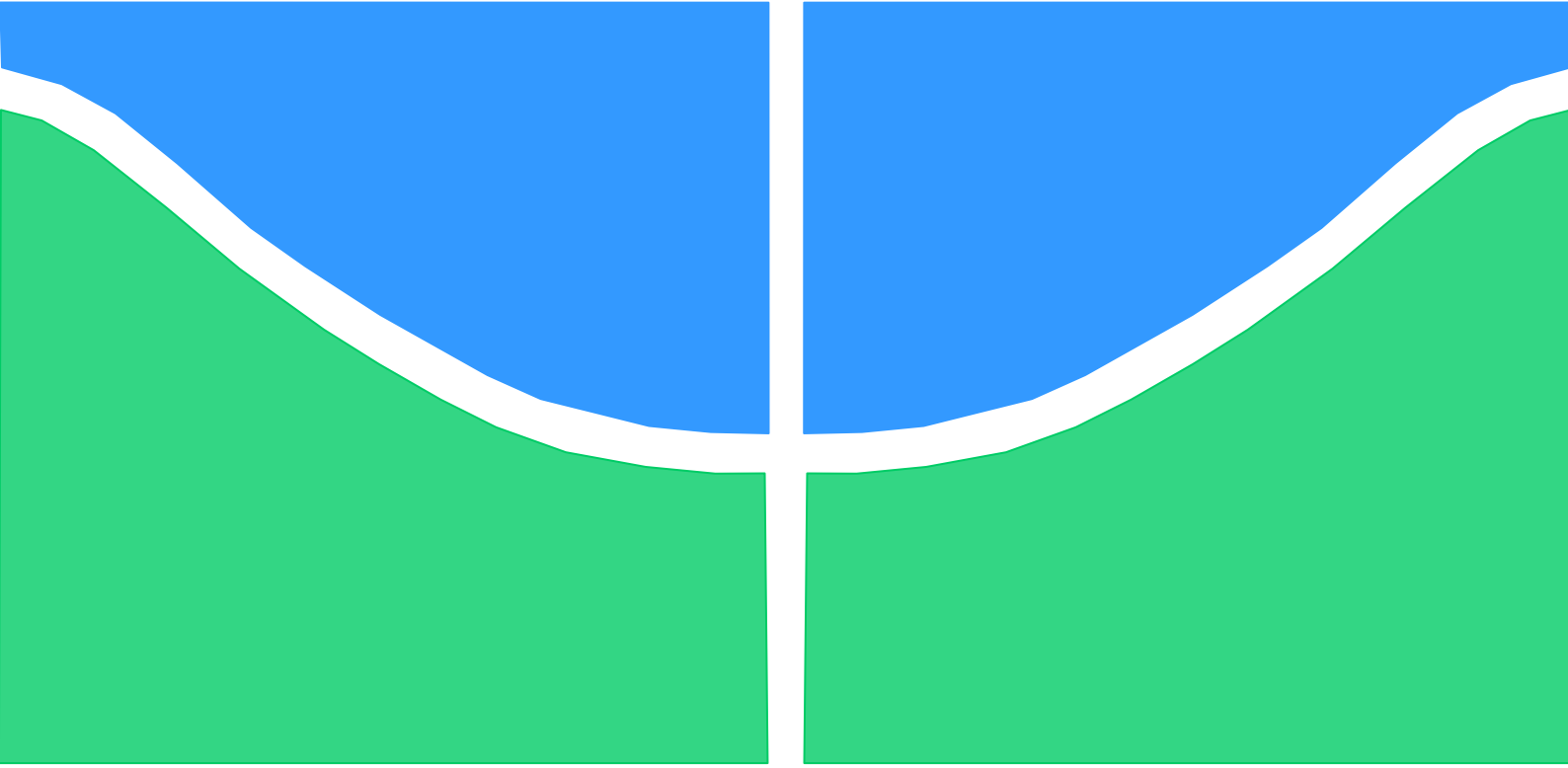


PROJETO DE GRADUAÇÃO

Aplicação de métricas de manutenibilidade para a garantia da qualidade do MVP da Plataforma Unificada de Metodologias Ativas (PUMA) no contexto PBL

Por,

Isabela Ribeiro Fontes

Brasília, novembro de 2021.

UNIVERSIDADE DE BRASILIA

Faculdade de Tecnologia
Curso de Graduação em Engenharia de Produção

UNIVERSIDADE DE BRASILIA
Faculdade de Tecnologia
Curso de Graduação em Engenharia de Produção

PROJETO DE GRADUAÇÃO

Aplicação de métricas de manutenibilidade para a garantia da qualidade do MVP da Plataforma Unificada de Metodologias Ativas (PUMA) no contexto PBL

Por,

Isabela Ribeiro Fontes

Relatório submetido como requisito parcial para obtenção do grau de Engenheiro de Produção

Banca Examinadora

Prof.ª Dr.ª Simone Borges Simão Monteiro (Orientadora)	_____
Prof.ª MSc. Ana Cristina Fernandes Lima Gomes (EPR/UnB)	_____
Prof. Everaldo Silva Júnior (EPR/UnB)	_____

Brasília, novembro de 2021.

"Levanta essa cabeça
Enxuga essas lágrimas, certo? (Você memo)
Respira fundo e volta pro ringue (vai)
Cê vai sair dessa prisão
Cê vai atrás desse diploma
Com a fúria da beleza do Sol, entendeu?
Faz isso por nóis
Faz essa por nóis (vai)
Te vejo no pódio

Ano passado eu morri
Mas esse ano eu não morro"

- Leandro Roque de Oliveira, o Emicida.

AGRADECIMENTOS

Primeiramente, agradeço ao meu pai que me mostrou desde sempre a importância do conhecimento e da ciência como forma de libertação. A minha mãe por sempre me lembrar de me conectar a minha espiritualidade e ancestralidade para encontrar conforto nos momentos difíceis. Ao meu irmão, meu maior exemplo de resiliência e disciplina na busca dos próprios sonhos.

Minha gratidão a minha orientadora, Professora Simone Borges, que desde o início da graduação me mostrou, principalmente pelo exemplo, como a empatia, atenção aos detalhes e a paixão pelo conhecimento científico formam excelentes profissionais. Agradeço, ainda, a todos os professores que durante essa jornada instigaram a sempre entregar o melhor de mim para o aprendizado constante.

Reconheço o trabalho árduo de toda a equipe que doou seu tempo e conhecimento ao PUMA, principalmente os futuros Engenheiros de Software que produziram o código que permitiu a realização deste projeto de graduação.

Agradeço a todos as pessoas incríveis que a UnB me deu a honra de conhecer, que foram meus maiores confidentes e apoiadores, e hoje, com orgulho, posso chamá-los de amigos.

Por fim, sempre serei eternamente grata à todas as mulheres que lutaram, muitas vezes com a preço da própria vida, para conquistar o direito que usufruo hoje ao me tornar engenheira.

Só sou, porque somos.

RESUMO

Desenvolver um aplicativo Web é uma tarefa difícil se os recursos e informações corretos não estiverem disponíveis, pois um determinado aplicativo durante seu ciclo de vida não está siplesmente sujeito à sua forma de ser operado, mas também à forma como foi desenvolvido. O objetivo deste estudo é aplicar métricas de qualidade interna de software para garantia da qualidade no desenvolvimento do Mínimo Produto Viável do PUMA em um contexto PBL. Uma pesquisa de caráter exploratório e natureza qualitativa que combina as estratégias de revisão sistemática da literatura e estudo de caso foi aplicada. A avaliação de indicadores e métricas relacionados a manutenibilidade foi realizada, mesmo que as métricas relacionadas ao código em si atingiram o valor esperado, aquelas relacionadas ao processo de desenvolvimento obtiveram os piores resultados possíveis. Os resultados demonstraram como o descompasso de qualidade do processo e do produto impactou na garantia da qualidade geral do MVP, além de fornecer outras percepções para a garantia da qualidade para o desenvolvimento contínuo do PUMA.

Palavras-chaves: Análise de Software; Manutenibilidade; Métricas de software; Lean Inception; PBL.

ABSTRACT

Developing a web app is a challenging task if the right resources and information are not available. This study aimed to apply software quality metrics for the quality guarantee in the PUMA's MVP development in a PBL context. The research was exploratory, qualitative, combining the strategies of systematic literature review and case study. The objective was accomplished by evaluating indicators related to maintainability. Even if the metrics related to the code itself reached the expected value, in contrast, those related to the development process had the worst possible results. This scenario demonstrated how the process and product quality mismatch impacted the overall MVP quality assurance. The results could provide additional insights into quality assurance for the ongoing development of the PUMA.

Keywords: Software Analytics; Maintainability; Software Metrics, Lean inception, PBL.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE QUADROS

LISTA DE SIGLAS

ABNT	*Associação Brasileira de Normas Técnicas*
AD	*Ausência de Duplicações*
ANB	*Arquivos Não-Bloqueados*
CRUD	*Creat, Read, Update, Delete*
DCM	*Densidade Comentários*
DNC	*Densidade Não Complexos*
ENFN	*É – Não É – Faz – Não Faz*
EPR	*Engenharia de Produção*
EPS	*Engenharia de Produto de Software*
IEC	*International Electrotechnical Commission*
ISO	*International Organization for Standardization*
MDS	*Métodos de Desenvolvimento de Software*
MVP	*Minimun Viable Product*
OWASP	*The Open Web Application Security Project*
PBD	*Problemas Bem Definidos*
PBL	*Problem Based Learning*
PPP	*Plano Político Pedagógico*
PSP	*Projeto de Sistema de Produção*
PUMA	*Plataforma Unificada de Metodologias Ativas*
TEMAC	*Teoria do Enfoque Meta Analítico Consolidado*
TIC	*Tecnologia de Informação e Comunicação*
US	*User Stories*
UX	*User Experience*
WBA	*Web Based Application*
WBAQM	*Web Based Aplication Quality Model*
WoS	*Web of Science*
WWW	*World Wide Web*

1. INTRODUÇÃO

1.1. CONTEXTUALIZAÇÃO

As aplicações web são componentes cruciais da vida e estão entre as classes de sistemas de software que são mais utilizados atualmente. São usados para realizar uma ampla variedade de funções, tanto de negócio quanto pessoais (BAKHTIN; AL-ZAGHEER, 2019). Uma aplicação web ou *webapp* são todos os softwares que podem ser processados e, por sua vez, armazenados nos servidores da Web e acessados por meio de navegadores com acesso à internet (CASTILLO et al., 2018).

A competitividade e o surgimento de novos negócios, que a cada dia ameaçam monopolizar o mercado, levam as organizações a buscarem meios inovadores no desenvolvimento de software (CASTILLO et al., 2018). O avanço da tecnologia aponta para um novo paradigma, o da interconectividade. Esse cenário de conexão entre as várias etapas da cadeia de valor, possibilita maior controle operacional, aliado a uma maior autonomia devido à capacidade de coleta de dados e intercomunicação (CGI.BR, 2020).

No Brasil, a pesquisa TIC Empresas 2019, identificou que 27% das empresas de médio e grande porte afirmaram usar algum tipo de software baseado na web. Além disso, 57% das empresas fazem negócios pela internet, mas apenas 16% delas o fazem pelo próprio canal (CGI.BR, 2020).

Nesse mundo de negócios globalizado, a qualidade dos produtos de software é considerada um elemento essencial para o sucesso empresarial, isso se dá, pois, a baixa qualidade pode levar a uma série de consequências para a construção de diferencial competitivo (AL-QUTAISH, 2010).

O Projeto PUMA tem como objetivo desenvolver uma plataforma web, denominada Plataforma Unificada de Metodologia Ativa (PUMA). A *webapp*, como o próprio nome sugere, aspira ser uma plataforma que unifique e dê suporte aos processos que envolvem a aplicação de metodologias ativas de aprendizado, principalmente nas disciplinas de Projeto de Sistema de Produção do curso de Engenharia de Produção da Universidade de Brasília (MONTEIRO; CAMPOS; LIMA; MARIANO, 2018).

O PUMA em sua concepção tem como balizador a aprendizagem baseada em problemas (PBL, do inglês Problem-Based Learning), então para o desenvolvimento da plataforma como um mínimo produto viável (MVP, do inglês *Minimum Viable*

Product) um grupo de trabalho multidisciplinar, dos cursos de Engenharia de Software e Engenharia de Produção da Universidade de Brasília, foi formatado para iniciar a geração da solução com base na aplicação do PBL no contexto de ensino em engenharia (MONTEIRO; CAMPOS; LIMA; MARIANO, 2018; NERI, 2021).

1.2. PROBLEMA DA PESQUISA

Segundo Castillo et al (2018), desenvolver um aplicativo Web é uma tarefa difícil se você não tiver os recursos e informações corretos sobre os aspectos de qualidade que ele deve ter para ser considerado um aplicativo robusto, confiável, seguro e operacional.

Embora a qualidade do *webapp* leve à avaliação de uma série de métricas para a análise de características e subcaracterísticas em relação ao seu uso, é de grande importância destacar a necessidade de verificar a qualidade no processo de desenvolvimento, pois um determinado aplicativo durante seu ciclo de vida não está simplesmente sujeito à sua forma de ser operado, mas também à forma como foi desenvolvido (CASTILLO et al., 2018).

Dessa forma, há duas questões motivadoras para a realização dessa pesquisa: existem modelos consolidados para a determinação e medição de critérios de qualidade em aplicações web? Se esses modelos existem, como podem ser aplicados de forma qualitativa durante o projeto do desenvolvimento do PUMA?

1.3. JUSTIFICATIVA

As características fundamentais ideais buscadas para o Engenheiro de Produção egresso da UnB é a capacidade de, como engenheiro, identificar, caracterizar e tratar adequadamente a criação de valor associada aos espaços econômico, político e cultural da sociedade. Nesse contexto, o egresso da UnB deverá ser capaz de entender a produção e alocação de bens e serviços por agentes privados e públicos nesses três espaços (SILVA; BALTHAZAR, 2010).

A fim de alcançar esse objetivo global, o Plano Político Pedagógico do Curso de Engenharia de Produção da UnB foi desenvolvido com base no método conhecido como PBL - “Problem Based Learning”, (Aprendizagem Baseada em Problemas). Atualmente o curso possui 8 disciplinas em sua grade curricular (6 obrigatórias e 2 optativas) chamadas de Projeto de Sistemas de Produção (PSP) nas quais o objetivo

é fazer com que os alunos apliquem os conhecimentos teóricos da engenharia de produção na resolução de problemas reais (SILVA; BALTHAZAR, 2010; MONTEIRO, 2020).

No processo de concepção das disciplinas de PSP, atualmente, é responsabilidade do professor orientador e/ou das equipes formadas encontrar partes interessadas que detenham projetos relacionados ao assunto base do projeto. Essa abordagem delimita a capacidade de prospecção de projetos para o limite do ciclo social e profissional dos alunos e professor e/ou para o limite do conhecimento do aluno sobre o tema, o qual ainda está em aprendizado.

Nesse contexto, a Plataforma Unificada de Metodologia Ativa (PUMA) busca integrar de forma automatizada os processos que envolvem o desenvolvimento de projetos pelos alunos nas disciplinas de PSPs, desde a captação de projetos até a divulgação das soluções desenvolvidas. A plataforma faz parte do Programa Aprendizagem para o 3º Milênio do Centro de Educação à Distância (CEAD/UnB) e está sendo concebida para a utilização de docentes e discentes da universidade, além de diversas empresas públicas e privadas. Logo, o Projeto PUMA tem uma dimensão social com alicerce no tripé universitário, porque é idealizada para fortalecer a integração aluno-empresa, facilitando a avaliação por parte dos professores e garantindo um maior feedback entre os envolvidos. (MONTEIRO; LIMA; MARIANO; SILVA JUNIOR; 2020)

Segundo o Plano Político Pedagógico do Curso de Engenharia de Produção (2010), Gestão da Qualidade, Gestão do Produto e Gestão da Tecnologia são áreas ou subáreas do conhecimento que são alinhadas à engenharia de produção que podem ser aplicadas no contexto do desenvolvimento de software. Além das necessidades apontadas na sessão de CONTEXTUALIZAÇÃO, este trabalho se apresenta como uma possibilidade de atuação do Engenheiro de Produção no mercado de TIC.

O estudo Formação Educacional e Empregabilidade em TIC da Associação das Empresas de Tecnologia da Informação e Comunicação (TIC) e de Tecnologias Digitais – Brasscom (2019), projeta em relação ao crescimento do mercado de TIC uma necessidade de 420 mil profissionais para atuar nesta área entre os anos de 2018 e 2024. Aponta, ainda, que o Brasil capacita 46 mil profissionais por ano com perfil tecnológico, caso essa média seja mantida, será gerado um déficit de 24 mil profissionais anualmente para o mercado.

1.4. OBJETIVOS

1.4.1. Objetivo geral

O objetivo deste estudo é aplicar métricas de qualidade interna de software para garantia da qualidade no desenvolvimento do Mínimo Produto Viável do PUMA.

1.4.2. Objetivos específicos

Para atingir o objetivo geral, faz-se necessário concluir os seguintes objetivos específicos:

1. Compreender os diferentes modelos de qualidade aplicáveis a *webapps;*
2. Priorizar critérios de avaliação da qualidade que possam ser aplicáveis ao PUMA;
3. Determinar métricas e forma de obtenção dos dados no contexto de desenvolvimento do Mínimo Produto Viável (MVP) do PUMA*;*
4. Analisar as métricas de qualidade interna previamente determinadas no desenvolvimento do MVP do PUMA.

1.5. ESTRUTURA DE CAPÍTULOS

Este trabalho estrutura-se em 6 capítulos, da seguinte forma:

- Capítulo 1 – Introdução: neste capítulo são apresentados os conceitos principais necessários para o entendimento do estudo, além da contextualização, problema da pesquisa e justificativa para o presente estudo além de seus objetivos gerais e específicos;
- Capítulo 2 – Levantamento do Estado da Arte: apresenta os resultados da revisão da literatura sobre modelos de qualidade aplicáveis a webapps e priorização de critérios.
- Capítulo 3 – Referencial Teórico: que aborda a fundamentação teórica em que se constrói a pesquisa, nesse caso a aprendizagem baseada em problemas no contexto de engenharia, a gestão da qualidade de software utilizando métricas de software, além dos frameworks ágeis de desenvolvimento de produtos *Lean Startup* e *Lean Inception*;
- Capítulo 4 – Metodologia: define os métodos e a estruturação da pesquisa necessários para o alcance dos objetivos;
- Capítulo 5 – Estudo de caso: na qual estão descritos os resultados obtidos a durante o estudo de caso, e, por fim;

- Capítulo 6 – Considerações Finais: onde está apresentada as conclusões, limitações e contribuições da pesquisa e sugestões para estudos futuros.

2. LEVANTAMENTO DO ESTADO DA ARTE

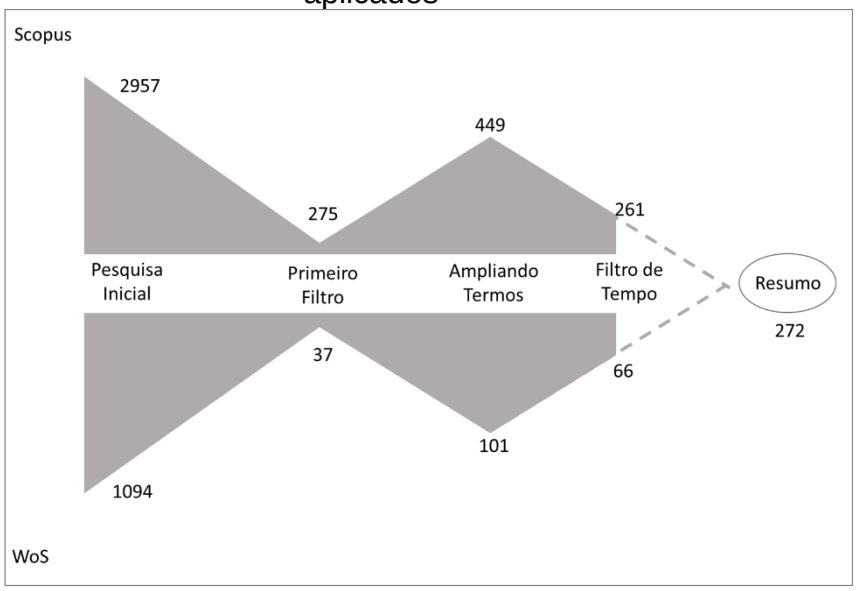
Com base no método de revisão sistemática da literatura segundo Mariano e Rocha (2017), chamado Teoria do Enfoque Meta Analítico – TEMAC, este capítulo apresenta um panorama dos modelos existentes para a avaliação de qualidade em aplicações web.

2.1.1. Etapa 1.1 – Preparação da Pesquisa

Como primeira parte da preparação da pesquisa foram selecionadas as bases de dados *Scopus* e *Web of Science (WoS)* como plataformas referenciais para a busca bibliométrica, não só por indexarem artigos revisados por pares, e concentrarem uma infinidade de publicações de diversas áreas, mas também por terem ferramentas de análises bibliométricas internas (ZUPIC; CARTER, 2015).

A busca pelo conjunto de palavras-chave ou descritores representativos para o objetivo da pesquisa se iniciou na base *Scopus*, por essa deter a maior coleção de publicações. Os campos da publicação consultados pela programação da busca foram o Título, Resumo e Palavras-chave. A evolução dos resultados em relação aos termos utilizados e as bases de dados estão apresentados na Figura 1.

Figura 1. Diagrama representativo do funil de resultados de acordo com os filtros aplicados



Fonte: Elaborada pela autora.

O primeiro conjunto testado, chamado de Pesquisa Inicial na Figura 1, foi o mais simplista em relação ao tema a ser abordado, resumindo-se aos termos *web aplication* e *quality*. Este conjunto retornou pouco menos de 3 mil publicações na base

Scopus, e grande parte das publicações se referiam a aplicações em que se avaliava a qualidade de outros processos de negócio, e não da própria aplicação web, o objetivo de fato desta busca. Nesse contexto, houve a necessidade de restringir a busca adicionando um termo referente ao desenvolvimento de software.

Considerando o Primeiro Filtro, adicionou-se a busca o termo *software engineering*, retornando assim 275 publicações na base *Scopus*. Para evitar que pesquisas relevantes sobre o tema não fossem identificadas pelos termos utilizados, ao fazer a leitura dos artigos mais relevantes em relação ao número de citações foram identificados termos sinônimos e variações gramaticais para identificar uma abrangência do funil de resultados, iniciando assim a fase ampliando termos.

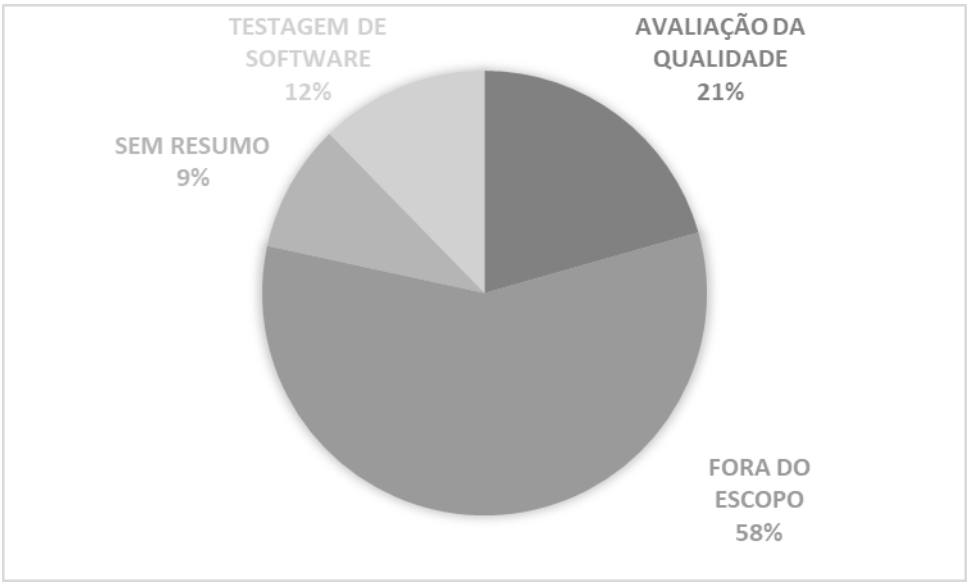
Os termos *webapp*, *web-based application*, além do termo *software development* foram acrescentados na programação resultando assim em quase 450 publicações. Por fim, a área de desenvolvimento de software está em constante mudanças, principalmente pelo avanço das pesquisas e novas tecnologias, então um lapso temporal de 2011-2021 foi delimitado, a fim de obter o que é de fato estado-da-arte para essa linha de pesquisa.

Ao final foram obtidos 261 resultados na base Scopus e 66 na base WoS. Como alguns editores são indexados em ambas as bases, foi aplicado um filtro de duplicidades entre as duas plataformas referenciais, identificando 55 publicações, resultando assim em 272 resultados distintos para a avaliação nas próximas etapas da pesquisa.

2.1.2. Etapa 1.2 – Agrupamento dos Estudos

Obtidos os 272 resultados a partir da busca bibliométrica, com base na leitura dos resumos as publicações foram avaliadas de acordo com a afinidade com o problema da pesquisa, ou seja, se o estudo em questão estava dentro do escopo da avaliação de qualidade de aplicações web e qual técnica, método, ou boa-prática estava sendo empregada. Os resultados dessa avaliação estão resumidos na Figura 2.

Figura 2. Gráfico Representativo do agrupamento de publicações a partir da avaliação dos resumos



Fonte: Elaborada pela autora.

Nesse processo, 28 publicações não foram consideradas por se tratar de anais de conferência sem resumo disponível. 174 das publicações restantes foram avaliadas como fora de escopo, em grande parte por se tratar de técnicas, boas práticas e ou modelos de desenvolvimento que proporcionavam webapps de qualidade, mas que não descrevem nenhum tipo de métrica para tal percepção. Além disso, 37 publicações avaliam a qualidade e eficiência de técnicas de testagem de software, uma boa-prática da garantia da qualidade de software, e que, por mais que a temática converse com a avaliação de qualidade e serviu de base para contextualização no tema, ainda não entregava o contexto principal pretendido. Por fim, 62 publicações tratavam especificamente de avaliação de qualidade das quais 57 estavam na base *Scopus* e 5 no *WoS*.

2.1.3. Etapa 1.3 – Análise de Dados

As análises bibliométricas possibilitam o auxílio no processo de tomada de decisões, pois permite explorar, organizar e analisar grandes massas de dados, e ainda, procura por padrões ou explicações para comportamentos não estruturados (YOSHIDA, 2010).

Os resultados obtidos após essa análise podem abranger os seguintes elementos: (1) identificação de tendências e o crescimento do conhecimento em uma área; (2) previsão da produtividade e identificação da influência de autores individuais,

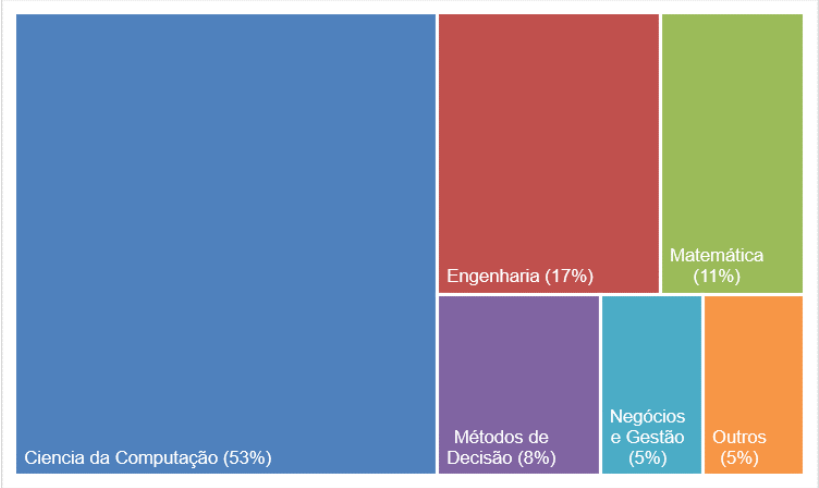
organizações ou países; (3) medição do surgimento de novos temas; (4) análise dos processos de citação e cocitação, além de outros temas (MEDEIROS; VITORIANO, 2015).

Com objetivo de caracterizar a busca obtida no contexto descrito por Medeiros e Vitoriano (2015), as análises serão divididas em duas: (a) Caracterização Geral das Publicações, contemplando os resultados referentes as áreas de pesquisa, localidades, distribuição cronológica e autores e (b) Detalhamento da Rede de Pesquisa, englobando as análises de citação, cocitação, coocorrência e acoplamento bibliográfico.

2.1.3.1. Caracterização Geral das Publicações

As bases de dados utilizadas categorizam as publicações de acordo com as áreas de pesquisa de diferentes formas, mas identificando os temas e agrupando nas grandes áreas, pode-se obter a visão geral apresentada na Figura 3.

Figura 3. Mapa de Árvore descritivo da predominância das áreas de pesquisa



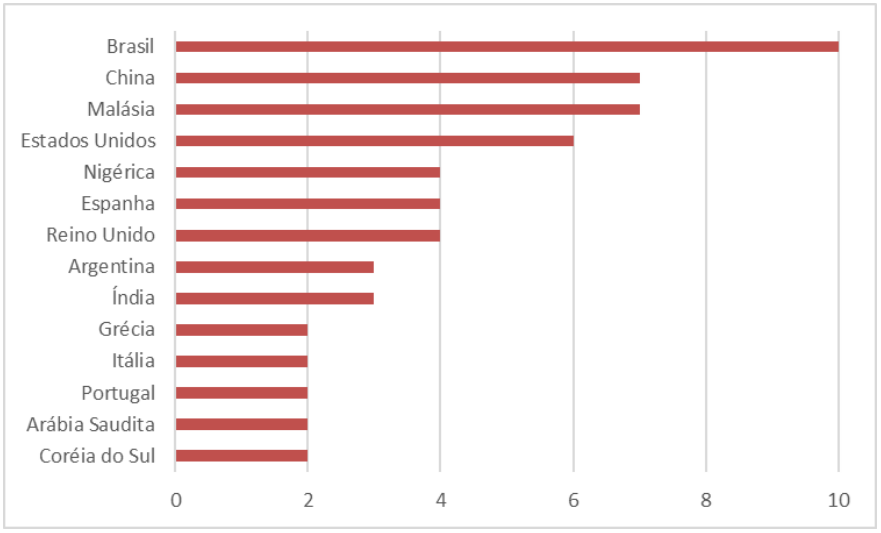
Fonte: Elaborada pela autora.

O foco da pesquisa em avaliação de qualidade em aplicações web, justifica a predominância do tema de Ciência da Computação, sem perder o paralelo importante da engenharia, matemática e métodos de decisão.

A análise de localidade pode apresentar indícios de como os fatores geopolíticos contribuem para o aparecimento de temas de pesquisa. A Figura 4 apresenta o número de publicações por país dos países que tiveram mais de uma publicação: Brasil (10); China e Malásia (7); Estados Unidos (6), Nigéria, Espanha e

Reino Unido (4); Argentina e Índia (3); Grécia, Itália, Portugal, Arábia Saudita e Coréia do Sul (2). Os demais locais apresentaram apenas uma publicação, sendo eles: Áustria, Canadá, Chile, Croácia, Equador, Egito, Finlândia, França, Alemanha, Indonésia, Japão, Jordânia, México, Mianmar, Peru, Polônia Suécia e Tunísia.

Figura 4. Número de publicações por país das publicações obtidas

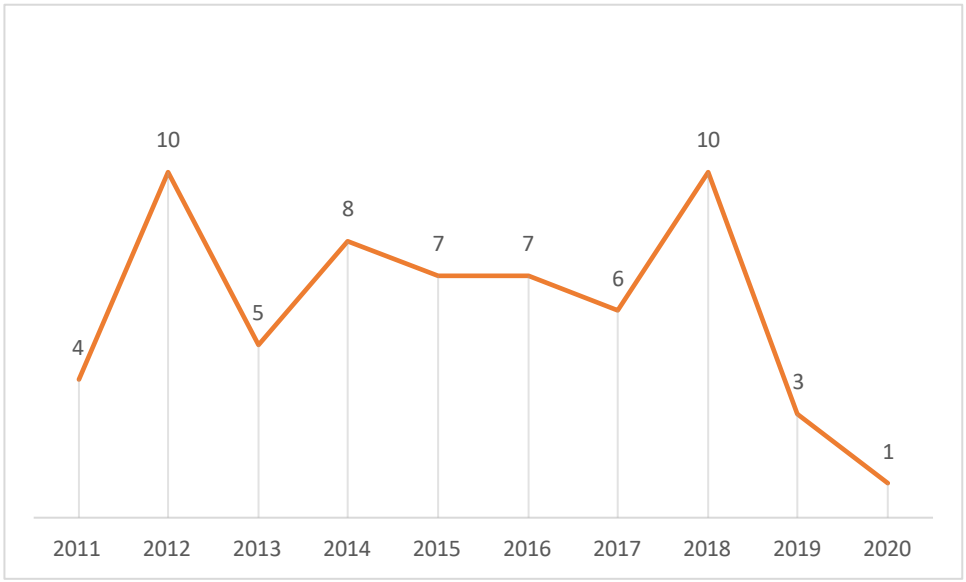


Fonte: Elaborada pela autora.

Segundo o Comitê Gestor da Internet no Brasil (2020) em 2008 apenas 18% dos domicílios brasileiros tinham acesso à internet, e apresentando um crescimento médio de 4,3 pontos percentuais por ano fechou 2019 com esse índice em 71%. Essa evolução se deu principalmente pelo surgimento e popularização dos celulares com acesso à internet. Em 2019, 58% dos brasileiros acessam a internet exclusivamente pelo celular, sendo essa uma tendência observada desde 2015. O Brasil apresentar o maior número de publicações em que se estudam avaliações de qualidade de aplicações web entre 2011-2021 pode estar relacionado com crescimento do acesso à internet da população, em principal, os dispositivos móveis.

A evolução da quantidade de publicações no período da busca está representada pela linha do tempo na Figura 5.

Figura 5. Distribuição da quantidade de publicações obtidas ao longo do tempo em relação ao lapso temporal de interesse



Fonte: Elaborada pela autora.

Numa perspectiva global a quantidade de publicações permaneceu sem grandes variações durante o período, e apresentou indícios de queda a partir de 2019. A evolução da pesquisa nos anos entre os 272 artigos iniciais permaneceu constante entre os anos de 2018 e 2019, o que indica que qualidade de software ainda é um tópico presente nas pesquisas, esse princípio de queda observado na Figura 5 pode indicar um novo foco em aspectos específicos de qualidade em comparação a modelos holísticos. Já considerando o perfil de autoria, os pesquisadores que aparecem em autoria de mais de uma publicação estão apresentados na Figura 6.

Figura 6. Quantidade de publicações por autoria

Fonte: Elaborada pela autora.

A principal autora dentre as publicações é Conte, T com autoria em 7 publicações, pesquisadora brasileira tem em seu trabalho pesquisas nas áreas de Engenharia de Aplicações Web, Avaliação de Usabilidade, Engenharia de Software Experimental, Usabilidade de Aplicações Web e Qualidade de Software.

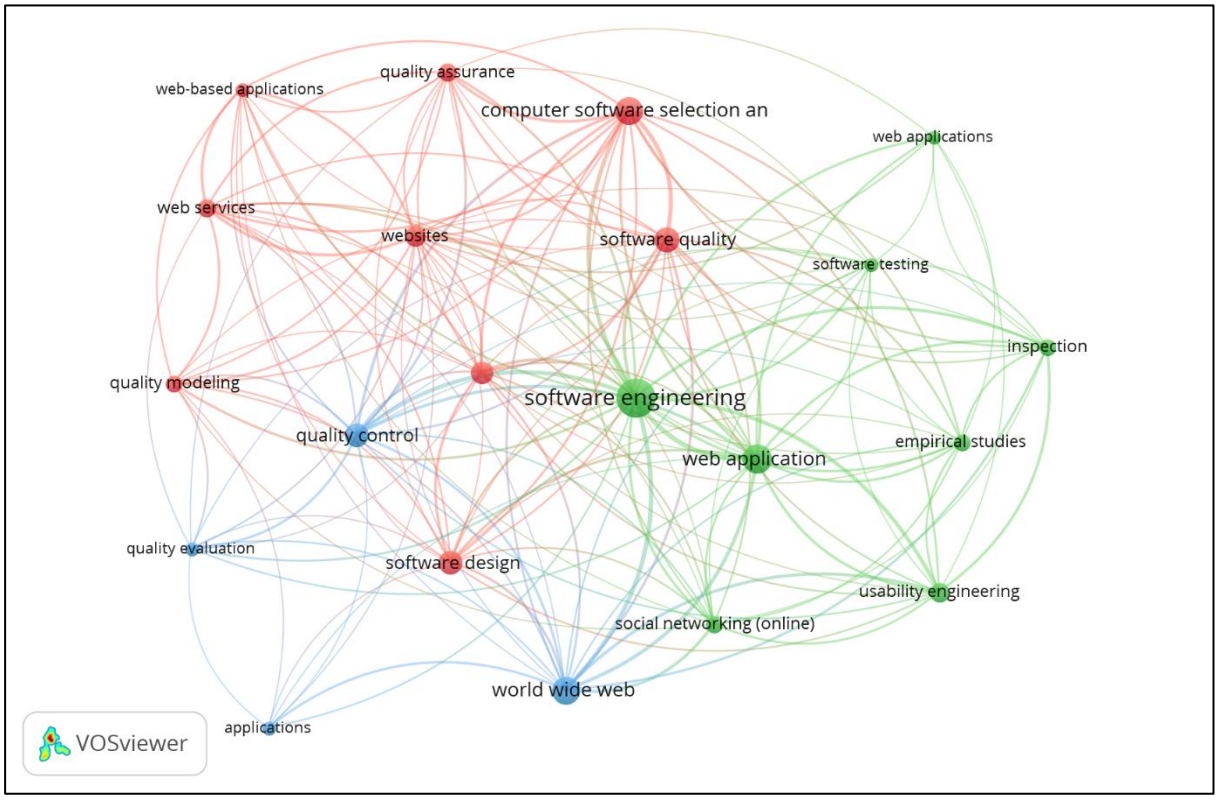
2.1.3.2. Detalhamento da Rede de Pesquisa

Os métodos de mapeamento bibliométricos permitem basear descobertas em dados bibliográficos agregados produzidos por outros cientistas, quando esses dados são agregados e analisados, insights sobre a estrutura do campo, conexões e tópicos podem ser obtidos (ZUPIC; CARTER, 2015).

Os mapas foram elaborados no *VOSviewer*, e por uma limitação das análises fornecidas pelo software a uma única base de dados, foram utilizados apenas os resultados obtidos pela busca da base *Scopus* por agrupar o maior número de resultados diferentes e relevantes para a pesquisa.

A análise de coocorrência de palavras-chave encontra conexões entre conceitos que aparecem simultaneamente, contribuindo para a estrutura conceitual do tema pesquisado (ZUPIC; CARTER, 2015). O mapa em rede das palavras-chave está apresentado na Figura 7, para a geração da imagem o software só imprime um número limitado de caracteres referentes a cada palavra-chave, então o nó indicado como c*omputer software selection an* na imagem representa a palavras-chave indexadoras determinada pela base *Scopus* chamada *computer software selection and evaluation* (17 ocorrências).

Figura 7. Mapa em rede das palavras-chaves atribuídas as publicações obtidas na base *Scopus*



Fonte: Elaborada pela autora.

Na Figura 7, pode-se observar que a rede apresenta três clusters que formam a estrutura conceitual da busca realizada: (a) Cluster 1: Representado pela cor vermelha, tem como centro da rede o termo *software quality* – 21 ocorrências; (b) Cluster 2: Em azul, expande a partir do termo *world wide web* – 17 ocorrências; (c) Cluster 3: Em verde, é centrado pelo termo de maior ocorrência *software engineering* – 33 ocorrências.

O estudo principal do Cluster 1, com 20 citações, é o apresentado por Concas et al. (2012), onde apresenta uma avaliação da qualidade associando o uso de métricas de software às métricas e práticas do desenvolvimento ágil para a garantia da qualidade de software.

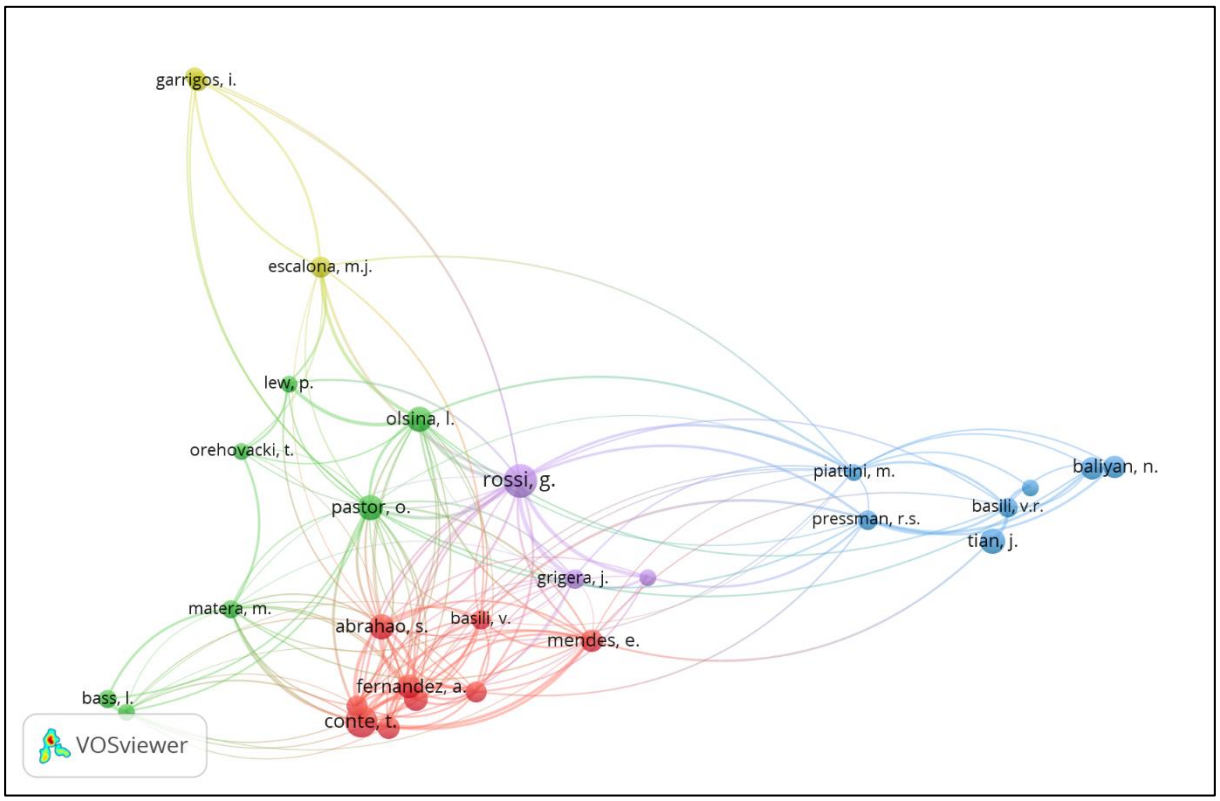
Já no Cluster 2, com 8 citações, o estudo de Rodríguez, Acuña e Juristo (2015) desenvolveu três aplicações web, ou seja, baseadas na WWW, para descobrir padrões de implementação de duas funcionalidades associadas a usabilidade com impacto na funcionalidade principal (cancelar a operação e feedback de progresso). Os arquétipos encontrados foram denominados de padrões de desenho e

programação, e determinaram que a implementação da funcionalidade de cancelar operação é mais factível de padronizar em relação a de feedback de progresso.

Por fim, o Cluster 3, com 10 citações, o estudo de Silva, Costa Valentim e Conte (2015) apresenta um mapeamento sistêmico que contribuiu para a categorização e sumarização de tecnologias que auxiliam na melhoria da usabilidade em aplicações interativas durante o processo de engenharia de software.

A análise de cocitação usa contagens de cocitação para construir medidas de similaridade entre autores, documentos ou periódicos. Em geral, a conclusão dessa análise é de que quanto mais dois itens são citados juntos, mais provável é que seu conteúdo esteja relacionado (ZUPIC; CARTER, 2015). A visualização em rede das cocitações por autor está apresentada na Figura 8.

Figura 8. Mapa em rede das cocitações por autor publicações obtidas na base *Scopus*



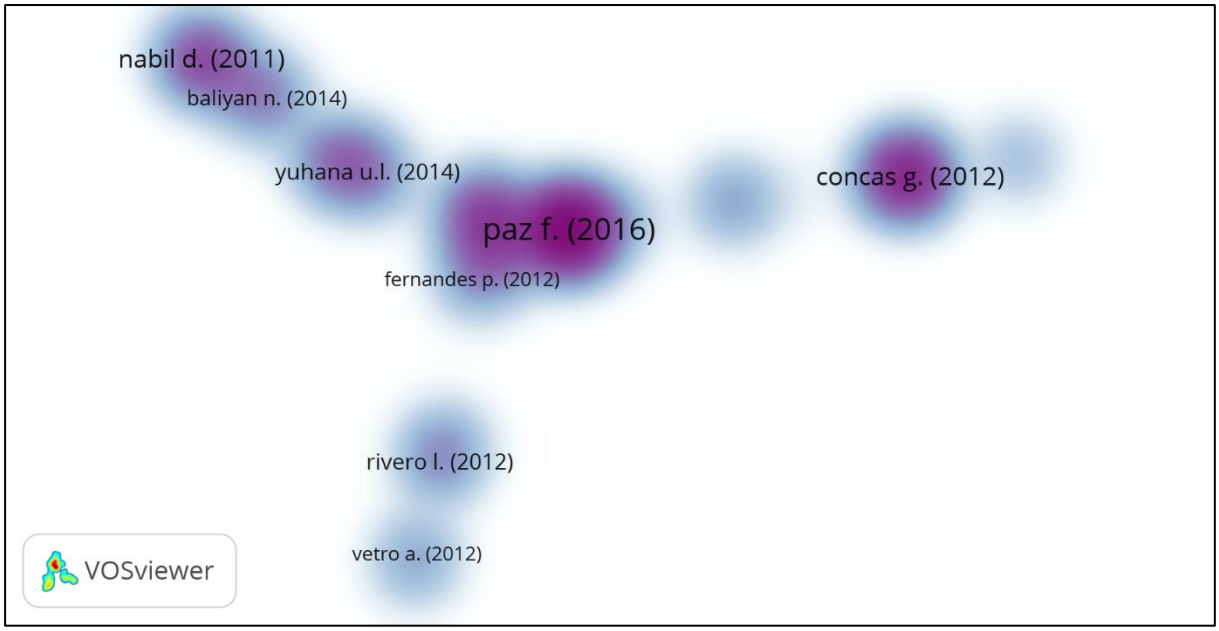
Fonte: Elaborada pela autora.

De acordo com a Figura 8 tem-se a formação de cinco clusters. O cluster vermelho (1), apresenta o estado do campo da construção da qualidade de software, principalmente voltada a usabilidade nos estudos apresentados principalmente por Conte, T. Já o cluster lilás (2), expande a partir dos estudos do autor mais citado (Rossi, G) que trata de métricas de software e avaliação da qualidade voltadas

principalmente para a usabilidade. O cluster verde (3), representa a conexão dentro do campo do desenvolvimento de aplicações web, se destacando o autor Pastor, O. No cluster amarelo (4), caracterizado pelos estudos de Garrigos, I., estão as pesquisas voltadas para o levantamento e caracterização de requisitos para o desenvolvimento. Por fim, o cluster azul (5), representa os pesquisadores de métricas de software e avaliação de qualidade, centrados nos estudos de Baliyan, N.

O acoplamento conecta documentos, autores ou periódicos com base no número de referências compartilhadas (ZUPIC; CARTER, 2015). Apesar de não retornarem resultados de relevância, são usados para analisar a estrutura intelectual de um campo de pesquisa científica (COBO et al., 2012). Na Figura 9 está apresentado o mapa de densidade de acoplamento, feito na perspectiva de documento das publicações obtidas da base de dados *Scopus.*

Figura 9. Mapa de densidade do acoplamento por documento dos resultados da base *Scopus*



Fonte: Elaborada pela autora.

A publicação de Paz e Pow-Sang (2016) se mostrou com o maior número de referências compartilhadas com os demais resultados (Figura 9). Se trata de uma revisão sistemática da literatura, e em geral esse tipo de publicação detém um número maior de referências e de temas e aplicações mais abrangentes, o que explica essa aparição na análise de acoplamento.

A revisão da literatura realizada por Paz e Pow-Sang (2016) foi voltada para os métodos de avaliação de usabilidade mais relevantes no momento, e permitiu

identificar que os questionários, testes de usuário, avaliação heurística, entrevistas e protocolos pensados em voz alta são as técnicas mais empregadas com esse objetivo.

O estudo de Fernandes, Conte e Bonifácio (2012) também é focado em usabilidade, mas o objetivo é descrever uma técnica chamada Avaliação Web – Técnica de Questões (WE- QT, do inglês, *Web Evaluation – Question Technique*). Além de apresentar a visão geral do desenvolvimento da técnica e sua aplicabilidade, foram realizados estudos empíricos para verificar a viabilidade, os quais forneceram indicativos que a técnica pode auxiliar iniciantes a realizar inspeções de usabilidade.

Já Baliyan e Kumar (2014) apresentam uma revisão da literatura sobre o processo de software e avaliação de qualidade voltado para aplicações de web semântica. Da mesma forma que as aplicações web evoluíram da interação dos softwares tradicionais com a *World Wide Web* (WWW), as aplicações de web semântica evoluíram das *webapps*. A principal evolução entre as duas possibilidades está na organização e compreensão dos dados. O objetivo é colocar uma camada de significado sobre os dados permitindo que computadores compreendam semanticamente a informação, em vez de interpretá-los apenas como um conjunto de caracteres, e a partir desse entendimento possam trabalhar em cooperação com as pessoas.

Os autores Baliyan e Kumar (2014) apresentam ainda que as aplicações baseadas em Web Semântica serão o futuro da engenharia da web, e por isso tinham como objetivo identificar as adaptações nas metodologias de desenvolvimento sistemático e nas métricas de qualidade para essa nova realidade.

O artigo de Nabil, Mosad e Hefney (2011) determinou os fatores que avaliam a qualidade das *web-based applications* (WBA), identificando os principais fatores de qualidade e seus subfatores com base nas visualizações e usos do WBA. O modelo teórico proposto aplicou os fatores e subfatores de qualidade da ISO 9126 para revisar as características comuns entre as WBA e o software tradicional, e depois propôs outros fatores particulares desenvolvendo o WBAQM – *Web Based Application Quality Model*.

Relacionado a mesma nuvem está o estudo de Yuhana, Raharjo e Rochimah (2014), que consiste numa aplicação adaptada do WBAQM e das normas ISO 9126 e ISO 25010 a sistemas de informação acadêmica, criando o modelo A*cademic Information System Quality Instruments* (AISQI) como um framework de medição de

qualidade implementada especificamente para os sistemas de informações acadêmicas.

Uma segunda publicação que aparece é a de Concas et al. (2012), é um estudo de caso no qual se apresenta uma revisão interessante sobre métricas de software utilizadas em linguagens de programação orientadas a objetos, além de verificar a correlação entre várias métricas, e avaliar como as práticas de desenvolvimento ágil permitem a melhoria da qualidade ao decorrer do projeto.

Outra densidade relevante é a do estudo de Rivero e Conte (2012) que descreve a técnica Web DUE – Avaliação de Usabilidade de Design da Web –, voltada para avaliação de usabilidade de aplicações web utilizando protótipos de baixa fidelidade. Os autores identificaram que a principal inovação da técnica é o guia do processo de inspeção por zonas das páginas web, além do auxílio em identificar problemas de usabilidade nos primeiros estágios do desenvolvimento, e assim, diminuindo custos de correção.

Por fim, Vetro et al. (2012) provou que a classificação de defeitos baseada no modelo de qualidade de produto de software da ISO/IEC 9126 é confiável e útil para vincular esses defeitos aos aspectos de qualidade impactados. E sugere que próximos estudos utilizem dessa classificação para determinar a priorização de correção de defeitos de acordo com o aspecto de qualidade pareado a diferentes *stakeholders*.

2.1.4. Etapa 1.4 – Modelo Integrador

A maioria dos estudos atuais está lidando com um número limitado de fatores de qualidade, focados em Segurança ou, principalmente, Usabilidade (NABIL; MOSAD; HEFNEY, 2011). Isso ocorre, provavelmente pela aceitabilidade de aplicações Web depender quase estritamente da usabilidade delas, além disso aplicações web com baixa usabilidade sendo historicamente e rapidamente substituídas por outras mais utilizáveis (FERNANDES; CONTE; BONIFÁCIO, 2012).

Como o interesse dessa revisão está na avaliação generalista, ou seja, que mantém a avaliação de qualidade na completude das dimensões das aplicações web, o modelo integrador será focado nos modelos de avaliação de qualidade de software de McCall (1977), Boehm (1976), Dromey's (1995), ISO/IEC 9126 (2003) e 25010 (2011) e o WBAQM (NABI; MOSAD; HEFNEY, 2011).

O primeiro modelo de avaliação de qualidade, um dos modelos mais conhecidos na literatura da engenharia de software (LIMA, 2019), é o Modelo de

McCall et al. (1977). Esse modelo definiu pela primeira vez um fator de qualidade de software como uma condição ou característica que contribui ativamente para a qualidade de um software. Nessa definição, também delimitou que todo fator está relacionado a um custo para realizar a atividade caracterizada por ele, ou para operar naquele grau de qualidade. O resumo dos fatores e os critérios do modelo de McCall para avaliação de qualidade de software estão descritos no Quadro 1.

Quadro 1. Relacionamento dos critérios e seus fatores conforme o Modelo de McCall et al. (1977)

Perspectiva	Revisão			Operação					Transição		
Critérios \ Fatores	Manutenibilidade	Flexibilidade	Testabilidade	Exatidão	Confiabilidade	Eficiência	Integridade	Usabilidade	Portabilidade	Reusabilidade	Interoperabilidade
Acurácia					x						
Auditoria de acesso							x				
Autodescritividade	x	x	x						x	x	
Comunicação comum											X
Comunicatividade								x			
Concisão	x										
Consistência	x			x	x						
Controle de acesso							x				
Dados comuns											X
Eficiência de armazenamento						x					
Eficiência de execução						x					
Expansibilidade		x									
Generalidade		x								x	
Independência da máquina									x	x	
Independência do sistema de software									x	x	
Instrumentação			x								
Integridade				x							
Modularidade	x	x	x						x	x	X
Operabilidade								x			
Rastreabilidade				x							
Simplicidade	x		x		x						
Tolerância					x						
Treinamento								x			

Fonte: Adaptado de McCall et al. (1977)

McCall et al. (1997), elicitou 3 perspectivas contendo 11 fatores de qualidade que descrevem a visão do software por parte dos usuários. Cada fator pode ser julgado e definido a partir de critérios orientados ao software (visão do desenvolvedor)

e cada um dos 23 critérios pode afetar mais de um fator. Além disso, um conjunto de métricas são definidas para fornecer uma escala e método de medição (AL-QUTAISH, 2010). O Modelo de Boehm (1976) é bastante semelhante ao modelo de McCall, pois também apresenta uma estrutura hierárquica. Esse modelo consiste em características de alto nível, nível intermediário e nível primitivo que contribuem a visão geral da qualidade (AL-QUTAISH, 2010). No alto-nível são 3 características que se desdobram em 7 características intermediárias e 15 características primitivas distintas conforme descrito no Quadro 2.

Quadro 2. Relacionamento dos níveis hierárquicos de características de qualidade conforme o modelo de Boehm (1978)

Alto Nível	Utilidade As-is			Manutena-bilidade			Portabi-lidade
Primitivas \ Intermediárias	Confiabilidade	Eficiência	Engenharia Humana	Testabilidade	Inteligibilidade	Flexibilidade	Portabilidade
Acessibilidade		x	x				
Accountability		x		x			
Acurácia	x						
Expansibilidade						x	
Comunicatividade			x	x			
Completude	x						
Concisão					x		
Consistência	x				x		
Eficiência do dispositivo		x					
Independência de dispositivo							x
Legibilidade							x
Robustez/ Integridade	x		x				
Autocontenção	x						x
Autodescritividade				x			
Estruturação				x	x	x	

Fonte: Adaptado de Boehm (1976)

O Modelo de Dromey (1995) trata-se de um modelo mais amplo porque reconhece que a avaliação da qualidade difere para cada produto e a amplitude permite a aplicação em diferentes sistemas.

Dromey (1995) partiu do pressuposto que as subcaracterísticas descritas na ISO/IEC 9126-1 não são mutualmente exclusivos para o impacto no critério de

qualidade em alto-nível, ou seja, um defeito identificado em uma subcaracterística pode impactar no zelo pela qualidade de um ou mais critérios. Esse comportamento das subcaracterísticas inspirou a principal diferença do modelo em relação a ISO/IEC 9126-1, a divisão em 4 propriedades de produto: concreto, interna, contextual e descritiva (LIMA, 2019).

Nessa divisão, Dromey (1995) relacionou os critérios de qualidade que impactam em determinadas propriedades de produto, sendo: (1) Concreto: Propriedades do produto em que alguma violação pode significar um desvio da função pretendida; (2) Interno: Propriedades relacionadas a como as declarações individuais e seus componentes são implementados e como são relacionadas e utilizadas no contexto micro; (3) Contextual: Propriedades que fornecem informações sobre como os módulos, em uma visão macro, relacionam entre si e outros aspectos da arquitetura, como bancos de dados etc; (4) Descritiva: Propriedades as quais impactam na interpretação das regras e funções do produto em um contexto de rastreabilidade e gestão do conhecimento durante o ciclo de vida do produto.

A representação esquemática da relação dos critérios de qualidade e propriedades de produto descritas por Dromey (1995) estão apresentadas no Quadro 3. Uma vantagem importante desse modelo é que ele pode auxiliar na busca sistemática de defeitos e indica as propriedades violadas que ocasionam os defeitos encontrados, facilitando o processo de correção (DROMEY, 1995).

Quadro 3. Relacionamento dos critérios de qualidade e propriedades de produto segundo o modelo de Dromey (1995)

	Propriedade do Produto			
	Concreto	Interno	Contextual	Descritiva
Critério de Qualidade	Confiabilidade	Confiabilidade	Confiabilidade	
		Manutenibilidade	Manutenibilidade	Manutenibilidade
			Portabilidade	Portabilidade
			Reusabilidade	Reusabilidade
	Funcionalidade	Eficiência		Usabilidade

Fonte: Adaptado de Dromey (1995)

Os modelos ISO são construídos em um consenso internacional e acordo de todos os países membros da organização. Em 1991, a ISO publicou seu primeiro consenso internacional sobre características de qualidade para avaliação de produto

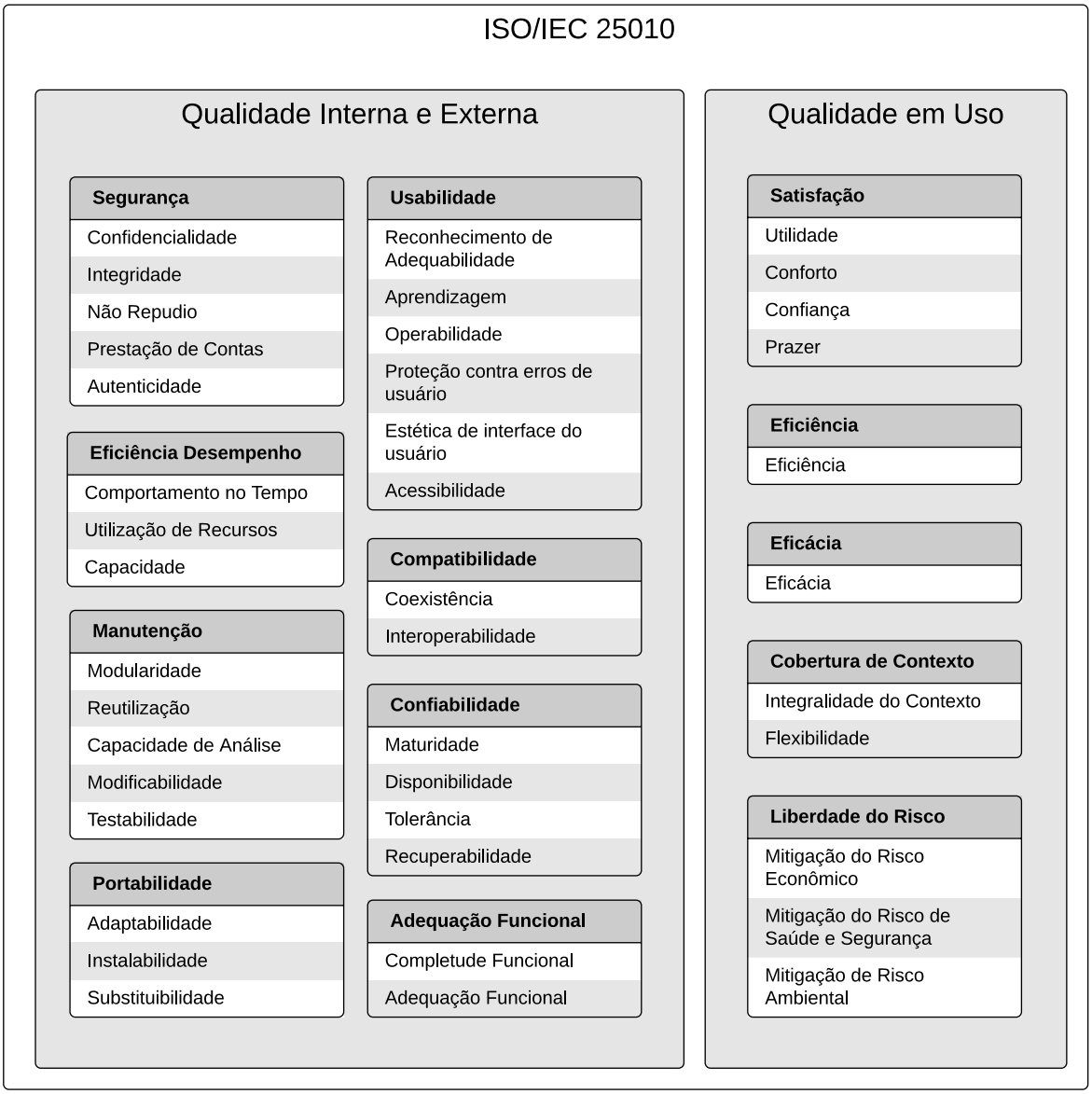
de software, a ISO/IEC 9126 denominada Engenharia de software - Qualidade de produto.

A ISO/IEC 9126 descreve um modelo de qualidade do produto de software, composto de duas partes: (a) Qualidade interna e qualidade externa: Categoriza os atributos de qualidade de software em seis características (funcionalidade, confiabilidade, usabilidade, eficiência, manutenibilidade e portabilidade) as quais são, por sua vez, subdivididas em subcaracterísticas que podem ser medidas por meio de métricas externas e internas; e (b) qualidade em uso: Categorizados em quatro características: eficácia, produtividade, segurança e satisfação.

O modelo da ISO/IEC 9126 foi e é utilizado amplamente, mas foi atualizado numa visão mais ampla na ISO/IEC 25010 chamada, Requisitos e Avaliação de Qualidade de Software (SQuaRE, do inglês *Software Quality Requirements and Evaluation*)

A norma atualizada descreve, ainda, as duas partes do modelo de qualidade: (a) Um modelo de qualidade em uso composto de, agora, cinco características, (algumas das quais, nesta atualização, são subdivididas em subcaracterísticas) que se relacionam ao resultado da interação quando um produto é usado em um determinado contexto de uso; e (b) Um modelo de qualidade de produto composto por oito características (que são subdivididas em subcaracterísticas) que se relacionam com propriedades estáticas do software e propriedades dinâmicas do sistema de computador. O resumo das duas partes da ISO/IEC 25010 está representado no Quadro 4.

Quadro 4. Esquema hierárquico dos contextos de avaliação de software na ISO/IEC 25010



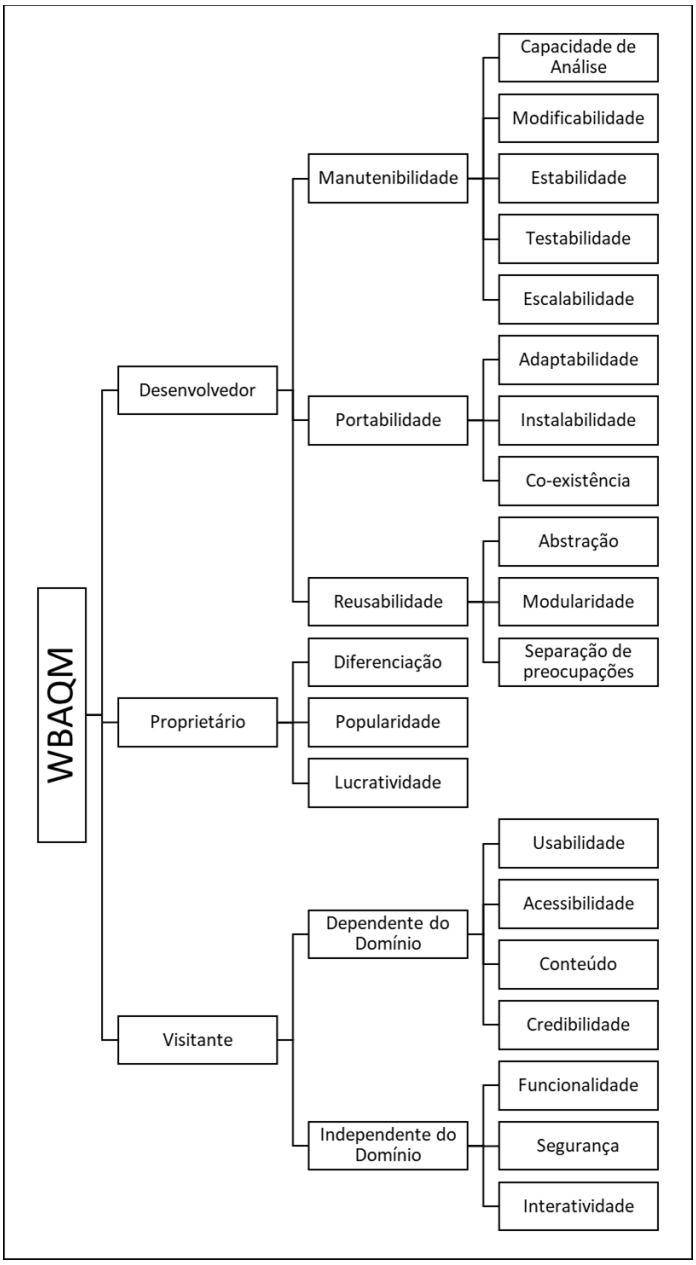
Fonte: Adaptado de ISO (2011)

O Modelo de Qualidade de Aplicações Baseadas na Web ou simplesmente WBAQM é um modelo teórico proposto por Nabil, Mosad e Hefney (2011) a partir dos fatores-critérios-métricas da ISO/IEC 9126 utilizando a perspectiva de caracterização de baixo para cima do modelo de Dromey (1995). O WBAQM é apresentado em camadas: (a) Camada 1: identificação de visualizações e usos de qualidade WBA; (b) Camada 2: categorização de fatores de qualidade para cada exibição de qualidade; e (c) Camada 3: Mapeamento de subfatores de qualidade para cada fator de qualidade.

No desenvolvimento da Camada 1, são mapeadas as preocupações de cada uma das partes interessadas no desenvolvimento de WBA. Para os desenvolvedores,

a necessidade de desenvolver uma WBA sólida, integrada às necessidades dos visitantes (usuários) e atendendo as necessidades do proprietário é o problema mais crucial. Considerando a percepção dos usuários, o objetivo final é que as aplicações web possam facilitar a busca de informações podendo melhorar seu desempenho e percepção em relação ao WBA que está sendo utilizado. Já o proprietário está preocupado principalmente com três fatores de qualidade: a diferenciação, a popularidade e a lucratividade (NABIL; MOSAD; HEFNEY; 2011).

Figura 10. Fatores de qualidade das perspectivas das partes interessadas no WBAQM



Fonte: Adaptado de Nabil, Mosad e Hefney (2011)

Na Camada 2 os fatores de qualidade da ISO 9126, foram revisados individualmente para descobrir se eles contribuíram para a natureza do WBA e foram distribuídos de acordo com as perspectivas de desenvolvedores, proprietários e visitantes conforme apresentado na Figura 10 (NABIL; MOSAD; HEFNEY; 2011).

A última etapa do modelo proposto, a Camada 3, tem como objetivo fornecer um conjunto de subfatores de qualidade que amplia as subcaracterísticas de qualidade apresentadas na ISO 9126. Essa expansão foi necessária, sobretudo, pelas diferenças das *webapps* em relação aos softwares tradicionais, o uso de alguns critérios é limitado e um novo conjunto de subfatores de qualidade foi sugerido, sendo eles: acessibilidade, consistência, expansibilidade, independência da máquina, modularidade, confidencialidade, credibilidade, independência do sistema de software, abstração, conteúdo, diferenciação, interatividade, lucratividade, popularidade e separação de preocupações (NABIL; MOSAD; HEFNEY; 2011).

Como forma de integralizar todos os fatores e critérios elicitados dos modelos apresentados, o Quadro 5 foi elaborado como forma de resumir e apresentar as evoluções na avaliação de cada modelo com base em seus critérios. Neste quadro foi incluído, além dos modelos de qualidade consolidados, a ferramenta Q-Rapids por além de agrupar os modelos de qualidade consolidados (baseados nas ISO/IEC 9126-1 e posteriormente a ISO/IEC 25010) e integrá-los a ferramentas de análise de software sendo a ferramenta com validação de uso na indústria de desenvolvimento mais recente (MARTÍNEZ-FERNÁNDEZ et al., 2019). Essa ferramenta será apresentada em mais detalhes no Capítulo 3.

Apesar de grande parte dos modelos terem sido traçados em um contexto de software tradicional, a construção abrangente pode permitir adaptações para fornecer aplicabilidade em diferentes produtos de software. Indícios dessa possibilidade de adaptação foram identificados pela variedade de publicações encontradas que utilizavam metodologias adaptadas desses modelos gerais, por mais que esses estudos tratavam de um fator de qualidade específico.

Quadro 5. Comparativo resumo entre os fatores elicitados nos modelos de qualidade avaliados

Fatores e Critérios \ Modelo de Qualidade	McCall	Boehm	DROMEY	ISO 9126	ISO 25010	WABQM	Q-Rapids	Total	Fatores e Critérios \ Modelo de Qualidade	McCall	Boehm	DROMEY	ISO 9126	ISO 25010	WABQM	Q-Rapids	Total
Manutenibilidade	x	x	x	x	x	x	x	7	Recuperabilidade				x	x			2
Usabilidade	x	x	x	x	x	x	x	7	Satisfação				x	x			2
Confiabilidade	x	x	x	x	x		x	6	Substituibilidade				x	x			2
Testabilidade	x	x		x	x	x	x	6	Abstração						x		1
Eficiência	x	x	x	x	x			5	Atratividade				x				1
Portabilidade	x	x	x	x	x			5	Auditoria de acesso	x							1
Funcionalidade			x	x	x	x		4	Autenticidade					x			1
Adequação				x	x		x	3	Autoconteção		x						1
Estabilidade				x		x	x	3	Cobertura de Contexto					x			1
Acessibilidade		x			x	x		3	Comportamento no Tempo					x			1
Acurácia	x	x		x				3	Comunicação comum	x							1
Adaptabilidade				x	x	x		3	Confidencialidade					x			1
Capacidade de Análise				x	x	x		3	Conforto					x			1
Co-Existência				x	x	x		3	Conteúdo						x		1
Completude	x	x			x			3	Controle de acesso	x							1
Eficiência de armazenamento	x			x	x			3	Credibilidade						x		1
Eficiência de execução	x			x	x			3	Dados comuns	x							1
Expansabilidade	x	x				x		3	Diferenciação						x		1
Flexibilidade	x	x			x			3	Disponibilidade					x			1
Independencia da máquina	x	x			x			3	Eficiência de Dispositivo		x						1
Integridade	x	x			x			3	Engenharia Humana		x						1
Inteligibilidade		x		x	x			3	Estética de interface					x			1
Interoperabilidade	x			x	x			3	Estruturação		x						1
Modificabilidade				x	x	x		3	Exatidão	x							1
Modularidade	x				x	x		3	Generalidade	x							1
Operabilidade	x			x	x			3	Instrumentação	x							1
Reusabilidade	x		x		x			3	Integralidade de Contexto					x			1
Segurança				x	x	x		3	Interatividade						x		1
Tolerância	x			x	x			3	Legibilidade		x						1
Produtividade				x			x	2	Lucratividade						x		1
Accountability		x			x			2	Mitigações de Risco					x			1
Aprendizagem				x	x			2	Não Repúdio					x			1
Autodescritividade	x	x						2	Popularidade						x		1
Comunicatividade	x	x						2	Prazer					x			1
Concisão	x	x						2	Proteção erros de usuário					x			1
Consistência	x	x						2	Rastreabilidade	x							1
Eficácia				x	x			2	Separação de preocupações						x		1
Independência do sistema de software	x				x			2	Simplicidade	x							1
Instalabilidade				x		x		2	Treinamento	x							1
Maturidade				x	x			2	Utilidade					x			1

Fonte: Elaborado pela autora.

O modelo integrador (Quadro 5) pôde ser considerado representativo para a objetivo da pesquisa porque cumpriu os quatro pontos de validação por evidências propostos por Mariano e Rocha (2017), sendo eles a presença de: (1) pelo menos uma publicação de revisão sistemática; (2) pelo menos uma publicação de estudo de caso com resultados apresentados; (3) estudos realizados por mais de um centro ou grupo de pesquisa e (4) opiniões de autoridades respeitadas.

O estudo de Paz e Pow-Sang (2016), uma revisão da literatura com 56 citações cumpre o primeiro ponto de validação. 37% das publicações consideradas para a construção do modelo foram identificadas como estudos de caso com resultados apresentados, além disso 72% foram provenientes de pesquisas de centros e grupos de pesquisa distintos, cumprindo, assim, os critérios 2 e 3. Três especialistas distintos validaram o modelo, sendo dois especialistas em engenharia de software e uma especialista em engenharia de qualidade.

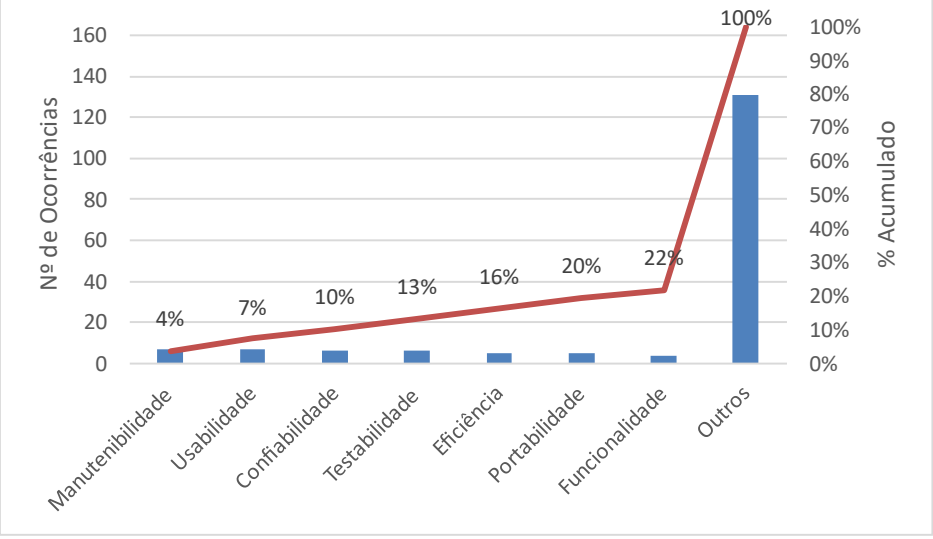
Indiscutivelmente o modelo normalizado mais recente, formalizado pela série ISO/IEC 25000, fornece a avaliação mais abrangente, com o maior número de fatores-critérios-métricas de avaliação. Ademais, o WBAQM detém perspectivas de avaliação específicas para o contexto das aplicações web, fornecendo *insights* complementares.

A ferramenta Q-Rapids utiliza de menos fatores em comparação com os modelos os quais serviram de base para sua elaboração, mas detém foco entre os que apresentam o maior número de ocorrências nessa avaliação global.

2.1.5. Etapa 1.5 – Priorização de Critérios

Analisando as ocorrências de cada um dos fatores de qualidade ao longo da evolução dos modelos de qualidade de software obtidos nesse modelo integrador (Quadro 5) pelo princípio de Pareto (Figura 11), pode-se determinar os principais fatores para avaliação de qualidade de produtos de software.

Figura 11. Diagrama de Pareto das ocorrências dos fatores de qualidade utilizados nos modelos de qualidade de software.



Fonte: Elaborado pela autora.

Os principais critérios de qualidade obtidos da aplicação do princípio de Pareto no número de ocorrência nos modelos de qualidade da análise foram: (1) Manutenibilidade; Capacidade do produto de software de ser modificado. As modificações podem incluir correções, melhorias ou adaptações do software devido a mudanças no ambiente e nos seus requisitos ou especificações funcionais; (2) Usabilidade: Capacidade do produto de software de ser compreendido, aprendido, operado e atraente ao usuário, quando usado sob condições especificadas; (3) Confiabilidade: Capacidade do produto de software de manter um nível de desempenho especificado, quando usado em condições especificadas; (4) Testabilidade: Capacidade do produto de software de permitir que o software, quando modificado, seja validado; (5) Eficiência: os recursos gastos em relação à precisão e integridade com que os usuários atingem as metas; (6) Portabilidade: Capacidade do produto de software de ser transferido de um ambiente para outro; (7) Funcionalidade: Capacidade do produto de software de prover funções que atendam às necessidades explícitas e implícitas, quando o software estiver sendo utilizado sob condições especificadas (ISO 25010, 2011).

3. REFERENCIAL TEÓRICO

3.1. Aprendizagem Baseada em Problemas no Ensino de Engenharia

A Aprendizagem Baseada em Problemas (PBL – do inglês Problem-Based Learning) é uma estratégia educacional. Um método para organizar o processo de aprendizagem de maneira que os alunos se envolvam ativamente na busca de respostas sozinhos (GRAAFF; KOLMOS, 2007).

O termo PBL foi originalmente cunhado por Don Woods, com base em seu trabalho com estudantes de química na Universidade McMaster no Canadá. No entanto, a popularidade e a disseminação mundial subsequente do PBL estão principalmente ligadas à introdução deste método educacional na escola de medicina da mesma universidade (GRAAFF; KOLMOS, 2007).

Os autores do tema geralmente concordam que o PBL tem seis características definidoras: (a) o uso de problemas como ponto de partida para o aprendizado, (b) colaboração em pequenos grupos e (c) orientação flexível de um tutor. Como os problemas orientam o aprendizado, (d) o número de aulas expositivas é limitado. Isso está alinhado com a ideia de que (e) a aprendizagem deve ser iniciada pelo aluno e que (f) deve haver tempo suficiente para o auto-estudo (SCHMIDT et al., 2009).

No PBL, o problema vem primeiro. Um problema geralmente é uma descrição de um conjunto de fenômenos ou eventos observáveis no mundo real que precisam de uma explicação em termos de uma teoria, um princípio, processo ou mecanismo subjacente. A tarefa dos alunos do PBL é construir essa explicação por meio de discussões em pequenos grupos e por meio de aprendizado autodirigido (SCHMIDT et al., 2009).

O desenvolvimento de competências de engenharia requer a aplicação de conhecimentos técnicos em contextos específicos vinculados à prática profissional. Além disso, as práticas de engenharia também exigem a aplicação de competências transversais, por exemplo: autonomia; liderança ou interação com outras pessoas em equipes interdisciplinares; negociação ou resolução de conflitos; comunicação de maneira eficaz; Gerenciamento de Projetos (SOARES et al., 2013).

O relatório da UNESCO (2010), sobre a área de engenharia, enfatiza que o resultado da aplicação de PBL não apenas aumentou a aprendizagem dos alunos, mas também a capacidade organizacional. Vários efeitos positivos do PBL na aprendizagem dos alunos foram identificados, como exemplo: (a) Promover

abordagens profundas da aprendizagem; (b) Desenvolver a visão crítica dos alunos; (c) Aumentar a consideração do conhecimento e das habilidades interdisciplinares; (d) Desenvolver habilidades de colaboração, comunicação e gestão; (e) Evoluir a identidade profissional e de responsabilidades.

O PBL no ensino de engenharia pode ser utilizado para introduzir uma infinidade de conceitos e competências transversais ao estudante, e para o foco deste estudo o objeto principal de aprendizagem está relacionado a gestão da qualidade de software.

3.2. Gestão da Qualidade de Software

Produtos de software são cada vez mais utilizados para realizar uma ampla variedade de funções comerciais e pessoais, e para que continuem a agregar valor às partes interessadas precisam deter uma preocupação com a qualidade (ISO 25010, 2011).

Existem várias definições sobre o termo qualidade de software, por exemplo o IEEE (1990) define como “o grau em que um sistema, componente ou processo atende aos requisitos especificados”. Já Pressman (2001) define como a “conformidade com requisitos funcionais e de desempenho explicitamente declarados, padrões de desenvolvimento explicitamente documentados e características implícitas que são esperadas de todos os softwares desenvolvidos profissionalmente”.

Segundo Lima (2019), a qualidade de software é uma área de conhecimento que tem como objetivo a garantia da qualidade através da definição e normalização de processos de desenvolvimento e demonstra que existem várias maneiras de definir qualidade de software por se tratar de um conceito dinâmico e relativo, e por isso precisa ser definida como um conceito amplo.

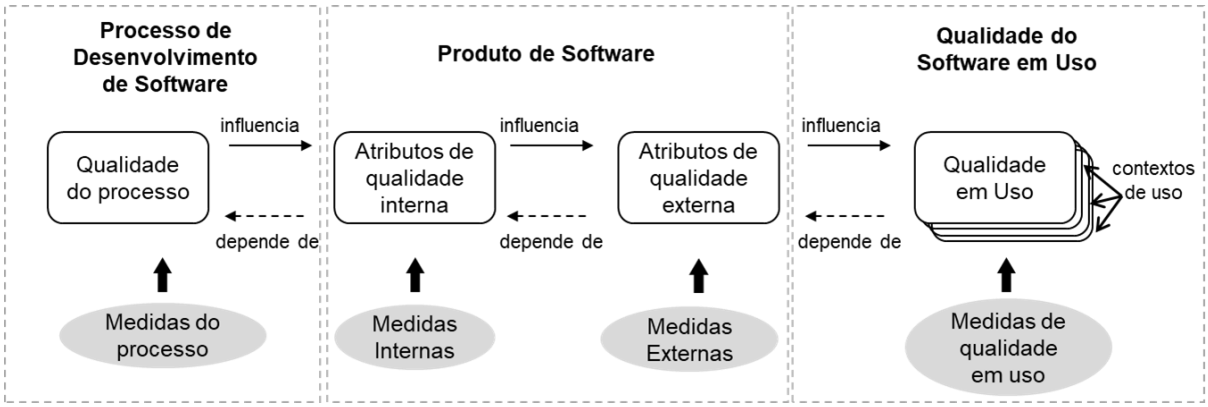
A avaliação e melhoria da qualidade de um software tem sido um dos principais alvos da comunidade de engenharia de software, gerando diversas propostas de padrões, modelos e frameworks para qualidade de software (MARTÍNEZ-FERNÁNDEZ et al., 2019).

Para a Modelo SQuaRE – Requisitos e avaliação de qualidade de produtos de software, a gestão da qualidade de software compreende o exame sistemático do produto de software com a missão de implementar e gerenciar as especificações dos requisitos de qualidade traçados a partir do objetivo do produto e da avaliação da qualidade do software por meio de técnicas, ferramentas e habilidades de gestão

como parte da garantia da qualidade, controle de qualidade e certificação de qualidade (ABNT NBR ISO/IEC 25001, 2009; ISO/IEC 25023, 2016).

A avaliação de qualidade de produtos de software é um dos processos do ciclo de vida de desenvolvimento, pois tem como objetivo satisfazer as necessidades de qualidade do produto. A qualidade do software pode ser avaliada medindo a qualidade interna (medidas estáticas de produtos intermediários), a qualidade externa (medida de comportamento do código quando executado) ou a qualidade do software em uso. O objetivo final do processo de desenvolvimento é a ocorrência do efeito requerido em um contexto específico de uso, conforme representado na Figura 12 (ABNT NBR ISO/IEC 25000, 2008).

Figura 12. Qualidade no ciclo de vida do software segundo o modelo SQuaRE



Fonte: Adaptado de ABNT NBR ISO/IEC 25000, 2008.

O modelo SQuaRE da ISO/IEC 25010 (2011) é amplamente adotado na indústria e na academia, e determina os aspectos de qualidade a serem levados em consideração ao avaliar as propriedades de um software (MARTÍNEZ-FERNÁNDEZ et al., 2019). Grande parte dos modelos de qualidade aplicados atualmente tem base conceitual no modelo proposto pela ISO/IEC, incialmente na ISO/IEC 9126-1 que foi substituída pelo modelo SQuaRE da série ISO/IEC 25000 (WAGNER et al, 2012; MARTÍNEZ-FERNÁNDEZ et al., 2019).

3.2.1. Métricas de Qualidade de Software

Segundo Pressman (2001), métricas de software se refere a uma ampla gama de medidas para softwares que podem ser aplicadas ao processo de desenvolvimento com a intenção de aprimorá-lo continuamente. Essas medições podem ser utilizadas em todo o projeto de software para contribuir com a estimativa e controle de qualidade além da avaliação de produtividade e controle do projeto.

Na gestão da qualidade de software, coletam-se medidas, desenvolvendo métricas para que indicadores sejam obtidos. Um indicador de qualidade é uma métrica ou combinação de métricas que fornecem uma visão sobre o processo, projeto ou o próprio produto (PRESSMAN, 2001).

Segundo o IEEE (1990), métrica de qualidade de software pode ser definida como “uma função cujas entradas são dados de software e cuja saída é um único valor numérico que pode ser interpretado como o grau em que o software possui um determinado atributo de qualidade.”

O uso de métricas estabelece como o software deve estar em conformidade com os requisitos implícitos e explícitos do cliente. Ou seja, a medição para que o sistema se adapte aos requisitos que o cliente estabelece é observada nas métricas de qualidade que são utilizadas para avaliar e controlar o processo de desenvolvimento de software, de forma que permita: (1) Indicar a qualidade do produto; (2) Avaliar a produtividade dos desenvolvedores; (3) Justificar o uso de novas ferramentas e formações adicionais; (4) Estabelecer uma linha base para estimar a qualidade; e (5) Avaliar os benefícios em termos de produtividade e qualidade (CASTILLO, 2018).

As métricas estão presentes em todo o ciclo de vida do software (Figura 12) podendo estar relacionadas ao processo de desenvolvimento (métricas de processo), a estrutura interna do código gerado (métricas internas), a performance do código quando é executado (métricas externas) e a interação usuário-software (métricas de qualidade em uso) (ABNT NBR ISO/IEC 25000, 2008).

Essa amplitude de métricas, possibilita a existência de vários estudos, modelos e ferramentas de aplicação dessas medidas para a avaliação da qualidade, contexto no qual se apresenta a ferramenta Q-Rapids (MARTÍNEZ-FERNÁNDEZ et al., 2019).

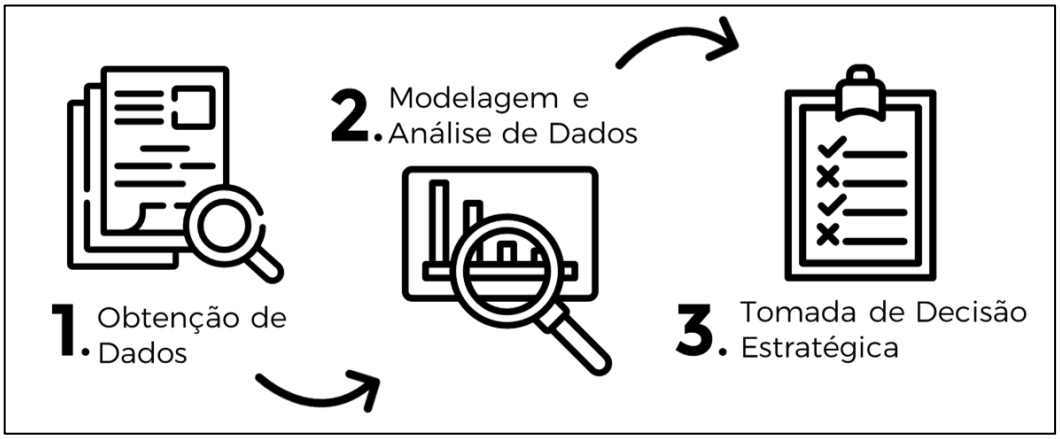
3.3. A ferramenta Q-Rapids de análise de software

A ferramenta do Desenvolvimento Rápido de Software Baseado em Qualidade – Q-Rapids, do inglês, *Quality-Aware Rapid Software Development* – tem como objetivo promover a integração dos modelos de qualidade consolidados existentes com ferramentas de análise de software, fornecendo informações compreensíveis, confiáveis, úteis e relevantes com certo nível de detalhe sobre a qualidade de um processo ou produto (MARTÍNEZ-FERNÁNDEZ et al., 2019).

A Q-Rapids foi desenvolvida em colaboração entre parceiros da academia e da indústria e, além de iterativamente unir conceitos de análise de dados voltado para o desenvolvimento de software, tem como característica principal a customização de aplicação, não só pela utilização de ferramentas de análise de acesso fácil disponíveis no mercado, mas também como um balizador no processo de definição de indicadores estratégicos relacionados à qualidade do produto de software, mas também as necessidades da empresa (MARTÍNEZ-FERNÁNDEZ et al., 2019).

A ferramenta foi construída com base em uma hierarquia de cinco entidades. (1) Indicadores Estratégicos, definidos como aspectos relacionados a qualidade em que a organização considera relevante para o processo de tomada de decisão. Um indicador estratégico pode ser influenciado por uma série de (2) Fatores de Processo/Produto, os quais são atributos ou partes do processo/produto que são concretos o suficiente para serem mensurados. Um fator de produto/processo pode ser composto por uma série de (3) Métricas Mensuradas, as quais consistem em descrições numéricas que são obtidas em determinado contexto. Para a obtenção de métricas mensuradas é preciso ter uma série de (4) Dados Puros que devem ser extraídos e obtidos por uma variedade similar de (5) Fonte de Dados (MARTÍNEZ-FERNÁNDEZ et al., 2019).

Figura 13. Módulos da aplicação da ferramenta Q-Rapids



Fonte: Adaptado de Martínez-Fernández et. al, 2019

A ferramenta foi construída para a aplicação em módulos de repetição contínua, conforme apresentado na Figura 13. O módulo de obtenção de dados (1) é composto por uma série de diferentes conectores *open-source* para obtenção de dados, sendo eles focados em análise de código, ferramentas de integração contínua, repositórios

de código, ferramentas de acompanhamento de tarefas e *logs* de uso de software (MARTÍNEZ-FERNÁNDEZ et al., 2019).

O módulo de modelagem e análise de dados usa os dados obtidos para avaliar a qualidade de software por meio de ferramentas de análise de dados. Esse módulo além das soluções apresentadas pelo próprio Q-Rapids tem alto grau de customização para realidade de desenvolvimento de várias organizações, podendo estas criarem indicadores estratégicos, fatores de processo/produto, métricas mensuradas e até outras formas de relação entre as entidades e a frequência de medição (MARTÍNEZ-FERNÁNDEZ et al., 2019).

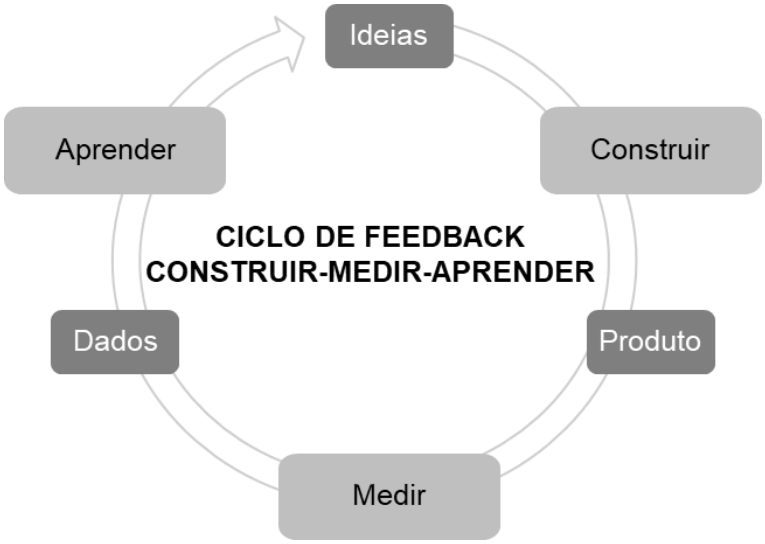
O módulo de tomada de decisão estratégica é responsável pela interface final do usuário, sendo esse usuário o tomador de decisão estratégica. Esse módulo é construído sobre dois dashboards, sendo: (1) Dashboard Estratégico: interface visual que compila e resume informações sobre a qualidade do produto de software; e (2) Dashboard de dados puros: interface que apresenta uma visualização específica sobre as informações reunidas do módulo de obtenção de dados. O dashboard de dados puros existe para o tomador de decisão tomar ações sobre aspectos específicos e melhorar a qualidade geral apresentada no dashboard estratégico (MARTÍNEZ-FERNÁNDEZ et al., 2019).

Um dos fatores relevantes apontados não só pela ferramenta, mas em boa parte dos modelos de qualidade, está relacionado a produtividade, a qualidade do processo de desenvolvimento desde a ideação até a avaliação da qualidade em uso. Existem métodos para a garantia da qualidade em todas as fases do ciclo de vida, e o Lean Startup e o Lean Inception representam os utilizados no contexto de desenvolvimento mais prevalente na indústria atualmente, o de metodologias ágeis, para as etapas iniciais do processo de desenvolvimento (LIZARELLI et al, 2021).

3.4. Lean Startup

O *Lean Startup* ou a Startup Enxuta se popularizou com as ideias de Ries (2012) e Blank (2013) e atraiu interesse de vários empreendedores, startups, inovadores e pesquisadores (LIZARELLI et al, 2021). A metodologia compreende uma abordagem de desenvolvimento de novos negócios baseado nos princípios da manutenção enxuta, ou *Lean Manufacturing,* desenvolvidos no Japão pela Toyota (RIES, 2012).

Figura 14. Representação do ciclo de feedback construir-medir-aprender do Lean startup



Fonte: Adaptado de Ries, 2012.

Ries (2012) descreve o método da startup enxuta com base em um ciclo básico de feedback construir-medir-aprender. Uma abordagem voltada para a experimentação, feedback do cliente e design iterativo (Blank, 2013). O contexto do startup enxuta foi construído em torno de princípios chave: (1) formular hipóteses sobre o negócio/ideia de produto; (2) desenhar um modelo de negócios incorporando essas hipóteses; (3) Construir um Mínimo Produto Viável – MVP, do inglês *Minimum Viable Product* – a fim de replicar as funções principais do produto e testar o modelo de negócios; (4) Rodar testes e experimentações com vários ciclos para a partir de dados, aprender sobre o produto e decidir permanecer nas hipóteses iniciais ou pivotar desse curso inicial de ação, isto é, criar novas hipóteses e recomeçar o ciclo, conforme apresentado na Figura 14.

3.4.1. Mínimo Produto Viável

O MVP é uma versão do produto com o menor agrupamento de funcionalidades, que permite uma volta completa no ciclo construir-medir-aprender, construído com o menor tempo, recursos e esforço, chegando logo ao mercado para realização de testes de hipótese (RIES, 2012; LIZARELLI et al, 2021).

Segundo Ries (2012), ao contrário de um produto tradicional, que envolve um período de incubação longo e ponderado, o objetivo do MVP é começar o processo de aprendizagem, projetado não só para responder as perguntas de viabilidade

técnica ou de design do produto, mas também para validade das hipóteses fundamentais do negócio.

Os MVPs variam em complexidade, podendo ser testes muito simples, como um vídeo ou anúncio, até protótipos iniciais reais. Não existe uma fórmula exata para determinar a complexidade do MVP, é necessário pesquisa e julgamento em relação a hipótese de valor, mas o próprio ciclo de feedback pode suportar esse processo. Uma vez desenvolvido um MVP em que os clientes não conseguiram entender como e porque utilizá-lo, isso confirmará a necessidade de investir em um projeto mais complexo (RIES, 2012).

Segundo Ries (2012), o papel da qualidade no MVP é diferente das noções tradicionais de qualidade. Para ele as energias devem estar concentradas exclusivamente na produção de resultados que o cliente percebe como dotados de valor, mas essa afirmação parte do pressuposto de que se conhece os atributos que serão valorizados pelo cliente. A partir disso, Ries (2012) afirma que mesmo um MVP lido pela perspectiva tradicional como de “baixa qualidade” pode fornecer insights importantes para o desenvolvimento de um produto de alta qualidade.

A construção de um MVP leva em consideração uma regra simples: é preciso eliminar todo recurso, processo ou esforço que não contribui para a aprendizagem pretendida. Ou seja, o MVP precisa ser suficientemente simples para prevenir retrabalho e desperdício de recursos antes da validação de hipóteses e suficientemente complexo para permitir a interação com o cliente e a coleta de dados para gerar as conclusões pretendidas (RIES, 2012).

Para Caroli (2018), o MVP é a intersecção entre o valioso, usável e factível representando, respectivamente, o interesse do negócio, a aceitação dos usuários e o que é possível construir. O MVP, ainda, proporciona uma construção de forma contínua e incremental, com funcionalidades validadas sendo adicionadas ao produto já existente, enquanto o processo de criação de produtos tradicional não fornece qualquer valor até o final, quando todo o processo está pronto. Nesse cenário, é importante ter uma visão ampla do produto, bastante abrangente, mas é necessário começar pequeno e aprender rápido, e uma forma de fazer isso é por meio do MVP. (CAROLI, 2018)

Segundo Caroli (2018), com a Lean Startup surgiram boas respostas sobre o que medir e aprender em um desenvolvimento ágil incremental, mas pouco se tinha informações em relação a como construir um MVP que garantisse a medição e o

aprendizado pretendido. Nesse contexto, Caroli (2018) desenvolveu uma sequência de atividades para balizar a construção, com foco na ideação, chamada de Lean Inception.

3.5. Lean Inception

Nos princípios de Lean Startup e Lean UX – Experiência do Usuário, do inglês *User Experience* (GOTHELF; SEIDEN, 2013), Caroli (2018) percebeu que apresentavam boas técnicas para os requisitos de aprender e medir o ciclo de feedback, mas faltava um direcionamento para o que construir. A Lean Inception surgiu como uma forma de definir uma sequência de atividades que fosse capaz de contribuir para a definição das funcionalidades de um MVP (CAROLI, 2018).

> Durante a Lean Inception, atividades dinâmicas acontecem para definir objetivos, estratégias e definições do produto, bem como mapear, priorizar as funcionalidades desejáveis para serem entregues gradualmente, construindo o MVP. O principal objetivo do workshop de Lean Inception é fazer com que a equipe descubra e compreenda coletivamente o que vai ser desenvolvido. No final, o time deve estar mais entrosado e deve obter uma visão clara do caminho a ser seguido (CAROLI, 2018).

Figura 15. Agenda da Lean Inception

Fonte: Caroli, 2018.

As atividades de uma Lean Inception são programadas para ocorrer durante uma semana, e por isso a agenda é definida em um cronograma de segunda à sexta, conforme apresentado na Figura 15.

Dia 1 – Segunda-feira – Construindo a Visão do Produto: Além de contextualizar o workshop para os participantes, é destinado a elaboração da visão do produto. Essa visão ajuda a trilhar o caminho inicial, entre a ideia e o lançamento,

que define a essência do valor de negócio e deve refletir uma mensagem clara e convincente para os clientes. Ao final do dia, após a utilização dos *templates* de Visão do Produto e O Produto É – Não é – Faz – Não faz (ENFN, abreviado), a equipe caminhará para a definição final dos objetivos do produto (CAROLI, 2018).

Dia 2 – Terça-feira – Personas e Jornada de Usuário: As atividades de caracterização das personas têm como resultado uma descrição inicial de usuários do produto, mas é importante ressaltar que esse resultado não é definitivo, pode ser revisitado após feedback do produto. A partir dos mapas descritivos das personas são levantadas as jornadas que serão realizadas por cada perfil de usuário traçado e suas respectivas interações com o produto em desenvolvimento (CAROLI, 2018).

Dia 3 – Quarta-feira – Funcionalidades e Revisão: Ao final desse dia a equipe deverá ter de forma clara quais funcionalidades o produto deve ter para atender os objetivos descritos no dia 1 e para aprender as necessidades das personas mapeadas no dia 2. Para isso é realizado um *brainstorming* de funcionalidades e em sequência a revisão técnica, de negócio e de *UX*. A revisão é necessária porque é preciso evoluir o entendimento sobre cada uma das funcionalidades levantadas. Então cada uma delas é avaliada em termos de esforço a ser empreendido no desenvolvimento, do valor daquela função para o negócio, do impacto na experiência do usuário (*UX*) e do nível de confiança da equipe de desenvolvimento em termos de como construí-la (CAROLI, 2018).

Dia 4 – Quinta-feira – Interação Função-Jornada e Sequenciador: As jornadas identificam os pontos de interação do usuário com o produto em ideação, e nesse processo contínuo de interação usuário-produto, permite verificar a sequência em que as funcionalidades são utilizadas pelos usuários. E, o processo de priorização realizado com base na revisão do dia 3, contribui para a criação de um sequenciador que apresente a ordem de implementação das funcionalidades levantadas. A partir das funcionalidades sequenciadas, e com foco no que precisa ser validado, é definido o escopo do MVP e seus respectivos incrementos (CAROLI, 2018).

Dia 5 – Sexta-feira – Canvas MVP e *Showcase*: Apresenta o ápice de uma *Lean Inception*, onde é detalhado o MVP e suas funcionalidades sob as perspectivas de *Design Thinking* (como apresentadas no Lean UX) e do *Lean Startup*. O Canvas MVP descreve as personas, as jornadas, a proposta de valor do MVP, as funcionalidades, o custo e cronograma, além do resultado esperado e as métricas para validação de

hipótese de negócio, fornecendo assim uma visão holística do MVP de forma concisa e esquematizada (CORALI, 2018).

4. METODOLOGIA

4.1. Método de Pesquisa

Esta pesquisa tem caráter exploratório e natureza qualitativa tendo como estratégia de pesquisa, um estudo de caso, o qual foi empregado para o acompanhamento da equipe de desenvolvimento de uma *webapp*, o PUMA - Plataforma Unificada de Metodologias Ativas que está sendo desenvolvida para a suportar a aplicação da aprendizagem baseada em problemas (PBL) ao longo das disciplinas descritas no Plano Político Pedagógico do curso de Engenharia de Produção da Universidade de Brasília (DUARTE, 2009; MONTEIRO, 2020).

As técnicas utilizadas para coleta de dados foram pesquisa bibliográfica, análise documental, entrevista semiestruturada, observação de reuniões de validação e planejamento.

4.2. Estruturação da pesquisa

Para a realização desde trabalho, os objetivos foram divididos em duas fases, sendo a Fase 1 – Revisão Sistemática da Literatura, a qual comporta as etapas utilizadas para alcançar os Objetivos específicos 1 e 2; e a Fase 2 – Estudo de caso e aplicação de métricas, comportando as etapas realizadas para atingir os Objetivos específicos 3 e 4, conforme apresentado no Quadro 6.

Quadro 6. Estruturação e etapas da pesquisa

<table>
<tr><th colspan="2">Fase 1</th></tr>
<tr><td colspan="2">Revisão Sistemática da Literatura e Priorização</td></tr>
<tr><td>Objetivo Específico 1</td><td>Etapa 1.1</td></tr>
<tr><td rowspan="5">Compreender os diferentes modelos de qualidade aplicáveis a webapps</td><td>Preparação da Pesquisa</td></tr>
<tr><td>Etapa 1.2</td></tr>
<tr><td>Agrupamento dos Estudos</td></tr>
<tr><td>Etapa 1.3</td></tr>
<tr><td>Análise dos Dados</td></tr>
<tr><td>Objetivo Específico 2</td><td>Etapa 1.4</td></tr>
<tr><td rowspan="3">Priorizar critérios de avaliação da qualidade que possam ser aplicáveis ao PUMA;</td><td>Modelo Integrador</td></tr>
<tr><td>Etapa 1.5</td></tr>
<tr><td>Priorização de Critérios</td></tr>
</table>

Fase 2	
Estudo de caso e Aplicação de métricas	
Objetivo Específico 3	Etapa 2.1
Determinar métricas e forma de obtenção dos dados no contexto de desenvolvimento do Mínimo Produto Viável (MVP) do PUMA	Contextualização do Produto
	Etapa 2.2
	Modelo de Desenvolvimento
	Etapa 2.3
	Qualidade no Processo de Desenvolvimento
Objetivo Específico 4	Etapa 2.4
Analisar as métricas de qualidade no desenvolvimento do MVP do PUMA	Obtenção de Dados
	Etapa 2.5
	Avaliação da Qualidade

Fonte: Elaborado pela autora.

A Fase 1 teve como objetivo desenvolver um modelo integrador capaz de descrever o estado da arte no processo de avaliação da qualidade de aplicações web e gerar uma priorização de critérios de qualidade de software com base nos modelos encontrados. Esta fase, denominada revisão sistemática da literatura, foi dividida em 5 etapas (Quadro 6), sendo:

Etapa 1.1 – Preparação da Pesquisa: Consiste na determinação e validação dos termos de pesquisa, o lapso temporal relevante para o estudo, além das bases de dados a serem utilizadas.

Etapa 1.2 – Agrupamento dos Estudos: Através da leitura dos resumos das publicações obtidas pela busca inicial, foram identificadas características comuns das pesquisas obtidas, principalmente em relação às metodologias, técnicas, normas e/ou frameworks utilizados no desenvolvimento da avaliação de web-apps e gerados agrupamentos distintos.

Etapa 1.3 – Análise dos Dados: Aplicação da apresentação e interrelação dos dados, ou seja, foram obtidos gráficos e mapas que descrevem o comportamento das linhas de pesquisa, suas relações de citação e autoria, além dos termos mais citados nas palavras-chaves, entre outras informações.

Etapa 1.4 – Modelo Integrador: Associação dos resultados obtidos nas etapas anteriores, descrevendo um panorama generalista dos modelos de avaliação de qualidade de aplicações web encontrados.

Etapa 1.5 – Priorização de Critérios: Identificação dos critérios prioritários a partir da análise qualitativa e quantitativa do modelo integrador.

Já a Fase 2 teve como objetivo determinar e calcular métricas de qualidade de software no contexto de desenvolvimento do Mínimo Produto Viável (MVP) do PUMA. Esta fase também dividida em 5 etapas (Quadro 6), sendo:

Etapa 2.1 – Contextualização do Produto: Levantamento de informações sobre o Projeto PUMA e sua missão como um produto de software.

Etapa 2.2 – Método de Desenvolvimento: Diagnóstico e caracterização do método de desenvolvimento sendo utilizado no projeto.

Etapa 2.3 – Qualidade no Processo de Desenvolvimento: Contextualização do modelo de gestão de qualidade a ser utilizado, com foco em critérios de qualidade de software, para a garantia da qualidade do MVP em desenvolvimento, unindo a priorização de critérios delimitada na Etapa 1.5 com contexto de projeto descrito nas Etapas 2.1 e 2.2.

Etapa 2.4 – Obtenção de dados: Determinação da ferramenta de obtenção de dados para cálculo das métricas de qualidade de software identificadas na Etapa 2.3.

Etapa 2.5 – Avaliação da Qualidade: Exposição dos resultados das métricas de qualidade calculadas durante a realização desse estudo segundo parte da ferramenta Q-Rapids, e as conclusões associadas.

5. ESTUDO DE CASO

5.1.1. Etapa 2.1 – Contextualização do Produto

O Projeto PUMA tem como objetivo desenvolver uma plataforma web, denominada Plataforma Unificada de Metodologia Ativa (PUMA). A *webapp*, como o próprio nome sugere, aspira ser uma plataforma que unifique e dê suporte aos processos que envolvem a aplicação de metodologias ativas de aprendizado, principalmente nas disciplinas de Projeto de Sistema de Produção do curso de Engenharia de Produção da Universidade de Brasília (MONTEIRO; CAMPOS; LIMA; MARIANO, 2018).

Nesse escopo, o PUMA tem como objetivo avaliar a eficiência da aplicação da metodologia PBL ao longo da graduação, automatizando o processo de submissão de propostas de projetos que apresentem problemas reais oriundos de diferentes stakeholders, a alocação dessas propostas às disciplinas adequadas até a avaliação e evolução das competências transversais dos alunos, podendo contemplar todas as disciplinas em que a metodologia é utilizada (MONTEIRO; CAMPOS; LIMA; MARIANO, 2018).

Procura-se, com a plataforma, ter ações de melhoria contínua com foco nas seguintes necessidades: (a) Diversificar a captação de projetos para desenvolvimento nas disciplinas; (b) Integrar de forma automatizada os projetos desenvolvidos nas disciplinas de PSPs; (c) Acompanhar o desempenho dos alunos ao longo da graduação com foco nas competências transversais; (d) Triar os projetos com o intuito de selecionar os mais pertinentes para o tripé universitário; (e) Coletar feedbacks de alunos, professores e stakeholders para aprimoramento da aplicação da metodologia ativa; (f) Emitir relatórios e indicadores referentes às disciplinas, evolução do aprendizado de discentes e atuação dos docentes, e (g) Atuar como uma plataforma de integração e divulgação científica com foco nos projetos desenvolvidos com intermédio da plataforma (MONTEIRO; CAMPOS; LIMA; MARIANO, 2018).

A equipe de idealizadores do PUMA está vinculada ao departamento de Engenharia de Produção e ao mestrado em Computação Aplicada da Universidade de Brasília. Para o desenvolvimento da plataforma, uma parceria com o curso de Engenharia de Software foi firmada e o mínimo produto viável começou a ser

desenvolvido por alunos do curso durante disciplinas em que se utilizam de metodologias ativas de ensino e aprendizagem.

Nesse contexto, a equipe de desenvolvimento do MVP é construída por alunos de duas disciplinas do curso de Engenharia de Software, sendo elas Métodos de Desenvolvimento de Software (MDS) e Engenharia de Produto de Software (EPS). Nessa conformação, os alunos de MDS precisam compreender os diferentes ciclos de vida e técnicas de desenvolvimento de software ao gerar código durante o desenvolvimento de um produto. Já os alunos de EPS precisam desenvolver competências como gerenciar de forma sistemática todo o ciclo de vida do software ao planejar, controlar e monitorar o projeto de desenvolvimento de software de forma a otimizar o uso de recursos envolvidos no projeto (NERI, 2021).

A parceria permite que a cada semestre letivo equipes mistas de alunos de MDS e EPS possam trabalhar em conjunto com os idealizadores no desenvolvimento e manutenção constante da plataforma no âmbito da aprendizagem ativa.

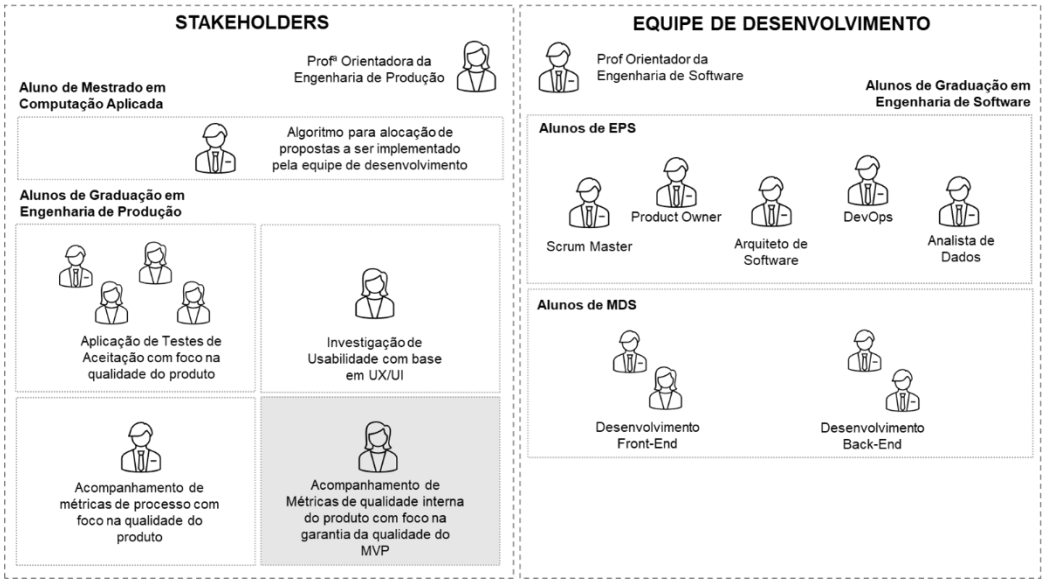
5.1.2. Etapa 2.2 – Modelo de Desenvolvimento

O primeiro semestre letivo de 2021 representa o processo de desenvolvimento do software no formato da parceria descrita anteriormente. Nesse período, o grupo de trabalho responsável pelo projeto era composto por 7 alunos do curso de Engenharia de Produção em estágios diferentes da graduação, 1 aluno do mestrado em Computação Aplicada, 4 alunos da disciplina de MDS, 5 alunos da disciplina de EPS, além dos dois professores orientadores sendo um do curso de Engenharia de Software e outra do curso de Engenharia de Produção, conforme esquema da Figura 16.

No contexto dos stakeholders, foram apresentadas as linhas de pesquisa individuais e, portanto, escopos de atuação no projeto de desenvolvimento de cada um dos participantes. A equipe responsável pela aplicação de testes de aceitação na solução em desenvolvimento é uma equipe de alunos da disciplina de Projeto de Sistemas de Produção 5, que tem como área do conhecimento associada a engenharia de qualidade (SILVA; BALTHAZAR, 2010; MONTEIRO, 2020).

Já para a equipe de desenvolvimento, estão apresentadas o papel de cada membro no desenvolvimento de acordo com as funções preconizadas no modelo de desenvolvimento ágil utilizado associado aos requisitos de avaliação apresentados nos planos de ensino das disciplinas de EPS/MDS (NERI, 2021).

Figura 16. Esquema dos participantes e campos de atuação para o desenvolvimento do MVP do PUMA



Fonte: Elaborada pela autora.

O escopo de atuação deste estudo de caso está apresentado na Figura 16 pelo quadro em cinza. Para o grupo de trabalho global, o escopo de desenvolvimento traçado era identificar o estágio de MVP da plataforma e desenvolver até esse ponto, e para isso, foi utilizada agenda da Lean Inception.

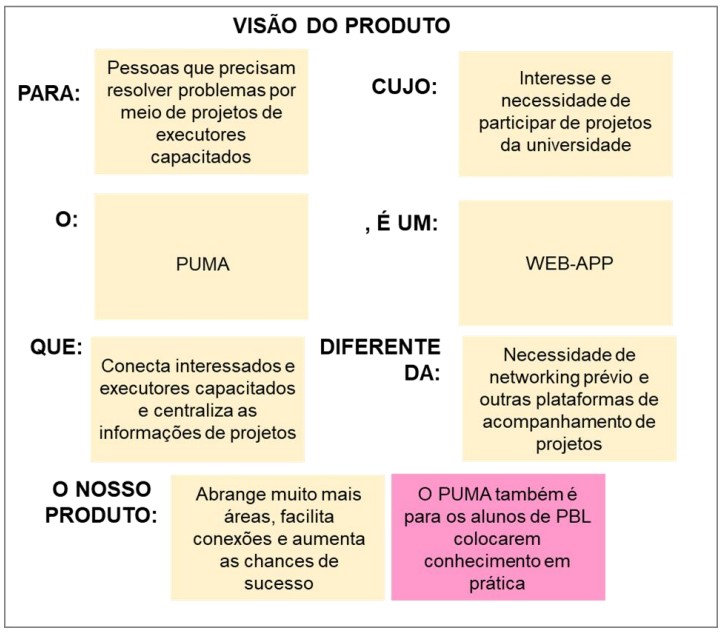
A Lean Inception foi construída pela colaboração de todo o grupo de trabalho apresentado na Figura 16, isso ocorreu, porque pelos próprios princípios da agenda é recomendado que os stakeholders participem ativamente do processo, para que haja o máximo alinhamento de expectativa para a construção do MVP (CAROLI, 2018).

O primeiro dia da agenda é crucial para o entendimento do produto por parte da equipe de desenvolvimento e por isso todo o grupo de trabalho estava presente para a realização e validação desse resultado (CORALI, 2018).

Pelo contexto de desenvolvimento que precisava ser feito durante o período letivo do semestre 1.2021 (SAA UNB, 2021), e a própria disponibilidade de horários dos participantes, os resultados esperados dos dias 1, 2 e o *brainstorming* de funcionalidades foram realizados em um único encontro, por vídeo chamada, utilizando a plataforma de colaboração visual MURAL.

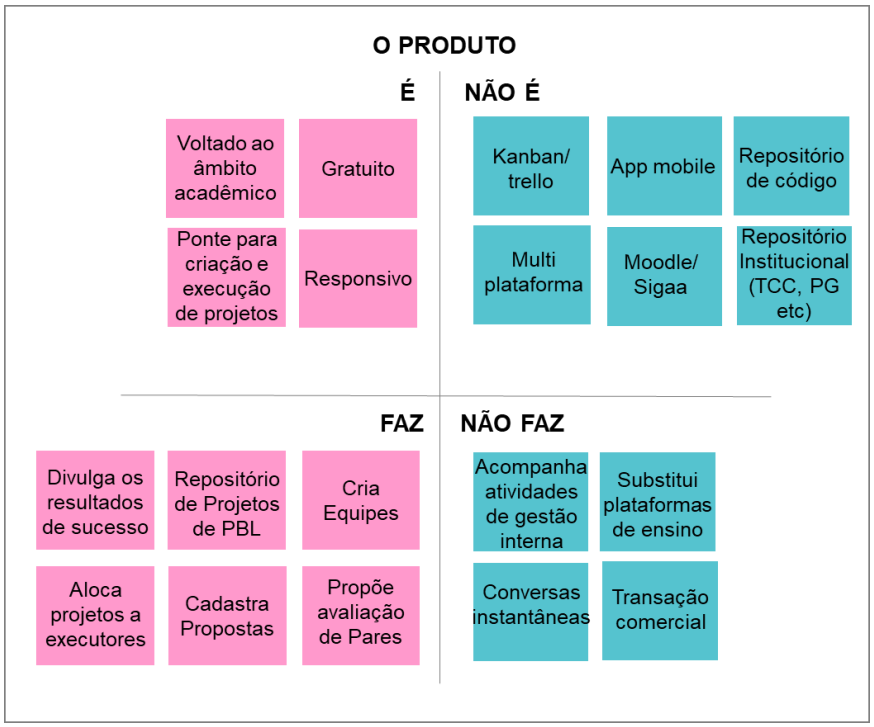
Para a obtenção dos agrupamentos de objetivos do produto como resultado do dia 1 da agenda, foram traçadas a visão do produto e a relação o produto é – não é – faz – não faz, conforme apresentado na Figura 17 e na Figura 18.

Figura 17. Visão do Produto do PUMA a partir da aplicação da Lean Inception



Fonte: Adaptado de PUMA WIKI, 2021.

Figura 18. Resultado da avaliação do Produto é – não é – faz – não faz para o PUMA



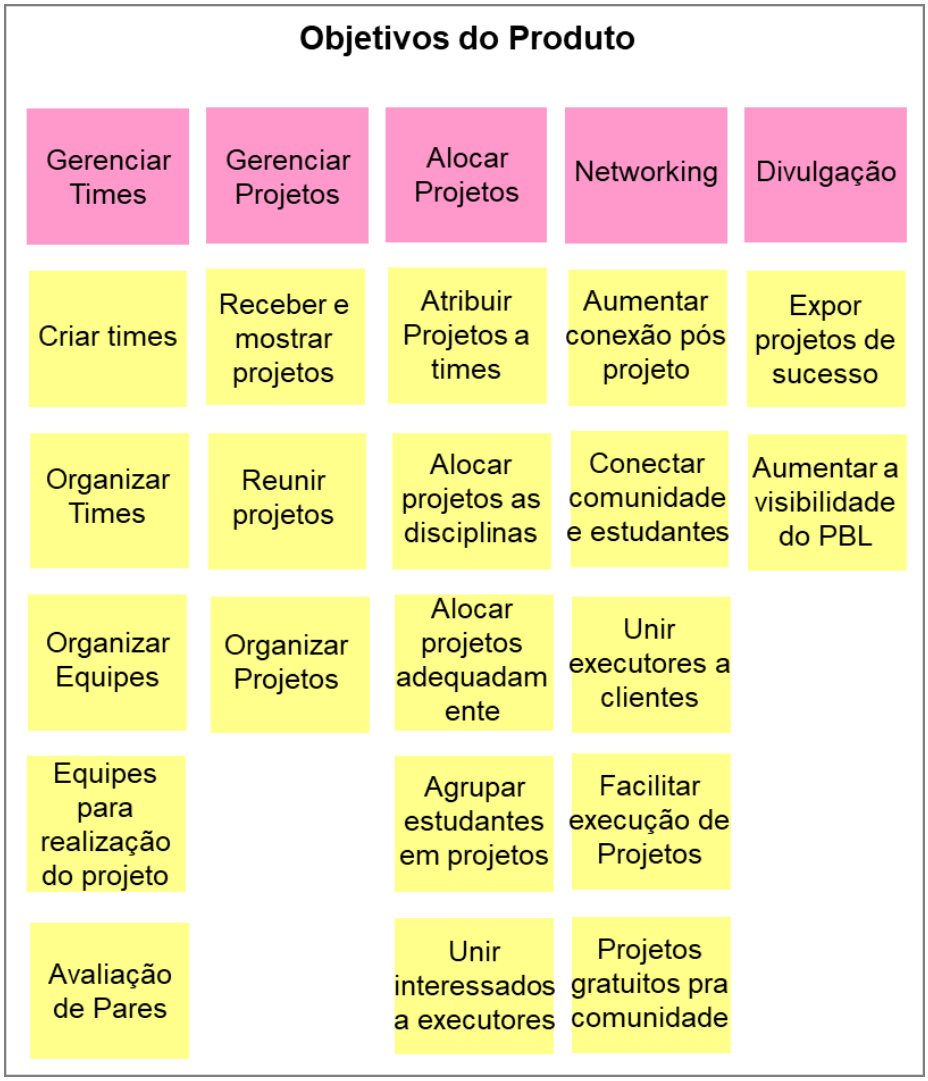
Fonte: Adaptado de PUMA WIKI, 2021.

Esses resultados foram importantes para delimitar o propósito geral da plataforma, não só para a equipe de desenvolvimento, mas também para alinhar o

escopo inicial de desenvolvimento do PUMA para os stakeholders. Segundo Corali (2018), em processos de ideação muitas vezes para delimitar o escopo de um produto que ainda não se sabe a fundo o que é, é mais fácil delimitar o que ele não é e não faz, por isso a atividade do ENFN é tão importante.

A partir do conhecimento alinhado sobre o que é o produto e seu escopo, os objetivos do desenvolvimento foram identificados em um *brainstorming* e agrupados de acordo com a temática principal, conforme apresentado na Figura 19.

Figura 19. Objetivos do PUMA ao final da aplicação da Lean Inception



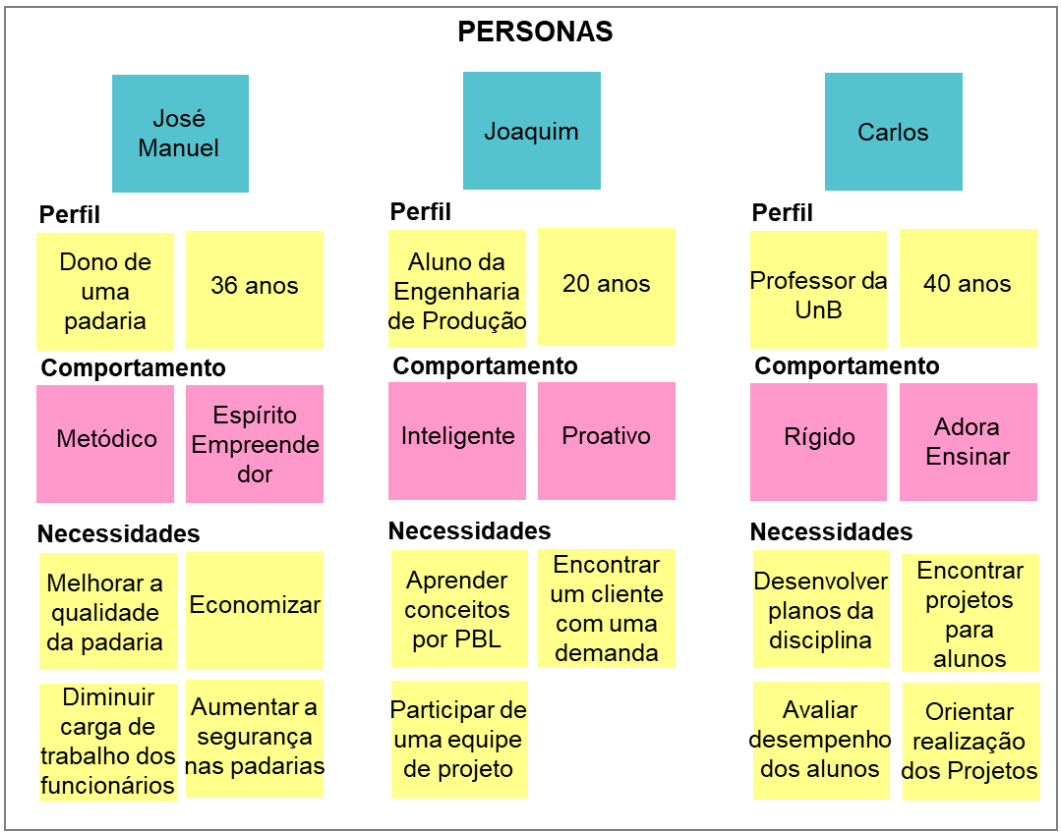
Fonte: Adaptado de PUMA WIKI, 2021.

Os cinco objetivos principais do PUMA identificados foram: (1) Gerenciar times de projeto no escopo amplo de atuação dos professores e alunos, agrupando as atividades de criar, organizar, acompanhar os grupos de alunos que serão executores de projeto; (2) Gerenciar os projetos em todas as suas fases na plataforma, desde a

submissão da proposta pelo agente externo até a execução dela como um projeto; (3) Alocar projetos tanto ao que se refere ao tema do projeto em consonância o tema da disciplina PBL quanto a determinação de executores; (4) Networking como ferramenta de conexão dos agentes externos com as universidade e o papel social do PUMA no tripé universitário; (5) Divulgação de projetos realizados como forma de aumentar a visibilidade do PBL e como plataforma de divulgação das produções científicas realizadas no âmbito dos projetos.

Finalizadas as atividades do dia 1 da agenda, as personas que serão usuárias da plataforma no contexto do MVP foram delimitadas utilizando a técnica dos quadrantes. Na técnica dos quadrantes é preciso determinar um apelido, características de um perfil, o comportamento típico dessa persona, e as necessidades dela relacionadas ao produto em ideação (CORALI, 2018). Os resultados dessa aplicação estão representados na Figura 20.

Figura 20. Personas identificadas pela técnica dos quadrantes



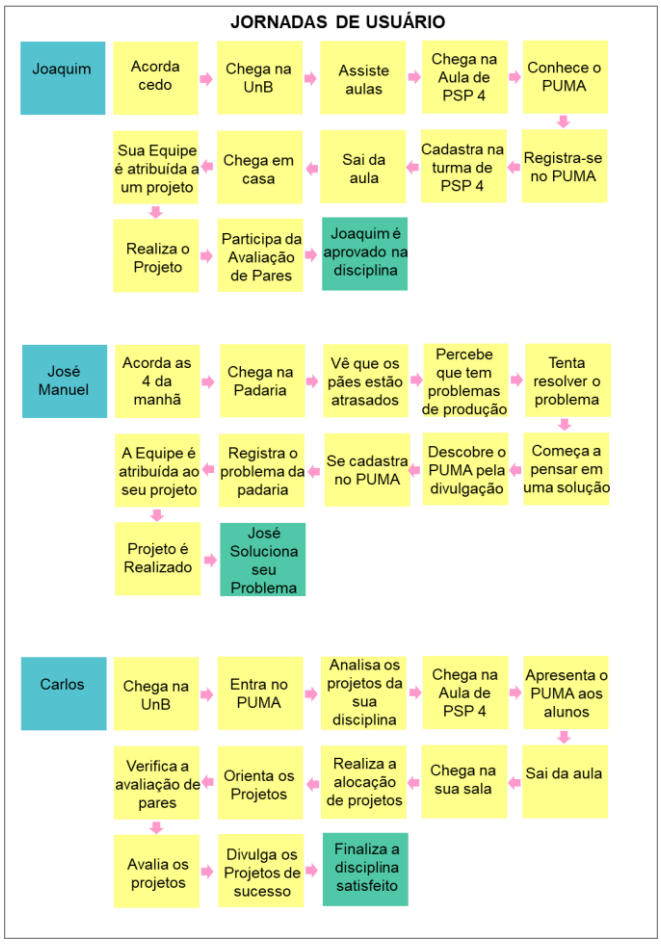
Fonte: Adaptado de PUMA WIKI, 2021.

Uma vez identificadas as personas, as jornadas de usuário foram traçadas. Jornadas de usuário descrevem o percurso de um usuário por uma sequência de passos dados para alcançar um objetivo. A ideia aqui é identificar em que pontos da

jornada que a persona pode traçar o produto pode atuar para auxiliar a suprir a necessidade identificada. Traçar personas e jornadas também é importante para humanizar os usuários, então por isso é relevante pontuar como a persona começa o dia e o que desencadeia o desejo de atingir seu objetivo (CORALI, 2018).

Existem várias formas de traçar jornadas, e a colaboração do grupo de trabalho pode gerar resultados diversos. Era possível e provável que várias jornadas, com graus de complexidade diferentes fossem determinadas, mas em colaboração foi decidido manter níveis mais simples de jornada, porque foi entendido que uma descrição simples já seria suficiente para determinar os pontos de interação persona-produto (Figura 21).

Figura 21. Representação simplificada das jornadas de usuário associadas as personas

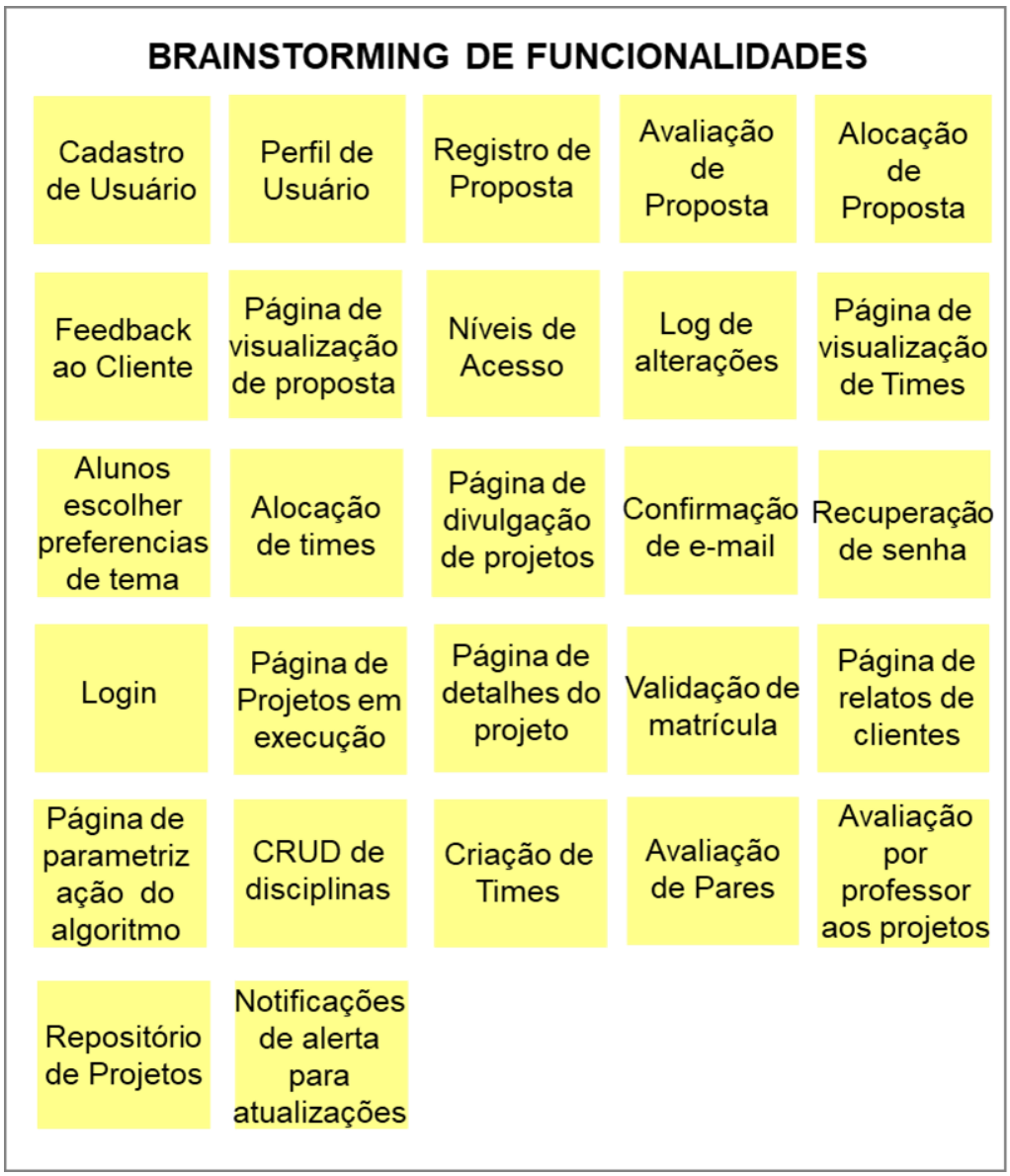


Fonte: Adaptado de PUMA WIKI, 2021.

Com as jornadas de usuário traçadas, as interações persona-produto já estavam claras, e cada ponto de interação é capaz de descrever funcionalidades do produto (CORALI, 2018). Nesse ponto foi aplicado o *brainstorming* de funcionalidades.

Cada membro do grupo de trabalho estava livre para gerar anotações no mural com toda funcionalidade que pudesse identificar não só pela jornada, mas lembrando da visão, o ENFN e os objetivos do produto. Cada funcionalidade era lida em voz alta pelo Scrum Master, e os demais participantes tiravam dúvidas. O objetivo desse debate é que todos tenham entendido a funcionalidade, além de determinar em conjunto se elas atendem o escopo apresentado nas atividades anteriores da Lean Inception. O resultado dessa etapa está apresentado na Figura 22.

Figura 22. Resultado do brainstorming de funcionalidades



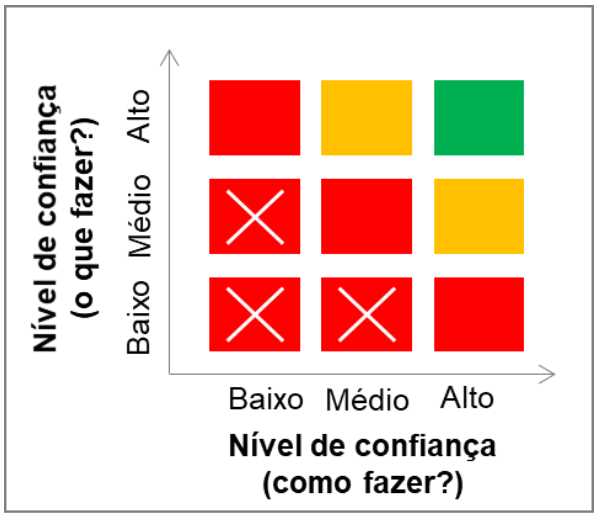
Fonte: Adaptado de PUMA WIKI, 2021.

Nesse ponto, o primeiro encontro de colaboração da lean inception foi encerrado, e as próximas etapas foram construídas pela equipe de desenvolvimento em colaborações pontuais dos stakeholders.

No mesmo painel de colaboração visual, a equipe de desenvolvimento em colaboração com o stakeholder responsável pela garantia da qualidade na usabilidade (Figura 16), aplicou a ferramenta da revisão técnica de negócio e de UX.

Nessa ferramenta cada funcionalidade precisa ser classificada sobre quatro critérios, o esforço, negócio, UX e nível de confiança. Para os três primeiros, a avaliação é mais simples, em uma escala de 1 a 3 é avaliado o esforço a ser desprendido para aquela funcionalidade (E, EE, EEE), o valor daquela funcionalidade para o negócio ($, $$, $$$) e o valor daquela funcionalidade para a experiência do usuário (❤️, ❤️❤️, ❤️❤️❤️). O nível de confiança é um pouco mais complexo de avaliar, porque a equipe de desenvolvimento precisa decidir se sabe qual a intenção desse item de trabalho e se sabe como fazê-lo (CORALI, 2018). Para auxiliar nesse processo de avaliação, existe o gráfico do semáforo (Figura 23).

Figura 23. Gráfico do semáforo



Fonte: Adaptado de Corali, 2018.

Determinando o nível de confiança sobre o que e como fazer no gráfico, cada funcionalidade ganha uma cor, como em um semáforo. Se a cor designada é verde, pode ir tranquilo, caso seja amarelo, precisa prestar atenção e talvez parar um pouco antes de prosseguir, vermelho, precisa parar antes de seguir em frente. Se uma das funcionalidades ficar em algumas das posições com o X, ou ela precisa ser descartada para a ideação do MVP, ou precisa ser esclarecida. O resultado dessa avaliação está apresentado na Figura 24.

Como pode-se observar pela Figura 24, as funcionalidades “Alunos escolher preferências de tema” e “Notificações de alerta para atualizações” foram excluídas. Ambas foram identificadas pelo gráfico do semáforo na região com o X, a primeira foi

excluída pois foi entendido que fazia parte do escopo da funcionalidade de alocação de times, a segunda, porque o nível de confiança era baixo, alto esforço, baixa relevância para o negócio, apesar da avaliação alta em UX.

Figura 24. Revisão técnica, de negócio e de UX

Fonte: Adaptado de PUMA WIKI, 2021.

Com os resultados da revisão técnica, de negócio e de UX finalizada, é preciso definir a sequência de desenvolvimento das funcionalidades, então é o momento fazer a construção do sequenciador.

O sequenciador é construído a partir de ondas numeradas, o intuito é executar o que é mais impactante o mais cedo possível, logo nas primeiras ondas, depois ir

completando o sequenciador onda a onda. Para auxiliar nesse processo, Corali (2018) define algumas regras.

- Regra 1: Uma onda pode conter três cartões, no máximo;
- Regra 2: Uma onda não pode conter mais de um cartão vermelho;
- Regra 3: Uma onda não pode conter três cartões somente amarelos ou vermelhos;
- Regra 4: A soma dos esforços dos cartões não pode ultrapassar cinco “E”;
- Regra 5: A soma dos cartões não pode ser menos de quatro “$” e quatro “❤️”;
- Regra 6: Se um cartão depende do outro deve estar em alguma onda anterior.

Esse processo é comparativo, caso surja alguma dúvida sobre qual funcionalidade incluir nas ondas conforme a sequência, a pergunta que deve ser respondida é: “Qual dessas duas é mais prioritária para o MVP?” (CORALI, 2018).

Os resultados do sequenciador estão apresentados na Figura 25. Como pode-se observar, a Regra 4 foi particularmente difícil de cumprir, e precisou ser ignorada para algumas ondas, principalmente pela análise de relevância ao MVP. Os alunos de MDS foram os principais responsáveis pela avaliação de esforço, por serem a parte equipe responsável pelo desenvolvimento das funcionalidades. Esses alunos estão nos primeiros estágios do processo de aprendizado, porque essa disciplina está prevista para a grade do segundo ano do curso de engenharia de software. Por se tratar de um contexto de aprendizado, é compreensível que a percepção desses alunos, relacionada ao nível de esforço, retorne uma avaliação mais próxima do peso 3 para cada funcionalidade em comparação com a avalição de um engenheiro de software formado e atuando no mercado.

É chegado o momento de entender o escopo do MVP, os incrementos e a possibilidade de evolução do produto. As hipóteses de negócio foram as maiores balizadoras para a decisão, mas o equilíbrio entre os pontos de interação nas jornadas de usuário, o contexto de PBL e a duração do período letivo também foram bastante relevantes na escolha do ponto de Mínimo Produto Viável (CORALI, 2018).

O fim da terceira onda foi determinado como o ponto de MVP (Figura 25), as demais ondas foram traçadas para já ter mapeado os próximos passos de incremento, principalmente para as próximas equipes de desenvolvimento.

Considerando que o grupo de trabalho já discutiu sobre o que compõe o MVP e já alinhou o que se espera dele, essas informações foram reunidas no Canvas MVP. Nele são detalhados os aspectos já conhecidos do MVP, como um resumo de tudo que foi feito até aqui (CORALI, 2018).

Figura 25. O resultado do sequenciador de funcionalidades que apresenta o escopo de desenvolvimento do MVP.

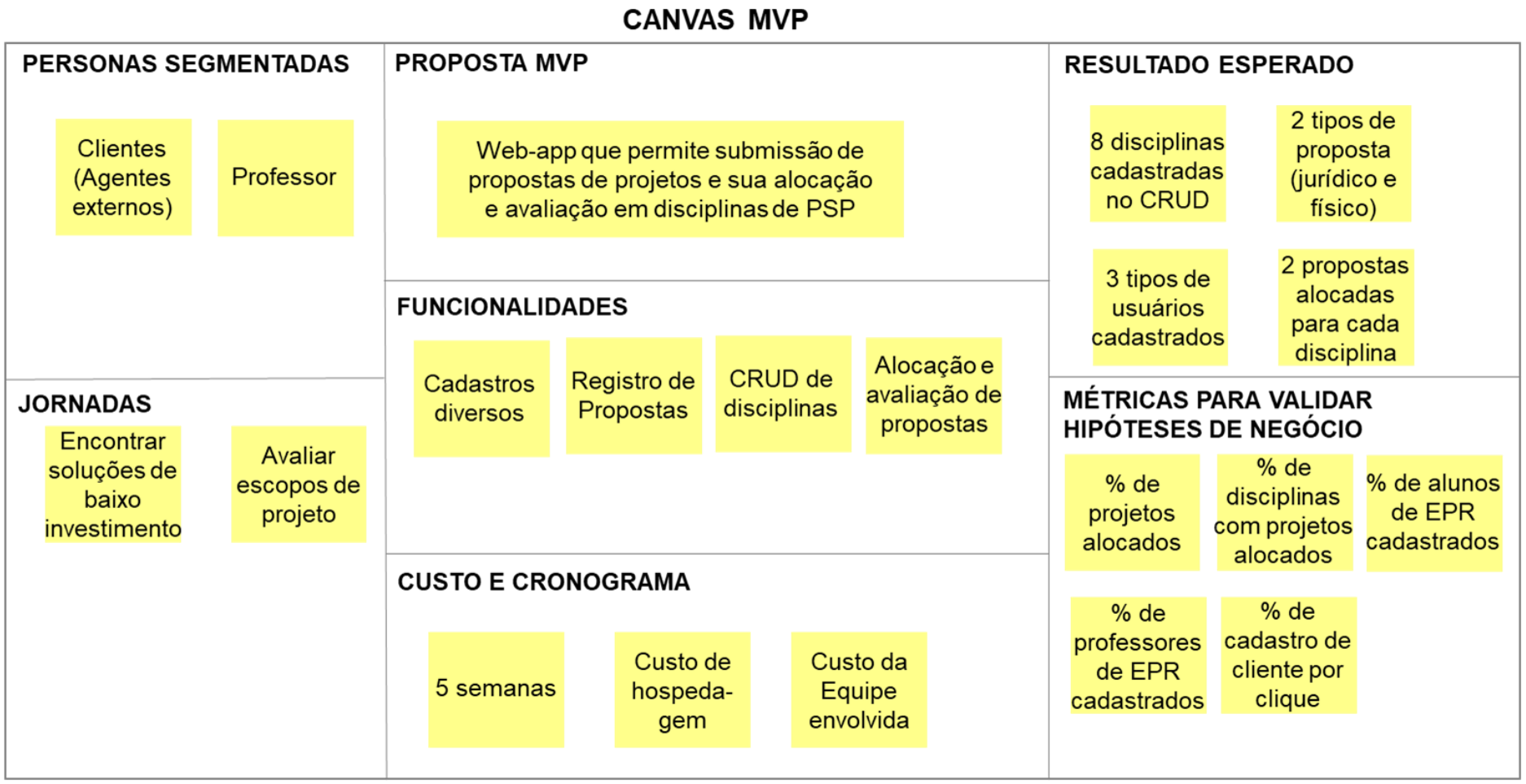
Fonte: Adaptado de PUMA WIKI, 2021.

Cada um dos sete blocos do Canvas MVP precisa responder uma pergunta de forma clara e objetiva, seguido a ordem indicada (CORALI, 2018):

1. Proposta MVP – Qual é a proposta deste MVP?
2. Personas segmentadas – Para quem é esse MVP? É possível segmentar e testar o MVP em um grupo menor?
3. Jornadas – Quais jornadas são atendidas com esse MVP?
4. Funcionalidades – O que vamos construir nesse MVP?
5. Resultado esperado – Que resultado estamos buscando neste MVP?
6. Métricas para validar hipóteses do negócio – Como podemos medir os resultados desse MVP?
7. Qual o custo e entrega prevista desde MVP?

Todas essas respostas estão apresentadas na Figura 26. O MVP foi definido até a alocação de projetos às disciplinas, então a proposta de valor é ser uma *webapp* que permite a submissão de propostas de projeto e a alocação delas a disciplinas de acordo com a área do conhecimento a partir do julgamento do professor. Com o escopo definido, as personas foram segmentadas como o agente externo e o professor – representados pelo apelido de José Manuel e Carlos na Figura 20. Essa definição está diretamente ligada aos pontos de interação persona-produto das jornadas de usuário (Figura 21) que compreendiam esse escopo, assim, já fornecia a resposta para o preenchimento da sessão jornadas do Canvas MVP.

Figura 26. O Canvas MVP apresentado no Showcase



Fonte: Adaptado de PUMA WIKI, 2021.

As funcionalidades presentes no sequenciador até o MVP (Figura 25) foram agrupadas em 4 de acordo com as características, sendo elas Cadastros diversos, Registro de propostas, o CRUD de disciplinas e a Alocação e avaliações de propostas às disciplinas.

No bloco resultado esperado, foram traçadas metas mensuráveis para o sucesso do desenvolvimento do MVP conforme apresentado na Figura 26. As métricas associadas para a validação do PUMA como proposta de valor geral de negócio são da aceitação do MVP, e por esse motivo elas foram associadas a adesão de possíveis usuários relacionados as personas.

5.1.3. Etapa 2.3 – Qualidade no Processo de Desenvolvimento

Uma vez delimitado o MVP para equipe de desenvolvimento, não só para a garantia da qualidade, mas também pela própria metodologia e programa das disciplinas de EPS e MDS (NERI, 2021), rituais e métodos ágeis foram utilizados para o gerenciamento do projeto/processo do desenvolvimento da solução.

Os resultados dessa aplicação não serão discutidos no detalhe neste trabalho, pois é foco de outra pesquisa do grupo de trabalho (Figura 16), entretanto é relevante apontar que o processo de desenvolvimento foi pautado em entregas parciais para validação gradual dentro da própria construção do MVP.

Cada uma das funcionalidades foi traduzida na perspectiva de histórias de usuário de cada uma das personas identificadas e seriadas em consonância com o sequenciador obtido pela Lean Inception.

Segundo Braga et al. (2020), a própria aplicação da Lean Inception é uma forma de garantir a qualidade para os stakeholders. E ainda, pelos resultados apresentados na Etapa 1.5 – Priorização de Critérios, os dois critérios mais relevantes para avaliação de qualidade foram Manutenibilidade e Usabilidade, o que corrobora com todos os princípios por trás da construção de um MVP. Ele precisa ter um certo nível de usabilidade para que os usuários em seus contextos de uso possam gerar dados e *insights* sobre o produto, e precisa de manutenibilidade o suficiente para que possam ser inseridas melhorias e funções graduais para construção de um produto agregado de alta qualidade (LIZARELLI et al, 2021).

Com base nos próprios conceitos do Lean Startup e consequentemente da Lean Inception, o objetivo do MVP é intermediário nos termos de qualidade do produto, porque em sua concepção não precisa ter níveis elevados de qualidade, mas precisa

ser considerado como um caminho para alcançar tais níveis (CORALI, 2018). Como o desenvolvimento do PUMA está sendo realizado por meio da aplicação das metodologias PBL no âmbito dos cursos de Engenharia de Produção e Engenharia de Software, o grupo de trabalho apresentado na Figura 16 é dinâmico pelo próprio fluxo de aprovação nas disciplinas e avanço dos alunos na graduação e mestrado.

Além de no grupo de trabalho (Figura 16) já existir uma pesquisa com foco em usabilidade, a dinâmica da mudança da equipe de desenvolvimento, fez com que o foco da aplicação de métricas fosse relacionado a manutenibilidade.

Para a equipe de desenvolvimento estava prevista a atuação de um analista de dados (Figura 16) como escopo da disciplina de EPS justamente para aplicação de métricas no âmbito global do desenvolvimento do produto segundo a ferramenta Q-Rapids, essa atuação ficou limitada durante o desenvolvimento. Isso ocorreu porque os membros da equipe que estavam matriculados na disciplina de MDS não engajaram o suficiente desde a aplicação da Lean Inception o que impactou na geração dos códigos e na própria atuação dos alunos de EPS.

Pelo acompanhamento das reuniões de revisão e de planejamento das sprints, e do próprio kanban e repositório de código do projeto nenhuma história de usuário foi completa sem que um aluno de EPS precisasse tomar responsabilidade individual e atuar diretamente na geração e correção de código do escopo dos alunos de MDS.

É importante ressaltar que devido a pandemia da Covid-19 essa atuação foi remota, e de acordo com a política da UnB para os semestres que ocorreram durante o modelo remoto para cursos presenciais – que é o caso da Engenharia de Software – os alunos poderiam realizar a retirada da disciplina sem que essa ação impactasse no índice de rendimento acadêmico dos alunos até o último dia previsto para o fim do período de aulas do 1/2021 (SAA UNB, 2021).

Nesse cenário um dos alunos da disciplina de EPS e dois alunos da disciplina de MDS utilizaram dessa retirada durante o andamento da disciplina, e a possibilidade de realização desse procedimento administrativo dificultou o engajamento dos alunos de MDS como parte da atuação dos alunos de EPS como gestores da equipe de desenvolvimento.

Nos pressupostos da gestão ágil, esse contexto fez com que as funções fossem redistribuídas e as expectativas de entrega por parte dos stakeholders foram realinhadas. Para um novo escopo de entrega, as funcionalidades de “Redefinição de

Senha” e “Alocação de Proposta” (Figura 25) deixaram de fazer parte do escopo da entrega de encerramento das disciplinas de MDS e EPS.

5.1.4. Etapa 2.4 – Obtenção de Dados

Não só por se tratar de uma ferramenta já ensinada e utilizada no contexto de ensino da disciplina de EPS, o método de obtenção e avaliação de dados escolhido para acompanhar a qualidade do desenvolvimento por meio de métricas foi a ferramenta Q-Rapids. A base conceitual da ferramenta ter sido construída na ISO/IEC 25010 e outros modelos baseados na norma, possibilidade de adequação e customização da ferramenta para diferentes realidades de desenvolvimento, também foi parte importante dessa escolha (MARTÍNEZ-FERNÁNDEZ et al., 2019).

O repositório de código Github integrado a ferramenta de gerenciamento ágil Zenhub foi utilizada para a gestão do conhecimento no andamento do projeto, sendo essas as maiores fontes de obtenção de dados para a aplicação de métricas. Os códigos no Github estão disponíveis em acesso livre, tornando este um projeto de *open-source*.

A ferramenta Q-Rapids apresenta a ligação com soluções para análise de software *open-source*, amplamente utilizadas e validadas na indústria de desenvolvimento, para obtenção de medidas relacionadas a métricas de qualidade. Por estarem inseridos na filosofia *open-source*, oferecem a utilização gratuita para projetos de código aberto. O alinhamento de filosofia entre as soluções indicadas pela Q-Rapids e o PUMA, além da contribuição para a diminuição do custo do projeto, também foram critérios relevantes para a seleção da ferramenta (MARTÍNEZ-FERNÁNDEZ et al., 2019).

Com o foco delimitado em métricas de manutenibilidade, a Q-Rapids apresenta o SonarQube como a ferramenta automatizada de revisão de código possível para a obtenção das medidas que compõe as métricas desse critério de qualidade. De fato, essa ferramenta de revisão, dentre as que tem versão gratuita de teste, é a que detém a documentação mais completa (SONARQUBE SA, 2021), mas não é a única disponível.

Todas as soluções Sonar oferecem avaliações de qualidade segundo uma série de boas práticas de codificação, incluindo guias altamente reconhecidos e utilizados por profissionais da área, como por exemplo o Guia de Referência – As 10

vulnerabilidades de segurança críticas conjunto de boas práticas de codificação do OWASP (SONARSOURCE SA, 2021; OWASP FOUNDATION, 2017).

O SonarCloud é uma ferramenta da mesma natureza, apresentada pela mesma empresa, que obtém a mesma série de medidas, e tem uma integração ao Github simplificada para projetos *open-source*. Essa integração, apresenta ainda a possibilidade de acompanhamento dos dados em um dashboard acessado pela web, e por essas facilidades e similaridades ao SonarQube foi a ferramenta escolhida para este projeto (SONARSOURCE SA, 2021).

O entendimento de grande parte dos modelos de qualidade é de que a Testabilidade se apresenta como uma sub-característica do critério de qualidade Manutenibilidade (vide Capítulo 2), na ferramenta Q-Rapids os indicadores relacionados a essas sub-características estão distribuídos sobre aspectos de qualidade Confiabilidade e Produtividade, por corroborar com o entendimento de Dromey (1995) de que os critérios/fatores não são mutualmente exclusivos.

A Testabilidade foi acompanhada durante o desenvolvimento, exclusivamente pela utilização do SonarCloud, a qual compreende avaliações sobre esse fator, entretanto a hierarquização da Q-Rapids limitou a apresentação desses resultados no escopo priorizado da Manutenibilidade e seus fatores.

Para a compreensão de cada pacote de dados, aqui é relevante uma pequena contextualização em relação a arquitetura de software traçada pela equipe de desenvolvimento.

O modelo de arquitetura selecionado foi o de microsserviços, nessa arquitetura a aplicação desenvolvida é dividida em pacotes que possuem pouca dependência entre eles, e se comunicam através de mecanismos leves (FOWLER, 2007).

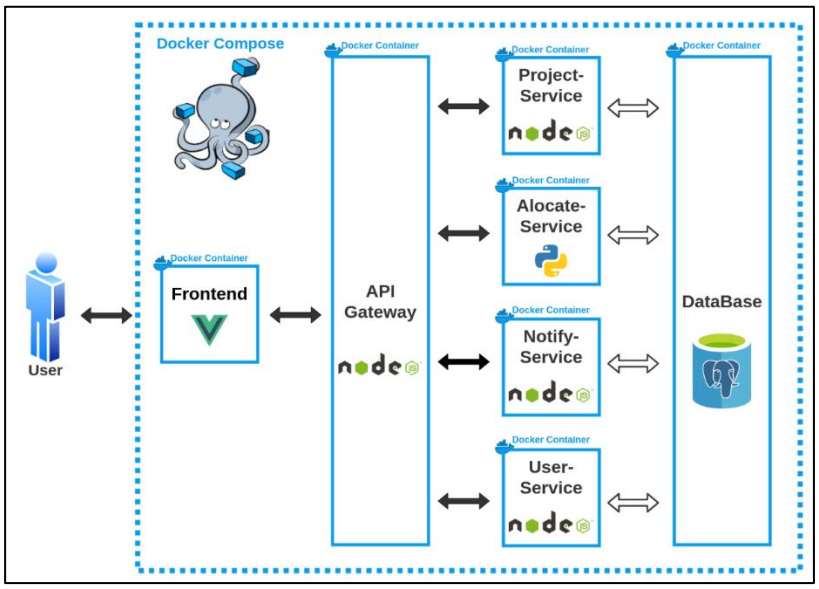
Segundo Rodrigues e Pinto (2019), esse tipo de arquitetura facilita a escalabilidade, não se limita a uma única linguagem, além de facilitar o processo de desenvolvimento de equipes com atuação remota.

Para o PUMA foi construída uma arquitetura em 7 pacotes (PUMA WIKI, 2021), conforme apresentado na Figura 27. Cada um deles tem a sua função delimitada:

- O Front End é a interface aonde o usuário irá se comunicar com o sistema;
- A API Gateway é uma ponte que recebe a requisição do front end e conecta ao serviço desejado;

- O Project-Service é o responsável por executar todas as tarefas envolvendo os projetos, como o envio de propostas, acompanhamento de projetos;
- O Alocate-Service é o alocador, onde o algoritmo essencial do produto para a alocação automática das propostas será executado;
- O Notify-Service é destinado a quaisquer necessidades de alerta aos usuários, tanto por e-mail ou por meio da própria plataforma;
- O User-Service é controle de usuários, desde a criação de usuários, controle de rotas de acesso e criação de times;
- O DataBase é banco de dados onde está armazenado todos os dados da aplicação.

Figura 27. Arquitetura do PUMA



Fonte: Adaptado de PUMA WIKI, 2021.

O código de cada serviço da arquitetura está armazenado em um repositório diferente no Github das disciplinas de EPS e MDS (2021), o SonarCloud mensura e extrai dados de cada repositório separadamente em momentos que são designados pelo analista de dados (Figura 16). Essas medições foram realizadas a cada momento em que uma história de usuário foi completamente entregue no ciclo ágil.

Por se tratar de serviços com pouca dependência, nem toda história de usuário gera alterações de código em todos os repositórios, e como o propósito do SonarCloud é ser uma ferramenta de revisão de código, quando não há alterações, não são realizadas medições (SONARSOURCE SA, 2021).

O Notify-Service foi criado vislumbrando as melhorias incrementais e escalabilidade do produto em cima de uma necessidade identificada durante a Lean Inception, mas pela priorização do MVP foi excluída do escopo, mas incluída no estacionamento de ideias para que fosse retomada quando fosse identificada relevante para a validação do negócio (CORALI, 2018). Assim, o repositório que contém os códigos desse serviço não foi atualizado durante o desenvolvimento e não apresenta dados de análise de código.

Até a finalização desse trabalho, foram realizadas cinco medições, com o intervalo médio de 11,5 dias. A Medição 1 (M1) foi realizada no repositório das documentações do projeto, a fim de validar a integração SonarCloud-Github. Como esse repositório não detém código da aplicação, não foram considerados para o estudo.

Uma vez bem-sucedida a integração, a Medição 2 (M2) foi realizada apenas no User-Service, à época, era o serviço com o desenvolvimento mais avançado, principalmente relacionado as funcionalidades priorizadas no sequenciador (Figura 25). Essa medição foi realizada no contexto de aprendizagem do analista de dados para verificar se as medidas estavam sendo obtidas e apresentadas corretamente no contexto da integração. No ponto em que a M2 foi realizada a primeira história de usuário “US1 – Eu como usuário posso fazer cadastro para ter acesso ao PUMA” estava em produção (ZENHUB PUMA, 2021; PUMA WIKI, 2021).

Então, apenas as Medições 3, 4 e 5 (M3, M4 e M5, respectivamente) foram as realizadas após o aceite de entrega de história de usuário. A M3 está associada a entrega da US1 e da “US2 – Eu usuário posso fazer login para acessar o PUMA”. A M4 está associada a entrega da “US3 – Eu como agente externo posso registrar proposta de projeto para resolver um problema”. A M5 está associada a entrega da “US4 – Eu como Professor posso avaliar a proposta” e a “US7 – Eu como usuário posso ver os detalhes do projeto”.

É importante ressaltar que o cenário de gestão ágil aplicado, quando uma US entra em processo de avaliação para aceite, os códigos das histórias seguintes já começam as ser desenvolvidos. Assim, as medições informadas pelo SonarCloud em cada um dos serviços já abarcam essas novas linhas de código, se elas existirem. Os repositórios em que foram retornados dados em cada uma das medições estão apresentados no Quadro 7.

Quadro 7. Relação das medições realizadas com os repositórios em que foram obtidos dados pelo SonarCloud

Repositório \ Medições	M1	M2	M3	M4	M5
Documentação	x				
Front End			x	x	
API Gateway			x	x	x
Project-Service				x	x
Alocate-Service				x	
Notify-Service					
User-Service		x	x	x	x

Fonte: Elaborado pela autora.

No escopo de avaliação da manutenibilidade apresentado pela ferramenta Q-Rapids são utilizados 7 dados puros obtidos do SonarCloud, sendo eles: o número de linhas de código, o número de funções, o número de arquivos, a complexidade ciclomática, a densidade de comentários, densidade de linhas duplicadas, número de violações críticas e bloqueadoras e 1 obtido do Zenhub, os campos de tarefas relacionadas a problemas de qualidade. Nesse ponto, é importante definir alguns desses dados para a compreensão de sua relevância para escopo da manutenibilidade.

A complexidade ciclomática é um cálculo baseado no número de caminhos através do código, e está associada a forma com que uma função se divide. Cada função tem uma complexidade mínima de 1, e a cada fluxo de controle dividido, a complexidade é incrementada em 1 (SONARQUBE AS, 2021). O limite proposto de complexidade para cada função é de 10 fluxos (MARTÍNEZ-FERNÁNDEZ et al., 2019).

Comentários são linhas de código que não tem efeito sobre o comportamento da aplicação, normalmente são utilizadas para auxiliar programadores a entender e documentar o código (MARIMOTO; HASHIMOTO, 2010). Martínez-Fernández et al. (2019) sugere que a densidade de comentários, ou seja, a relação entre linhas de comentário e a soma de linhas de código e linhas de comentário, deve estar entre 10% e 30%.

A duplicidade é quando o mesmo bloco de código aparece identicamente em mais de uma parte de código. É comum que se reuse fragmentos de códigos via copiar-colar sem que haja nenhuma modificação (DANG, 2012), mas já foi identificado que esse tipo de prática é prejudicial a qualidade do código (HERMANS;

AIVALOGLOU, 2017). Para duplicidade, é recomendado que a densidade deve ser menor que 5% (MARTÍNEZ-FERNÁNDEZ et al., 2019).

Segundo Rasool e Arshad (2015), o conceito de code smell foi introduzido por Martin Fowler em seu livro sobre refatoração. Fowler identificou 22 tipos de características estruturais do software que poderiam indicar um problema mais profundo de código ou de design que torna o software difícil de escalar e manter, e nomeou essas estruturas como Code Smells (RASOOL; ARSHAD, 2015).

A presença de qualquer code smell é um indicador de baixa qualidade de código em algum aspecto, mas eles podem ser classificados pelo nível de severidade em relação ao impacto na qualidade. O SonarCloud classifica qualquer problema de código (bugs, vulnerabilidades ou code smells) como crítico, bloqueador, maior ou menor (SONARQUBE AS, 2021). Na perspectiva da Q-Rapids, aqueles classificados como críticos ou bloqueadores precisam de mais atenção, e por isso compõe os dados puros para avaliação de manutenibilidade (MARTÍNEZ-FERNÁNDEZ et al., 2019).

Por fim, as tarefas relacionadas a problemas de qualidade são aquelas oriundas da identificação de algum problema de código (bugs, vulnerabilidades ou code smells) durante o desenvolvimento. Esse processo de identificação de problemas e associá-los a tarefas permite que o trabalho seja acompanhado, e que nenhum problema de código identificado fique sem solução (GITHUB INC, 2021). As tarefas são definidas em campos, que podem conter uma infinidade de informações, mas é sugerido que para uma que uma tarefa esteja bem definida ela precisa ter os campos de descrição e data de entrega preenchido, além do título que é um campo obrigatório (MARTÍNEZ-FERNÁNDEZ et al., 2019; ZENHUB PUMA, 2021).

5.1.5. Etapa 2.5 – Avaliação da Qualidade

Nesta etapa, os dados puros foram organizados e agrupados, para fornecer as informações sobre a qualidade relacionada a manutenibilidade, e para contribuir com a interpretação dos indicadores estratégicos. Na Tabela 1 estão apresentados os dados referentes a quantidade de linhas de código, arquivos e funções totais a cada medição, com exceção da M1 realizada apenas como teste no repositório de documentação.

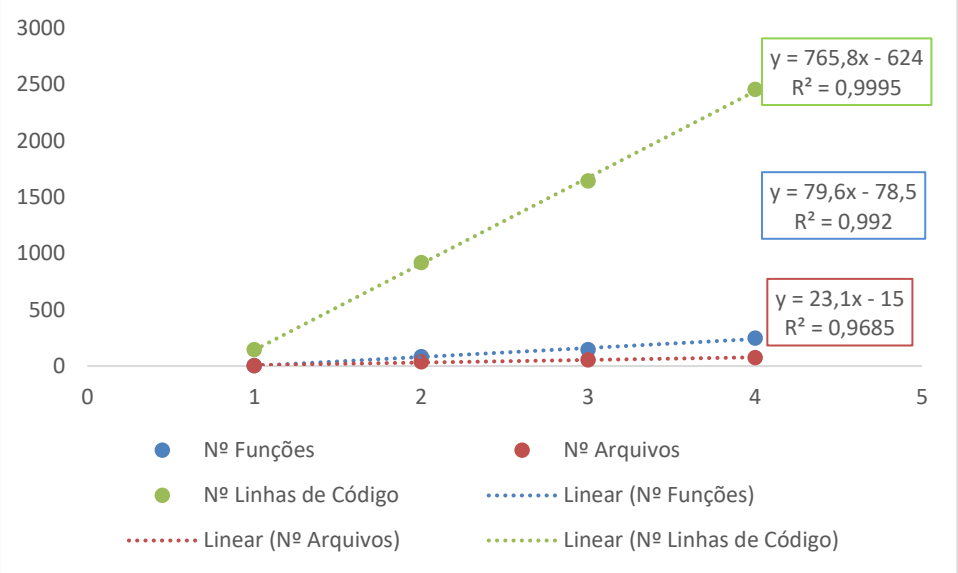
Tabela 1. Número total de funções, arquivos e linhas de código nos repositórios para cada medição

Dado	M2	M3	M4	M5
Nº Funções	4	83	147	248
Nº Arquivos	3	38	56	74
Nº Linhas de Código	145	918	1643	2456

Fonte: Elaborada pela autora.

Ao aplicar uma regressão linear, pode-se perceber que a evolução do tamanho do código respeitou uma equação linear a uma taxa de 765,8 linhas por medição, com um R^2 de 0,99. Já o número de funções cresceu numa taxa de 79,6 funções por medição, também com um R^2 de 0,99. Já o número de arquivos até a M5 também apresentou um comportamento linear crescente na taxa de 23,1 arquivos por medição, com uma correlação ligeiramente menor (Figura 28).

Figura 28. Gráfico de dispersão da quantidade de linhas de código, funções e arquivos por medição com o destaque para o comportamento linear



Fonte: Elaborada pela autora

Esse comportamento linear indica um nível de produtividade e esforço constante da equipe de desenvolvimento ao decorrer das entregas. Um resultado numérico do que foi observado durante o desenvolvimento, do momento em que as entregas começaram a ser tácitas, o reforço positivo fornecido pelos stakeholders por essa clareza de resultados, a equipe de desenvolvimento conseguiu manter-se em um nível constante de produtividade na perspectiva do tamanho do código.

Tabela 2. Complexidade ciclomática média avaliada para cada um dos repositórios e medições

Repositório	M2	M3	M4	M5
Alocate-Service			2,25	
ApiGateway		1,05	1,04	1,05
Front End		2,26	2,26	1,64
Project-Service			2,14	1,81
User-Service	2,25	1,38	1,26	1,49
Média	2,25	1,56	1,79	1,50

Fonte: Elaborada pela autora

A complexidade ciclomática média, ou seja, a complexidade geral no repositório dividida pelo número de funções dele, decresceu a cada medição, indicando uma tendência de manter uma complexidade que garanta a manutenibilidade (Tabela 2).

Tabela 3. Densidade de comentários no decorrer das medições segmentado por repositório

Repositório	M2	M3	M4	M5
Alocate-Service			0%	0%
ApiGateway		21%	6%	3%
Front End		43%	43%	24%
Project-Service			8%	6%
User-Service	0%	23%	23%	11%
Média	0%	29%	16%	9%

Fonte: Elaborada pela autora

Na Tabela 3 estão apresentados os dados relacionados a densidade de comentários ao longo das entregas. É preocupante observar que à medida que o código avança a densidade de comentários diminui, sendo que no contexto de garantia da manutenibilidade deveria existir um esforço em gerar linhas de comentário que trouxessem informações sobre o funcionamento do código.

Em relação a medida de densidade de linhas duplicadas, a medida permaneceu constante em 0% para todas as medições e repositórios, exceto a M5 relacionada ao repositório da API Gateway que retornou um resultado de 13% para uma relação de 15 arquivos (Tabela 4).

Tabela 4. Densidade de Linhas Duplicadas

Repositório	M2	M3	M4	M5
Alocate-Service			0%	0%
ApiGateway		0%	0%	13%
Front End		0%	0%	0%
Project-Service			0%	0%
User-Service	0%	0%	0%	0%
Média	0%	0%	0%	3%

Fonte: Elaborada pela autora

O número total de violações críticas ou bloqueadoras teve um comportamento crescente, com uma pequena queda entre M2 e M3 (Tabela 5). À primeira vista, esse comportamento dá sinais de que à medida que o código cresce, o número de arquivos e funções aumentam, a probabilidade de surgir uma violação de qualidade também cresce. O que importa aqui é identificar essas violações gerando respostas no processo de desenvolvimento incremental.

Tabela 5. Número total de violações críticas ou bloqueadoras nos repositórios para cada medição

Repositório	M2	M3	M4	M5
Alocate-Service			4	4
ApiGateway		0	0	8
Front End		1	3	3
Project-Service			4	4
User-Service	4	1	1	0
Total	4	2	12	19
Média	4,0	0,7	2,4	3,8

Fonte: Elaborada pela autora

Até a realização da M5, a equipe de desenvolvimento não havia documentado no Zenhub, nenhuma tarefa associada diretamente a um problema de qualidade de código identificado. Pela própria evolução dos outros dados, é possível concluir que esse cenário não significa que não foram identificados problemas no código, muito menos que as correções não foram efetuadas. Uma causa provável é a de que os alunos de EPS, que estariam a princípio responsáveis por atualizar a ferramenta de acompanhamento de tarefas, precisaram priorizar a entrega do MVP e reunir esforços para produzir código e, ao final, não documentaram todas as tarefas executadas.

Com os dados tratados, é preciso calcular os indicadores estratégicos da ferramenta Q-Rapids conforme esquema apresentado no Quadro 8.

Quadro 8. Esquema associado do aspecto de qualidade Manutenibilidade com os respectivos fatores e métricas calculadas segundo a ferramenta Q-Rapids

Manutenibilidade	Qualidade de Código	Complexidade (DNC)	Arquivos abaixo do limiar da complexidade ciclomática	Quanto mais o DNC se aproxima de 1 melhor para a garantia da manutenibilidade	Densidade de aquivos não complexos (DNC) , onde um arquivo não é complexo se a complexidade ciclomática é menor que 10. $DNC = \frac{nº\ de\ arquivos\ não\ complexos}{total\ de\ arquivos}$
		Comentários (DCM)	Arquivos em que se tem a densidade de comentários fora do limiar	Quanto mais o DCM se aproxima de 1 melhor para a garantia da manutenibilidade	Densidade de arquivos comentádos (DCM), onde um arquivo é comentado quando a densidade de comentários (DLC) está entre 10% e 30%. $DCM = \frac{nº\ de\ arquivos\ comentados}{total\ de\ arquivos}$ $DLC = \frac{nº\ linhas\ de\ comentário}{\left(\begin{matrix}nº\ de\ linhas\ de\ código \\ +nº\ de\ linhas\ de\ comentário\end{matrix}\right)}$
		Duplicação (AD)	Arquivos abaixo do limiar percentual de linhas duplicadas	Quanto mais o AD se aproxima de 1 melhor para a garantia da manutenibilidade	Ausência de Duplicações (AD), onde um arquivo não está duplicado quando a densidade de duplicação (DD) é menor que 5%. $AD = \frac{nº\ de\ arquivos\ não\ duplicados}{total\ de\ arquivos}$ $DD = \frac{nº\ de\ linhas\ duplicadas}{total\ de\ linhas}$
	Bloqueio de Código	Arquivos Não-Bloqueados (ANB)	Arquivos sem fatores críticos ou bloqueios de qualidade	Quanto mais o ANB se aproxima de 1 melhor para a garantia da manutenibilidade	Adequação de fatores críticos e bloqueios de qualidade (ANB), onde um arquivo cumpre as regras de qualidade quando não há bugs, vulnerabilidades e Code Smells classificados como críticos ou de bloqueio. $ANB = \frac{\begin{matrix}nº\ de\ arquivos\ sem\ problemas \\ críticos\ ou\ de\ bloqueio\end{matrix}}{total\ de\ arquivos}$
	Especificação de Problemas de Qualidade	Problemas Bem definidos (PBD)	Desidade de Problemas de qualidade que detém boa definição	Quanto mais o PBD se aproxima de 1 melhor para a garantia da manutenilidade	Densidade de problemas bem definidos (PBD), onde um problema é bem definido quando os campos de descrição e data de entrega não estão vazios. $PBD = \frac{nº\ de\ problemas\ bem\ definidos}{total\ de\ problemas\ identificados}$

Fonte: Adaptado de Martínez-Fernández et al., 2019.

Relacionado a Qualidade de Código, os indicadores de complexidade (DNC) e duplicidade (AD) permaneceram constantes iguais a 1 em todas as medições. O indicador de comentários (DCM) também permaneceu constante, mas no outro extremo, nenhum arquivo tinha um % de DLC entre 10 e 30%, fazendo com que o indicador fosse avaliado como 0. Dessa forma, quanto ao fator Qualidade de Código, 67% dos indicadores atingiram o valor esperado.

Como nenhum problema de qualidade foi traduzido em tarefas no Zenhub, o indicador estratégico PBD é zero, o mais distante possível do valor esperado, impactando diretamente o fator Especificações de problemas de qualidade, prejudicando a gestão de conhecimento sobre o desenvolvimento da aplicação.

Quanto ao fator bloqueio de código, o indicador ANB foi calculado e os resultados estão apresentados na Tabela 6.

Tabela 6. Avaliação do indicador ANB calculado para cada medição e repositório

Repositório	M2	M3	M4	M5
Alocate-Service			75%	75%
ApiGateway				73%
Front End		86%	89%	89%
Project-Service			85%	85%
User-Service	67%	92%	92%	100%
Média	67%	89%	85%	84%

Fonte: Elaborada pela autora

A proximidade de 100% indica que a maioria dos arquivos não tem violações de qualidade. Em nenhuma das medições foram encontrados bugs ou vulnerabilidades, apenas code smells. E segundo análise do SonarCloud (2021), a dívida técnica é baixa e portando simples de resolver em um processo de melhoria incremental.

A fim de obter um indicador agrupado que pudesse traduzir a avaliação da manutenibilidade, a média dos resultados dos indicadores estratégicos em relação aos repositórios foi determinada como a Manutenibilidade média da aplicação, conforme apresentado na Tabela 7.

Tabela 7. Resultados dos Indicadores Estratégicos Médios em relação aos repositórios para cada uma das medições

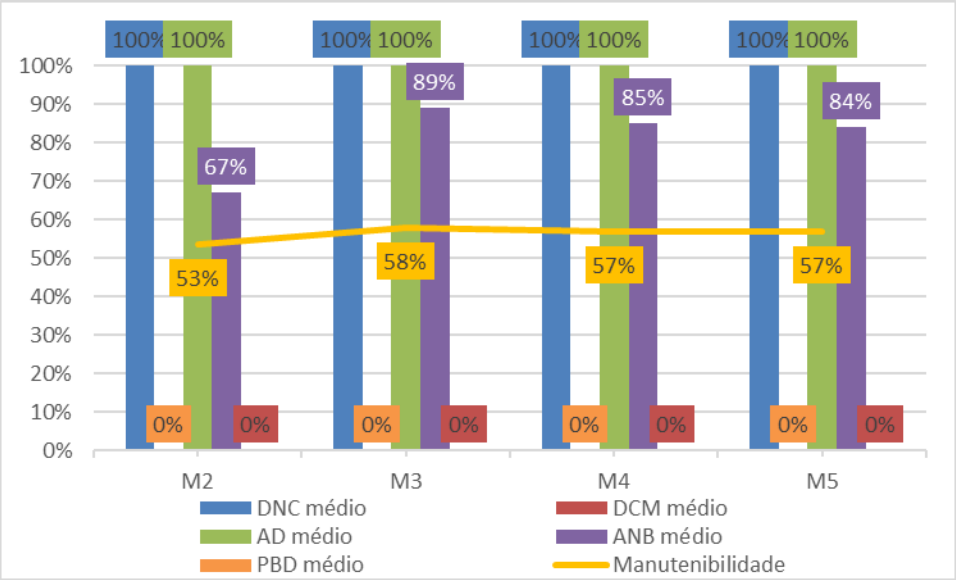
Indicador Estratégico	M2	M3	M4	M5
DNC médio	100%	100%	100%	100%
DCM médio	0%	0%	0%	0%
AD médio	100%	100%	100%	100%
ANB médio	67%	89%	85%	84%
PBD médio	0%	0%	0%	0%
Manutenibilidade (Média)	53%	58%	57%	57%

Fonte: Elaborada pela autora

Esse resultado significa que ao estágio de entrega da US4 e US7, 57% do nível de manutenibilidade ótimo traçado pela ferramenta Q-Rapids foi alcançado. Cabe

destacar que esse indicador, por definição, não é estático, mas sim acompanha o ambiente dinâmico de desenvolvimento de software (Figura 29)

Figura 29. Gráfico do acompanhamento dos indicadores médios em cada medição



Fonte: Elaborada pela autora

Os indicadores de DNC, AD e ANB estão mais focados na qualidade da estrutura de código e são os que tiveram as melhores avaliações, validando a aprendizado da equipe de desenvolvimento, e possibilidade de sucesso do desenvolvimento da plataforma no contexto de equipes multidisciplinares para PBL.

Já os indicadores de DCM e PBD além de estarem associados a boas práticas de programação para a garantia da manutenibilidade, tem ligação com a qualidade do processo de desenvolvimento. O princípio definido por Dromey (1995) em seu modelo de qualidade, em que os critérios de qualidade não são mutualmente exclusivos, nesse contexto foi facilmente observado.

Na perspectiva PBL e do desenvolvimento do MVP, um dos fatores elencados para a priorização das métricas de manutenibilidade era a relação delas com a gestão do conhecimento sobre o código, e as duas métricas mais relevantes para esse processo são as de DCM e PBD, as quais tiveram o pior desempenho possível.

No contexto geral de qualidade de software, não é suficiente que apenas o código detenha nível esperado de qualidade, todo o processo de desenvolvimento precisa ser realizado com esse foco. Nesse ponto, os resultados dos indicadores DCM e PBD apontam que a garantia da qualidade do processo de desenvolvimento não se

concretizou, podendo gerar impactos na qualidade do produto em uso, conforme esquema apresentado na Figura 12.

Essas lacunas de processo e boas práticas podem representar um risco para o sucesso das melhorias incrementais do MVP do PUMA nesse contexto em que grande parte do grupo de trabalho será outro a cada semestre letivo.

Com o encerramento do período letivo, 83,3% das funcionalidades do escopo delimitado para a entrega final das disciplinas de MDS e EPS foram entregues. Essas funcionalidades representam 62,5% daquelas determinadas durante a Lean Inception para o MVP (PUMA WIKI, 2021).

A desatenção sobre a garantia da qualidade no processo de desenvolvimento contribuiu para que o escopo de entrega não fosse cumprido integralmente, apesar indicadores de qualidade de código terem atingido o valor esperado. Esse resultado deixa como lição aprendida a necessidade de manter não só a qualidade do produto, como também a qualidade do processo para a garantia da qualidade como um todo.

6. CONSIDERAÇÕES FINAIS

A partir da priorização de critérios de qualidade, e adequação ao contexto de desenvolvimento, o produto desenvolvido obteve 57% de manutenibilidade média. Resultado abaixo do esperado no contexto de garantia da qualidade, principalmente devido a desatenção às boas práticas relacionadas ao processo de desenvolvimento.

O objetivo geral deste projeto de graduação de aplicar métricas de qualidade interna para garantia da qualidade do MVP do PUMA foi alcançado a partir do cumprimento de seus objetivos específicos.

Durante a compreensão dos modelos de qualidade aplicáveis a webapps, caracterizada pela revisão sistemática da literatura, pôde-se ter uma visão abrangente do contexto de modelos de qualidade, com todos os conceitos de gestão da qualidade de software, características e métricas. O modelo integrador, além de fornecer uma percepção qualitativa para priorização de critérios de qualidade, contribuiu para a integração de novos paradigmas e abstrações ao pensamento conceitual do Engenheiro de Produção.

Já no estudo de caso, como uma representação de como a aprendizagem baseada em problemas se retroalimenta gerando conhecimento coletivo e competências individuais, foi possível verificar que a aplicação de métricas de qualidade traz informações relevantes para a garantia da qualidade no desenvolvimento. E esse tipo de abordagem é altamente customizável a contexto e equipes diversas de desenvolvimento.

Uma atenção equilibrada entre os indicadores relacionados à qualidade do código em si (Complexidade, Duplicação e Arquivos Não-Bloqueados) e os indicadores relacionados à qualidade do processo de desenvolvimento (Comentários e Problemas Bem Definidos) fazem a diferença para a garantia da qualidade geral.

Com as limitações da pesquisa no acompanhamento de uma única característica de qualidade, a manutenibilidade, relacionada a uma única ferramenta e a um projeto de desenvolvimento muito específico, já foi factível para, nas devidas proporções, a garantia da qualidade como um processo cíclico e de identificação de riscos, as possibilidades de aplicações de métricas e modelos de qualidade são diversas.

Outra limitação está na aplicação do princípio da não exclusividade de Dromey’s (1995), em que outros fatores de qualidade também influenciam na

avaliação da Manutenibilidade não foram avaliados nessa pesquisa. Como por exemplo a Testabilidade, que aparece entre os 4 critérios prioritários determinados pelo modelo integrador durante o levantamento do estado da arte, mas não foram incluídos neste estudo.

Estudos futuros podem verificar de forma mais aprofundada a relação entre as utilizações de métricas de qualidade como parte da gestão de riscos pertinentes a produtos de software.

7. REFERÊNCIAS BIBLIOGRÁFICAS

ABNT NBR ISO/IEC 25001 Engenharia de software - Requisitos e avaliação da qualidade de produto de software (SQuaRE) - Planejamento e gestão, 2009;

ABNT. NBR ISO/IEC 9126 Engenharia de software - Qualidade de produto**.** NBR ISO/IEC 9126. 2003

AL-QUTAISH, R. E. Quality Models in Software Engineering Literature: An Analytical and Comparative Study. **Journal of American Science**, 2010.

BAKHTIN, S., AL-ZAGHEER, H. The Influance of CASE Tools Traits on The Successful Design of Web Applications. International Journal of Scientific & Technology Research. v. 8, n. 12, p 2007-2010, 2019

BALIYAN, N.; KUMAR, S., Software Process and Quality Evaluation for Semantic Web Applications, **IETE Technical Review**, v. 31 n. 6, 452-462, 2014.

BLANK, S., Why the lean start-up changes everything, **Harvard Business Review**, Vol. 91 No. 5, pp. 63-72, 2013. <Disponível em: https://hbr.org/2013/05/why-the-lean-start-up-changes-everything>

BOEHM, B. W., BROWN, J. R., LIPOW, M. Quantitative Evaluation of Software Quality **ICSE '76: Proceedings of the 2nd international conference on Software engineering,** p 592–605, 1976.

BRASSCOM. **Formação Educacional e Empregabilidade em TIC – Achados e Recomendações**. Relatório de Inteligência e Informação, 2019.

CAROLI, Paulo. Lean Inception: como alinhar pessoas e construir o produto certo. 1ª Edição atualizada - São Paulo: Editora Caroli, 2018.

CASTILLO, F. F. R.; MORA, N. M.L., Elizades, K. D. C.; OROZCO, J. I. P. Comparación de métricas de calidad para el desarrollo de aplicaciones web. 3C **Tecnología. Glosas de Innovación aplicadas a la pyme**, 7(3), 94-113. 2018.

COBO, M. J., LÓPEZ- HERRERA, A. G., HERRERA- VIEDMA, E., & HERRERA, F. SciMAT: A new science mapping analysis software tool. **Journal of the Association for Information Science and Technology**, 2012.

COMITÊ GESTOR DA INTERNET NO BRASIL Pesquisa sobre o uso das tecnologias de informação e comunicação nos domicílios brasileiros: **TIC Domicílios 2019 [livro eletrônico]** Núcleo de Informação e Coordenação do Ponto BR., 1. ed., São Paulo. 2020.

COMITÊ GESTOR DA INTERNET NO BRASIL Pesquisa sobre o uso das tecnologias de informação e comunicação nas empresas brasileiras: **TIC empresas 2019 [livro eletrônico]**. Núcleo de Informação e Coordenação do Ponto BR. -- 1. ed. -- São Paulo: 2020

CONCAS, G; MARCHESI, M; DESTEFANS, G.; TONELLI, R., An Empirical Study of Software Metrics for Assessing the Phases of An Agile Project. International **Journal of Software Engineering and Knowledge Engineering, v**. 22, n. 4, p.525-548, 2012.

DANG, Y.; ZHANG, D.; GE S.; CHU C.; QIU Y.; XIE T. XIAO: Tuning code clones at hands of engineers in practice. **ACM International Conference Proceeding Series** 28th Annual Computer Security Applications Conference, ACSAC 2012 pp 369 – 378, 2012.

DROMEY, R. G. A model for Software Product Quality**. IEEE Transactions on Software Engineering**, v. 2, n. 2, p. 146-162, 1995.

DUARTE, E. N., RAMALHO, F. A., AUTRAN, M. M. M., PAIVA, E. B. P., ARAUJO, M. B. S. A. Estratégias Metodológicas Adotadas Nas Pesquisas De Iniciação Científica Premiadas Na UFPB: Em Foco A Série “Iniciados” **R. Eletr. Bibliotecon. Ci. Inf**., ISSN 1518-2924, Florianópolis, v. 14, n. 27, p.170-190, 2009. Disponível em: <10.5007/1518-2924.2009v14n27p170>

EPS MDS. Repositório de Código. GitHub, 2021. Disponível em: < https://github.com/fga-eps-mds>

FERNANDES, P., CHAVES, L., BONIFÁCIO, B., CONTE, T., “WE-QT Tool: Uma Ferramenta de Apoio a Inspeção de Usabilidade de Aplicações Web”. X Simpósio Brasileiro sobre Fatores Humanos em Sistemas Computacionais, v. 1. p. 6-10. 2012.

FOWLER, M. Padrões de arquitetura de aplicações corporativas (e-book) 1ª Edição. Porto Alegre: Bookman, 2007.

GARRIGOS, I., MAZON, J.-N., TRUJILLO, J., A requirement analysis approach for using in web engineering. **ICWE 2009**. LNCS, 5648, pp. 151-165, 2009.

GITHUB INC. GITHUB DOCS. Problemas do GitHub. 2021. Disponível em: < https://docs.github.com/pt/issues/tracking-your-work-with-issues/about-issues>

GOTHELF, J., SEIDEN, J.; Lean UX: Applying Lean Principles to Improve User Experience. O’Reilly Media, Inc., Newton, 2013.

GRAAFF, E. De; KOLMOS, A. **History of Problem-Based and Project-Based Learning**. [s.l: s.n.]. 2007.

HERMANS, F.; AIVALOGLOU, E., "Teaching Software Engineering Principles to K-12 Students: A MOOC on Scratch," 2017 IEEE/ACM 39th International Conference on Software Engineering: Software Engineering Education and Training Track (ICSE-SEET), 2017, pp. 13-22, 2017

IEEE. IEEE Std. 610.12: Standard Glossary of Software Engineering Terminology. The Institute of Electrical and Electronics Engineers, New York, NY, USA, 1990. Disponível em: <10.1109/ieeestd.1990.101064>

ISO/IEC 25010. **Software Quality Requirements and Evaluation – SQuaRE**. ISO/IEC 25010, 2011.

ISO/IEC 25023: Systems and software engineering – System and software Quality Requirements and Evaluation (SQuaRE) – Measurement of system and software quality, 2016

LIMA, A. C. F. Ferramenta de gestão de riscos aplicada a ambientes de desenvolvimento de software com foco na garantia da qualidade do produto. 2019. XVIII, 185 f., il. Dissertação (Mestrado Profissional em Computação Aplicada) — Universidade de Brasília, Brasília, 2019.

LIZARELLI, F.L. AND TORRES, A.F. AND ANTONY, J. AND RIBEIRO, R. AND SALENTIJN, W. AND FERNANDES, M.M., AND CAMPOS, A.T., Critical success factors and challenges for Lean Startup: a systematic literature review, **TQM Journal**, 2021. Disponível em: <10.1108/TQM-06-2021-0177>

MARIANO, A.M; Rocha, M.S. Revisão da Literatura: Apresentação de uma Abordagem Integradora. **AEDM International Conference – Economy, Business and Uncertainty: Ideas for a European and Mediterranean industrial policy**. Reggio Calabria (Itália), 2017.

MARIMOTO, C. H. HASHIMOTO, R. F. Introdução a Ciência da Computação em C. Apostila IME/USP. 2010. Disponível em: <https://www.ime.usp.br/~hitoshi/introducao/> Acesso em: 4 nov. de 2021.

MARTÍNEZ-FERNÁNDEZ, S et al., "Continuously Assessing and Improving Software Quality with Software Analytics Tools: A Case Study," in **IEEE Access**, vol. 7, pp. 68219-68239, 2019. Disponível em: <10.1109/ACCESS.2019.2917403.>

MCCALL, J. A., RICHARDS, P. K., WALTERS, G. F. Factors in Software Quality, Volume I. **US Rome Air Development Center Reports**, US Department of

Commerce, USA, 1977 Disponível em <https://apps.dtic.mil/dtic/tr/fulltext/u2/a049014.pdf >

MEDEIROS, J. M. G. de; VITORIANO, M. A. V. A Evolução da Bibliometria e sua Interdisciplinaridade na Produção Científica Brasileira. **Revista Digital de Biblioteconomia e Ciência da Informação**, Campinas, v. 13, n. 3, p. 491-503, set. 2015. Disponível em: <https://periodicos.sbu.unicamp.br/ojs/index.php/rdbci>.

MONTEIRO, S. B. M., CAMPOS, M. R.M., LIMA, A. C. F., MARIANO, A. M., Avaliando resultados diretos e indiretos da metodologia ativa na aprendizagem: proposta de um desenho integrador em 360º via plataforma unificada. Em **International Symposium on Project Approaches in Engineering Education**, volume 8, páginas 768–775, 2018.

MONTEIRO, S. B. M., LIMA, A. C. F., MARIANO, A. M., JÚNIOR, E. S. Plataforma Unificada de Metodologia Ativa (PUMA): um projeto multidisciplinar. **RISTI**, N. º E28, pp 766-778, 2020.

NABIL, D.; MOSAD, A.; HEFNY, H., Web-Based Applications Quality Factors: A Survey and a Proposed Conceptual Model. **Egyptian Informatics Journal** 12, pp. 211-217, 2011.

NERI, H. R. Engenharia de Produto de Software - Plano de ensino. 1/2021 Disponível em < https://github.com/fga-eps-mds/A-Disciplina-MDS-EPS/blob/master/PlanosDeEnsino/EPS_Plano_de_Ensino.md>

NERI, H. R. Métodos de Desenvolvimento de Software - Plano de ensino. 1/2021 Disponível em <https://github.com/fga-eps-mds/A-Disciplina-MDS-EPS/blob/master/PlanosDeEnsino/MDS_plano_ensino.md>

OWASP FOUNDATION. OWASP Top 10 – 2017: The Ten Most Critical Web Application Security Risks. 2017.

PAZ, F.; POW-SANG, J. A. A Systematic Mapping Review of Usability Evaluation Methods for Software Development Process. **International Journal of Software Engineering and Its Applications**, v. 10, n. 1, p. 165-178, 2016.

PRESSMAN, R. S., Software engineering: a practitioner’s approach. 5ª Edição. McGraw-Hill series in computer Science, 2001

PUMA WIKI. Documentos de Arquitetura. Disponível em < https://fga-eps-mds.github.io/2021-1-PUMA/#/planejamento/doc-de-arquitetura> Acesso em: 20 out. de 2021.

RASOOL, G.; ARSHAD, Z. “A review of code smell mining techniques” **Journal of Software: Evolution and Process** V. 27, ed 11, pp. 867 – 895, 2015.

RIES, E., A startup enxuta: como os empreendedores atuais utilizam inovação contínua para criar empresas extremamente bem-sucedidas, São Paulo: Lua de Papel, 2012.

RIVERO L.; CONTE T., Using an Empirical Study to Evaluate the Feasibility of a New Usability Inspection Technique for Paper Based Prototypes of Web Applications. **Journal of Software Engineering Research and Development**, v. 1. p. 81-90, 2012.

RODRIGUES, J. L.; PINTO, G. S. Análise Da Arquitetura De Microserviços. **SIMTEC - Simpósio de Tecnologia da Fatec Taquaritinga**, v. 5, n. 1, p. 39-49, 21 dez. 2019.

RODRÍGUEZ, F. D., ACUÑA, S. T., JURISTO, N. Design and programming patterns for implementing usability functionalities in web applications. **Journal of Systems and Software**, 105, 107–124. 2015. Disponível em <doi:10.1016/j.jss.2015.04.023>

ROSSI, G., PASTOR, O., SCHWABE, D., OLSINA, L., Web Engineering Modelling and Implementing Web Applications, London: Springer, 2010.

SAA UNB. Calendário Universitário de Graduação. 2021.1 Disponível em <https://saa.unb.br/images/documentos/graduacao/atividade/Calendrio_de_Atividades_GRADUAO_1_2021_-_09_08_2021.pdf>

SCHMIDT, H. G. et al. Constructivist, problem-based learning does work: A meta-analysis of curricular comparisons involving a single medical school. **Educational Psychologist**, [s. l.], v. 44, n. 4, p. 227–249, 2009.

SILVA, JOÃO MELLO DA SILVA E BALTHAZAR, JOSÉ CARLOS: Projeto político pedagógico do curso de engenharia de produção, 2010. https://sig.unb.br/sigaa/verProducao?idProducao=2029779&&key=4b019ab5789b0cde43fe99a209ac6943,

SILVA, W.; COSTA VALENTIM, N. M.; CONTE, T. Integrating the Usability into the Software Development Process: A Systematic Mapping Study. **ICEIS 2015 - 17th International Conference on Enterprise Information Systems, Proceedings**. 3. 2015. Disponível em <DOI 10.5220/0005377701050113.>

SOARES, F. O. et al. An integrated project of entrepreneurship and innovation in engineering education. Mechatronics, [s. l.], v. 23, n. 8, p. 987–996, 2013. Disponível em: <http://dx.doi.org/10.1016/j.mechatronics.2012.08.005>

SONARQUBE SA. SonarQube Documentation. Disponível em: < https://docs.sonarqube.org/latest/>. Acesso em: 17 de set. de 2021.

SONARSOURCE SA. SonarCloud: Clean Code, Rockstar Status. 2021 Disponível em: <https://sonarcloud.io> Acesso em: 17 de set. de 2021.

UNESCO. Engineering: issues, challenges, and opportunities for development. United nations educational, scientific, and cultural organization, Paris (France); 2010.

VETRO, A; ZAZWORKA, N; SEAMAN, C.; SHULL, F., Using the ISO/IEC 9126 product quality model to classify defects: a Controlled Experiment. In: **Evaluation & Assessment in Software Engineering (EASE 2012),** 16th International Conference on, Ciudad Real (Spagna), Spain, pp. 187-196, 2012.

WAGNER, S et al. Proc. The Quamoco Product Quality Modelling and Assessment Approach. **34th International Conference on Software Engineering (ICSE'12). IEEE**, 10 pp., 2012.

YOSHIDA, Nelson D. Análise Bibliométrica: Um Estudo Aplicado à Previsão Tecnológica. **Future Studies Research Journal Trends and Strategies**, São Paulo, v. 2, n. 1, p. 52-84, Jan. 2010. Disponível em: <https://www.revistafuture.org/FSRJ/article/viewFile/45/68>.

YUHANA, U. L.; RAHARJO A. B.; ROCHIMAH, S, Academic information system quality measurement using quality instrument: A proposed model, **International Conference on Data and Software Engineering (ICODSE**), pp. 1-6, 2014.

ZENHUB. PUMA: Board. 2021 Disponível em: <https://app.zenhub.com/workspaces/puma-611a97719904c20012bf1866/board?labels=epic&repos=393792332> Acessado em: 27 set. 2021.

ZUPIC, I.; ČATER, T. Bibliometric methods in management and organization. **Organizational Research Methods**, v. 18, n. 3, p. 429-472, 2015.