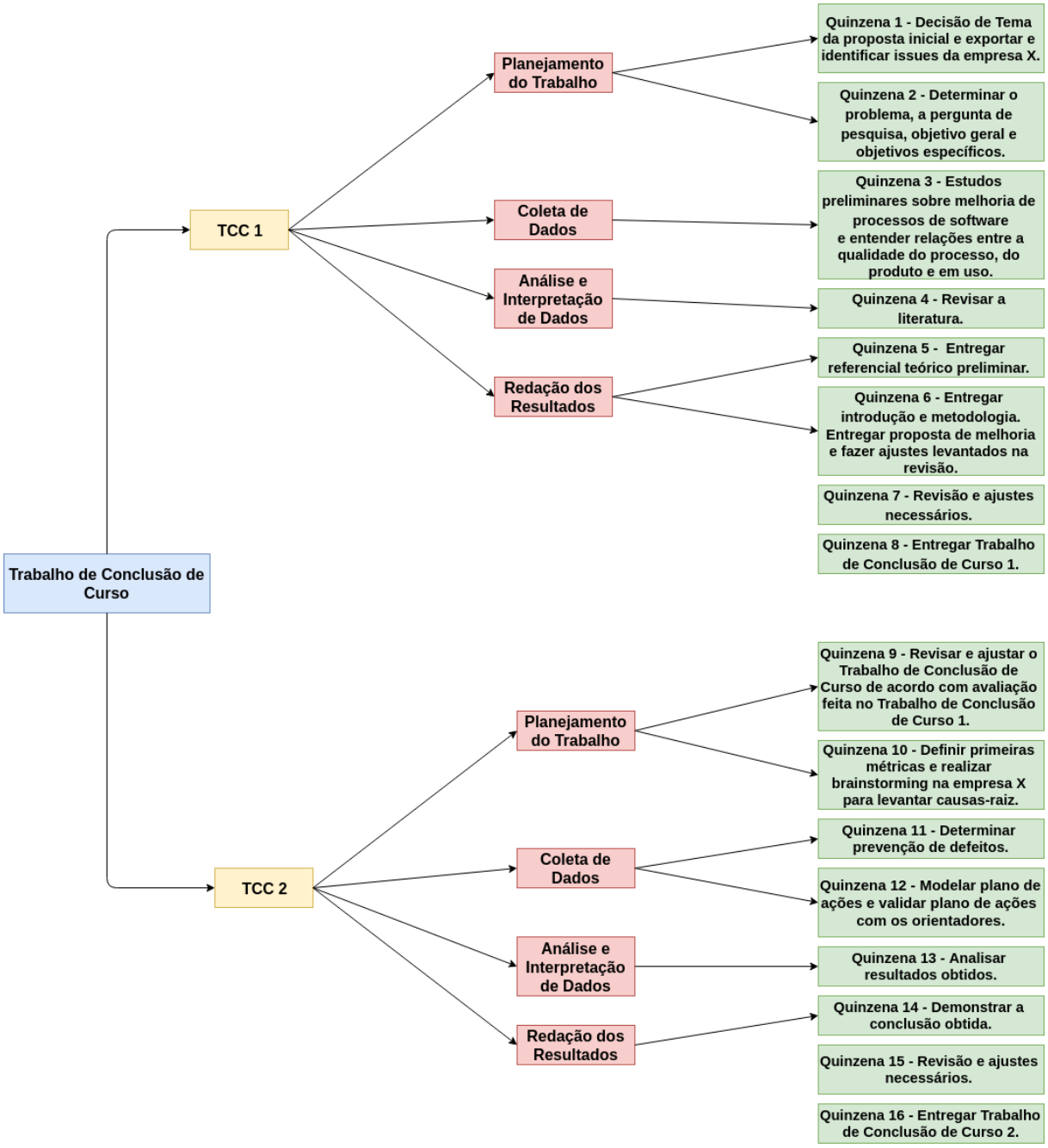

Figura 7 – Estrutura Analítica do Projeto de Trabalho de Conclusão de Curso.

4 Proposta de Melhoria de Processo na Empresa X

4.1 Considerações iniciais do capítulo

A Qualidade só pode ser conhecida quando avaliada. Para isso, é preciso basear-se em dados extraídos de processos e produtos que, quando cruzados entre si, resultam em valiosas informações capazes de auxiliar em tal avaliação. Mas, para que se obtenha bons resultados são necessários procedimentos que os garantam.

A fim de garantir a qualidade do desenvolvimento do produto é imprescindível a utilização de metodologias e técnicas que permitam uma melhor organização de idéias e fatos, dando maior objetividade ao processo de obtenção de dados e à análise que se fizer necessária.

Nesta seção, serão detalhadas algumas das principais técnicas e conceitos hoje utilizadas na melhoria contínua de processos, algumas das quais, aplicam-se diretamente na qualidade de software. Será apresentado o processo da organização X juntamente com a proposta de solução.

4.2 Contexto organizacional

A organização X, conta com uma equipe altamente qualificada para desenvolver as melhores soluções para certificação digital. Trabalham com organizações públicas ou privadas, pequenas ou grandes que possuem necessidades de desenvolvimento de software.

A organização X adota algumas técnicas do modelo scrum, por compreender que, equipes ágeis oferecem vantagens potenciais em relação às equipes tradicionalmente gerenciadas. Sabendo disso, os métodos Scrum foram englobados nos projetos. A Daily é feita diariamente no horário mais cômodo para toda a equipe, ao final de cada Sprint é feito o planejamento, onde definimos todas as issues que serão implementadas no decorrer das duas semana visto que temos um Backlog de produtos, o time de desenvolvimento tem um papel muito importante, pois garante que todos as funcionalidade que foram propostas sejam implementadas, sem sobrecarga dos desenvolvedores e tornando o trabalho mais satisfatório possível para todos. Essa é uma das preocupação da empresa, de que seus funcionários estejam inseridos em um ambiente de trabalho que seja o mais agradavel possível.

A empresa faz uso da prática ágil de releases a cada sprint, já que a equipe de

desenvolvimento é madura determinamos que cada sprint tenha a duração de duas semana, estabelecemos que o final das sprints seriam às sextas e o início seria nas segundas. Temos os ambientes de desenvolvimento e homologação, a cada release fechada uma nova versão é gerada, o sistema atualizado vai para o ambiente homologação onde será apresentado para cliente. O ambiente de desenvolvimento e homologação são para os testes serem feitos, o usuário final não tem acesso a tais ambientes, apenas o de produção.

Se faz uso da ferramenta de gerência de projetos Jira da empresa Atlassian, o Jira Software oferece suporte a qualquer metodologia de gerenciamento de projeto ágil para o desenvolvimento do produto de software. A organização planeja as sprints, distribui tarefas para os desenvolvedores e usa as issue que estão documentados na ferramenta JIRA. Visto na Figura 8

A organização se preocupa com a satisfação do cliente, onde conversas são feitas semanalmente e diretamente com o cliente. A gerência de projetos tem o papel de sanar todas as eventuais dúvidas e questões que os desenvolvedores tenham, isso evita o desperdício de tempo dos desenvolvedores. A gerência de projetos tem a função de entender todas as funcionalidades que irão compor o sistema.

Depois de passada as tarefas para os desenvolvedores é feito o acompanhamento de perto de todos os detalhes do desenvolvimento para evitar o retrabalho. Toda a equipe discute e prioriza as tarefas que serão feitas no dia, em seguida entramos na fase de testes. A gerência de projetos é responsável pelos testes funcionais do sistema, é de inteira responsabilidade da gerência de projetos entender o sistema, entender o que o cliente quer e passar para os desenvolvedores os requisitos do sistema e ajustes.

A priorização de funcionalidades é feito junto ao cliente, onde reunidos podemos ter uma visão ampla do sistema e assim decidirmos quais issues são essenciais naquela sprint. A análise de gargalos é feita pela gerência de projetos, onde identificamos as issues que estão a muito tempo no Backlog do produto. Entramos em contato com o cliente pela própria issue deixando comentários ou por email, onde, discutimos sobre a prioridade de tal funcionalidade. O cliente determina o momento oportuno para a finalização das issues.

Na Figura 8, foi modelado o processo usado na empresa, mostrando todas as práticas ágeis que foram citadas acima e que são utilizadas na organização X.

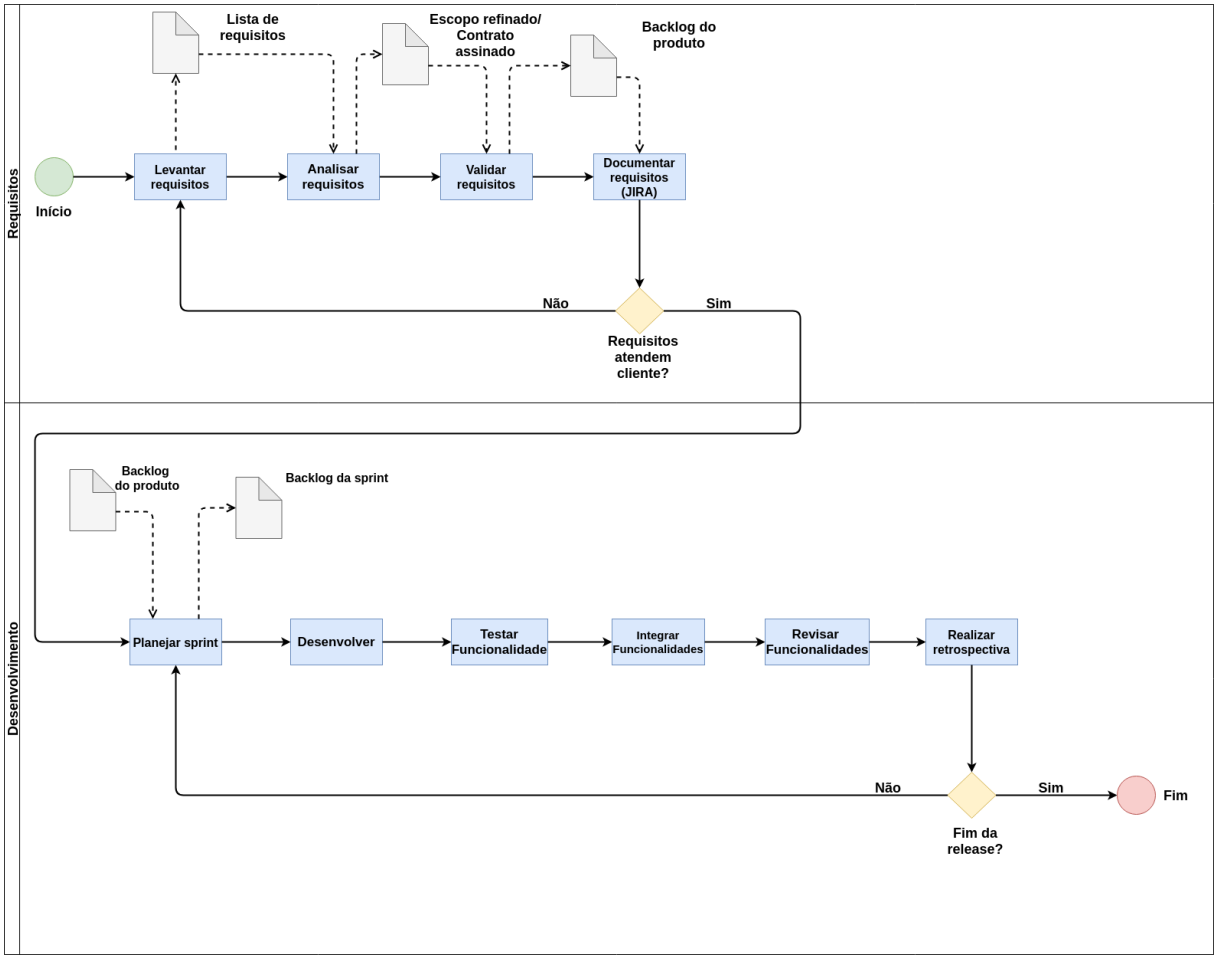

Figura 8 – Processo de desenvolvimento da organização X. Fonte: Autora

1. Levantar Requisitos

   - **Descrição:** Nesta atividade é feita a coleta dos requisitos funcionais e não funcionais do projeto.

2. Analisar Requisitos

   - **Descrição:** Logo após é feito uma análise e interpretação dos requisitos do projeto.

3. Validar Requisitos

   - **Descrição:** Os requisitos são validados como cliente, nesta fase é feita a especificação dos requisitos.

4. Documentar Requisitos (JIRA)

   - **Descrição:** Este documento é embasado na fase de coleta de requisitos e é melhorado ao longo de sprint. Tal documentação é transformada em issues

e é feita toda no Jira Software. Uma issue pode representar uma tarefa do projeto, um funcionalidade, um solicitação de ajuste de funcionalidade, falhas do sistema ou de uma funcionalidade.

5. Planejar Sprint

   - **Descrição:** Nesta etapa do processo é elaborado um documento de backlog do projeto, onde estão e são descritas as issues do sistema.

6. Desenvolver

   - **Descrição:** O processo de desenvolvimento é onde são implementados as funcionalidade de software.

7. Testar

   - **Descrição:** Depois de implementada a funcionalidade é testada, se for observado que a funcionalidade foi mal implementada ou algum bug e falha; é registrado uma issue para que seja feita posteriormente pelos desenvolvedores.

8. Integrar Funcionalidades

   - **Descrição:** Depois de devidamente testada é feito o merge das funcionalidades implementadas. Tais funcionalidades são derivadas das issues da sprint.

9. Revisar Funcionalidades

   - **Descrição:** Esta atividade é feita junto ao cliente. A cada versão lançada do sistema é validada com o cliente se estiver de acordo com o esperado é feito o merge para o sistema de produção.

10. Revisar Retrospectiva

    - **Descrição:** Ao final de cada sprint é feita a retrospectiva, onde é detalhado tudo o que foi feito e o que não foi feito, e os motivos que levaram a esses resultados.

4.3 Proposta de Solução

O objetivo desta seção é descrever a proposta de solução para a melhoria de processo de software na organização X. Para definir a Estratégia de planejamento do programa de melhoria de processo de software na organização X, foi selecionada a abordagem IDEAL (MCFEELEY, 1996) de estabelecimento de programas de melhoria contínua, apresentada no Capítulo 2.

Com base nas fases da Abordagem IDEAL, são definidas as etapas e atividades que constituem a Estratégia de planejamento do programa de melhoria a partir de issues.

4.3.1 Etapa 1 - INICIAÇÃO

Esta etapa é composta pelas atividades:

1. Definir os esforços para mudança e os objetivos de melhoria;
    - **Descrição:** Nesta atividade será definido quais seriam os objetivos de melhoria do processo e quanto seria o esforço para implementação da melhoria.
    - **Responsáveis:** Orientadores e autora.
    - **Entradas:** —
    - **Saídas:** Objetivos de melhoria.
2. Estabelecer o patrocinador do programa de melhoria;
    - **Descrição:** Ficará estabelecido qual organização será levantado e implementado a melhoria de processo.
    - **Responsáveis:** Autora.
    - **Entradas:** —
    - **Saídas:** Patrocinador.
3. Estabelecer a infraestrutura mínima necessária para condução das melhorias.
    - **Descrição:** Esta atividade consistirá em estabelecer qual será a infraestrutura da melhoria de processo. Será utilizado a base de dados da ferramenta Jira Software.
    - **Responsáveis:** Orientadores e autora.
    - **Entradas:** —
    - **Saídas:** Base de dados da ferramenta Jira Software.

4.3.2 Etapa 2 - DIAGNÓSTICO

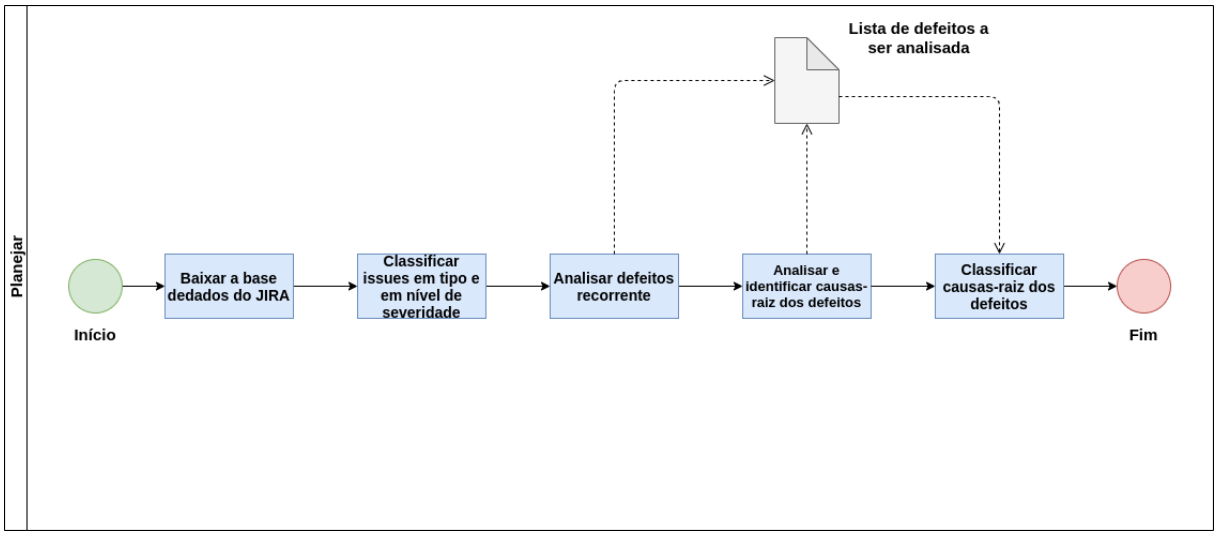

Figura 9 – Proposta de processo de planejamento da análise de dados para a organização X. Fonte: Autora

4.3.2.1 Baixar a base de dados do JIRA

- **Descrição:** Nesta atividade, é considerada fundamental pois irá listar as issues da Organização X a serem analisadas. Para definir e agrupar defeitos que são recorrentes.
- **Responsáveis:** Autora.
- **Entradas:** Base de dados do JIRA.
- **Saídas:** Planilha de issues da Organização X.

4.3.2.2 Classificar issues em tipo e em nível de severidade

- **Descrição:** A partir da planilha de issues da atividade anterior, é feita a classificação de issues em tipo e em nível de severidade, para que possa agrupar e assim levantar causas-raiz das issues.
- **Responsáveis:** Autora.
- **Entradas:** Planilha de issues da Organização X.
- **Saídas:** Agrupamento de issues.

4.3.2.3 Analisar defeitos recorrentes

- **Descrição:** Nesta atividade são analisados e selecionados os defeitos que mais aparece no projeto.
- **Responsáveis:** Autora.
- **Entradas:** Agrupamento de issues.
- **Saídas:** Lista de defeitos recorrente a ser analisada.

4.3.2.4 Analisar e identificar causas-raiz dos defeitos

- **Descrição:** Com base na classificação das causas-raiz dos defeitos, será feita uma análise dessas causas-raiz e seleção dos defeitos recorrentes para que ocorra a aplicação da prevenção de defeitos.
- **Responsáveis:** Autora.
- **Entradas:** Lista de defeitos recorrente a ser analisada.
- **Saídas:** —

4.3.2.5 Classificar causas-raiz dos defeitos

- **Descrição:** A partir da atividade anterior é feito uma classificação de causas-raiz dos defeitos recorrentes. É nesta atividade que se define o grau de complexidade das causas-raiz, visto que as técnicas e métodos já foram definidas no Capítulo 3.
- **Responsáveis:** Equipe de desenvolvimento.
- **Entradas:** Agrupamento de issues.
- **Saídas:** Lista de defeitos recorrente a ser analisada.

4.3.3 ETAPA 3 - ESTABELECIMENTO

1. Definir as ações de melhoria;
    - **Descrição:** Nesta atividade será definido quais serão as ações vão ser implementadas na melhoria de processo da organização X. Onde são levantados e definidos os pontos de melhoria de processo para a organização X.
    - **Responsáveis:** Equipe de desenvolvimento.
    - **Entradas:** Pontos de melhorias.

- **Saídas:** Ações de melhorias.

2. Priorizar as ações de melhorias de critérios estabelecidos junto à organização X.

   - **Descrição:** A partir da definição de ações melhorias e metodologia de priorização para a organização X, será feito uma classificação dos pontos de melhorias acordo com o seu nível de importância, para a realização do processo.
   - **Responsáveis:** Equipe de desenvolvimento.
   - **Entradas:** Ações de melhorias.
   - **Saídas:** Relatório com ações de melhorias priorizadas.

3. Elaborar o planejamento detalhado o 1º ciclo do programa de melhoria de processo de software a partir das priorização feita na tarefa anterior.

   - **Descrição:** Nesta atividade será feito um planejamento do primeiro ciclo do programa de melhoria de processo de software a partir das priorização feita na tarefa anterior.
   - **Responsáveis:** Autora.
   - **Entradas:** Relatório com ações de melhorias priorizadas.
   - **Saídas:** Planejamento do primeiro ciclo do processo.

4. Estabelecer as alternativas de solução técnica;

   - **Descrição:** Será estabelecido algumas possibilidade de solução técnica para a melhoria de processo.
   - **Responsáveis:** Autora.
   - **Entradas:** —
   - **Saídas:** Alternativas de solução técnica.

5. Elaborar o plano de medição para avaliar os benefícios das melhorias a serem implementadas.

   - **Descrição:** Nesta atividade será elaborado um plano de medição, onde será coletados dados do ciclo de melhoria para analisar e interpretar os dados gerando um relatório de feedback apontando os benefícios das melhorias.
   - **Responsáveis:** Autora.
   - **Entradas:** Métricas.
   - **Saídas:** Plano de medição.

4.3.4 Etapa 4 - AÇÃO

Aqui define-se a solução técnica associada à prevenção de defeitos e executa-se o plano de ações. Tal etapa ficará a critério da organização X e será de inteira responsabilidade da mesma.

4.3.5 Etapa 5 - LIÇÃO

Esta etapa ficará a critério da organização X e será de inteira responsabilidade da mesma.

5 Aplicação da Proposta de Solução

5.1 Considerações iniciais do capítulo

A pesquisa tratou do conhecimento das ferramentas da qualidade e sua utilização dentro das organizações. Foi de grande valia, para o conhecimento de como é importante a organização e utilização das ferramentas de qualidade de um processo. As organizações que se preocupam com a qualidade renovam a sua história, e baseiam-se os seus propósitos além de reavaliar a sua trajetória no mercado.

Todavia um resultado eficiente, o ambiente onde as pessoas trabalham em equipe, possibilita um desempenho mais eficaz na busca dos objetivos a que a instituição propôs.

Percebe-se que estas ferramentas possuem estratégias em alcançar os efeitos e causas de um problema dentro de um processo através do uso de algumas mais simples, todavia são complementadas com outras, obtendo maior eficiência na solução dos problemas detectados.

Toda e qualquer organização deve fazer uso das ferramentas básicas da qualidade, juntamente com outras, na busca de melhorias contínuas em seu processo de produção.

5.2 ETAPA 1 - Iniciação

A fase de iniciação do modelo IDEAL é o ponto de partida. Aqui é onde a infraestrutura de melhoria inicial é estabelecida, e suas funções e responsabilidades são definidas e os recursos iniciais são atribuídos. Nessa fase, um plano de melhoria de processo de software é criado para orientar a organização X durante as fases de inicialização, diagnóstico e estabelecimento.

Os objetivos gerais do programa de melhoria de processo de software são definidos durante a fase de iniciação. Eles são estabelecidos a partir das necessidades de negócios da organização e serão aperfeiçoados e tornados específicos durante a fase de estabelecimento do IDEAL.

5.2.1 Atividade 1 - Definir os esforços para mudança e os objetivos de melhoria

Nesta atividade o contexto da melhoria do processo foi estabelecido, foram definidos os esforços para a mudança e os objetivos da melhoria de processos. Foi feito um

estudo preliminar para identificar oportunidades de melhoria de processo de software na organização X, com o intuito analisar as issues.

5.2.2 Atividade 2 - Estabelecer o patrocinador do programa de melhoria

A escolha se deu por a organização X já adotar algumas técnicas do modelo scrum, porque trazem autoridade de decisão ao nível de problemas operacionais e incertezas e, assim, aumentam a velocidade e a precisão da resolução de problemas. Foi realizado um estudo sobre os princípios e características das principais metodologias ágeis aplicadas tanto no âmbito industrial quanto no acadêmico, ao mesmo tempo foi feito um levantamento sobre ferramentas e práticas utilizadas por tais metodologias.

Sabendo disso, foi comunicado minha pretensão de estudo com os sócios da organização X, onde tiveram bastante interesse e ficaram interessados no produto final do meu estudo, pois, seria muito enriquecedor para a organização e para os colaboradores.

5.2.3 Atividade 3 - Estabelecer a infraestrutura mínima necessária para a condução das melhorias

A ferramenta adotada na organização X é o Jira Software, este sistema foi desenvolvido para ajudar equipes de todos os tipos à gerenciar o trabalho, que vai desde gerenciamento de requisitos e casos de teste até o desenvolvimento ágil de software. O Jira Software combina técnicas de desenvolvimento com recursos ágeis e nos oferece várias funcionalidades, se faz necessária para a gerência das melhorias.

5.3 ETAPA 2 - Diagnóstico

A fase de diagnóstico do modelo IDEAL iniciará a organização X no caminho da melhoria contínua do processo de software. Nessa fase, o plano de ação da melhoria de processo de software é iniciado de acordo com a visão da organização X, plano estratégico de negócios, principais problemas de negócios enfrentados pela mesma.

5.3.1 Atividade 1 - Baixar a base de dados do JIRA

O Jira Software tem a funcionalidade de exportar os dados de um projeto. Foi gerado uma planilha indicando o nome da issue e os dados pertinentes a ela. Foi criado mais uma coluna de nível de severidade na planilha, para fazer a classificação dessas issues.

5.3.2 Atividade 2 - Classificar issues em tipo e nível de severidade

Os tipos de issues especificam diferentes tipos de trabalho e de maneira única ajudando a identificar, categorizar e relatar o trabalho da equipe. Uma issue pode ser, classificada como:

- ***Bug:*** Um bug é um problema que prejudica ou impede as funções de um produto. Algo que foi testado e não aprovado.
- ***Task:*** Uma tarefa representa o trabalho que precisa ser realizado. Atividades que envolvem a participação da equipe ou do cliente (se houver).
- ***New feature:*** Solicitação de uma nova funcionalidade ou recurso de software. Manutenção evolutiva, quando inclui, exclui, ou altera alguma funcionalidade.
- ***Improvement:*** Manutenção perfectiva - quando melhora algum requisitos da funcionalidade.
- ***Epic:*** Usamos épicos para agrupar bugs, new features, tasks e improvements para exibir o progresso de um objetivo maior. Mantendo a agilidade ao organizar tarefas. Segundo a documentação da Atlassian (2020), no desenvolvimento ágil, os épicos geralmente representam uma entrega significativa, como um novo recurso ou experiência no software que sua equipe desenvolve.

Os níveis de severidade foram estabelecidos, segundo os critérios:

- O esforço para resolução da issue.
- O número de desenvolvedores para resolver ou complexidade técnica da issue.
- Como a issue afeta o sistema.

5.3.3 Atividade 3 - Analisar defeitos recorrentes

Os dados utilizados para análise, foram as issues retirados da base de dados do Jira Software referente aos meses de Fevereiro de 2019 à Junho de 2020 da organização X. Com base na classificação já feita no Jira Software, durante todo o processo de desenvolvimento, foram levantados os dados e construído os diagramas de Pareto abaixo.

A contagem dos dados coletados foi realizada com base na planilha obtida da plataforma do Jira Software.

5.3.3.1 Análise de Pareto por tipo de issue

O diagrama de Pareto com os diferentes tipos de issues, pode ser visto na Tabela 3 e Figura 10.

Por ser utilizado para agrupar os diferentes tipos de issues, os Épicos foram discriminados no levantamento e classificações dos diagramas de Pareto mostrados neste trabalho.

Situação	**Frequência**	**%**	**% Acumulada**
Improvement	323	35,69%	35,69%
Bug	197	21,77%	57,46%
Task	195	21,55%	79,01%
New Feature	190	20,99%	100%
Total	905	100%	100%

Tabela 3 – Diagrama de Pareto por tipos de issues. Fonte: Autora

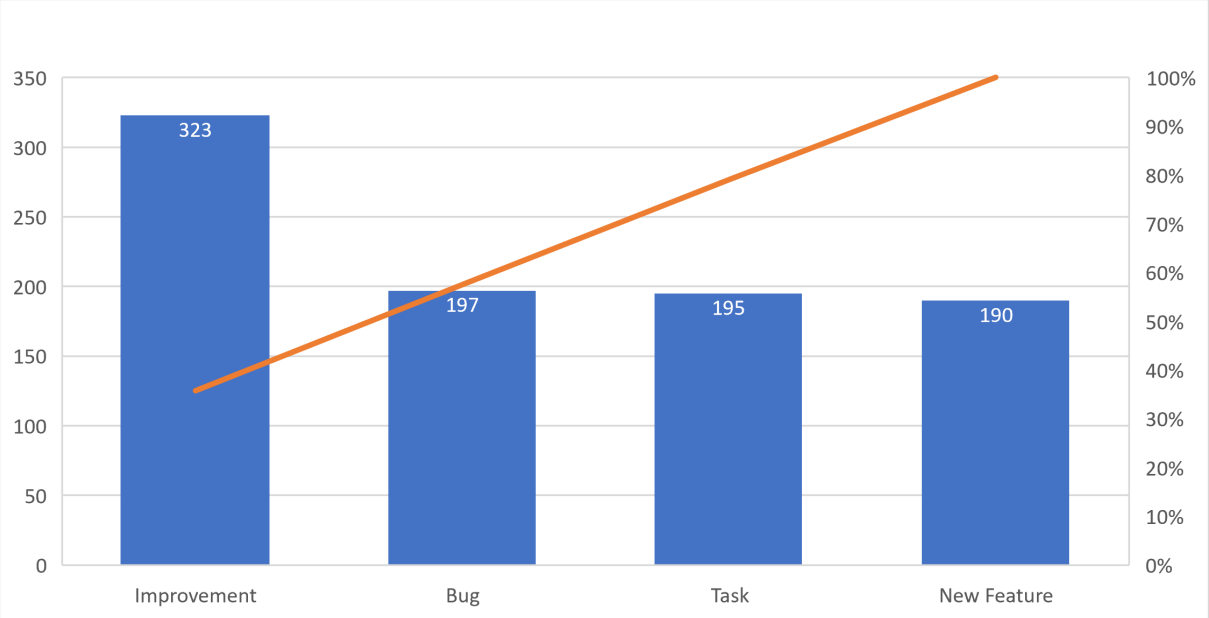

Figura 10 – Diagrama de Pareto por tipos de issues. Fonte: Autora

5.3.3.2 Análise de Pareto por Severidade

Com intuito de levantar falhas recorrentes, foi realizada uma classificação por nível de severidade de cada issue.

Os níveis de severidade foram estabelecidos, segundo os critérios da Seção "5.3.2. Atividade 2 - Classificar issues em tipo e nível de severidade", como salientado na Tabela 4:

Nível de severidade	**Esforço**	**Impacto da issue no sistema**
SEVERIDADE BAIXA	Até 8hs ou 1 dia de trabalho	- *Issues* de pequenas alterações no frontend e de experiência de usuário. - *Issues* de pequenos defeitos ou falhas encontrados no frontend. - *Issues* cujo sistema funciona, mas que envolve a manipulação de dados.
SEVERIDADE MÉDIA	Até 16h ou 2 dias de trabalho	- *Issues* que envolvem ajustes em performance e dependências de API's.
SEVERIDADE ALTA	Até 24h ou 3 dias de trabalho	- A *issues* implicaria diretamente no funcionamento do sistema. Precisa de mais cooperação entre as parte de auxílio externo.

Tabela 4 – Categorização dos níveis de severidade. Fonte: Autora

Em seguida foi feita uma tabela para traçar o diagrama de pareto, onde serão mostradas as situações de falhas na organização X ordenada pela sua respectiva frequência, o nível de severidade, a multiplicação da frequência das falhas com o nível de severidade, percentual de cada situação e percentual acumulado.

Situação	**Frequência**	**Severidade**	**Frequência*Severidade**	**%**	**% Acumulada**
Severidade 1	691	1	691	58,07%	58,07%
Severidade 2	143	2	286	24,03%	82,10%
Severidade 3	71	3	213	17,90%	100%
Total	905	-	1190	100%	100%

Tabela 5 – Diagrama de Pareto com nível de severidade. Fonte: Autora

Com essa organização dos dados na Tabela 5, foi possível traçar o diagrama de Pareto.

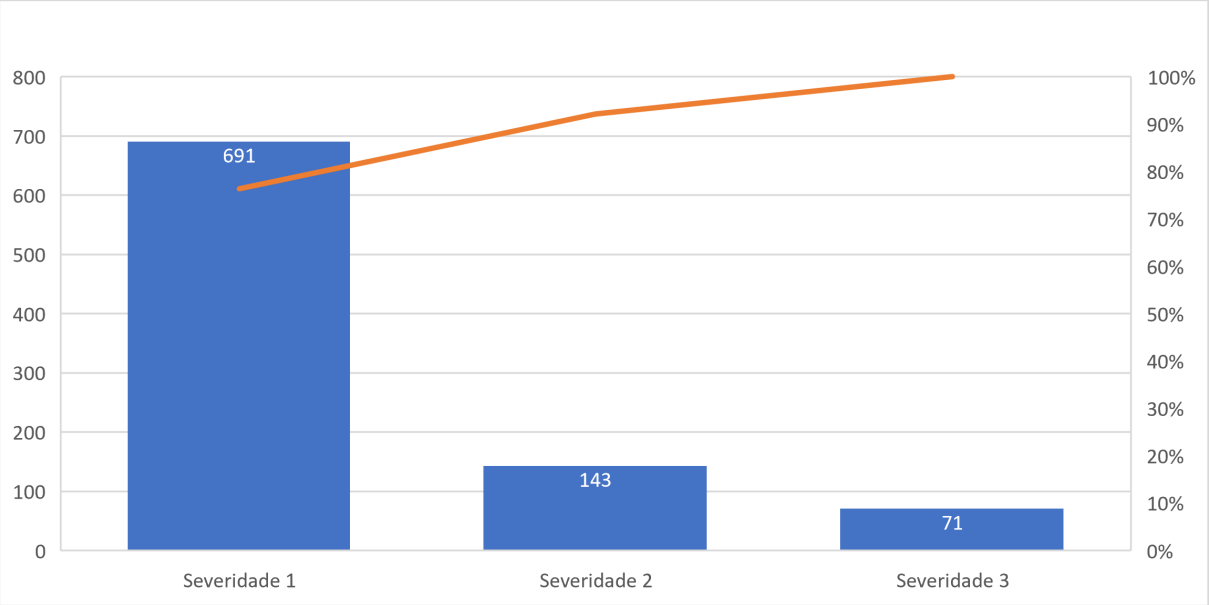

Figura 11 – Diagrama de Pareto por Níveis de Severidade da Issues. Fonte: Autora

Foi observado no Diagrama de Pareto na Figura 11, que 58,07% das issues são de severidade 1, visto que as issues pequenas foram as mais recorrentes, para melhor verificar e analisar os problemas, foi feito um novo diagrama de pareto usando as classificações existentes no Jira Software, ou seja, os tipos de issues.

5.3.3.3 Análise de Pareto de issues com nível de severidade 1

Situação	**Frequência**	**%**	**% Acumulada**
Improvement	253	36,61%	36,61%
Bug	193	27,93%	64,54%
Task	175	25,33%	89,87%
New Feature	70	10,13%	100%
Total	691	100%	100%

Tabela 6 – Diagrama de Pareto de tipos de issues com nível de severidade 1. Fonte: Autora

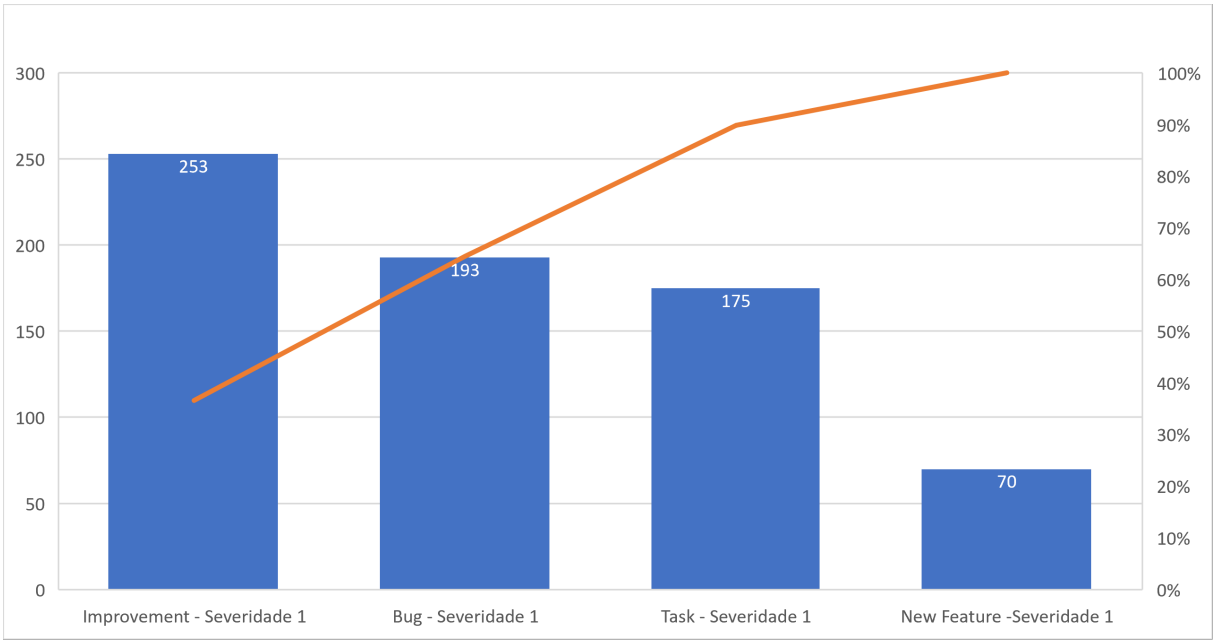

Figura 12 – Diagrama de Pareto de tipos de issues com nível de severidade 1. Fonte: Autora

A nova análise vista na Tabela 6 e na Figura 12, indica que 36,61% dessas issues são de Improvement, 27,93% são de Bug, 25,33% são de Task e 10,13% são de New Feature.

5.3.4 Atividade 4 - Analisar e identificar causa-raiz dos defeitos

Para obter as causas-raiz do problema ocorrido no processo atual, foi realizado um brainstorming com três desenvolvedores da organização X.

Com base no Diagrama de Pareto (issues por nível de severidade), foram selecionadas aquelas que foram criadas em decorrência de requisitos e testes de sistemas. Neste subconjunto, foram analisadas 10 issues, sendo 2 de improvement e bug; e três issues de task e new feature. Sobre as issues analisadas as que foram do tipo:

- *Improvement,* que foi as issues de melhoria de funcionalidade já implementadas no sistema;
- *Bug,* que são issues que apresentaram problemas que prejudicam ou impedem o bom funcionamento das funcionalidades do sistema;
- *Task,* que é uma tarefa significava um trabalho ou atividade que precisa ser realizada individual ou em conjunto;
- *New feature,* que representa uma nova funcionalidade ou recurso de software do sistema.

É possível concentrar forças nos pontos certos para solucionar o problema apresentados, como mostrou análise de Pareto. Dessa forma, gerou o Diagrama de Ishikawa visto nas Figuras 13 e 14.

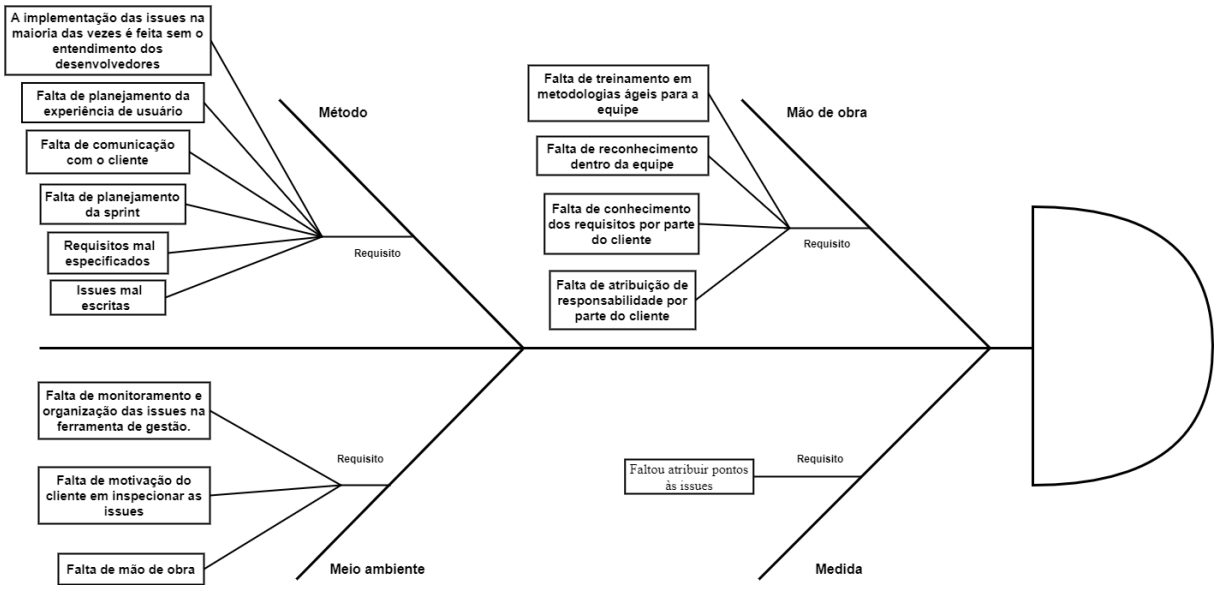

Figura 13 – Diagrama de Ishikawa - Causas de Requisito. Fonte: Autora

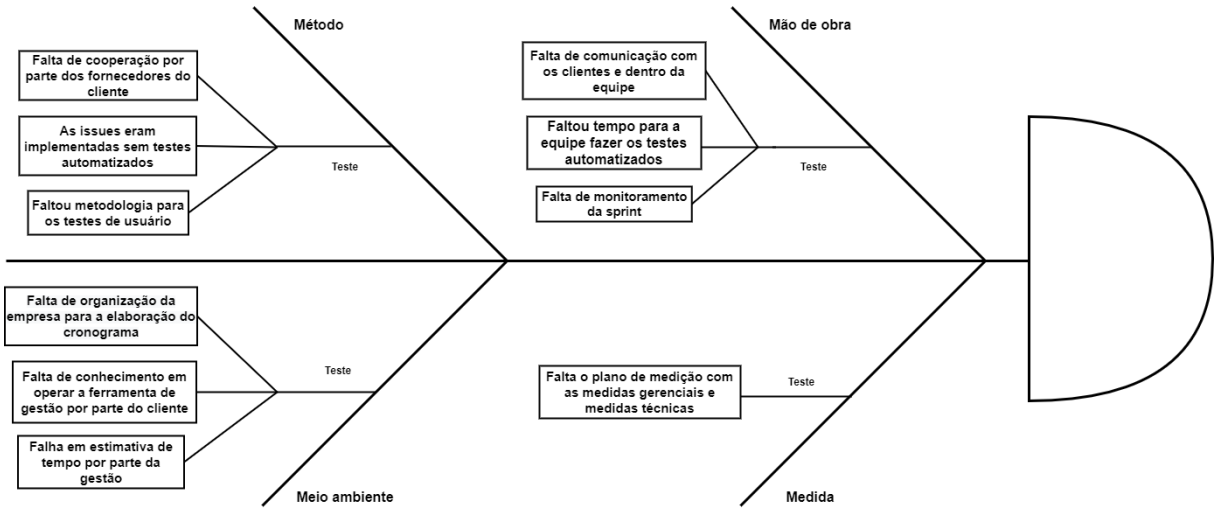

Figura 14 – Diagrama de Ishikawa - Causas de Teste. Fonte: Autora

Os Diagramas foram divididos em quatro categorias: método, meio ambiente, mão de obra e medida. E classificadas quanto a causas de requisitos e causas de teste.

Quanto ao método foram levantados tais causas:

- Requisito:

  - A implementação das issues na maioria das vezes é feita sem o entendimento dos desenvolvedores.
  - Falta de planejamento da experiência de usuário.
  - Falta de comunicação com o cliente.
  - Falta de planejamento da sprint.
  - Requisitos mal especificados.
  - Issues mal escritas.

- Teste:
  - Falta de cooperação por parte dos fornecedores do cliente.
  - As issues eram implementadas sem testes automatizados.
  - Faltou metodologia para os testes de usuário.

Quanto ao meio ambiente foram levantados tais causas:

- Requisito:
  - Falta de monitoramento e organização das issues na ferramenta de gestão.
  - Falta de motivação do cliente em inspecionar as issues.
  - Falta de mão de obra.

- Teste:
  - Falta de organização da empresa para a elaboração do cronograma.
  - Falta de conhecimento em operar a ferramenta de gestão por parte do cliente.
  - Falha em estimativa de tempo por parte da gestão.

Quanto a mão de obra foram levantados tais causas:

- Requisito:
  - Falta de treinamento em metodologias ágeis para a equipe.
  - Falta de reconhecimento dentro da equipe.
  - Falta de conhecimento dos requisitos por parte do cliente.
  - Falta de atribuição de responsabilidade por parte do cliente.

- Teste:
  - Falta de comunicação com os clientes e dentro da equipe.

  - Faltou tempo para a equipe fazer os testes automatizados.
  - Falta de monitoramento da sprint.

Quanto a medida foram levantados tais causas:

- Requisito:
  - Faltou atribuir pontos às issues.
- Teste:
  - Falta o plano de medição com as medidas gerenciais e medidas técnicas.

5.3.5 Atividade 5 - Classificar causa-raiz dos defeitos

As causas-raiz estão relacionadas com os seguintes processos do MPS.BR (2020):

- Gerência de Projetos – GPR.
- Engenharia de Requisitos - REQ.
- Verificação – VER.
- Validação – VAL.
- Medição – MED.
- Gerência de Recursos Humanos – GRH.
- Gerência Organizacional - ORG.

Os processos listados acima resolvem os problemas identificados na análise de causa-raiz, tais processos estão listados abaixo com seus respectivos problemas.

Os resultados analisados mostram que, os principais problemas relacionados à issues de severidade 1, são:

- ***Problema 1*** - Falta de Comunicação e engajamento do cliente para o desenvolvimento do projeto acarretou em desentendimento pela equipe, impactando no gerenciamento e desenvolvimento das funcionalidades do sistema.

  Os processos do MPS.BR (2020) que solucionam este problema estão listados na Tabela 7.

Processo	**REP**	**Descrição**
GPR	GPR9	O envolvimento das partes interessadas no projeto é planejado (a partir do nível G).
GPR	GPR13	O Plano de Projeto é revisado com todos os interessados, incluindo e o compromisso com ele é obtido (até o nível E).
GPR	GPR13+	O Plano do Projeto é revisado com todos os interessados, incluindo o tratamento de dependências críticas, e o compromisso com ele é obtido (a partir do nível D).
GPR	GPR15	O envolvimento das partes interessadas no projeto é monitorado e tratado em relação ao planejado.
GPR	GPR17	Os riscos do projeto são monitorados e seus resultados são comunicados às partes afetadas (até o nível E).
REQ	REQ1	As necessidades, expectativas e restrições das partes interessadas, tanto em relação ao produto quanto a suas interfaces, são identificadas (a partir do nível G). *NOTA: Parte interessada pode envolver cliente, gestores do produto, usuários interessados no produto, entre outros.*
REQ	REQ3	(Até nível E) Os requisitos são entendidos e analisados junto a fornecedores de requisitos.
REQ	REQ3+	(A partir do nível D) Os requisitos são entendidos e analisados para garantir que sejam claros, necessários e suficientes e para balancear as necessidades das partes interessadas com as restrições existentes.
REQ	REQ4	(Até Nível E) Os requisitos são aprovados pelos fornecedores de requisitos. *NOTA: O objetivo deste resultado é assegurar que a equipe do projeto e os fornecedores de requisitos têm um entendimento comum sobre os requisitos eliminando as ambiguidades.*
REQ	REQ4+	(A partir do Nível D) Os requisitos são validados. *NOTA: O objetivo da validação de requisitos é garantir que a solução irá funcionar como esperado no ambiente alvo. A validação deve ser realizada com pessoas afetadas pelo produto do projeto e deve confirmar que os requisitos são necessários e suficientes. A validação pode ser realizada, entre outros, com uso de protótipos, demonstrações ou revisões.*

Tabela 7 – Processos que solucionam o Problema 1. Fonte: MPS.BR (2020)

- ***Problema 2*** - Requisitos mal escritos:

  1. A equipe não entende a especificação do requisito;
  2. Falha no levantamento de requisitos, dificultando o desenvolvimento das funcionalidades.

Os processos do MPS.BR (2020) que solucionam este problema estão listados na Tabela 8.

Processo	**REP**	**Descrição**
GPR	GPR1	O escopo do trabalho para o projeto é estabelecido, mantido atualizado e utilizado (a partir do nível G)
REQ	REQ1	As necessidades, expectativas e restrições das partes interessadas, tanto em relação ao produto quanto a suas interfaces, são identificadas (a partir do nível G). *NOTA: As necessidades, expectativas e restrições das partes interessadas, tanto em relação ao produto quanto a suas interfaces, são identificadas (a partir do nível G).*
REQ	REQ2	(Até Nível E) Os requisitos são especificados, priorizados e mantidos atualizados a partir das necessidades, expectativas e restrições identificadas para o produto e suas interfaces.
REQ	REQ2+	(A partir do nível D) Os requisitos são especificados, priorizados, refinados, alocados para implementação e mantidos atualizados a partir das necessidades, expectativas e restrições identificadas, o que inclui a especificação de conceitos operacionais, cenários e interfaces internas e externas.
REQ	REQ3	(Até nível E) Os requisitos são entendidos e analisados junto a fornecedores de requisitos.
REQ	REQ3+	(A partir do nível D) Os requisitos são entendidos e analisados para garantir que sejam claros, necessários e suficientes e para balancear as necessidades das partes interessadas com as restrições existentes.
REQ	REQ 4	(Até Nível E) Os requisitos são aprovados pelos fornecedores de requisitos. *NOTA: O objetivo deste resultado é assegurar que a equipe do projeto e os fornecedores de requisitos têm um entendimento comum sobre os requisitos eliminando as ambiguidades.*
REQ	REQ4+	(A partir do Nível D) Os requisitos são validados. *NOTA: O objetivo da validação de requisitos é garantir que a solução irá funcionar como esperado no ambiente alvo. A validação deve ser realizada com pessoas afetadas pelo produto do projeto e deve confirmar que os requisitos são necessários e suficientes. A validação pode ser realizada, entre outros, com uso de protótipos, demonstrações ou revisões.*
REQ	REQ5	REQ 5 (A partir do nível G) O compromisso da equipe técnica com a implementação dos requisitos é obtido.
REQ	REQ6	REQ 6 (A partir do nível G) A rastreabilidade bidirecional entre requisitos, atividades e produtos de trabalho do projeto é estabelecida e mantida.

Processo	**REP**	**Descrição**
REQ	REQ7	REQ 7 (A partir do nível G) Os planos, atividades e produtos de trabalho relacionados são revisados visando identificar e corrigir inconsistência em relação aos requisitos.
VER	VER1	(A partir do nível G) Produtos de trabalho a serem verificados por meio de testes e de revisão por pares são selecionados. *NOTA: Considera-se "par"uma outra pessoa que tem conhecimento equivalente ou superior sobre o assunto do produto de trabalho a ser verificado.*
VER	VER2	(A partir do nível G) Procedimentos e material de apoio são definidos, mantidos atualizados e usados para preparação e realização de revisões por pares.
VER	VER3+	(A partir do Nível D) Métodos, procedimentos, critérios e ambientes para realização de testes são definidos, mantidos atualizados e usados durante as atividades de teste.
VER	VER 5+	(A partir do nível D) Os resultados das atividades de verificação são analisados, registrados e comunicados.
VAL	VAL1	(A partir do nível G) Produtos de trabalho a serem validados são selecionados.
VAL	VAL2+	(A partir do nível D) Métodos, procedimentos, critérios e ambientes para validação são definidos, mantidos atualizados e usados durante as atividades de validação para garantir que a solução atingirá seus objetivos no ambiente operacional.
VAL	VAL3	(A partir do nível G) Atividades de validação são conduzidas para garantir que o produto irá funcionar conforme especificado no ambiente operacional. *NOTA: A resolução de problemas identificados pode ser realizada fora do escopo do projeto e, se o problema não for resolvido, é necessário o registro com a justificativa.*

Tabela 8 – Processos que solucionam o Problema 2. Fonte: MPS.BR (2020)

- ***Problema 3*** - Falta de estimativas de prazo e estimativas de custos das issues.

  Os processos do MPS.BR (2020) que solucionam este problema estão listados na Tabela 9.

Processo	**REP**	**Descrição**
GPR	GPR3	(Até nível E) Estimativas de dimensão de tarefas e produtos de trabalho do projeto são estabelecidas e mantidas atualizadas. *NOTA: HH (Homem Hora) é medida de esforço e não pode ser considerada como dimensão.*
GPR	GPR3+	(A partir do nível D) Estimativas de dimensão de tarefas e produtos de trabalho do projeto são estabelecidas com a utilização de métodos apropriados e documentados, e são mantidas atualizadas. *NOTA: As estimativas de dimensão podem ser realizadas por pontos por função (FPA), Story Points, método de estimativa definido pela empresa etc.*
GPR	GPR4	(Até nível E) Estimativas de esforço, duração e custo para a execução das tarefas e dos produtos de trabalho do projeto são estabelecidas e justificadas. *NOTA: As estimativas de esforço, duração e custo devem ser derivadas das estimativas de dimensão de tarefas e produtos de trabalho. Para justificativa das estimativas, essas devem conter um racional de como foram realizadas.*
GPR	GPR5	(A partir do nível G) O orçamento e o cronograma do projeto, incluindo a definição de marcos, são estabelecidos e mantidos atualizados.
GPR	GPR14	(Até nível E) O escopo, as tarefas, as estimativas, o orçamento, o cronograma, os recursos materiais e humanos e o ambiente de trabalho são monitorados em relação ao planejado.

Tabela 9 – Processos que solucionam o Problema 3. Fonte: MPS.BR (2020)

- ***Problema 4*** - Falta de métricas para gestão do projeto.

  Os processos do MPS.BR (2020) que solucionam este problema estão listados na Tabela 10.

Processo	REP	Descrição
MED	MED2	(A partir do nível F) Medidas são identificadas e documentadas por meio de definições operacionais, a partir dos objetivos organizacionais de medição e de desempenho, seguindo os processos pertinentes, e são atualizadas quando pertinente. *NOTA: Uma definição operacional de medida deve conter pelo menos as seguintes informações: nome; objetivo relacionado; fórmula de cálculo; medidas básicas (se for medida derivada); forma, momento e responsabilidade pela coleta; forma de armazenamento; procedimento e responsabilidade pela análise.*
MED	MED3	(A partir do nível F) Medidas são coletadas, verificadas e armazenadas de acordo com as definições operacionais. *NOTA: As medidas coletadas precisam ser verificadas para garantir acurácia, precisão e completeza.*
MED	MED4	(A partir do nível F) Medidas são analisadas e os resultados da análise são armazenados de acordo com as definições operacionais. *NOTA: Nesse nível, a análise de medida é realizada em nível de projeto.*
MED	MED5	(A partir do nível F) A partir do resultado da análise das medidas, ações corretivas são realizadas visando alcançar os objetivos de desempenho estabelecidos.
MED	MED6	(A partir do nível C) A partir do resultado da análise das medidas, necessidades de melhoria no desempenho são determinadas.

Tabela 10 – Processos que solucionam o Problema 4. Fonte: MPS.BR (2020)

- ***Problema 5*** - Falta de monitoramento da sprint.

  Os processos do MPS.BR (2020) que solucionam este problema estão listados na Tabela 11.

Processo	REP	Descrição
GPR	GPR18	(A partir do nível G) Ações para corrigir desvios em relação ao planejado são identificadas, implementadas e acompanhadas até a sua conclusão.
GPR	GPR18+	(A partir do nível D) Ações para corrigir desvios em relação ao planejado e outras questões relacionadas ao projeto são identificadas, tratadas com as partes interessadas, implementadas e acompanhadas até a sua conclusão. *NOTA: Exemplos de questões a serem tratadas com as partes interessadas são: requisitos incompletos, problemas no design, ausência de recursos.*
GPR	GPR19	(Até nível E) Aspectos do projeto com resultados positivos ou negativos significativos são analisados e as causas do desvio são tratadas.
GPR	GPR19+	(A partir do nível D) Aspectos do projeto com resultados positivos ou negativos significativos são analisados e tratados em relação à causa-raiz, utilizando um procedimento organizacional e documentando os seus resultados.

Tabela 11 – Processos que solucionam o Problema 5. Fonte: MPS.BR (2020)

- ***Problema 6*** - Falta de testes automatizados e ferramentas de apoio.

  Os processos do MPS.BR (2020) que solucionam este problema estão listados na Tabela 12.

Processo	REP	Descrição
VER	VER4	(A partir do nível G) Atividades de verificação são realizadas e problemas identificados são resolvidos. *NOTA: A resolução de problemas identificados pode ser realizada fora do escopo do projeto, neste caso é necessário o registro/justificativa se o problema não for resolvido.*
VAL	VAL4+	(A partir do nível D) Os resultados da validação são analisados, registrados e comunicados.

Tabela 12 – Processos que solucionam o Problema 6. Fonte: MPS.BR (2020)

- ***Problema 7*** - Falta de metodologia de testes.

  Os processos do MPS.BR (2020) que solucionam este problema estão listados na Tabela 13.

Processo	REP	Descrição
VER	VER3	(Até Nível E) Métodos, procedimentos e ambientes para realização de testes são definidos, mantidos atualizados e usados durante as atividades de teste.
VAL	VAL2	(Até nível E) Métodos, procedimentos e o ambiente para validação são definidos, mantidos atualizados e usados durante as atividades de validação para garantir que a solução atingirá seus objetivos no ambiente operacional.

Tabela 13 – Processos que solucionam o Problema 7. Fonte: MPS.BR (2020)

- ***Problema 8*** - Falta de um plano de medição para gestão do processo de testes.

  Os processos do MPS.BR (2020) que solucionam este problema estão listados na Tabela 14.

Processo	REP	Descrição
VER	VER5	(Até nível E) Os resultados das atividades de verificação são registrados e comunicados
VAL	VAL4	(Até nível E) Os resultados da validação são registrados e comunicados.

Tabela 14 – Processos que solucionam o Problema 8. Fonte: MPS.BR (2020)

- ***Problema 9*** - Falta de conhecimento da equipe em relação a técnicas de metodologia ágil afeta o desenvolvimento e a gestão do projeto.

  Os processos do MPS.BR (2020) que solucionam este problema estão listados na Tabela 15.

Processo	REP	Descrição
GRH	GRH1	(No nível F) As necessidades de capacitação e recrutamento relacionadas aos processos e projetos são identificadas.
ORG	ORG2	(A partir do nível F) Recursos e treinamento para definição, apoio, execução, avaliação da aderência e melhoria dos processos são garantidos pela gerência da organização. *NOTA: Recursos abrangem recursos humanos, materiais e financeiros.*

Tabela 15 – Processos que solucionam o Problema 9. Fonte: MPS.BR (2020)

- ***Problema 10*** - Falta de treinamento da equipe afeta a implementação da equipe.

  Os processos do MPS.BR (2020) que solucionam este problema estão listados na Tabela 16.

Processo	REP	Descrição
GRH	GRH2	(A partir do nível F) Os treinamentos identificados como necessários para capacitação dos colaboradores são realizados e registrados.
GRH	GRH3	(A partir do nível E) A efetividade dos treinamentos é avaliada.
GPR	GPR6	(A partir do nível G) Os recursos humanos para o projeto são planejados considerando as habilidades e os conhecimentos necessários para executá-lo. *NOTA: Caso não existam recursos humanos, na organização, com as habilidades e conhecimentos necessários estes podem ser contratados ou serem realizados treinamentos para superar as deficiências.*

Tabela 16 – Processos que solucionam o Problema 10. Fonte: MPS.BR (2020)

- ***Problema 11*** - Falta de reconhecimento da equipe.

  A equipe considera que para ser reconhecido precisa que a organização:

  - Valorize as pessoas que não atrasam as entregas.
  - Receber incentivos por estar fazendo um bom trabalho e por estar fazendo a mais do que o seu trabalho exigiria.
  - De funcionários que não precisam fazer horas extras para entregar um projeto ou issue e que se o funcionário entregar issues ou tarefas com muita antecedência por veze passa despercebido pelos chefes.

  O reconhecimento teria que vir através de pessoas que trabalham com você e percebem o quão positivamente você afetou o processo produtivo/de trabalho. Eles sugerem que o reconhecimento seja feito por meio de:

  - Agradecimentos.
  - Feedback positivo, bônus e promoções.
  - Percebendo seu potencial e te ajudando a crescer profissionalmente.
  - Oportunidades de trabalhar com projetos diferentes e maiores (que eventualmente levem a pessoa do 1 ao 2).

5.4 Caracterização das Práticas de Implementação do modelo MPS.BR

Em função da análise feita anteriormente foi verificado que as práticas do modelo MPS.BR (2020) estavam definidas e implementadas no processo de desenvolvimento da organização X.

Foi utilizado a Escala de FiLIPINY do método SCAMPI (2006) para identificar o grau de definição, vista na Tabela 17, bem como o grau de implementação das atividades do processo.

Caracterização	**Sigla**	**Comentário**
Totalmente implementado (*Fully Implemented*)	FI	- Artefatos diretos e adequados; - Nenhuma deficiência na aplicação prática, pois faz tudo o que é esperado.
Largamente Implementado (*Largely Implemented*)	LI	- Artefatos diretos e adequados; - Nenhuma deficiência na aplicação prática, mas existe algum ponto fraco.
Parcialmente implementado (*Partially Implemented*)	PI	- Artefato não diretos ou são julgados inadequados; - Artefatos ou afirmação indicam que alguns aspectos da práticas estão implementados; - Uma ou mais fraquezas encontradas na aplicação prática.
Não implementado (*Not implemented*)	NI	- Artefatos não diretos e/ou julgados inadequados; - Uma ou mais fraquezas encontradas na aplicação prática.
Não implementado ainda (*Not Yet*)	NY	- O projeto não alcançou o estágio esperado no ciclo de vida.

Tabela 17 – Escala de FiLIPINY. Fonte: Adaptação do método SCAMPI (2006) e MPS.BR (2020)

5.4.1 Processo 1 - Gerência de Projetos - GPR

Propósito: *O propósito do processo Gerência de Projetos é estabelecer e manter atualizados planos que definam as atividades, recursos, riscos, prazos e responsabilidades do projeto. Também é propósito deste processo prover informações sobre o andamento do projeto que permitam a realização de correções quando houver desvios significativos no desempenho do projeto, incluindo análise de causa-raiz.* (MPS.BR, 2020)

Prática	**Grau de Definição**	**Grau de Implementação**
GPR 1 - (A partir do nível G) O escopo do trabalho para o projeto é estabelecido, mantido atualizado e utilizado.	FI	FI
GPR 2 - (Até nível E) Um processo a ser utilizado para a execução do projeto é descrito, mantido atualizado e utilizado. *NOTA: Do nível G ao Nível E, a descrição do processo para o projeto deve conter,pelo menos, o ciclo de vida do projeto e a lista de tarefas que serão executadas.*	FI	LI
GPR 2+ - (A partir do nível D) Um processo definido para o projeto, derivado da estratégia para adaptação do processo da organização, é estabelecido, mantido atualizado e utilizado.	PI	PI
GPR 3 - (Até nível E) Estimativas de dimensão de tarefas e produtos de trabalho do projeto são estabelecidas e mantidas atualizadas. *NOTA: HH (Homem Hora) é medida de esforço e não pode ser considerada como dimensão.*	LI	PI
GPR 3+ - (A partir do nível D) Estimativas de dimensão de tarefas e produtos de trabalho do projeto são estabelecidas com a utilização de métodos apropriados e documentados, e são mantidas atualizadas. *NOTA: As estimativas de dimensão podem ser realizadas por pontos por função (FPA), Story Points, método de estimativa definido pela empresa etc.*	PI	PI
GPR 4 - (Até nível E) Estimativas de esforço, duração e custo para a execução das tarefas e dos produtos de trabalho do projeto são estabelecidas e justificadas. *NOTA: As estimativas de esforço, duração e custo devem ser derivadas das estimativas de dimensão de tarefas e produtos de trabalho. Para justificativa das estimativas, essas devem conter um racional de como foram realizadas.*	NI	PI

Prática	**Grau de Definição**	**Grau de Implementação**
GPR 4+ - (A partir do nível D) Estimativas de esforço, duração e custo para a execução das tarefas e dos produtos de trabalho do projeto são estabelecidas utilizando métodos apropriados, baseadas no repositório organizacional de medidas e no conjunto de ativos de processo organizacional. *NOTA: As estimativas devem ser derivadas das estimativas de dimensão de tarefas e produtos de trabalho.*	NI	PI
GPR 5 - (A partir do nível G) O orçamento e o cronograma do projeto, incluindo a definição de marcos, são estabelecidos e mantidos atualizados.	PI	PI
GPR 6 - (A partir do nível G) Os recursos humanos para o projeto são planejados considerando as habilidades e os conhecimentos necessários para executá-lo. *NOTA: Caso não existam recursos humanos, na organização, com as habilidades e conhecimentos necessários estes podem ser contratados ou serem realizados treinamentos para superar as deficiências.*	PI	PI
GPR 7 - (Até nível E) Os recursos e o ambiente de trabalho necessários para executar o projeto são planejados e mantidos atualizados.	FI	FI
GPR 7+ - (A partir do nível D) Os recursos e o ambiente de trabalho necessários para executar o projeto são planejados a partir dos ambientes padrão de trabalho da organização, e são mantidos atualizados.	FI	FI
GPR 12 - (Até nível E) Um plano geral para a execução do projeto é estabelecido com a integração consistente dos planejamentos realizados, e é mantido atualizado.	NI	PI
GPR 12+ - (A partir do nível D) Um plano geral para a execução do projeto é estabelecido a partir do processo do projeto, dos ativos de processo e do repositório de medidas, e é mantido atualizado.	NI	PI

Tabela 18 – Processo 1 - Gerência de Projetos. Fonte: MPS.BR (2020)

5.4.2 Processo 2 - Engenharia de Requisitos - REQ

Propósito: *O propósito do processo Engenharia de Requisitos é definir, gerenciar e manter atualizado os requisitos das partes interessadas e do produto, garantindo que inconsistências entre os requisitos, os planos e os produtos de trabalho sejam identificadas.* (MPS.BR, 2020)

Prática	**Grau de Definição**	**Grau de Implementação**
REQ 1 - (A partir do nível G) As necessidades, expectativas e restrições das partes interessadas, tanto em relação ao produto quanto a suas interfaces, são identificadas. *NOTA: Parte interessada pode envolver cliente, gestores do produto, usuários interessados no produto, entre outros.*	FI	LI
REQ 2 - (Até Nível E) Os requisitos são especificados, priorizados e mantidos atualizados a partir das necessidades, expectativas e restrições identificadas para o produto e suas interfaces.	FI	FI
REQ 2 + - (A partir do nível D) Os requisitos são especificados, priorizados, refinados, alocados para implementação e mantidos atualizados a partir das necessidades, expectativas e restrições identificadas, o que inclui a especificação de conceitos operacionais, cenários e interfaces internas e externas.	FI	LI
REQ 3 - (Até nível E) Os requisitos são entendidos e analisados junto a fornecedores de requisitos.	NI	PI
REQ 3+ - (A partir do nível D) Os requisitos são entendidos e analisados para garantir que sejam claros, necessários e suficientes e para balancear as necessidades das partes interessadas com as restrições existentes.	NI	PI
REQ 4 - (Até Nível E) Os requisitos são aprovados pelos fornecedores de requisitos. *NOTA: A objetivo deste resultado é assegurar que a equipe do projeto e os fornecedores de requisitos têm um entendimento comum sobre os requisitos eliminando as ambiguidades.*	NI	PI
REQ 4 + - (A partir do Nível D) Os requisitos são validados. *NOTA: O objetivo da validação de requisitos é garantir que a solução irá funcionar como esperado no ambiente alvo. A validação deve ser realizada com pessoas afetadas pelo produto do projeto e deve confirmar que os requisitos são necessários e suficientes. A validação pode ser realizada, entre outros, com uso de protótipos, demonstrações ou revisões.*	FI	FI
REQ 5 - (A partir do nível G) O compromisso da equipe técnica com a implementação dos requisitos é obtido.	PI	FI
REQ 7 - (A partir do nível G) Os planos, atividades e produtos de trabalho relacionados são revisados visando identificar e corrigir inconsistência em relação aos requisitos.	PI	FI

Tabela 19 – Processo 2 - Engenharia de Requisitos. Fonte: MPS.BR (2020)

5.4.3 Processo 3 - Gerência de Recursos Humanos – GRH

Propósito: *O propósito do processo Gerência de Recursos Humanos é prover a organização e os projetos com os recursos humanos necessários e manter suas competências adequadas às necessidades do negócio.* (MPS.BR, 2020)

Prática	**Grau de Definição**	**Grau de Implementação**
GRH 1 - (No nível F) As necessidades de capacitação e recrutamento relacionadas aos processos e projetos são identificadas.	FI	FI
GRH 1+ - (A partir do nível E) As necessidades de capacitação e recrutamento, do ponto de vista estratégico e de curto prazo, relacionadas aos processos e projetos são identificadas e planos para seu atendimento são definidos, seguidos e mantidos atualizados.PI	PI	PI
GRH 2 - (A partir do nível F) Os treinamentos identificados como necessários para capacitação dos colaboradores são realizados e registrados.	PI	LI
GRH 3 - (A partir do nível E) A efetividade dos treinamentos é avaliada.	LI	LI
GRH 4 - (A partir do nível E) A partir da análise dos registros e avaliações da efetividade dos treinamentos, as habilidades dos instrutores e os recursos para treinamento dos colaboradores são desenvolvidos e aprimorados. *NOTA: As habilidades estão relacionadas à capacitação dos instrutores. Os recursos estão relacionados ao desenvolvimento, manutenção e disponibilização de materiais de apoio e da infraestrutura necessária.*	LI	LI

Tabela 20 – Processo 3 - Gerência de Recursos Humanos. Fonte: MPS.BR (2020)

5.4.4 Processo 4 - Verificação

Propósito: *O propósito do processo Verificação é confirmar que os produtos de trabalho selecionados atendem aos requisitos especificados pela execução de testes e revisão por pares.* (MPS.BR, 2020)

Prática	**Grau de Definição**	**Grau de Implementação**
VER 1 - (A partir do nível G) Produtos de trabalho a serem verificados por meio de testes e de revisão por pares são selecionados. *NOTA: Considera-se "par"uma outra pessoa que tem conhecimento equivalente ou superior sobre o assunto do produto de trabalho a ser verificado.*	FI	LI
VER 2 - (A partir do nível G) Procedimentos e material de apoio são definidos, mantidos atualizados e usados para preparação e realização de revisões por pares.	FI	LI
VER 3 - (Até Nível E) Métodos, procedimentos e ambientes para realização de testes são definidos, mantidos atualizados e usados durante as atividades de teste.	LI	LI
VER3+ - (A partir do Nível D) Métodos, procedimentos, critérios e ambientes para realização de testes são definidos, mantidos atualizados e usados durante as atividades de teste.	LI	LI
VER 4 - (A partir do nível G) Atividades de verificação são realizadas e problemas identificados são resolvidos. *NOTA: A resolução de problemas identificados pode ser realizada fora do escopo do projeto, neste caso é necessário o registro/justificativa se o problema não for resolvido.*	NI	LI
VER 5 - (Até nível E) Os resultados das atividades de verificação são registrados e comunicados.	LI	PI
VER 5+ - (A partir do nível D) Os resultados das atividades de verificação são analisados, registrados e comunicados.	LI	PI

Tabela 21 – Processo 4 - Verificação. Fonte: MPS.BR (2020)

5.4.5 Processo 5 - Validação

Propósito: *O propósito do processo Validação é confirmar que um produto ou componente do produto atenderá a seu uso pretendido quando colocado no ambiente operacional.* (MPS.BR, 2020)

Prática	**Grau de Definição**	**Grau de Implementação**
VAL 1 - (A partir do nível G) Produtos de trabalho a serem validados são selecionados.	FI	FI
VAL 2 - (Até nível E) Métodos, procedimentos e o ambiente para validação são definidos, mantidos atualizados e usados durante as atividades de validação para garantir que a solução atingirá seus objetivos no ambiente operacional.	LI	LI
VAL 2+ - (A partir do nível D) Métodos, procedimentos, critérios e ambientes para validação são definidos, mantidos atualizados e usados durante as atividades de validação para garantir que a solução atingirá seus objetivos no ambiente operacional.	LI	LI
VAL 3 - (A partir do nível G) Atividades de validação são conduzidas para garantir que o produto irá funcionar conforme especificado no ambiente operacional. *NOTA: A resolução de problemas identificados pode ser realizada fora do escopo do projeto e, se o problema não for resolvido, é necessário o registro com a justificativa.*	FI	FI
VAL 4 - (Até nível E) Os resultados da validação são registrados e comunicados.	LI	LI
VAL 4+ - (A partir do nível D) Os resultados da validação são analisados, registrados e comunicados.	LI	LI

Tabela 22 – Processo 5 - Validação. Fonte: MPS.BR (2020)

5.5 ETAPA 3 - Estabelecimento

Durante a fase de estabelecimento, ocorre a priorização das atividades de melhoria e se criam estratégias para buscar as soluções. O esboço do plano de ação da melhoria de processo de software será concluído de acordo com a visão da organização, plano estratégico de negócios, principais problemas de negócios enfrentados pela organização X e metas de longo prazo.

Durante a fase de estabelecimento, metas mensuráveis são desenvolvidas a partir dos objetivos gerais definidos na fase de iniciação.

5.5.1 Atividade 1 - Definir ações de melhoria

Esses serão os principais pontos que iremos atacar no plano de melhoria de processo da organização X.

5.5.1.1 Processo 1 - Gerência de Projetos - GPR

1. Planejar o gerenciamento dos custos

   Descrição: Baseia-a em, estruturar documentos orçamentários dos projetos.

2. Pontuar as issues

   Descrição: Na atividade de Planejar Sprint a equipe estima os pontos de esforço dá issue.

3. Planejamento de riscos dos projetos

   Descrição: O gerenciamento de riscos é essencial para a comunicação entre as partes interessadas e para a obtenção de acordos, dentre outros benefícios para uma boa gestão da equipe.

4. Definição de um plano de projeto

   Descrição: O plano de projeto conta com informações fundamentais para o bom desempenho do projeto, como o cronograma, estudo inicial de custos e o escopo do sistema.

5.5.1.2 Processo 2 - Engenharia de Requisitos - REQ

1. Refinamento de requisitos

   Descrição: Deve ser feito internamente na equipe, como definição de escopo e de viabilidade do projeto. Especificar tarefa de viabilidade do projeto. A ausência de uma tarefa de viabilidade do projeto prejudica a definição de um escopo de maior valor.

2. Definir melhor a granularidade dos requisitos

   Descrição: O maior detalhamento dos requisitos permite uma melhor compreensão dos mesmos.

3. Definir prioridade dos requisitos

   Descrição: Indicar quais funcionalidades devem ser priorizadas.

5.5.1.3 Processo 3 - Gerência de Recursos Humanos – GRH

1. Elaborar um plano de carreira

   Descrição: Para que o colaborador veja se a expectativa dentro da empresa o atende e que se sinta motivado a fazer um bom trabalho, visto que, ele pode estipular o tempo determinado para alcançar os objetivos na organização.

5.5.1.4 Processo 4 - Verificação – VER

1. Estabelecer um "padrão" de escrita de issues

   Descrição: Tornar a escrita da issue padrão e detalhada para o entendimento dos colaboradores.

5.5.1.5 Processo 5 - Validação – VAL

1. Testes automatizados

   Descrição: Efetuar os testes automatizados nos sistemas e torná-los obrigatório para o aceite da funcionalidade.

2. Teste de usabilidade

   Descrição: Realizar testes de usabilidade com diversos clientes e usuários, antes do sistema entrar em produção.

5.5.2 Atividade 2 - Priorização de melhorias

A priorização será feita de acordo com a classificação dos pontos de melhorias assim com o seu nível de importância, para a realização do processo na organização X. A classificação será feita de acordo com a Tabela 23:

Caracterização	**Comentário**
Alta	Os pontos de melhoria serão executados imediatamente.
Média	Seria indispensável que os pontos de melhoria fossem implementados, porém não impediria a realização do processo na organização X.
Baixa	Seria bom que os pontos de melhoria fossem implementados, mas com não de imediato.

Tabela 23 – Priorização de Melhorias. Fonte: Autora

5.5.2.1 Processo 1 - Gerência de Projetos - GPR

1. Planejar o gerenciamento dos custos

   Priorização: Alta.

2. Pontuar as issues

   Priorização: Alta.

3. Planejamento de riscos dos projetos

   Priorização: Baixa.

4. Definição de um plano de projeto

   Priorização: Média.

5.5.2.2 Processo 2 - Engenharia de Requisitos - REQ

1. Refinamento de requisitos

   Priorização: Alta.

2. Definir melhor a granularidade dos requisitos

   Priorização: Média.

3. Definir prioridade dos requisitos

   Priorização: Alta.

5.5.2.3 Processo 3 - Gerência de Recursos Humanos – GRH

1. Elaborar um plano de carreira

   Priorização: Média.

5.5.2.4 Processo 4 - Verificação – VER

1. Estabelecer um "padrão" de escrita de issues

   Priorização: Média.

5.5.2.5 Processo 5 - Validação – VAL

1. Testes automatizados

   Priorização: Alta.

2. Teste de usabilidade

   Priorização: Média.

5.5.3 Atividade 3 - Elaborar o planejamento do 1º ciclo do programa de melhoria de processos de software

O planejamento do primeiro ciclo do programa de melhoria de processo de software, será feito com base nas priorização feita na atividade anterior.

As melhorias que foram incluídas, para facilitar o desenvolvimento desse 1º ciclo do programa são:

- Planejar o gerenciamento dos custos.
- Pontuar as issues.
- Refinar de requisitos.
- Definir melhor a granularidade dos requisitos.
- Definir prioridade dos requisitos.
- Testes automatizados.
- Teste de usabilidade.

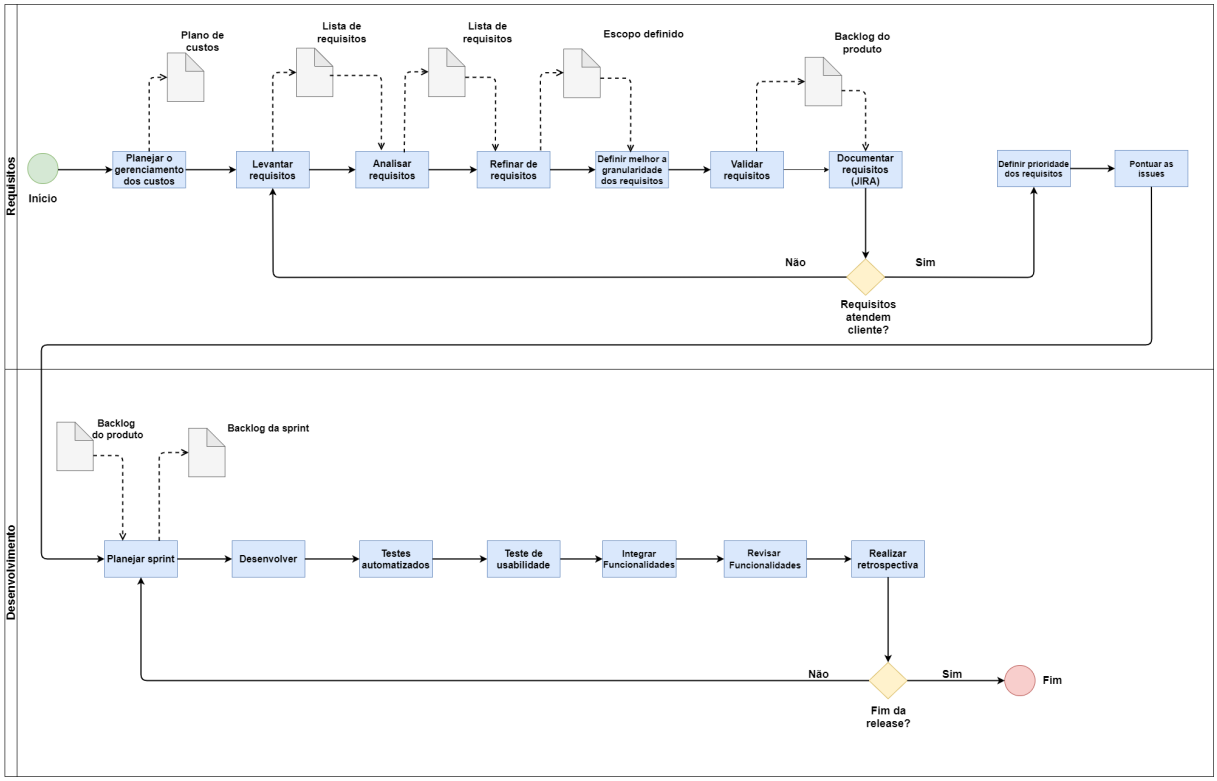

Figura 15 – Planejamento do 1º ciclo do programa de melhoria de processos de software. Fonte: Autora.

- Processo de requisito:
  - Planejar o gerenciamento dos custos.
  - Levantar requisitos.
  - Analisar requisitos.
  - Refinar de requisitos.
  - Definir melhor a granularidade dos requisitos.

  - Validar requisitos.
  - Documentar requisitos (Jira).
  - Definir prioridade dos requisitos.
  - Pontuar as issues.

- Processo de desenvolvimento:
  - Planejar sprint.
  - Desenvolver.
  - Testes automatizados.
  - Teste de usabilidade.
  - Integrar funcionalidades.
  - Revisar funcionalidades.
  - Realizar retrospectiva.

5.5.4 Atividade 4 - Estabelecer as alternativas de solução técnica

Esta atividade ficará a critério da organização X e será de inteira responsabilidade da mesma.

5.5.5 Atividade 5 - Elaborar o plano de medição

No plano de medição, é onde será analisada e interpretada as informações coletadas no 1º ciclo do programa de melhoria, gerando um relatório de apontamentos e os benefícios das melhorias.

5.5.5.1 Objetivo de medição - Escopo

Aspecto geral do diagrama do GQM de escopo, sendo mostrado na Figura 16, considerando o objetivo de medição, as suas respectivas questões e seus correspondentes indicadores e métricas.

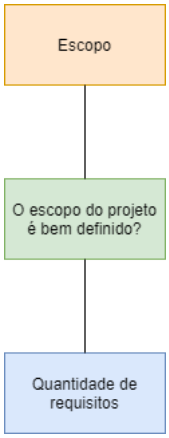

Figura 16 – GQM de Escopo. Fonte: Autora.

Na tabela 24, está sendo exposto o objetivo de medição do escopo.

Objetivo:	Definição de escopo.
Propósito:	Melhorar.
Foco de qualidade:	Melhor definição.
Ponto de vista:	Equipe de desenvolvimento.
Contexto:	Organização X.

Tabela 24 – Objetivo de medição - Escopo. Fonte: Autora

- **Q1.1 Como é definido o escopo do projeto?**

  O cliente é o principal ator na definição do escopo do projeto, cabe o gerente de projetos guia-lo da melhor forma possível, o cliente deve sempre expressar de forma clara os seus anseios e expectativas quanto ao projeto.

  Métricas e Indicadores

  Em relação ao escopo:

- **Quantidade de requisitos** - É aguardado que o número de requisitos definido no início seja igual ou próximo do número de requisitos implementados ao final do projeto.

5.5.5.2 Objetivo de medição - Tempo

Aspecto geral do diagrama do GQM de tempo, sendo mostrado na Figura 17, considerando o objetivo de medição, as suas respectivas questões e seus correspondentes indicadores e métricas.

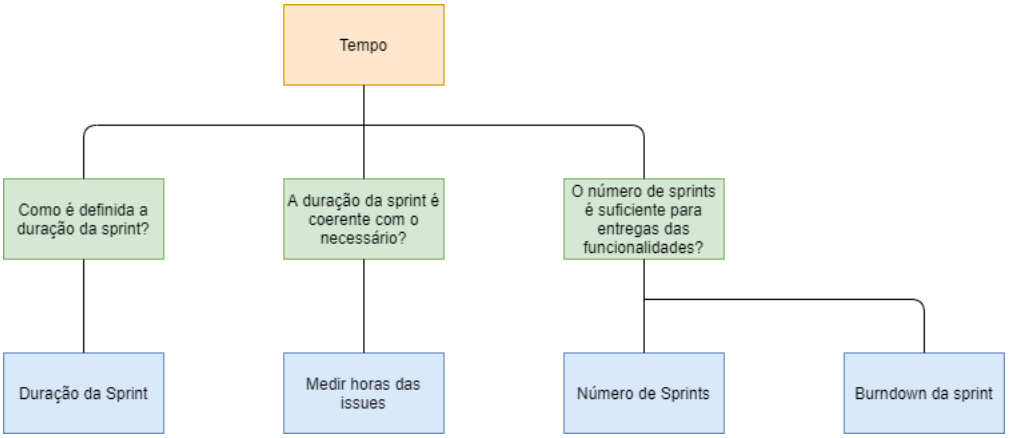

Figura 17 – GQM de Tempo. Fonte: Autora

Na tabela 25, está sendo exposto o objetivo de medição do prazo de tempo.

Objetivo:	A estimativa de tempo.
Propósito:	Melhorar.
Foco de qualidade:	Aproximação da estimativa.
Ponto de vista:	Equipe de desenvolvimento.
Contexto:	Organização X.

Tabela 25 – Objetivo de medição - Tempo. Fonte: Autora

- **Q2.1 Como é definida a duração da sprint?**

  A duração de cada sprint depende de maturidade da equipe de desenvolvimento.

- **Q2.2 A duração da sprint é coerente com o necessário?**

  O tempo de duração de cada sprint normalmente é de 1 à 4 semanas. Depende da maturidade da equipe do projeto.

- **Q2.3 O número de sprints é suficiente para entregas das funcionalidades?**
  O número de sprint terá que ser compatível para a entrega de todas as funcionalidade, caso contrário deverá replanejar o cronograma.

Métricas e Indicadores

Em relação ao tempo:

- **Duração da Sprint** - É esperado que a duração de cada sprint permita que as entregas sejam feitas com qualidade e dentro do prazo planejado. Se a duração da sprint influenciar no desenvolvimento do projeto, o gerente de projeto deve conversar com a equipe de desenvolvimento a fim de alterar a duração caso necessário e sanar o problema.

- **Medir horas das issues (Quantas horas são necessárias para concluir uma issue.)** - É esperado que o número horas necessárias para cada ponto de esforço auxilie a fazer um planejamento melhor da sprint. Se o número de horas necessário para 1 ponto de esforço da issue for muito alta, aquela sprint deverá ter poucos pontos a serem desenvolvidos, possibilitando assim que a sprint seja entregue como planejado.
- **Número de Sprints** - É esperado que o número de sprints permita que o projeto seja entregue no prazo estipulado. Se o número de sprints não for satisfatório para a entrega de um produto de qualidade ao final do projeto, o gerente de projeto deverá entrar em contato com a equipe de desenvolvimento para replanejar o cronograma do projeto.
- **Velocidade média da equipe (Burndown da sprint)** - o gráfico Burndown, indica no eixo y os pontos da sprint e no eixo x os dias subsequentes da semana, o que facilita o inspeção e a execução da sprint.

5.5.5.3 Objetivo de medição - Produtividade

Aspecto geral do diagrama do GQM de produtividade, sendo mostrado na Figura 18, considerando o objetivo de medição, as suas respectivas questões e seus correspondentes indicadores e métricas.

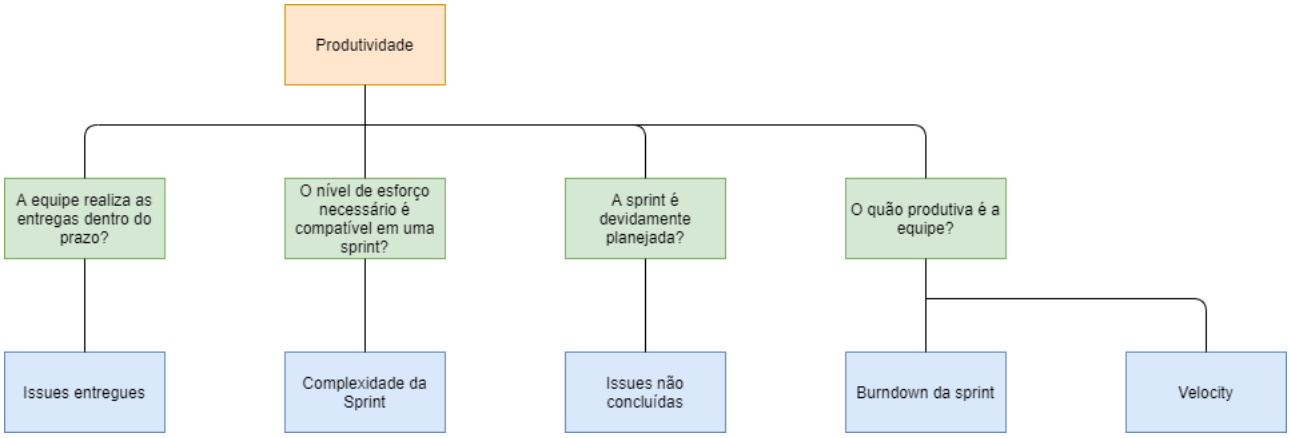

Figura 18 – GQM de Produtividade. Fonte: Autora

Na tabela 26, está sendo exposto o objetivo de medição da produtividade.

Objetivo:	Medir a produtividade da equipe.
Propósito:	Aumentar.
Foco de qualidade:	Eficiência da equipe.
Ponto de vista:	Equipe de desenvolvimento.
Contexto:	Organização X.

Tabela 26 – Objetivo de medição - Produtividade. Fonte: Autora

- **Q3.1 A equipe realiza as entregas dentro do prazo?**

  Se o número de histórias entregues é compatível ao planejado, caso contrário as entregas deverão ser analisadas e posteriores deverão ser replanejadas e as issues repontuadas.

- **Q3.2 O nível de esforço necessário é compatível em uma sprint?**

  O planejamento deve ser realizado de acordo com a produtividade da equipe. Se a equipe não entregar todos os pontos da sprint, os problemas, serão analisados os possíveis problemas para que possamos atacá-los.

- **Q3.3 A sprint é devidamente planejada?**

  Se for identificado que a equipe fez as entregas antes do término da sprint, ou não conseguiu entregar todas as issues, será indicado os possíveis problemas para que sejam analisados e acometidos.

- **Q3.4 O quão produtiva é a equipe?**

  Para que se possa melhorar a produtividade da equipe, é fundamental, em primeiro lugar, entender sua produtividade, e fazer planejamentos de acordo com esta.

Métricas e Indicadores

Em relação a produtividade:

- **Issues entregues** - É esperado que o número de issues planejadas sejam entregues na sprint. Caso contrário o gerente do projeto deverá analisar o motivo e replanejar a próxima sprint.

- **Complexidade da Sprint** - Esperamos que a complexidade seja levada em consideração no planejamento da sprint do projeto. Se a complexidade da Sprint for muito alta e isso impactar na entrega do produto, a Sprint deverá ser replanejada a fim de diminuir a sua complexidade.

- **Issues não concluídas (transferidas para outra sprint)** - É desejável que a entrega esteja sempre de acordo com o planejado. Se a quantidade de pontos

entregues for diferente do planejado, a sprints deverá ser replanejadas e se necessário as issues deverão ser repontuadas.

- **Burndown da sprint** - o gráfico Burndown, indica no eixo y os pontos da sprint e no eixo x os dias subsequentes da semana, o que facilita o inspeção e a execução da sprint.
- **Velocity** - é uma média de pontos que foram entregues pelo time na sprint. Isso auxilia no planejamento da próxima sprint, ajuda na estimativa dos pontos.
- **Pontos** - os pontos são usados para avaliar o esforço da issue que será entregue. Cada issue será pontuada, usando a sequência de fibonacci: 1, 2, 3, 5, 8, 11, 13 e 21.

5.6 Etapa 4 - AÇÃO

Aqui será aplicada a definição da solução técnica associada à prevenção de defeitos e execução do plano de ações. Tal etapa ficará a critério da organização X e será de inteira responsabilidade da mesma.

5.7 Etapa 5 - LIÇÃO

Esta etapa ficará a critério da organização X e será de inteira responsabilidade da mesma.

6 Considerações Finais

Com o intuito de minimizar os problemas a indústria de software, têm geralmente adotado metodologias para o desenvolvimento de software. Todavia, os paradigmas metodológicos para desenvolvimento de software têm sido considerados somente em termos de "um método" de análise, projeto, implementação, isto é, um conjunto de conceitos, técnicas e notações. Essa visão elimina os aspectos tecnológicos, contextuais e organizacionais que potencialmente existem dentro de um processo de software. Os ambientes tradicionais das indústrias de software geralmente suportam apenas a engenharia do produto, assumindo um processo implícito e tendo como foco principal o produto. Essa visão tem limitado a indústria de software no que diz respeito à tomada de decisões, ao estabelecimento e arquivamento de metas organizacionais, à determinação de pontos para melhoria, à estipulação de prazos para entrega de produtos e à obtenção de uma certificação.

Um projeto de software pode ser visto como uma coleção de atividades que criam um resultado identificável de valor. Portanto, o desenvolvimento de software com metodologia ágil consiste em representar uma nova abordagem para planejar e gerenciar projetos de software. Ele coloca menos ênfase em planos iniciais e em controle rigoroso, desse modo, depende mais da colaboração informal da coordenação e da aprendizagem.

Como motivação para este trabalho, foi usado a seguinte pergunta:

"Como identificar oportunidades de melhoria de processo de software a partir da análise das Issues cadastradas no sistema de administração de issues da empresa?"

Foi definido o seguinte objetivo geral:

"O objetivo geral deste trabalho: é propor melhorias de processo software a partir da análise das issues cadastradas no sistema de administração de issues da empresa."

Os cinco objetivos específicos foi elaborado com o propósito de atender ao presente objetivo geral.

Em conformidade com o primeiro objetivo específico *"Identificar e mapear o processo atual de desenvolvimento de software da empresa"*, foi mapeado o processo atual da organização X, apresentado na Capítulo 4. A organização X faz uso de algumas técnicas do o método *Scrum*, foi feita uma descrição detalhada de como funciona a execução das etapas do processo na organização, que pode ser vista na seção 4.1 deste trabalho.

Em conformidade com o segundo objetivo específico *"Identificar e mapear os problemas que têm ocorrido na execução do processo atual de desenvolvimento de software da empresa"*, para tal objetivo foi feito um *brainstorming* com alguns membros da equipe de desenvolvimento para assim obter as causas-raiz dos problemas ocorrido no processo

atual, os resultados do *brainstorming*, nos deu um diagrama de *Ishikawa* que pode ser visualizado na seção 5.3.4.

Em conformidade com o terceiro objetivo específico *"Analisar os problemas registrados como issues"*, para isso, foi realizado uma análise e classificação de *issues* que estão na ferramenta Jira Software da organização X, as classificações das *issues* podem ser vistas na seção 5.3.2.

Em conformidade com o quarto objetivo específico *"Identificar necessidades de melhorias dos processos de software"*, após a análise dos problemas recorrentes ocorreu o planejamento da prevenção de defeitos, logo após foi feito o levantamento e identificação de pontos melhoria, com base na execução da prevenção de defeitos, todo o processo de levantamento e execução dos pontos de melhorias podem ser vistos na seção 5.5.1.

Em conformidade com o quinto objetivo específico *"Estabelecer uma estratégia para planejamento de programa de melhoria de processo de software"*, conta com a análise de *issues* e o levantamento de defeitos recorrentes para assim poder apontar as causas-raiz dos defeitos. Com isso, ocorrerá a execução de prevenção de defeitos, para posteriormente ser feita uma análise dos dados obtidos, toda a proposta de melhoria está descrita Capítulo 4 e o planejamento do 1º ciclo de melhoria de software pode se visto na seção 5.5.3.

Por fim, não existe a ferramenta totalmente capaz de solucionar todos os problemas. Caberá aos profissionais a arte de combiná-las, criando novas abordagens e possibilidades. A qualidade depende muito mais de pessoas comprometidas com o desenvolvimento de todas as suas potencialidades, do que de um conjunto de técnicas.

Foi realizado um estudo aprofundado sobre o quais seriam os pontos de melhoria a para a organização X, visto a importância dos problemas levantados no *brainstorm* realizado pelos membros da equipe de desenvolvimento. Resolvemos atacar os problemas que precisam ser sanados imediatamente. Utilizamos o modelo MPS-BR como base para a melhoria, indicamos quais o processos do MPS-BR poderiam solucionar os problemas encontrados.

Foi elaborado e estabelecido um plano de medição, com as métricas necessárias para a comprovação ou não da melhoria. Assim que acabar o 1º ciclo, será feito uma análise dos dados pela organização X, que poderá explorar e avaliar se as ações de melhoria é adequado para a organização em questão.

Referências

ATLASSIAN. *What are issue types?*. [S.l.], 2020. Disponível em: <https://support.atlassian.com/jira-cloud-administration/docs/what-are-issue-types/>. Citado na página 53.

BASILI, V.; CALDEIRA, G.; ROMBACH, H. D. The goal question metric approach. 1994. Citado 2 vezes nas páginas 33 e 34.

BAZZANA, G.; ANDERSEN, O.; JOKELA, T. Is0 9126 and is0 9000: Friends or foes? 1993. Citado na página 18.

BECK, K. Extreme programming explained. Massachusetts: Addison-Wesley, 2000. Citado na página 30.

BEHKAMAL, B.; KAHANI, M.; AKBARI, M. K. Customizing iso 9126 quality model for evaluation of b2b applications. In: *Information and Software Technology*. [S.l.: s.n.], 2009. p. 599–609. Citado 2 vezes nas páginas 15 e 22.

BERGIN, J. et al. Teaching software development methods: the case of extreme programming. In: *Revista T.I.S. - Tecnologia, Infraestrutura e Software*. [S.l.: s.n.], 2004. v. 36, p. 448–449. Citado na página 29.

CAMPOS, V. F. Gerenciamento da rotina do trabalho do dia-a-dia. Minas Gerais: INDG, 2004. Citado 2 vezes nas páginas 15 e 31.

CHIAMULERA, F. et al. Ferramentas de gestão da qualidade no processo de regularização fundiária de um Órgão público federal no estado do amazonas. In: *Revista Científica Multidisciplinar Núcleo do Conhecimento*. [S.l.: s.n.], 2017. v. 01, p. 542–565. Citado 2 vezes nas páginas 30 e 31.

CMMI-DEV. *Melhorando processos para obter melhores produtos. CMMI para Desenvolvimento, Versão 1.2*. [S.l.], 2006. Citado 4 vezes nas páginas 23, 24, 26 e 27.

DIMAGGIO, L. Reflections on scheduling software tests, the project life cycle, and the last minute bug. In: *SQA Magazine*. [S.l.: s.n.], 1998. Citado na página 28.

D'OLIVEIRA, F. Planejamento de testes na homologaÇÃo de software: Uma abordagem orientada ao cliente. Brasília, 2003. Citado na página 28.

GENCEL, C. et al. A decision support framework for metrics selection in goal-based measurement programs: Gqm-dsfms. In: *The Journal of Systems and Software*. [S.l.: s.n.], 2013. p. 3091–3108. Citado 2 vezes nas páginas 32 e 33.

GUERRA, A. C.; COLOMBO, R. M. T. Tecnologia da informação: qualidade de produto de software. Brasília: PBQP, 2009. Citado 6 vezes nas páginas 16, 18, 19, 20, 21 e 23.

HIGHSMITH, J. Agile software development ecosystems. Boston: Addison Wesley, 2002. Citado na página 30.

HUANG, F.; LIU, B. Software defect prevention based on human error theories. In: *Chinese Journal of Aeronautics*. [S.l.: s.n.], 2017. p. 1054–1070. Citado 2 vezes nas páginas 24 e 25.

ISO/IEC. *Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) – Measurement of quality in use.* [S.l.], 2012. Citado 2 vezes nas páginas 19 e 20.

KOSLOSKI, R. A. D. Melhoria contínua de estimativa de esforço para o desenvolvimento de software: Uma abordagem sobre produtividade. UNIVERSIDADE CATÓLICA DE BRASÍLIA, 2005. Citado na página 31.

LAWANNA, A. Test case based selection for the process of software maintenance. In: *Department Of Information Technology, Faculty Of Science And Technology*. Assumption University, Bangkok, Thailand.: [s.n.], 2013. Citado na página 28.

LEITE, L. M.; LUCRÉDIO, D. Desenvolvimento de software utilizando o framework scrum: um estudo de caso. In: *Revista T.I.S. - Tecnologia, Infraestrutura e Software*. [S.l.: s.n.], 2014. v. 03, p. 112–122. Citado na página 30.

LOPEZ, R. E. Agile software development applied to the management of business projects. Proceedings of the 2015 IEEE 35th Central American and Panama Convention , CONCAPAN, 2016. Citado 2 vezes nas páginas 29 e 30.

MARTINS, R.; BASTINI, J. A. D. Diagrama de ishikawa. blog da qualidade. 2012. Disponível em: <https://blogdaqualidade.com.br/diagrama-de-ishikawa/>. Citado na página 26.

MARTINS, R.; BASTINI, J. A. D. Diagrama de pareto. blog da qualidade. 2012. Disponível em: <https://blogdaqualidade.com.br/diagrama-de-pareto/>. Citado na página 25.

MCFEELEY, B. Ideal, a users guide for software process improvement. In: *CMU/SEI*. [S.l.: s.n.], 1996. Citado 4 vezes nas páginas 9, 31, 32 e 45.

MPS.BR. *Guia Geral MPS de Software. MPS.BR - Melhoria de Processo do Software Brasileiro*. [S.l.], 2016. Citado na página 15.

MPS.BR. *Guia Geral MPS de Software. MPS.BR - Melhoria de Processo do Software Brasileiro*. [S.l.], 2020. Citado 16 vezes nas páginas 10, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 71, 72, 73, 74 e 75.

PUTRA, I.; YULIAWATI, A.; MURSANTO, P. Industrial extreme programming practice's implementation in rational unified process on agile development theme. In: *International Conference on Advanced Computer Science and Information Systems, ICACSIS*. [S.l.: s.n.], 2012. p. 137–142. Citado na página 30.

RISING, L.; JANOFF, N. The scrum software development process for small teams. In: *IEEE*. [S.l.: s.n.], 2000. v. 17, p. 26–32. Citado na página 30.

SADIKIN, M. Data forecasting to assist software verification validation strategy preparation. In: *International Conference on Computer Information Science (ICCIS)*. [S.l.: s.n.], 2012. Citado na página 28.

SCAMPI. *Standard CMMI Appraisal Method for Process Improvement (SCAMPI), Versão 1.2*. [S.l.], 2006. Citado 2 vezes nas páginas 10 e 69.

SCHALKEN, J.; VLIET, H. V. Measuring where it matters: determining starting points for metrics collection. In: *Journal of Systems and Software*. [S.l.: s.n.], 2008. p. 603–615. Citado na página 33.

SCHWABER, K.; SUTTHERLAND, J. The scrum guide. 2015. Disponível em: <http://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-US.pdf#zoom=100>. Citado na página 30.

SILVA, S. R.; MEDEIROS, J. T. O ciclo pdca como ferramenta para alcanÇar a eficiÊncia e eficÁcia na gestÃo da manutenÇÃo. In: *XI CONGRESSO NACIONAL DE EXCELÊNCIA EM GESTÃO*. [S.l.: s.n.], 2015. Citado na página 31.

SOUZA R. T.; ZORZO, S. D. . D. S. D. A. D. Evaluating capstone project through flexible and collaborative use of scrum framework. 2015. Citado 2 vezes nas páginas 29 e 30.

VLAANDEREN, K. et al. The agile requirements refinery: Applying scrum principles to software product management. In: *Information and Software Technology*. [S.l.: s.n.], 2011. v. 53, p. 58–70. Citado na página 29.

WERKEMAN, M. C. C. Ferramentas estatísticas básicas para o gerenciamento de processos. Belo Horizonte: Fundação Christiano Ottoni, 1995. Citado na página 30.