Uma proposta de rotinas e processos de gestão, comunicação e qualidade aplicados a projetos de pesquisa e desenvolvimento

Izaura D'angela Oliveira Magalhães

CENTRO DE INFORMÁTICA
UNIVERSIDADE FEDERAL DA PARAÍBA

João Pessoa, 2021

Izaura D'angela Oliveira Magalhães

Uma proposta de rotinas e processos de gestão, comunicação e qualidade aplicados a projetos de pesquisa e desenvolvimento

Monografia apresentada ao curso Ciência da Computação do Centro de Informática, da Universidade Federal da Paraíba, como requisito para a obtenção do grau de Bacharel em Ciência da Computação

Orientador: Raoni Kulesza
Co-Orientador: Rafael M. Toscano

Julho de 2021

Ficha catalográfica: elaborada pela biblioteca do CI.

Será impressa no verso da folha de rosto e não deverá ser contada.
<Se não houver biblioteca, deixar em branco>

CENTRO DE INFORMÁTICA
UNIVERSIDADE FEDERAL DA PARAÍBA

Trabalho de Conclusão de Curso de Ciência da Computação intitulado ***Uma proposta de rotinas e processos de gestão, comunicação e qualidade aplicados a projetos de pesquisa e desenvolvimento*** de autoria de **Izaura D'angela Oliveira Magalhães**, aprovada pela banca examinadora constituída pelos seguintes professores:

_____

Prof. Dr. Raoni Kulesza
Universidade Federal da Paraíba - UFPB

_____

Prof. Dr. Valdecir Becker
Universidade Federal da Paraíba - UFPB

_____

Prof. Dr. Tiago Maritan Ugulino de Araújo
Universidade Federal da Paraíba - UFPB

_____

Coordenador(a) do Departamento de Informática
Gustavo Henrique Matos Bezerra Motta
CI/UFPB

João Pessoa, 9 de julho de 2021

Centro de Informática, Universidade Federal da Paraíba
Rua dos Escoteiros, Mangabeira VII, João Pessoa, Paraíba, Brasil CEP: 58058-600
Fone: +55 (83) 3216 7093 / Fax: +55 (83) 3216 711

Dedico este trabalho ao meu pai, Franco Magalhães.
Pai, muito obrigado por ser o meu melhor amigo e meu maior exemplo de força e dedicação.

AGRADECIMENTOS

Primeiramente, quero agradecer a Deus, por ter feito os meus caminhos ainda mais bonitos do que imaginei. Aos meus pais, Franco e Ivonete, como também à minha madrasta, Kilvia, obrigada por sempre me incentivar, torcer e apoiar todas as minhas decisões, eu não estaria aqui se não fosse por vocês. A Caio, muito obrigada por me dar forças e sonhar junto comigo. E aos meus amigos, nunca esquecerei como vocês torceram por mim.

Agradeço ao professor Valdecir Becker, por ter sido o melhor mentor que eu poderia ter durante o curso. Obrigada por me ensinar tanto, por me fazer encontrar a minha verdadeira paixão na computação e por estar ao meu lado até na visita ao São Januário. Carregarei para sempre todos os ensinamentos profissionais e de vida que você me passou.

A melhor amizade que essa universidade me proporcionou, Rafael Toscano, obrigada por acreditar no meu potencial até quando eu mesma não acreditava, é uma honra trabalhar com você todos os dias.

Ao meu orientador, Raoni Kulesza, eu não teria como ser mais grata por todas as oportunidades que me foram proporcionadas, aplicar este trabalho no LAVID me trouxe ensinamentos e uma alegria que carregarei para sempre.

Aos professores Tiago Maritan, Danielle Rousy e Guido Lemos, sou extremamente agradecida por todo o conhecimento técnico que vocês me passaram e por toda a confiança que me foi dada.

Aos professores Ulysses Oliveira e Laura Alves, vocês foram essenciais. Obrigado por tirarem o tempo de vocês para ensinar e resolver questões de disciplinas alheias às suas, professores como vocês são divisores de água na experiência de alguns alunos durante a universidade.

A todos os servidores do centro de informática, meus dias não seriam os mesmos sem o bom dia de vocês.

Ao LAVID, por confiar e embarcar na minha proposta, por me possibilitar crescer profissionalmente todos os dias, obrigada.

A todos os bolsistas que cruzaram o meu caminho, sou grata por ter trabalhado com vocês e gostaria de dizer que nada disso seria possível sem a colaboração que vocês proporcionaram.

RESUMO

Desde sua concepção inicial, o desenvolvimento de software passou por mudanças em seu fluxo de processos e escopo. A busca pela sistematização gerou modelos de processos, como o modelo de cascata. Com o passar do tempo, o desenvolvimento de software necessitou de abordagens mais ágeis e flexíveis. Tais modelos impactam significativamente os processos de gestão e comunicação utilizados pelas equipes para garantir a qualidade dos softwares. Com o intuito de compreender melhor esse cenário foi realizado um levantamento dos conceitos e práticas documentadas na literatura. A partir desses itens este trabalho propõe um conjunto de rotinas e processos que podem ser aplicados em uma equipe de software nas áreas de comunicação, gestão e qualidade. Para tal, foi realizado um estudo de caso com cinco projetos de desenvolvimento de software do Laboratório de Aplicações de Vídeo Digital. Após a aplicação desses itens, percebeu-se através de uma sondagem qualitativa com os bolsistas que as mudanças tiveram impacto positivo no dia a dia de trabalho e na percepção da entrega de qualidade dos produtos. Por fim, acreditamos que as práticas sistematizadas por essa pesquisa podem ser aplicadas em outros contextos e times de desenvolvimento de software.

Palavras-chave: Desenvolvimento de software, gestão, comunicação, qualidade de software.

ABSTRACT

Since its initial conception, software development has undergone changes in its process flow and scope. The search for systematization generated process models, such as the waterfall model. Over time, software development required more agile and flexible approaches. Such models significantly impact the management and communication processes used by teams to ensure software quality. In order to better understand this scenario, a survey of the concepts and practices documented in the literature was carried out. Based on these items, this work proposes a set of routines and processes that can be applied in a software team in the areas of communication, management and quality. To this end, a case study was carried out with five software development projects from the Digital Video Applications Laboratory. After applying these items, it was realized through a qualitative survey with the scholarship holders that the changes had a positive impact on their daily work and on the perception of quality delivery of the products. Finally, we believe that the practices systematized by this research can be applied in other contexts and software development teams. The impacts and consequences are in charge of future studies in the continuation of this research.

Key-words: Software development, management, communication, software quality.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

XP	–	*Extreme Programming*
RUP	–	Processo Racional Unificado
IHC	–	Interação Humano-computador
LAVID	–	Laboratório de Aplicações de Vídeo Digital
DevOps	–	*Development Operations*
UX	–	*User Experience*
DSR	–	*Design Science Research*
QA	–	*Quality Assurance*
V4H	–	*Video for Health*
PHI	–	Paraíba Humana Inteligente
CTs	–	Casos de teste

SUMÁRIO

1. INTRODUÇÃO

Desde sua concepção, o desenvolvimento de software passou por mudanças em seu fluxo de processos e escopo. Um dos primeiros movimentos nessa área introduziu a noção de método. Essa, por sua vez, propôs um processo disciplinado ao desenvolvimento de software, para que tal atividade fosse mais previsível e eficiente, desenvolvendo um conjunto de rotinas detalhadas para todo o processo [1].

> Um processo de software (ou metodologia de desenvolvimento de software) é um conjunto de atividades e resultados associados que auxiliam na produção de software. Dentre as várias atividades associadas, existem por exemplo a análise de requisitos e a codificação. O resultado do processo é um produto que reflete a forma como o processo foi conduzido. [2]

Ao longo dos anos, com o intuito de padronizar as métricas de trabalho, foram surgindo várias metodologias, mas que em sua maioria, acabou tornando o processo muito mais longo e burocrático.

As metodologias ágeis surgem com o passar dos anos para ser um meio termo entre a falta de processos e uma quantidade exagerada dos mesmos, diminuindo algumas etapas e tornando menos custoso e com um melhor tempo de entrega. Mais recentemente temos a adoção do *DevOps*, que surgiu como alternativa ao modelo ágil, com o intuito de manter um processo com rotinas rigorosas de integração e entrega contínua e grande quantidade de testes, sem perder a flexibilidade de gestão proposta no contexto ágil.

Tendo em vista a evolução que ocorreu no ambiente de desenvolvimento de software, como também o aumento da complexidade de projetos desenvolvidos no Laboratório de Aplicações de Vídeo Digital (LAVID), da Universidade Federal da Paraíba (UFPB), foi identificada a necessidade de estudar um processo de gestão e organização de projetos e pessoas do laboratório, como também uma estrutura de trabalho que auxiliasse uma maior qualidade de software, dos entregáveis solicitados em projetos atuais.

1.1 Objetivos gerais e específicos

O principal objetivo deste trabalho é organizar uma lista de rotinas e processos que auxiliam o desenvolvimento de software nas áreas de gestão, comunicação interna e qualidade no contexto de gerenciamento de projetos de uma organização de pesquisa e desenvolvimento. Para tanto, os seguintes objetivos específicos foram tratados:

1. Identificar os modelos, métodos e processos do desenvolvimento de software listados na literatura técnica e científica.
2. Sistematizar um conjunto de rotinas para auxiliar nos processos de gestão, comunicação interna e qualidade de software.
3. Aplicar a proposta em um estudo de caso de cinco projetos do LAVID.
4. Verificar os impactos da aplicação da proposta.
5. Sintetizar as lições aprendidas e os desdobramentos do estudo.

1.2 Metodologia

Para o desenvolvimento dessa pesquisa, foi utilizado o método *Design Science Research* (DSR), que une camadas científicas à produção tecnológica, tendo como um dos seus princípios fundamentais o desenvolvimento de artefatos, tais como *frameworks*, métodos, modelos, teorias de projetos, entre outros. Podemos compreender o DSR como:

> Método de pesquisa orientado a solução de problemas, a Design Science Research (DSR) busca, a partir do problema, construir e avaliar artefatos que permitam transformar situações, alterando suas condições para estados melhores ou desejáveis. [18]

A figura 1 apresenta a divisão feita para abordar todos os tópicos necessários para a consolidação da pesquisa, baseando-se no *Design Science Research*.

Figura 1: Aplicação do Design Science

No primeiro ponto, referente a conscientização do problema, o estudo será dividido em ambiente externo (ver capítulo 2), que por meio de uma revisão da literatura existente , irá apresentar uma visão geral dos métodos e processos existentes, e ambiente interno do estudo de caso (ver capítulo 3), que tem como objetivo entender o funcionamento do laboratório e identificar pontos de melhoria, por meio de entrevistas, tanto com alunos que trabalham no laboratório, quanto com gestores e professores. Já o tópico de proposição do artefato (ver capítulo 5) se divide em apresentar os conceitos que fundamentam a abordagem escolhida para propor o artefato, seguido pela proposta de práticas a serem utilizadas no LAVID, que tem como objetivo solucionar os problemas encontrados na etapa de conscientização do problema. Para a avaliação da proposta (Ver capítulo 6), serão descritos os casos de uso em que o processo foi aplicado. Esses casos de uso serão cinco projetos do laboratório, sendo eles o Vídeo for Health (V4H), projeto voltado ao desenvolvimento de um sistema especializado em teleconsultas; o Hope, que tem como objetivo desenvolver uma plataforma que integra dependentes químicos; o Paraíba Humana Inteligente (PHI), que teve como foco reformular a plataforma educacional SABER, além de criar um aplicativo móvel com o intuito de facilitar o

uso da plataforma para três personas: Aluno, Responsável e Professor; o VLibras, suíte focada na tradução de conteúdos em português para Libras e o projeto com a OvermediaCast, que visou melhorar e automatizar a produção de videobots. Para a última etapa, foi realizada uma análise e discussão dos resultados (Ver capítulo 8), baseando-se na experiência obtida nos projetos, como também apresentado os impactos e desdobramentos futuros para novos estudos.

2. REVISÃO DA LITERATURA

Neste capítulo, apresentam-se os tópicos que fundamentam essa pesquisa.

2.1 A evolução dos modelos

O primeiro modelo de processos para auxiliar o desenvolvimento de software foi criado em 1970, por Winston Walker Royce, o Modelo em Cascata que acabou recebendo esse nome pela forma com que o processo é dado dentro das fases de trabalho. Nesse tipo de modelo, o trabalho é baseado em planos, iniciando pela especificação e planejamento, passando pela elaboração da arquitetura do projeto, seguindo para a implementação e findando na realização de testes unitários e a integração das funcionalidades ao sistema [3]. As novas fases do projeto só devem ser iniciadas quando as anteriores forem concluídas, uma por vez, o que pode acabar sendo um ponto negativo nesse modelo, pois não existe flexibilidade e adaptações no processo. A entrega para o cliente acaba sendo bastante demorada e possíveis problemas podem acabar aparecendo ao longo do caminho, considerando que se algum processo quebrar todo ele precisará ser refeito do topo. De todo modo, segundo Sommerville, autor do estudo [4], “o modelo em cascata deve ser usado apenas quando os requisitos são bem compreendidos e pouco provavelmente venham ser radicalmente alterados durante o desenvolvimento do sistema.”

Para realizar o controle de processos imprevisíveis, temos o desenvolvimento iterativo e incremental, que tem como foco produzir versões funcionais do sistema final para a realização de testes, fazendo com que a aplicação real do projeto mostre as suas funcionalidades, mas principalmente, descubra suas falhas, sejam elas em termos de bugs no sistema, como também, em requisitos incompreendidos durante à coleta. Em processos adaptativos, voltados para desenvolvimento por iteração, a cada iteração realizada, é possível apresentar para o cliente o que está sendo feito e realizar a aprovação ou solicitar alterações no percurso, promovendo uma maior visibilidade do estado do projeto, além de melhorar a relação cliente-empresa.

O desenvolvimento iterativo e incremental, tem como vantagens no desenvolvimento à obtenção de um *feedback* mais rápido por parte dos clientes, além da diminuição na quantidade de documentos e análise de requisitos, quando comparado ao modelo em cascata, o que promove um menor custo para que esses requisitos sejam adaptados no decorrer do processo, sem contar com a redução de riscos entre as etapas. Ademais, a entrega do sistema acontece em um período de tempo bem menor [4].

Um processo bastante conhecido é o Processo Racional Unificado ou RUP, que apareceu quase ao mesmo tempo que os métodos ágeis, gerando uma discussão sobre a compatibilidade dos mesmos. Este, foi criado por Ivar Jacobson, com a contribuição de Philippe Krutchten e Walker Royce [5]. RUP baseia-se em diversos princípios de desenvolvimento de software, sendo assim não só um processo, mas uma estrutura de processos, por combinar desenvolvimento iterativo, arquitetura baseada em componente, gerenciamento de requisitos, controle de qualidade de software, entre outros princípios [6]. O RUP é constituído por quatro fases, sendo elas: Concepção, elaboração, construção e transição. Essas fases unitárias podem ser executadas de forma iterativa e ter os resultados desenvolvidos de forma incremental, enquanto que todo o conjunto de fases pode ser executado de forma incremental. Contudo, este processo era muito grande, prescritivo e de difícil adoção ampla nos projetos.

O desenvolvimento ágil que sucede o processo citado acima, envolve muitas abordagens específicas, como *Extreme Programming* (XP), *Scrum*, Desenvolvimento *Lean*, etc. Onde cada uma dessas tem suas próprias ideias, estruturas e líderes [1]. São várias comunidades dentro do ágil, utilizando os mesmos princípios gerais e isso é o que promove a versatilidade do método. O termo ‘ágil’ começou à ser utilizado em 2001, após à criação do Manifesto ágil, originalmente escrito em 2000, por 17 autores, que defendia abordagens de desenvolvimento iterativas e que tinha como um dos seus objetivos definir um nome para atuar de forma abrangente em várias abordagens, o que acabou levando ao termo ‘ágil’ [7].

A difusão da XP[1] ocorreu em 1997, por Kent Beck, sendo uma abordagem fundamentada em cinco valores: Comunicação, Simplicidade, Feedback, Coragem e Respeito. Além de catorze princípios e vinte e quatro práticas, tendo como ideia que as práticas são ações concretas que podem ser realizadas durante a rotina de trabalho de uma equipe, enquanto os valores são o conhecimento que dependem da aplicação da prática e o pilar de fundamentação da abordagem.

O XP envolve diversos princípios que se baseiam nos métodos ágeis. Nesse tipo de abordagem, os requisitos são expressos como cenários e são chamados de 'história do usuário', que depois de coletados são divididos em tarefas. O trabalho do desenvolvimento é feito em pares e os testes são desenvolvidos para cada tarefa antes mesmo do código ser escrito. Assim que o código é integrado ao sistema, os testes são realizados e obtendo sucesso, o *release* é feito. Nesse modelo, várias versões podem ser implementadas, testadas e liberadas em um único dia, por programadores diferentes [4].

Assemelhando-se ao método citado anteriormente, o *Scrum* se desenvolveu entre as décadas de 80 e 90, por Ken Schwaber e Jeff Sutherland e é uma das metodologias mais utilizadas. Scrum e Ágil costumam ser confundidos por várias pessoas, porém, a ideia do scrum é fornecer um processo/estrutura para a organização do seu trabalho, enquanto que o ágil tem uma cultura, requer mudanças estruturais no time. O *Scrum* é utilizado majoritariamente em projetos de desenvolvimento de software, apesar de proporcionar o uso em diversas áreas. A ideia central é diminuir o risco, promover uma maior transparência em todos os processos, aumentar a velocidade de entrega e tornar o processo ainda mais flexível e adaptável do que os outros apresentados [8]. Esta metodologia tem como foco o gerenciamento de desenvolvimento de software e trabalha com '*sprints*', que consistem em dividir o processo em iterações com duração de trinta dias, onde serão realizadas entregas contínuas de software funcionando. Todavia, o *Scrum* é flexível e pode ser adaptado de formas diferentes de acordo com o que será trabalhado, isso inclui quantidade de sprints, duração de interações, entre outras práticas.

[1] *Extreme Programming* é um método de desenvolvimento de software, leve, não é prescritivo, e procura fundamentar as suas práticas por um conjunto de valores.

Existem alguns artefatos como, *product backlog*, *sprints, sprint backlog e entrega,* além de processos necessários para a todo o trabalho, que são: *Sprint planning, daily scrum, sprint review* e a retrospectiva, que é reunião final. Para que haja um controle de trabalho e um *feedback* rápido dos processos que estão sendo feitos, são realizadas reuniões diárias de alinhamento, com duração de quinze minutos em geral, são essas as 'daily Scrum'. Dentro desse *framework*, existem três papéis centrais, que são: *Scrum Master, Product Owner e Development Team*. O primeiro papel é responsável por conduzir os processos de toda a equipe, é quem ensina, educa e corrige, como é dito no Guia do Scrum. Vale salientar que esse papel é realizado por uma única pessoa. O segundo, é o responsável por todos os requisitos do projeto, regras de negócio, prioridades, prazos, todo o *backlog* do produto. É também um papel individual. Por último, o time de desenvolvimento é composto por todas as pessoas responsáveis por colaborar com o desenvolvimento do produto. Estes podem e devem se reorganizar de acordo com as tarefas que vão ser realizadas, baseando-se sempre em tudo que foi estabelecido pelo *product owner*.

O desenvolvimento *Lean*, embora não seja considerado puramente um método ágil, possui 7 princípios como base que se alinham perfeitamente aos princípios ágeis. Eliminar desperdício, fortalecer o time, entregas rápidas, otimizar o todo, construir qualidade, adiar decisões e ampliar conhecimento. Fatores como evitar retrabalho, fazer entregas contínuas e rápidas, realizar testes, refatoração e trabalhar com integração contínua de software, não deixar para último momento decisões que são urgentes, facilitar e promover uma comunicação entre equipes e usuários durante o processo de desenvolvimento, são todos pontos de alta importância para um bom funcionamento do método.

2.2 Método Ágil

Em 2001, ocorreu um movimento formado por Kent Beck e outros dezesseis desenvolvedores, que destacavam métodos modernos e flexíveis com o intuito de promover sempre uma entrega e recebimento de feedbacks rápidos. Métodos ágeis promoveram uma nova visão e criaram um novo caminho a ser seguido dentro da produção de software. A busca pela agilidade era algo habitual.

A metodologia ágil propõe o foco em interações e nos indivíduos, promove a ideia de que ter um software funcional é mais importante que uma documentação completa, além de sugerir uma comunicação e parceria contínua entre clientes e desenvolvedores do projeto. Essa parceria torna-se possível pois essa metodologia faz uso do desenvolvimento incremental, de modo que a cada feature apresentado, é possível adaptar-se e incluir outros requisitos necessários [4]. A adaptação é um fator essencial nesse tipo de processo, dado o fato que os requisitos do sistema irão sempre sofrer algumas mudanças. Acima de tudo, essa metodologia exige uma adaptação e mudança na cultura de trabalho, de modo que toda a equipe consiga adaptar-se aos valores e princípios que a mesma carrega.

A ascensão do método ágil durou um longo período e enquanto ocorria, o método foi aclamado por algumas empresas. A 37signals, hoje chamada de Basecamp, escreveu um livro chamado "Caindo na Real", que retrata como os projetos de sua empresa tiveram sucesso utilizando do ágil. Os autores ressaltam alguns fatores como importantes. Minimizar reuniões, funcionalidades e pessoas nas equipes, de forma que o projeto seja sempre tratado como algo pequeno e simples, porém flexível, com o foco de obter um produto ótimo, mas que tenha somente funções essenciais é de alta importância para um bom processo de desenvolvimento. Trabalhar com equipes multifuncionais e sempre integradas e ter boa comunicação com os seus clientes são mais algumas das dicas de processo de trabalho que a empresa nos ensina e diz ter utilizado nas suas criações de sucesso.

Apesar do grande sucesso do ágil, é iniciado um movimento de declínio, onde este passa a ser considerado um método morto por vários autores. Em seu primeiro artigo sobre o assunto, Agile is Dead, Matthew Kern faz um apanhado com outros autores, indagando esse fato e curiosamente recebe várias afirmações que confirmam a ideia [9]. Alguns consideram que esse fato se deu pela má utilização do método, outros pela inaplicabilidade em certos problemas ou grupos de empresas, visto que este tinha um ponto ideal e são diversas as diferenças existentes dentro desses grupos.

No ágil, um dos problemas comuns é encontrado quando paramos para observar o projeto por perspectivas da IHC[2]. A queda no sucesso do método ágil pode ser justificada

[2] A interação humano-computador (IHC) é uma área de pesquisa e prática que surgiu no início dos

levando em consideração os problemas apresentados. A partir daí é iniciado uma possível substituição ou adoção de uma abordagem híbrida. Segundo Armitage,

> A comunidade de métodos ágeis raramente menciona os usuários ou a interface com usuário como um todo, o que significa que ou eles negligenciam a experiência de uso, ou estão focando projetos com menor necessidade de uma experiência de uso sofisticada. [15]

Em Agile is Dead 2, é apresentada uma visão de mercado para o modelo. É dito que o ágil pode ser representado pela curva S, iniciando de forma lenta seu processo mercadista, engatando em um período de crescimento e logo depois passando por uma baixa de vendas e/ou clientes, onde é iniciado o processo de saturação do método, devido a falta de espaço para crescimento. Kern e Matthew destacam que, em 2014, 94% das empresas utilizavam do ágil, enquanto que em 2009, essa utilização já encontrava-se em 19% [16]. Então, desse ponto em diante, a tecnologia começa a ser substituída. Uma das novas técnicas que vem ganhando espaço com toda essa história é o *DevOps*, que apoia-se bastante na metodologia de Scrum e requer integração, implantação e entrega contínua, processos de fundamental importância para o desenvolvimento e manutenção de um sistema.

> "Você nunca deve tentar impor o trabalho ágil em uma equipe que não deseja experimentá-lo." (FOWLER, Martin)

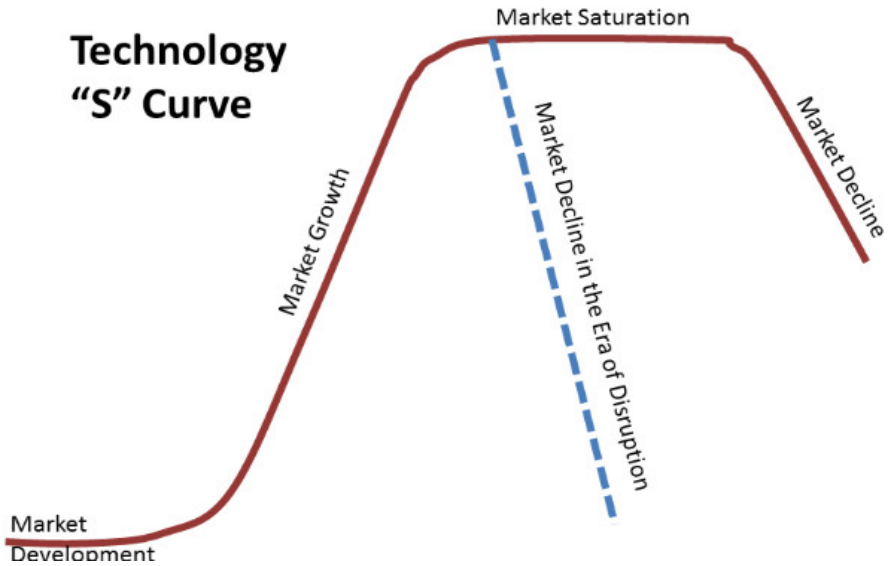

Figura 2: Curva "S"

Devido a más práticas, o falso-ágil, como ficou conhecido, é resultado do uso do nome ágil como um método, sem colocar suas práticas e valores em ação. Todo método ágil sofre mudanças contínuas, fazendo com que não sejam feitas programações a longo prazo, para que tudo possa ser adaptado ao longo das necessidades que surgem no processo. Como citado anteriormente, todo tipo de método deve ser maleável e passível de mudanças, para que seja possível um bom funcionamento em diversos tipos de equipes e projetos. Infelizmente, muitos métodos foram adotados como religião pelos seguidores, sem antes fazer uma avaliação e comparação dos métodos e de suas práticas. Esse fato é um dos responsáveis por perfazer o falso-ágil.

2.3 Um novo método

Atualmente, os processos encontram-se muito mais integrados do que no início do desenvolvimento de software. Desenvolvimento e testes não são mais atividades divididas por equipes e finalizadas em tempos diferentes de execução. Atualmente, estas atividades estão atreladas em um só processo e devem ser realizadas de forma simultânea. É aqui onde entra o movimento *DevOps,* que chega com o intuito de unir os processos de desenvolvimento com os de análise, teste, requisitos, operações e manutenção, sem que estes sejam quebrados por equipe, promovendo uma cultura de responsabilidade compartilhada, com o intuito de distribuir serviços rapidamente.

Para esse fim, as equipes de desenvolvimento e operações trabalham de forma colaborativa, algumas vezes, combinadas em uma só. Algumas mudanças na organização e funcionamento da equipe são necessárias para que o modelo *DevOps* funcione. Em alguns modelos do movimento, as equipes de *Quality Assurance* e segurança podem integrar-se com o desenvolvimento, e quando um determinado produto requer a segurança como prioridade de todos da equipe, pode-se chamar o processo de *DevSecOps* [17].

Uma dessas mudanças é possuir e apoiar equipes autônomas, de alta confiabilidade, que tenham a capacidade de tomar decisões e aplicar alterações sem grandes complicações no processo. Ter um bom controle de versão, além de um sistema de gerenciamento de projeto com tickets, fazem muita diferença para todo esse processo. Para garantir que as implantações

sejam feitas com qualidade durante todo o processo de desenvolvimento, também é necessário abraçar técnicas de Entrega Contínua com testes automáticos, de modo que o processo ocorra com implementações mais confiáveis e com baixo risco de erro [10]. A automação, é portanto, fundamental para o movimento *DevOps* e facilita todo o processo de trabalho das equipes, fazendo com que o processo ocorra de forma automática desde o lançamento do software, da fase de criação, até a fase de implantação. A figura 3 apresenta todos os processos relacionados a esse movimento.

O aumento da excelência técnica torna-se de extrema importância, para realizar uma Refatoração de Software e solucionar o problema do falso-ágil. Esta, é fundamental para mudança tardia de requisitos, mantendo o software funcionando, além de facilitar o processos como, encontrar bugs e tornar o software mais claro. Todavia, para que esta aconteça, faz-se necessário testes, que são responsabilidade da Integração Contínua, representada na figura 2, a qual promove um conceito de agregação, onde a execução de *builds*[3] e testes são realizados juntos e adicionados à base de código existente, promovendo segurança em relação a mudanças, sincronização e agilidade para a solução e correção de problemas [11]. O conceito de Entrega Contínua reúne um conjunto de práticas que tem o intuito de assegurar a aptidão de um novo código a ser disponibilizado em ambiente de produção a qualquer momento. Essa, por sua vez, promove menos riscos no lançamento e mais velocidade na entrega do produto. Por fim, a implantação contínua traz a capacidade de automatizar essas entregas, de forma que o produto seja implantado no servidor automaticamente, finalizando assim o ciclo do *DevOps*, que está apresentado na figura 3.

[3] "*Build*, no contexto do desenvolvimento de software, é uma versão "*compilada*" de um software ou parte dele que contém um conjunto de recursos que poderão integrar o produto final." (Wikipédia, 2020)

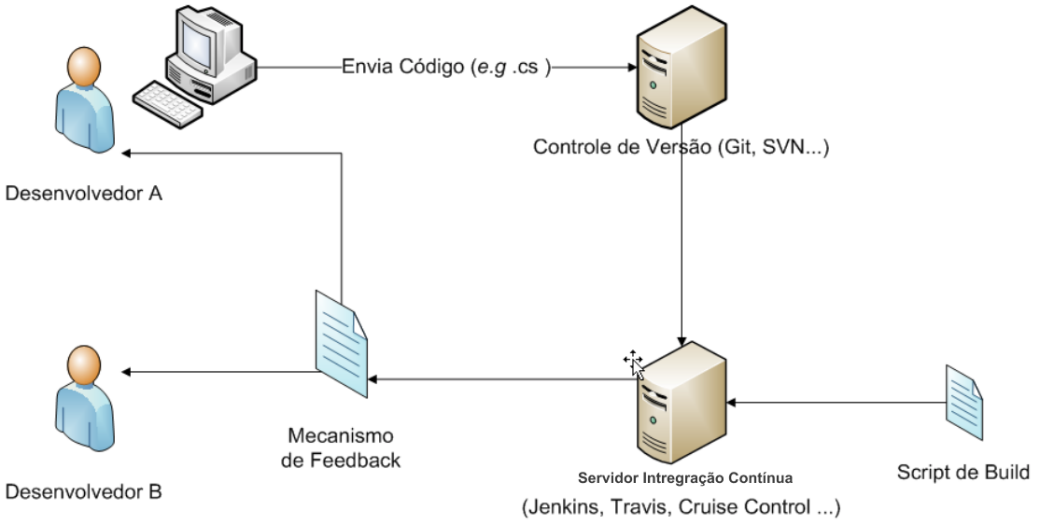

Figura 3: Processo de integração contínua

Aprofundando sobre a integração contínua, pode-se afirmar que essa representa um processo de extrema importância para um bom desenvolvimento de software. O termo teve origem no processo de desenvolvimento de Programação Extrema de Kent Beck, sendo uma de suas doze práticas originais. O processo consiste, resumidamente, em versionar todo o código-fonte do projeto em um repositório, dividindo-o entre versão de 'linha principal', 'candidato à lançamento' e 'desenvolvimento'.

A integração contínua possui uma alta camada de testes automáticos em sua compilação, que devem ser realizados e minimamente aprovados antes de liberar a *release* para a camada de 'candidato à lançamento'. Esses testes são importantes para identificar os principais problemas e reportar as falhas, com o intuito de que todos os problemas sejam corrigidos e assim o *release* seja permitido. Para completar o ciclo, a compilação é feita de forma automática, fazendo com que o código da camada mais baixa, onde tudo é desenvolvido, seja vinculado à um executável e os testes automáticos sejam rodados, de modo que se tudo for criado e testado de forma funcional, tem-se uma boa construção como resultado, rodando a compilação de confirmação e fazendo o merge na master [12]. No geral, o maior benefício da integração contínua é o risco reduzido. Esse processo fornece um

software estável, com bom funcionamento e uma quantidade ínfima de bugs, não pelo fato de eliminá-los, mas por facilitar o processo de encontrar e remover os mesmos.

Algumas boas práticas de integração são [12]:

- **Utilizar repositórios** com todas as informações essenciais do projeto como, configurações de IDE, um bom sistema para controle de suas ramificações, entre outros.
- Possuir um **ambiente automatizado** para compilação, que inclui todas as funções, desde a retirada do esquema do banco de dados do repositório, até a inicialização no ambiente de execução.
- Incluir **testes automatizados** no ciclo. Esses testes devem ser iniciados com um comando e verificados automaticamente, de modo que se o resultado da execução dos testes apresentar alguma falha, deve ser causada também uma falha na compilação.
- **Integração da equipe** também é facilitada pela integração contínua, de modo que um sempre pode acompanhar o trabalho que está sendo executado pelo outro, desde que todos os desenvolvedores estejam devidamente comprometidos com o repositório.
- ***Builds* quebradas devem ser corrigidas imediatamente**, especialmente se elas pertencem à linha principal do código.
- Manter uma **compilação rápida** para fornecer um feedback rápido é um dos pontos chaves. Para isso, pode ser estabelecida uma compilação faseada, onde existem várias compilações feitas em sequência.
- **Testes secundários** são importantes, pois esses testes feitos em um ambiente controlado fazem com que o risco existente neles seja evitado no ambiente de produção.
- **Implantação automática** acelera o processo e reduz erros. Todavia, quando se utiliza disso, é necessário que também tenha um recurso de reversão automática, para possíveis problemas.

A entrega contínua juntamente com o implantação contínua finalizam esse ciclo, onde o software tem como prioridade ser implantável durante todo o processo de forma automatizada, estando sempre pronto para implantação, além de promover um feedback rápido para toda a equipe. Durante todo o processo deve ser mantido um grande trabalho colaborativo da equipe, a cultura do *DevOps*, pois a entrega contínua incorpora todo o processo de integração contínua, responsabilizando-se unicamente pelos estágios finais de implantação da produção [13]. Esse tipo de abordagem promove alguns benefícios em comum com a integração, pelo fato dos dois processos evoluírem juntos. O processo de implantação contínua também incorpora ferramentas de monitoramento, com o intuito de obter métricas como tempo de resposta, quantidade de acesso e volume de tráfego. Dentre os benefícios, risco de implementação reduzido, pelo fato de estar sendo implementada alterações de pequeno porte, reduzindo os erros e promovendo um maior controle de qualidade, além de feedback do usuário, pois o quanto antes o programa estiver rodando com usuários reais, mais rápido você terá uma resposta sobre os seus resultados.

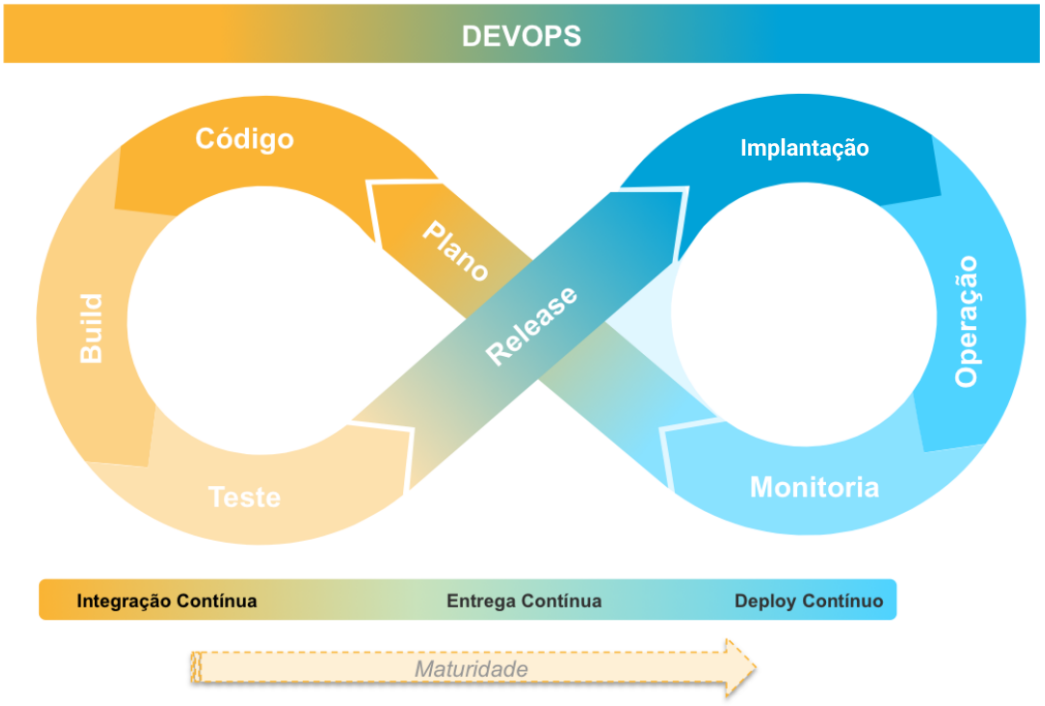

Figura 4: Ciclo de um produto utilizando *DevOps*

3. CONSCIENTIZAÇÃO DO PROBLEMA

O contexto interno, isto é, o cenário de aplicação desta pesquisa é o LAVID e os processos de trabalho já existentes no laboratório. Para entender o cenário geral para o primeiro ponto, foi feita uma entrevista com três professores e gestores de projeto, como também, com três ex-alunos de computação que ainda trabalham no laboratório. E com o intuito de ter esse mesmo entendimento sobre um projeto e tentar aplicar melhorias, foi definido e aplicado alguns pontos de *quality assurance*[4].

Para ter resultados um pouco mais assertivos, foram escolhidos três perfis de gestores para serem entrevistados, entre eles, uma pessoa que é responsável pela gestão geral do laboratório e outras duas que geriram os principais produtos do LAVID. Ademais, dois bolsistas que trabalharam em projetos distintos, seguindo a linha de trabalho de diferentes gestores. O perfil dos 5 participantes das entrevistas está sintetizado na Tabela 1.

Vale salientar o fato de que todas as entrevistas foram realizadas entre o dia 24/07/2020 à 28/08/2020.

Tabela 1: Perfil dos participantes da entrevista

Entrevistado	Vínculo UFPB	Função em projetos	Experiência no LAVID
E1	Servidor	Gestor geral	9 anos
E2	Professor	Coordenador de projeto	20 anos
E3	Professor	Coordenador de projeto	15 anos
E4	Estudante	Desenvolvedor	3 a 4 anos
E5	Estudante	Desenvolvedor	5 anos

Fonte: A autora

[4] A área de Quality Assurance (QA), tem como objetivo promover métricas para garantir a qualidade de um produto, prevenindo erros ou falhas que possam vir a existir.

3.1 Entrevistas com os perfis E1, E2 e E3

A fim de obter respostas mais técnicas e focadas no funcionamento do laboratório, foram abordados alguns questionamentos que estão detalhados nas seções seguintes.

3.1.1 Modelo de gestão utilizado

Com o intuito de entender o cenário geral de gestão existente no laboratório, foi indagado se havia algum modelo de gestão já utilizado em todos os projetos, além de quais as maiores dificuldades enfrentadas.

Para tanto, o perfil E1 informou que a organização do projeto não passava pela gestão geral, sendo somente coordenada pelo professor responsável em cada projeto, não havendo portanto um modelo geral de trabalho. O perfil E2, por sua vez, indicou que os projetos são desenvolvidos de uma maneira livre, sem seguir nenhum formato específico, e solucionando as demandas da forma que o grupo mais se adaptasse. Enquanto isso, o perfil E3, informou que nos projetos coordenados pelo próprio, normalmente é utilizado Scrum. Além disso, informou que, para o seu principal projeto, foi necessário durante o andamento estabelecer um processo de trabalho, por meio de divisão de times, onde cada um deles tinha um líder, que respondia sempre ao gestor principal do projeto. Foi comentado também que não era usado nenhum processo para organização e gerenciamento de tarefas, e que por esse motivo, era bastante trabalhoso o gerenciamento de todo o projeto.

Ademais, ficou clara a dificuldade de especificar um ambiente destinado à comunicação da equipe, que é toda baseada em grupos de Whatsapp, fator que interfere nas limitações de ambiente de trabalho e ambiente pessoal.

3.1.2 Padrão de entregas e garantia de qualidade

Sobre o funcionamento das entregas, foi questionado se existia algum padrão para as entregas e se havia uma garantia de qualidade sobre as mesmas.

O perfil E1, indicou que não existe um controle específico do que terá em cada entrega, já que os entregáveis variam de projeto para projeto, afirmando também que não é utilizado nenhuma formalização que seja geral do laboratório, independente de projeto. E2 trouxe a informação de que o laboratório nunca deixou de realizar nenhuma entrega, pois mesmo que haja alguma dificuldade, pessoas de outros projetos são remanejadas de modo a suprir aquilo que é necessário. O mesmo também citou o fator relatado pelo entrevistado E1, sobre a diferença entre os entregáveis de um projeto para outro. A questão da garantia de qualidade foi mais abordada pelo entrevistado E3, que informou que a qualidade da entrega é um ponto fraco do laboratório, devido a não existência de QA em todos os projetos.

A preferência do laboratório em direcionar os investimentos em perfis alheios a qualidade ficou bastante clara, principalmente pelo fato de que em um projeto específico, a equipe de QA acabou sendo cortada por dificuldades de orçamento, mesmo sendo sabido a importância da mesma quanto às entregas.

Em maio de 2020, devido uma necessidade do projeto V4H, a coordenação do LAVID decidiu criar um núcleo de qualidade para prestar serviços aos projetos que os desejarem. Esse grupo, até o presente momento realiza trabalho em cinco projetos que estão em andamento, sendo eles: V4H, OvermediaCast, Hope, PHI e Vlibras.

3.1.3 Necessidade de estabelecer um processo

Após ter uma visão geral da abordagem existente, foi interrogado se os entrevistados sentiam a necessidade de criar um processo básico de gestão, comum a todos os projetos. E porque esse sentimento ocorria.

O entrevistado E1 afirma que sente essa necessidade clara no dia a dia de trabalho, pela dificuldade que acaba ocorrendo quando se trata de acompanhar as atividades de todos os projetos. Acrescentando ainda que ter esse acompanhamento mais formal facilitaria bastante o controle e registro de todas as entregas realizadas, fato que se justifica baseado no nível de exigência atual existente nos projetos.

O E2, relatou que essa era uma necessidade tão presente que o fez criar o núcleo de qualidade, mencionado na seção 3.1.2, para obter uma maior exigência com relação às entregas, passando sempre pelo crivo de testes e respeitando o processo definido entre as equipes. Para o entrevistado E3, existe sim essa necessidade, principalmente para a realização do acompanhamento das atividades, tanto em grandes equipes de um único projeto, quanto dos outros projetos existentes no laboratório.

> Saber o que ocorre em cada projeto, tal qual suas necessidades, facilitaria muito tanto na reutilização de artefatos, quanto na solicitação e divisão de materiais de acordo com a verba disponibilizada (E3, sobre pergunta 3);
>
> Existe a necessidade de ter pessoas dedicadas somente à gerenciar cada projeto, pois muitas vezes os professores acabam não tendo o tempo necessário para a função de gestão (E3, sobre pergunta 3);

Ficou claro o interesse dos entrevistados em um plano mais organizado para a execução do trabalho, tanto quanto a disponibilidade e o intuito de colaborar com os processos estabelecidos futuramente.

3.2 Entrevistas com os perfis E4 e E5

Para os perfis E4 e E5, foi indagado a visão do estudante/bolsista com relação às facilidades e dificuldades para realização do trabalho. Os questionamentos feitos estão detalhados nas seções 3.2.1, 3.2.2 e 3.2.3.

3.2.1 Participação em projetos

Para entender a participação dos entrevistados, foi perguntado em quais projetos do LAVID os mesmos trabalham e/ou trabalharam, como também seu cargo e a duração em cada projeto.

Tabela 2: Projetos e data de participação

Entrevistado	Cargo	Projeto	Ano de início	Ano de fim
E4	Desenvolvedor Web	Vlibras	2019	Em andamento
		Corning-Inspec	2018	2018
		GTAAS 2.0	2017	2017
		Open Signs	2016	2017
E5	Desenvolvedor Aplicativo Móvel	PHI	2020	Em andamento
		V4H	2020	2021
		TCE - Espaço Cidadania	2018	2018

Fonte: A autora

O perfil E4, informou que além de trabalhar com desenvolvimento *Web* para front e back, além de ter trabalhado com IA em alguns projetos. Já o E5, teve além do papel de desenvolvedor, a função de líder do time de desenvolvimento móvel, fazendo participações em reuniões com clientes e tendo liberdade para organizar as tarefas da forma que achasse adequado.

> [...] escolhi utilizar o trello, pois começamos sem, mas ficou um pouco bagunçado, então migramos (E5, sobre pergunta 1 e 2).

3.2.2 Empresa X LAVID

Contando com o fato de que os dois entrevistados além de possuírem experiência em alguns projetos do laboratório, atualmente trabalham também em empresas, foi abordado o questionamento sobre quais as maiores diferenças encontradas nos dois ambientes de trabalho.

O E4 explicou que a falta de acompanhamento dos professores/gestores nas atividades, tanto como a organização e divisão das mesmas, deixa muito a desejar quando se compara uma empresa com o LAVID.

> [...] às vezes falta organização quanto à essas coisas de padrão e projeto. [...] quando algum projeto ia iniciar, geralmente era falado "você ficará com isso e isso" e escrevia a tarefa no quadro e seguia (E4, sobre pergunta 3).

O entrevistado E5 ressalta como maior diferença a falta de uma pessoa responsável unicamente para gerenciar e destinar tarefas entre os projetos, acrescentando ainda que as atividades do laboratório são sempre separadas em blocos de notas ou só ouvindo aquilo que é falado e tendo que organizar-se sozinho.

> No LAVID não tem nenhuma linha bem definida de gerenciamento [...]. Eu acho que deveria ter pelo menos uma base, para cada um seguir, para ficar mais organizado (E5, sobre pergunta 3).

Apesar dos pontos citados com relação a planejamento e gerenciamento de atividades, ambos os estudantes ressaltam o ponto de terem conquistado uma diversidade de conhecimentos de extrema importância para a carreira, fato que só ocorreu devido a participação nesses projetos.

3.2.3 Pontos positivos e negativos

Foi pedido que os estudantes falassem sobre os pontos positivos e negativos sobre a experiência geral no laboratório.

Tabela 3: Pontos positivos e negativos

Entrevistado	Pontos Positivos	Pontos Negativos
E4	Comunicação fácil e acessível com os professores	Não atribuição formal de tarefas
	Flexibilidade de horários	Prazos não fixados

E5	Flexibilidade de horários	Falta de fluxo de teste
	Incentivo à pesquisa	Infraestrutura não oferece o suporte ideal
	Grupos muito acessíveis e dispostos a ajudar	

Fonte: A autora

Dado as respostas fornecidas pelos mesmos, ficam claros alguns pontos de melhoria, como também a sintonia com relação aos pontos positivos. Ambos informaram que todos do laboratório estão sempre disponíveis para ajudar e acrescentar nos seus conhecimentos, fazendo dessa uma rede muito boa para o trabalho.

3.3 Classe de problemas

Tomando como base a caracterização dos ambientes externos e internos surge a classe de problemas de um estudo para o DSR. Na figura 9 é apresentada uma síntese dos aprendizados obtidos na caracterização destes ambientes. A partir desses dois elementos identificou-se dois problemas a serem resolvidos.

Para a classe de problemas pode-se perceber dois itens. O primeiro é a definição de um processo com o intuito de melhorar o controle e definição de atividades, dentre outros aspectos. Este problema é fundamentado primordialmente pela falta de uma metodologia específica e fácil a ser seguida, além da pouca disponibilidade de tempo das pessoas existentes no projeto, considerando as atribuições exigidas dos professores e/ou coordenadores e a falta de profissional especializado nesse tipo de trabalho, como foi relatado pelos entrevistados.

O segundo item traz à tona as dificuldades de trabalho quando trata-se de garantia de qualidade. Essas dificuldades mostram-se presentes principalmente pela falta de mão de obra com vertentes de desenvolvimento em testes de software. Considerando o fato de que não são todos os projetos que possibilitam a alocação de vagas e recursos disponíveis para pesquisadores e bolsistas na área de QA, o cenário ideal é formar pessoas com um conhecimento misto, que consigam não só desenvolver o código, como também testar os

mesmos. Ademais, o fato de incluir o *DevOps* no processo além de facilitar e organizar o trabalho, garante a execução contínua de testes.

Sendo assim, a classe de problemas compreende dois tópicos que determinam o escopo a ser trabalhado neste estudo. As próximas etapas da pesquisa ocupam-se em construir um conjunto de rotinas capaz de promover uma solução satisfatória, a estes dois requisitos.

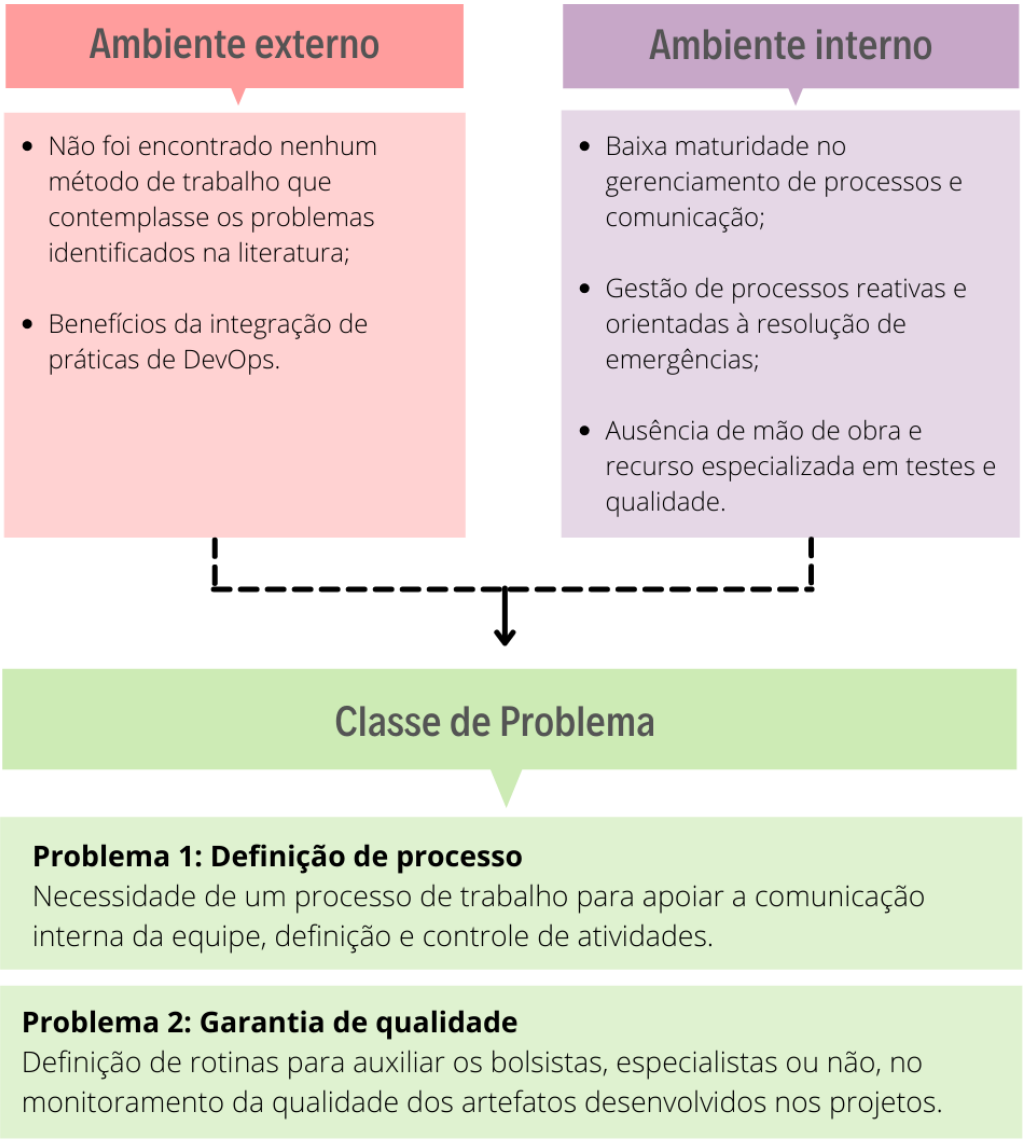

Figura 5: Classe de problemas

4. PROPOSIÇÃO DO ARTEFATO

Em face à classe de problemas apresentada no capítulo anterior, foi desenvolvido o artefato desta pesquisa. Neste trabalho de conclusão de curso propõe-se a organização de uma rotina de processos, com foco na melhorias para o ciclo de criação e desenvolvimento de novos projetos. O ciclo necessário para a construção de um artefato para o DSR passa por alguns pontos:

1. Levantamentos de artefatos que de algum modo solucionam as problemáticas do estudo;
2. Definição das bases teóricas que fundamentam a solução;
3. Desenvolvimento do artefato;
4. Avaliação de desempenho do artefato;

Nas seções 4.1 e 4.2 cada um desses processos serão aprofundados.

4.1 Trabalhos relacionados

É possível especificar alguns estudos que discutem modelos de processo para o desenvolvimento de software por uma ótica mais ampla, como é citado abaixo:

> "Um modelo de processo de software é uma representação abstrata de um processo de software. Cada modelo de processo representa um processo a partir de uma perspectiva particular, de uma maneira que proporciona apenas informações parciais sobre o processo (SOMMERVILLE 1995)."

Como apresentado anteriormente no capítulo 2 deste trabalho, existem alguns modelos base que podem ser aplicados em diversos aspectos no processo desenvolvimento como, cascata, incremental, ágil, entre outros. Todavia, para cada um deles deve ser feita uma análise particular do ambiente onde será aplicado, com o intuito de compreender a viabilidade, além de assegurar-se os prós e contras que à inclusão do mesmo trará a empresa/projeto.

Para o caso específico do LAVID, que possui uma dinâmica não tão restrita quanto a de uma empresa, mas que ainda assim necessita de um processo para à organização e

planejamento de entregas, algumas metodologias podem ser usadas como base de definição do processo, como é o caso do Scrum, que é uma metodologia ágil com o objetivo de gerir e planejar software.

São diversos os estudos de caso existentes sobre a utilização dos modelos de processo de software. Esses estudos tendem a elencar práticas para ajudar profissionais e pesquisadores das áreas de Computação e Gestão de Projetos. Os modelos apresentam contribuições pertinentes ao problema, todavia, como já falado no capítulo 2, as abordagens concentram-se em perspectivas gerais e com focos normalmente voltados ao ciclo de trabalho empresarial.

4.2 A proposta

Com o objetivo central de criar rotinas de trabalho, melhorar o gerenciamento de atividades e garantir a qualidade do software produzido, foram estabelecidas algumas práticas a serem utilizadas no LAVID.

A essência da proposta é indicar uma ferramenta de gerenciamento para definição e acompanhamento das atividades, de modo que o controle e coordenação do projeto seja facilitado. Além disso, é sugerida a inclusão de um time para controle de qualidade, responsável pela realização de testes e contribuições com o processo.

4.2.1 Gerenciamento de atividades

Almejando a escolha de uma ferramenta para a gestão das atividades de desenvolvimento de software, foram considerados os seguintes requisitos:

- Fácil utilização;
- Fácil acesso à artefatos gerais do projeto;
- Acesso gratuito e sem limites de usuários;
- Compatibilidade ou integração com as plataformas Google e Gitlab.[5]

[5] O Laboratório já possui a assinatura desses serviços para viabilizar o armazenamento e versionamento dos arquivos e códigos gerados pelos projetos.

- Gestão visual e colaborativa das atividades;
- Possibilidade de criação e compartilhamento de documentação;

Dentre as ferramentas analisadas, tivemos: *Trello*[6]*, Jira*[7]*, Microsoft Teams*[8] *e ClickUp*[9]. Foram encontrados alguns problemas como, limitações nos aspectos de organização entre projetos, sincronia com o *GitLab*, controle de *Issues* para a equipe de qualidade, altos custos para assinatura, limitações para versão gratuita e complexidade de uso. Considerando esses itens, a ferramenta sugerida para dar início à nova abordagem foi o *ClickUp*.

Após a escolha da ferramenta, sugere-se uma sequência de etapas para gerenciar o andamento das atividades.

- *BACKLOG* - Atividades que devem ser desenvolvidas em algum momento;
- *TO DO* - Atividades que devem ser iniciadas no ciclo de trabalho;
- *DOING* - Atividades que estão em andamento;
- *REVIEW* - Atividades finalizadas que precisam da revisão dos stakeholders e/ou líder técnico da equipe;
- *REOPEN* - Atividades que foram revisadas ou concluídas e que precisam de ajustes;
- *CLOSED* - Atividades finalizadas.

Essas etapas podem ser visualizadas na plataforma tanto no formato de *boards* quanto na forma de listas. As figuras 6 e 7 demonstram a visualização de *board* e organização de um projeto considerando essas etapas.

A feature de documentação do ClickUp, pode ser utilizada para fornecer uma base introdutória à novos integrantes, ser o local registro das documentações de referência dos projetos, que inclui desde templates essenciais ao projeto, até links com materiais importantes para o andamento das atividades. A figura 8 apresenta um exemplo da aba de documentação, nomeada “Doc and Files”.

[6] Trello, https://trello.com/pt-BR.
[7] Jira Software, https://www.atlassian.com/br.
[8] Microsoft Teams, https://www.microsoft.com/pt-br/microsoft-teams/free.
[9] ClickUp - Project Management Onboarding, https://clickup.com/onboarding.

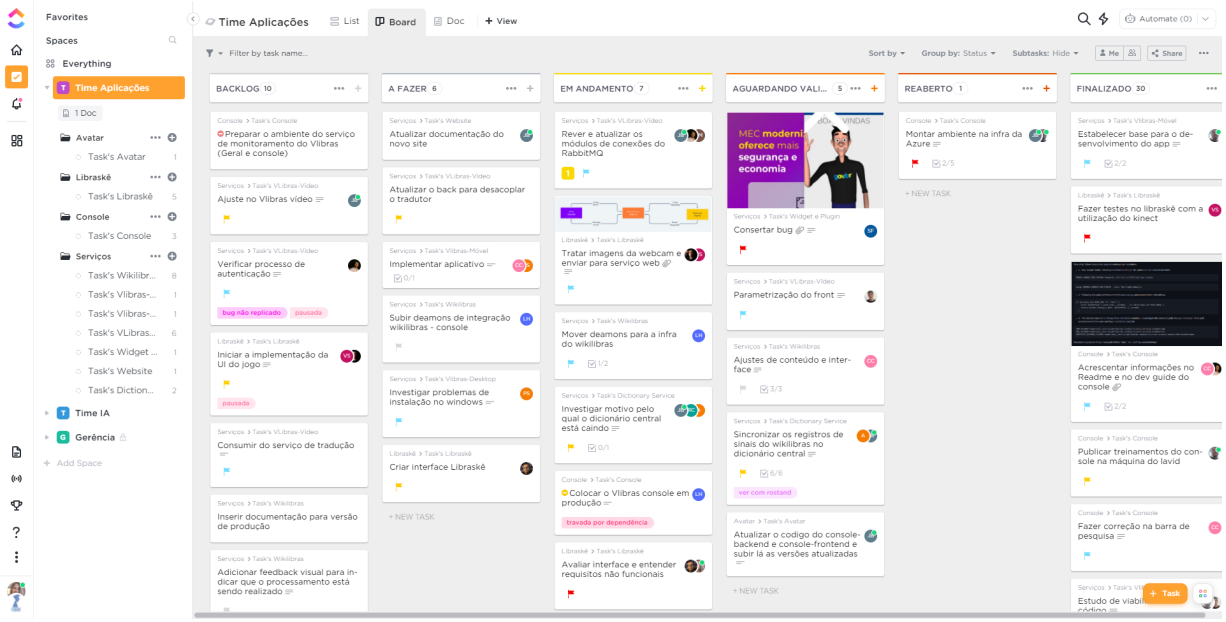

Figura 6: Boards do ClickUp para o projeto Vlibras

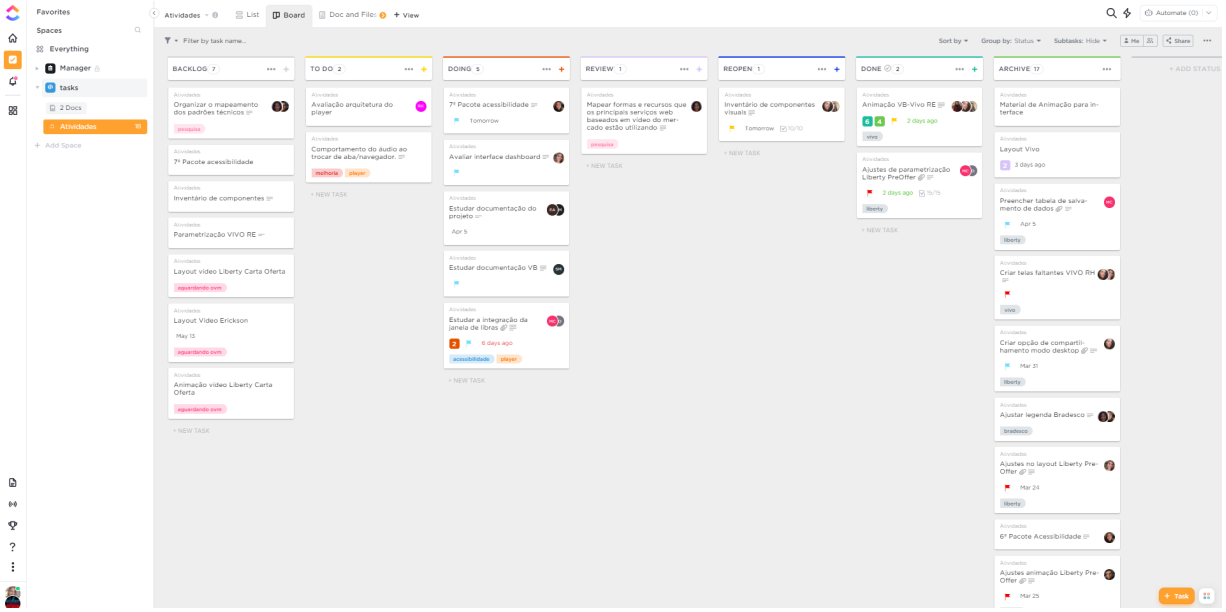

Figura 7: Boards do ClickUp para o projeto Overmediacast

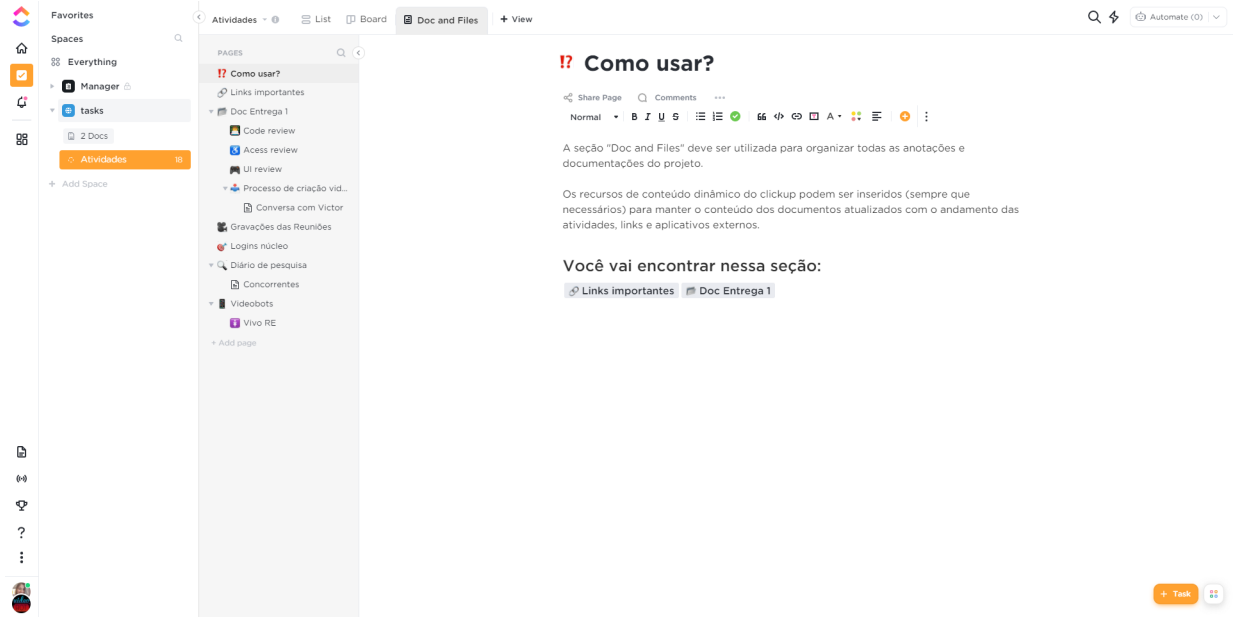

Figura 8: Documentação do projeto Overmediacast

No quesito comunicação interna entre bolsistas e pesquisadores, sugere-se um software de comunicação que atenda o seguintes requisitos:

- Comunicação síncrona e assíncrona a partir de textos, imagens e vídeos;
- Criação de enquetes ou formas de coletar a reação da equipe;
- Canal institucional ao invés de contato privado;
- Criação de grupos ou canais por ponto focal;
- Integração com os softwares de gestão de atividades e gerência de arquivos;
- Fácil acesso à artefatos gerais do projeto;
- Acesso gratuito e sem limites de usuários.

Nesse contexto, foi sugerida a ferramenta *Slack*[10] por atender os requisitos necessários. A figura 9 apresenta um exemplo de uso dos canais por temas ou funções comuns no projeto

[10] Slack, https://slack.com/intl/pt-br/.

em parceria com a Overmedia. Na imagem temos canais próprios para as discussões do time de desenvolvimento, design, gestão interna, dúvidas e reunião diária (*daily*).

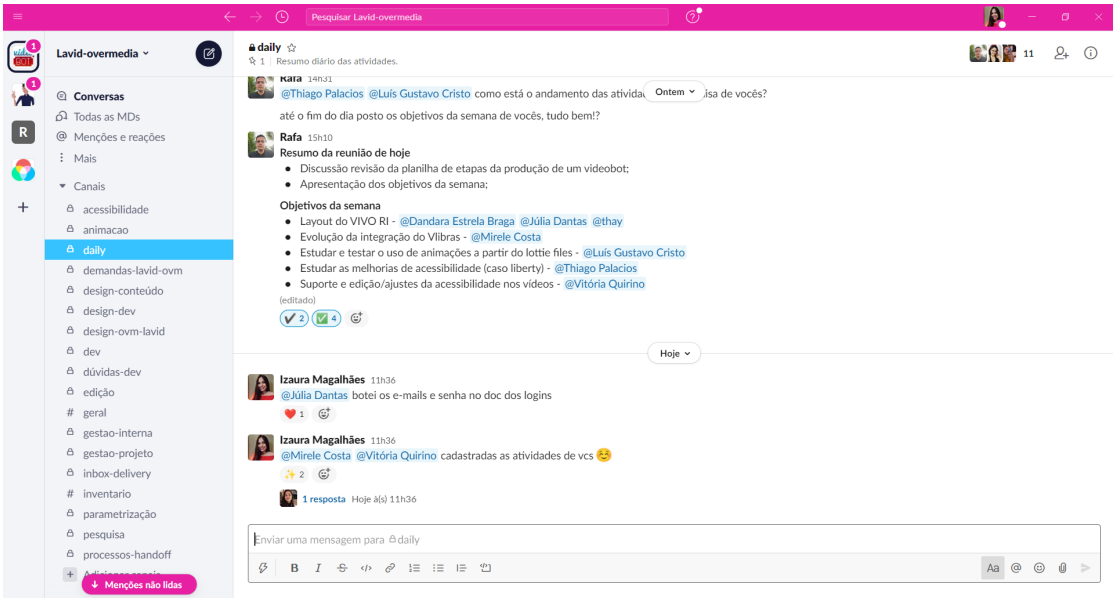

Figura 9: Slack do projeto Overmediacast

4.2.2 Quality Assurance

A fim de proporcionar uma maior qualidade para as entregas e garantir a execução dos processos, foi inserido no LAVID um time de qualidade, responsável por testar os softwares produzidos pelos projetos, como também planejar e guiar a execução dos mesmos. Para isto, estabeleceu-se alguns passos básicos a serem seguidos em todos os projetos.

A. Elaboração do plano de testes

   Neste documento devem estar contidas todas as informações gerais relacionadas ao projeto, uma referência dos requisitos que devem ser testados, os tipos de teste que serão realizados, ferramentas necessárias, recursos que serão utilizados nos testes e plataformas que serão utilizadas para a execução.

B. Mapeamento de casos de teste

Dado o fato que os casos de teste serão responsáveis por ser um guia na hora de criação e execução dos testes, o mapeamento deve ser baseado no documento de requisitos e na referência visual do projeto (protótipo de alta fidelidade), com o intuito de ter a maior cobertura possível do artefato proposto.

C. Cadastro de issues

As *issues* encontradas durante a execução dos casos de teste devem ser cadastradas no ambiente de desenvolvimento dos projetos (*GitLab*), seguindo os status da máquina de estados apresentada na figura 9 e utilizando das severidades apresentadas no guia de prioridade e severidade proposto pelo time de qualidade.

O guia fornece um detalhamento de como deve ser feita a classificação e ordenação de execução das *issues*. Os níveis de severidade são fornecidos pelo time de QA no momento do cadastramento na máquina de estados e indicam o quão preocupante a *issue* é. Já as prioridades devem ser aplicadas pelo time de desenvolvimento de acordo com a severidade recebida pelo QA e com o tempo disponível da equipe para a realização das tarefas. Este guia apresenta quatro níveis de prioridade, sendo elas, Critica, Alta, Média e Baixa, além de quatro opções de severidade, que são classificados como, Crítica, Grave, Moderada e Pequena.

D. Execução

A execução dos testes mapeados deve ser realizada mediante a entrega de um *release* notes vindo da equipe de desenvolvimento, que constará todas as funcionalidades que devem ser testadas. Essa execução deve ser feita em no mínimo dois navegadores para fornecer uma maior cobertura aos testes.

E. Documentação

Ao fim de cada execução deve ser feito um relatório com os resultados obtidos e o parecer dado para os resultados em questão. Exemplo de relatório do projeto V4H.

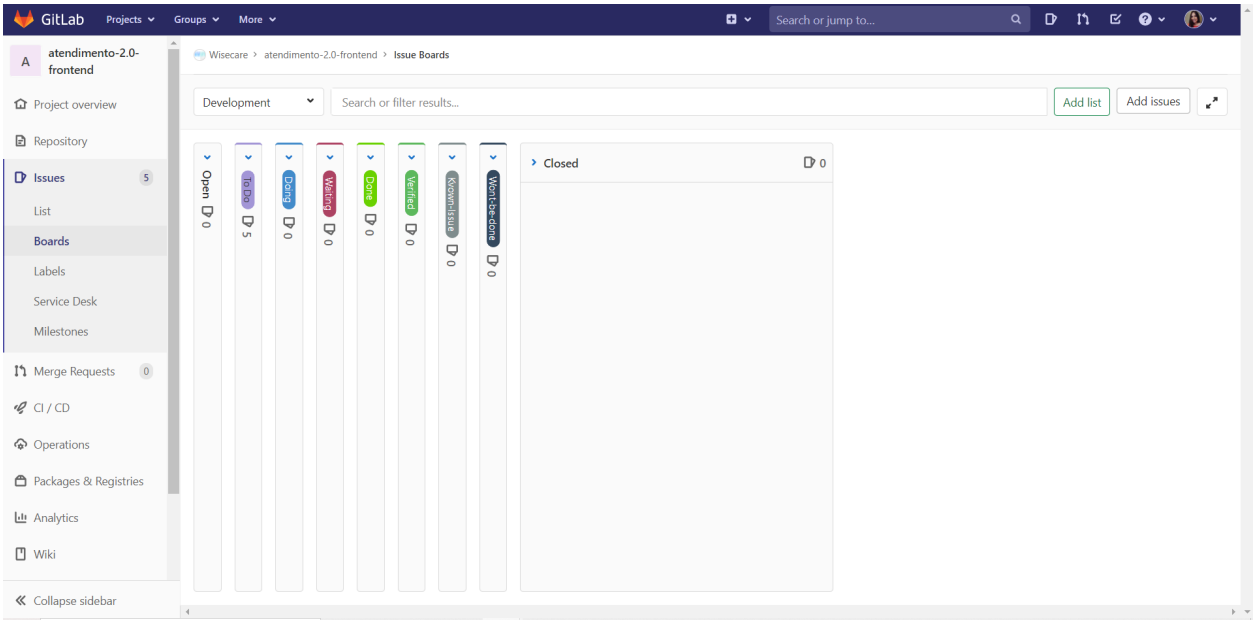

Figura 10: Aplicação da máquina de estados no V4H

Em prol de um maior controle e organização no mapeamento dos testes, foi proposto e utilizado em alguns projetos o apoio da ferramenta Testlink, que promove o controle dos casos de teste e geração de relatórios com os dados de execução. Para a automação dos testes, foi utilizado o *Robot Framework* para testes Web e o *Detox* para testes End-to-End em React Native. Vale salientar que tanto as ferramentas quanto os tipos de testes que serão realizados devem ser avaliados de acordo com cada projeto.

Por fim, seguindo os princípios do *DevOps* e para garantir a qualidade do software desenvolvido, indica-se a utilização de CI/CD, de modo que todos os testes automatizados sejam executados a cada merge na branch. A definição de em qual e quantas branches esse processo deve ocorrer varia de acordo com a necessidade e estrutura arquitetural de cada projeto. A figura 10 apresenta a proposta do time de qualidade que foi executada no projeto PHI.

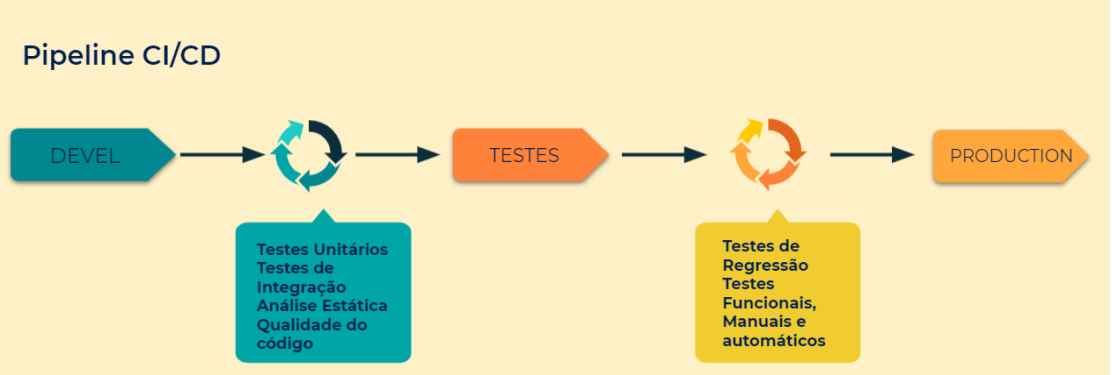

Figura 11: Pipeline CI/CD para o projeto PHI

5. APLICAÇÃO DA PROPOSTA

Como ambiente de estudo para este trabalho de conclusão de curso, foram utilizados cinco projetos do LAVID com focos distintos, sendo eles: V4H - Video for Health; PHI - Paraíba Humana Inteligente; VLibras; Overmediacast e Hope.

A tabela 4 apresenta um resumo de quais pontos propostos foram aplicados nos projetos citados acima.

Tabela 4: Ferramentas utilizadas nos projetos

Projeto	Comunicação	Gestão de atividades	QA	Documentação
V4H	Grupo de *Whatsapp*	Clickup	*GitLab*	*G Suite*
				Docs Clickup
			Testlink	*Figma*
PHI	*Slack*	*Clickup*	*GitLab*	*G Suite*
			Planilhas	*Figma*
Overmediacast	*Slack*	*Clickup*	-	*G Suíte*
				G *Docs*, *Clickup*
		G Suite		*Adobe XD*
VLibras	*Slack*	*Clickup*	-	G *Suite*
				G Docs, Clickup
	Grupo de *Whatsapp*			*Figma*
Hope	Grupo de *Whatsapp*	*Clickup*	*GitLab*	*G Suíte*
			Planilhas	G Docs, *Clickup*

Fonte: A autora

5.1 V4H - Vídeo for Health

O projeto V4H é uma parceria entre LAVID e a Wisecare Tecnologia para desenvolver uma plataforma de videochamadas seguras para o contexto da saúde e demais cenários que possuam conteúdo em vídeo sensível. A plataforma permitirá a realização de atividades de telessaúde como consultas, consultorias e ensino.

A aplicação da proposta deste trabalho foi incluída no projeto V4H somente após um ano de andamento, que anteriormente era concentrado somente na equipe de desenvolvimento. Por isso, o ponto de contato diário permaneceu via whatsapp e foi acrescido uma rotina de reuniões semanais, com o objetivo de trabalhar de forma multidisciplinar e organizada, considerando o acréscimo de dois novos times: design e qualidade. Para consolidar essa integração e atender as necessidades de todos, foi elaborado um processo de trabalho comum, que incluiu a migração da ferramenta *Trello* para o *ClickUp*.

O time de qualidade foi extremamente necessário para a organização de rotinas, planejamento e execução dos testes, que começaram com foco manual e como resultado do crescimento do produto e com o amadurecimento da equipe foi acrescido de testes automatizados. Para este projeto, foi utilizado o Testlink para o controle e execução dos casos de teste, além do *Robot Framework* para a automação de testes de aceitação. Devido a falta de um documento de requisitos, foi tido como base para a escrita dos CT's o protótipo criado pelo time de design do projeto, que utilizou a ferramenta *Figma*[11] para ideação e entrega de assets, tanto pela capacidade organização que a ferramenta possui, quanto pela sua gratuidade.

5.2 PHI - Paraíba Humana Inteligente

O objetivo geral deste projeto foi propor o desenvolvimento de um conjunto de soluções digitais para o governo no contexto da Paraíba, mais especificamente, nas áreas:

[11] Figma, https://www.figma.com/ui-design-tool/.

acessibilidade, gestão cidadã e engajamento e interação digital a partir de serviços que podem ser aplicados inicialmente no contexto da Secretaria da Educação, Ciência e Tecnologia do Governo da Paraíba. A experiência que será retratada sobre a adesão da proposta feita neste trabalho, tem como foco o WP7 - Interação Digital, que propôs uma remodelação do sistema Saber Web e desenvolveu o Saber Móvel, com versões para iOS e Android.

Para o projeto em questão, foi possível incluir todas as ferramentas pensadas para promover uma rotina de trabalho. Todas as atividades foram registradas e acompanhadas com o auxílio do Clickup, que nesse caso específico, devido aos prazos e com o intuito de ter uma maior produtividade, focou-se na estrutura de sprint com duração de uma semana. No quesito comunicação interna dos times, ficou definido uma reunião semanal para abertura da sprint e o Slack foi aderido por todos os membros do WP7 como o único ponto de comunicação ágil utilizado pelo time.

No quesito qualidade, este projeto contou com duas integrantes responsáveis pelas atividades de QA, além do apoio dos desenvolvedores do aplicativo. Para a refatoração do sistema Web do Saber, foram realizados estudos do código com o intuito de entender, propor e estabelecer um processo de CI/CD para ambas as plataformas. O desenvolvimento do aplicativo móvel teve como base *React Native*[12]. Nele, foi possível unir testes manuais com a automação, utilizando o *Detox*[13], que é uma biblioteca destinada à criação de testes funcionais/E2E. Além disso, todo o planejamento e mapeamento de casos de teste foi realizado para as duas plataformas.

Os protótipos fornecidos pelos responsáveis pelo design, juntamente com o documento de requisitos criado, serviram como base para o mapeamento. Toda a criação e repasse de assets foi realizado utilizando o *Figma*.

[12] React Native é uma biblioteca Javascript criada pelo Facebook. É usada para desenvolver aplicativos para os sistemas Android e iOS de forma nativa. Link: https://reactnative.dev/.
[13] Detox, https://github.com/wix/detox.

5.3 VLibras

A suíte VLibras é um conjunto de ferramentas gratuitas e de código aberto que traduz conteúdos digitais em Português para Libras, tornando computadores, celulares e plataformas Web mais acessíveis para as pessoas surdas. O Vlibras é um dos maiores projetos do LAVID, possuindo mais de 12 anos de duração, resultado de uma parceria entre o Governo Federal e a UFPB.

Devido ao longo tempo de existência do VLibras, já havia um nível de gerenciamento. As atividades eram planejadas e registradas no *Trello*, apesar da baixa adesão da equipe e o Slack também já estava incluso. Todavia o ponto principal de comunicação do time até o ano 2019-2020 eram os grupos de *Whatsapp*. Para a continuação do projeto no ano de 2021, foram incluídas algumas das rotinas de trabalho sugeridas anteriormente nesta pesquisa. O *ClickUp* passou a ser a principal ferramenta destinada ao gerenciamento das demandas. Além disso, o G Suíte foi extremamente importante para a organização e registro de documentações e animações. Apesar da dificuldade de excluir o whatsapp da rotina de trabalho da equipe, ele estabeleceu-se como um ponto de contato para casos de urgência e o *Slack* passou a ser protagonista como ferramenta de comunicação.

Sempre com foco em padronizar e promover bases sólidas para o futuro, foi refatorado e incluído no projeto em questão, um *Dev Guide* para os novos bolsistas, explicando tanto as mudanças na rotina de trabalho, quanto métricas de uso para o *GitLab*. Além disso, para facilitar o reuso de software e aumentar a curva de aprendizado dos novos bolsistas, foi incluído como padrão a necessidade de criação de documentação, tanto para os serviços que necessitam de mudanças/atualizações, quanto para os em desenvolvimento.

Considerando a baixa necessidade de testes devido os serviços já estarem bem estabelecidos para este projeto, como também limitações no orçamento, não foi incluído um time de QA. Para o design, foi utilizada na prototipação de software a mesma ferramenta citada nos projetos anteriores.

5.4 Overmediacast

Este projeto é uma parceria entre o LAVID e a empresa Overmedia. O principal objetivo é a melhoria no processo de produção de videobots. Tal contexto envolve desde os desafios computacionais para a integração e geração de interfaces em tempo de execução a criação de projetos de interação a acessibilidade para conteúdos multimídia.

Considerando a natureza do produto a ser trabalhado e a interdisciplinaridade necessária para a integração da equipe, que inclui bolsistas para criação de design de interface, animadores, editores e desenvolvedores, o projeto iniciou o gerenciamento de atividades diretamente no *ClickUp.* Com o objetivo de promover uma continuidade ao trabalho já executado na Overmedia e facilitar a adesão às mudanças de rotina, foi utilizada a suíte Adobe [14] para a criação de um banco de conteúdos destinados à produção audiovisual, e o apoio do *G Suite* para registro e entrega de elementos. Além disso, a feature de documentação do *ClickUp* foi utilizada para concentrar fichas com os dados de cada videobot trabalhado, tentando sempre fornecer uma comunicação sólida e fácil acesso às demandas solicitadas pela empresa ao LAVID.

A necessidade de uma produção ágil fez com que as bases deste projeto fossem as rotinas do Scrum, que consistiram em uma reunião semanal para abertura das Sprint e reuniões diárias com duração de 15 minutos. Toda a comunicação e solicitações entre o LAVID e os bolsistas, e entre a empresa e o LAVID foi concentrada em reuniões e na ferramenta Slack.

Para este projeto não existiu uma equipe específica de garantia de qualidade, devido à natureza específica de produção pôde-se perceber que as atividades básicas de teste poderiam ser realizadas pelos desenvolvedores e pela líder de equipe.

[14] Adobe, https://www.adobe.com/br/.

5.5 Hope

O Núcleo LAVID em parceria com a empresa Fuze.cc, deu início ao projeto Hope, que visa o desenvolvimento de uma rede social focada em ajudar grupos que sofrem exclusão comportamental por serem toxicodependentes, fazendo uma ponte entre esses grupos e casas de apoio, profissionais de saúde, entre outros.

As necessidades de garantia de qualidade, realização de planejamento e execução de testes apareceram ao longo do projeto, por isso, durante o andamento foi substituído o *Trello* pelo *Clickup*, que forneceu uma maior organização e adesão da equipe, porém toda a comunicação permaneceu no whatsapp. Para este projeto em específico, todo o design da interface foi projetado antes da execução do projeto e portanto todos os assets ficaram concentrados no *G Suite.*

Considerando a proposta da empresa, que tinha como um dos seus objetivos principais para essa parceria treinar desenvolvedores full stack e com expertise em testes automatizados, foi necessário somente uma integrante especialista em qualidade para planejar e treinar o time de desenvolvimento, que utilizou para os testes unitários do backend o *framework* Jest[15]. Para os testes automáticos de navegação da plataforma Web foi utilizada a ferramenta *Cucumber*[16], e o *Appium* [17]para esses mesmos testes no mobile (iOS e android). Visando a completude do processo, a garantia de qualidade do software e todos os benefícios que esse método pode fornecer, foi utilizada a plataforma Heroku [18]para realizar a execução do pipeline de CI/CD.

5.6 Validação com os bolsistas

A atividade de validação consistiu em um formulário para os participantes de projetos do LAVID que experimentaram tanto as novas rotinas de trabalho, quanto às dinâmicas

[15] Jest, https://jestjs.io/pt-BR/.
[16] Cucumber, https://cucumber.io/.
[17] Appium, https://appium.io/.
[18] Heroku, https://www.heroku.com/.

anteriores. Dentre os participantes, foram obtidas respostas de pelo menos duas pessoas para cada um dos projetos citados anteriormente. Vale salientar que algumas pessoas que responderam participaram de mais de um dos projetos utilizados para esse estudo. Este questionário ficou disponível do dia 02 de junho de 2021 até o dia 08 de junho de 2021, e teve 17 respostas. Nesta seção, apresenta-se a síntese dos resultados obtidos.

As primeiras perguntas do questionário tinham como objetivo entender o perfil dos participantes, que comprovaram a participação da maioria em outros projetos e também o perfil de desenvolvedor como predominante.

Tabela 5: Respostas sobre o perfil dos participantes

Pergunta	Qtde. de respostas SIM	Qtde. de respostas NÃO
Você já participou anteriormente de outro projeto do LAVID?	11	6

Fonte: A autora

Tabela 6: Respostas sobre o perfil dos participantes

Pergunta	Desenvolvedor	Analista de testes	Produtor audiovisual	Especialista em acessibilidade	Outros
Qual papel você desempenhou no projeto?	9	1	2	1	4

Fonte: A autora

Com o foco de investigar os objetivos propostos neste trabalho, o questionário foi dividido em três seções: Gestão de atividades, dinâmica de comunicação e *quality assurance.* Estas tiveram respectivamente quatro, três e cinco perguntas voltadas a cada um dos temas, com o intuito de avaliar os impactos que as rotinas de trabalho propostas ofereceram para os bolsistas. Os resultados referentes às três seções estão apresentados nas figuras 12, 13 e 14.

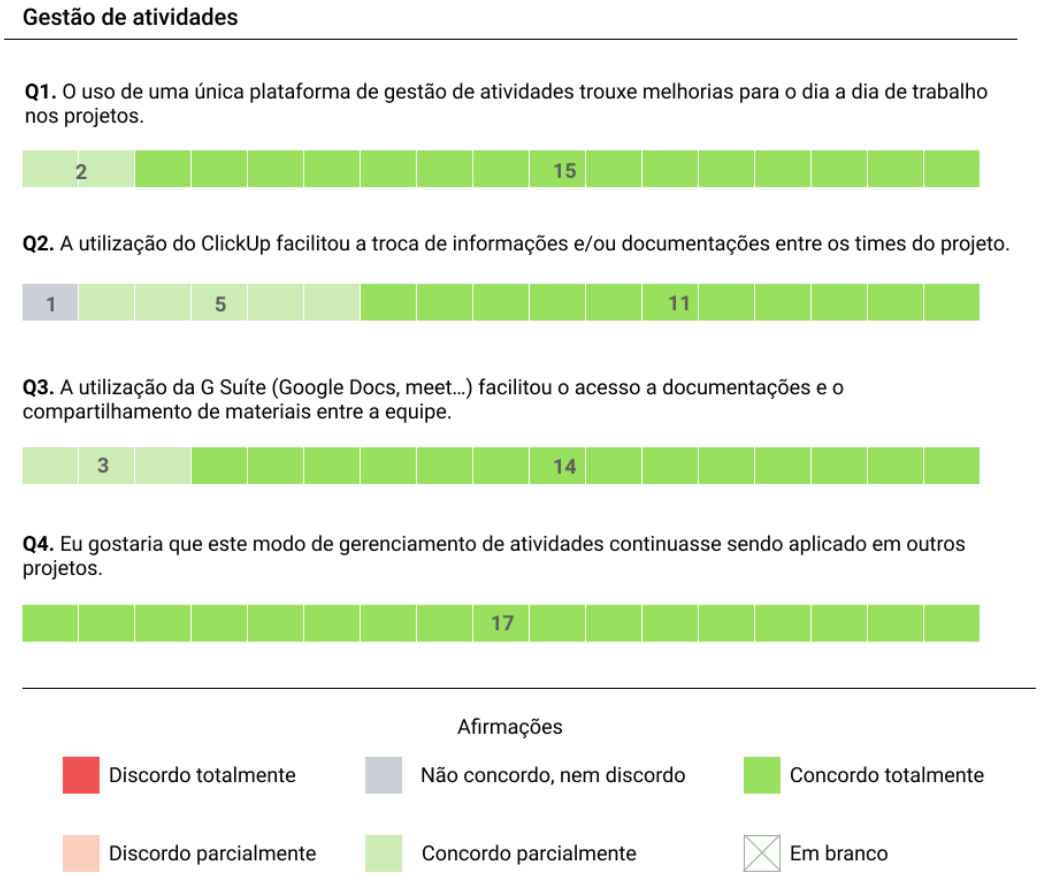

Figura 12: Gestão de atividades com utilização do clickup

Sobre o potencial para replicabilidade, todos os participantes declaram concordar plenamente que o novo modo de gerenciamento deve ser aplicado em outros projetos, podendo trazer melhorias para o trabalho e entregas. De modo geral, todas as perguntas feitas com relação ao gerenciamento de atividades mostraram uma boa aceitação e tiveram como resposta “concordo” (totalmente ou parcialmente).

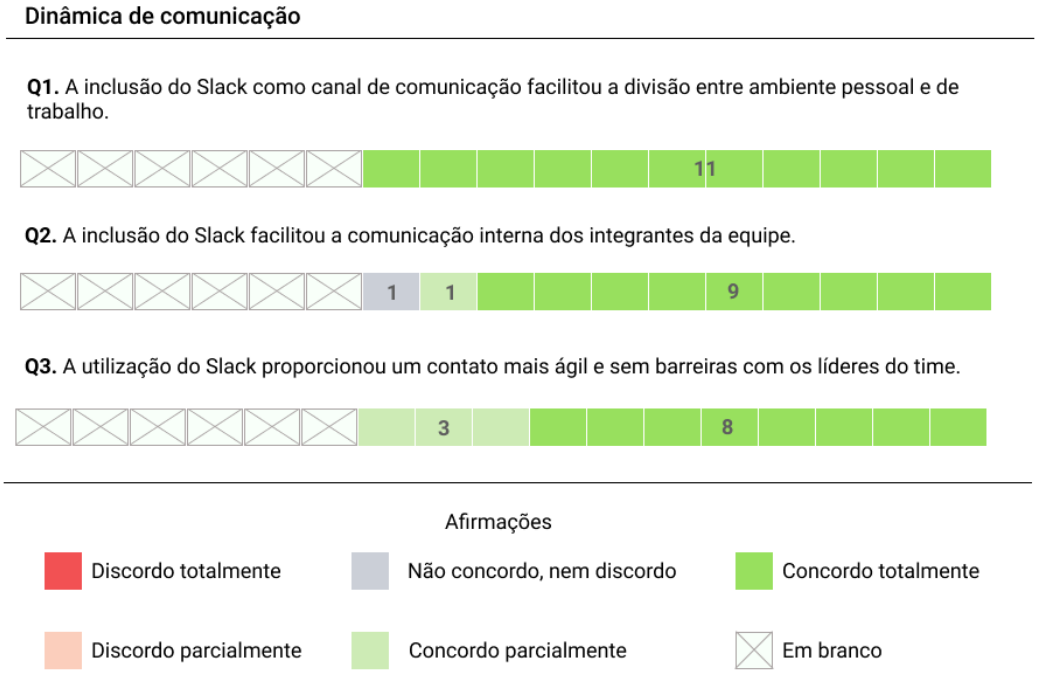

Figura 13: Dinâmica de comunicação entre os bolsistas utilizando o slack

No quesito comunicação interna, seis dos participantes não responderam, já que tratava-se de uma experiência não experimentada no projeto do qual o mesmo participa. Os outros participantes, em sua maioria, concordaram (totalmente ou parcialmente) com o fator do slack auxiliar na divisão de ambientes (trabalho e pessoal) e também com a facilidade da comunicação entre a equipe.

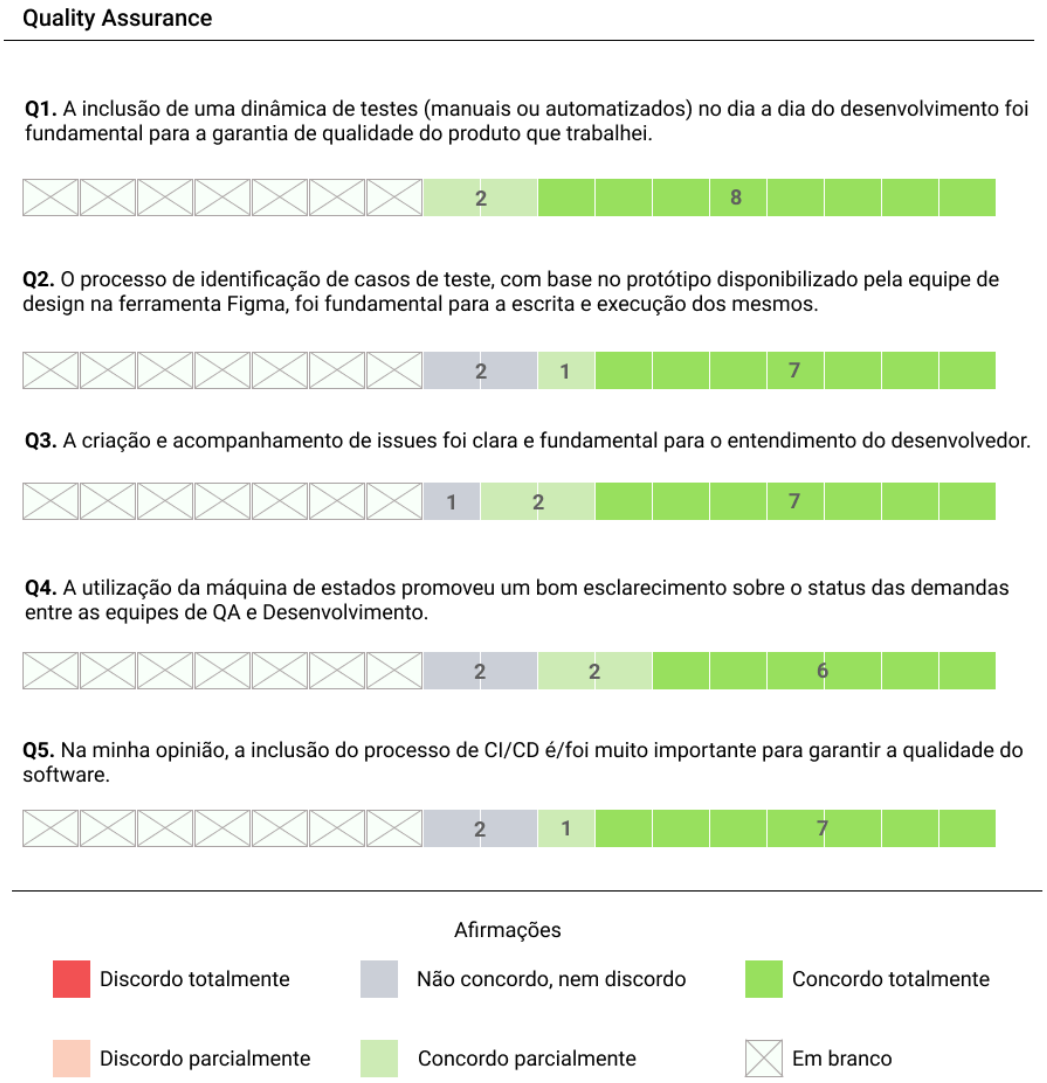

Figura 14: Aplicação de uma rotina de testes e controle de qualidade

Para os questionamentos relacionados à garantia de qualidade e aplicação de testes, sete participantes não responderam, considerando que tratava-se de uma abordagem ou dinâmica não experimentada no projeto do qual o mesmo participa. Os outros participantes, em sua maioria, concordaram com as rotinas de teste e métodos de acompanhamento.

Em seguida foi apresentado a questão: "Você considera que as rotinas e práticas aplicadas nos projetos atuais trouxeram melhorias em comparação a suas experiências anteriores no laboratório?". Esta pergunta tem como objetivo verificar se houve entre os

participantes dos projetos uma boa aceitação para a proposta. Doze participantes responderam a pergunta e essas respostas estão visíveis na abaixo.

Tabela 7: Você considera que as rotinas atuais trouxeram melhorias?

ID Participante	Respostas
P4, P13 e P15	Sim
P5	Considero que as rotinas e práticas aplicadas nos projetos trouxeram diversas melhorias, a exemplo da otimização do tempo de trabalho, de uma comunicação mais efetiva entre a equipe e de mais eficácia ao realizar as demandas solicitadas.
P17	Sim, projetos com equipes especializadas (QA, DEV e DESIGN) e uma clara comunicação entre as partes, facilitaram para um processo mais saudável e correto.
P9	Sim. Dessa maneira foi possível organizar melhor as tarefas com entregas factíveis e seguras.
P16	Sim, pois deram celeridade e melhoraram a qualidade dos produtos desenvolvidos.
P10	Sim, principalmente na organização de tarefas e comunicação com o grupo.
P8	Sim, melhorias substanciais de qualidade e principalmente eficiência.
P14	Não participei de outros projetos, mas achei a rotina muito produtiva.
P11	Sim, tá bem mais organizado.
P6	Sim, bastante.

Fonte: A autora

As respostas apresentadas foram baseadas nas experiências dos participantes em projetos utilizados como caso de uso e por isso refletem melhorias iniciais em aspectos práticos como a otimização do tempo dos bolsistas e a qualidade dos produtos desenvolvidos, até a comunicação entre os integrantes dos projetos.

6. ANÁLISE E DISCUSSÃO

Ao término deste ciclo da pesquisa é preciso analisar os impactos que as rotinas propostas nesta pesquisa promoveram nos ambientes externo e interno da classe de problemas. As rotinas funcionam como esperado? Os problemas foram resolvidos satisfatoriamente? Estes itens serão discutidos ao longo deste capítulo.

Considerando os resultados observados durante o período de aplicação da proposta em cinco projetos do laboratório, como também as respostas ao questionário aplicado durante a etapa de validação, o resultado pode ser considerado como satisfatório, visando que de modo geral, as rotinas trouxeram melhorias nos dois âmbitos propostos, sendo eles, gerenciamento de atividades e garantia de qualidade de software. Vale salientar que devido a natureza e orçamento de alguns dos projetos utilizados para esta pesquisa, não foi possível incluir todas as rotinas propostas nos cinco projetos.

6.1 Gerenciamento de atividades

Cumprindo o objetivo inicial, foi incluída nos cinco projetos participantes desta pesquisa a utilização da ferramenta *ClickUp* para o gerenciamento de atividades. Em todos eles, foi facilmente identificado um maior controle das demandas, maior agilidade no trabalho das equipes, além de uma troca de materiais mais fluida entre todos os membros. Esses resultados obtidos a partir da experiência prática, também foram citados pelos participantes do questionário de validação. Além disso, foi relatado o desejo dos participantes de terem este mesmo modelo gerenciamento de atividades replicado em outros projetos.

> "Considero que as rotinas e práticas aplicadas nos projetos trouxeram diversas melhorias, a exemplo da otimização do tempo de trabalho, de uma comunicação mais efetiva entre a equipe e de mais eficácia ao realizar as demandas solicitadas." (P5, sobre a pergunta "Você considera que as rotinas atuais trouxeram melhorias?")

Um problema recorrente dentro de empresas é a má comunicação que ocorre entre equipes de design e desenvolvimento. Por diversas vezes essas duas equipes não estão trabalhando em sincronia, fato que pode acarretar grandes problemas para o produto final como, questões de má usabilidade e interação do indivíduo. Segundo Nadia Udalova, a

alocação da equipe de design na organização dos processos e alinhamento e comunicação entre as equipes, são pilares para um bom resultado [14]. Considerando esse fator e a interdisciplinaridade presente na maioria dos projetos do LAVID, o *Slack* foi incluído e teve grande destaque na organização, sendo um facilitador para a comunicação diária com todos os integrantes do projeto, promovendo um ambiente mais claro e dividido de acordo com cada time ou produto a ser entregue, além de ter sido fundamental para estabelecer um ambiente próprio para as atividades de trabalho. Ademais, no projeto da Overmedia, que necessita de uma integração contínua entre os projetistas de interface, animadores e desenvolvedores, foram incluídas algumas práticas que tiveram resultados muito bons quanto falamos da facilitação das escolhas técnicas e alinhamento das expectativas da equipe de audiovisual com as limitações do desenvolvimento.

Para dois dos cinco projetos utilizados (V4H e Hope), não foi possível aplicar essa ferramenta de comunicação. Esse fato se deu devido o projeto já ter sido previamente iniciado e os integrantes e coordenadores já estarem confortáveis com a utilização da rede social *WhatsApp* como veículo de comunicação diária.

> "A respeito da plataforma de comunicação slack, eu sou um grande adepto da ideia de separar profissional e particular, porém ainda sinto que exista uma grande barreira na utilização de ferramentas deste tipo já que vivemos no mundo onde o WhatsApp é a principal ferramenta de troca de mensagens e com baixo tempo de resposta, lutar contra isso é difícil [...]" (P17, sobre a pergunta "Você gostaria de dizer mais alguma coisa?")

Ainda no quesito comunicação, a quantidade de reuniões variou de acordo com a necessidade e os prazos determinados para os entregáveis. Todavia, todos os projetos tiveram no mínimo uma reunião semanal. Devido à natureza do produto fornecido e a rapidez necessária para as entregas no projeto da Overmedia, foram incluídas reuniões diárias de quinze minutos, com o intuito de facilitar o acompanhamento e a distribuição de atividades.

Um dos participantes do questionário comentou sobre a frequência das reuniões, além de apresentar uma dificuldade na utilização do *ClickUp* para os casos onde há arquivamento de atividades.

> "Visando melhorar a rotina de trabalho, sugiro duas práticas, sendo: a primeira, a contínua visualização de atividades já atualizadas como "DONE" no ClickUp, visto que, em certas demandas, é necessário o acesso de informações que costam em tarefas anteriores, de modo a facilitar a resolução dos novos objetivos; e a segunda, a

> realização de reuniões virtuais dia sim, dia não, a fim de possibilitar um maior espaço de tempo para a resolução dos objetivos semanais." (P05, sobre a pergunta "Você gostaria de dizer mais alguma coisa?")

6.2 Quality Assurance

A inclusão de uma rotina baseada em testes foi muito bem recebida pelo laboratório. Esta, pôde ser aplicada em três (PHI, V4H e Hope) dos cinco projetos utilizados nesta pesquisa. Devido a quantidade de tempo do projeto VLibras no mercado e por consequência a estabilidade do mesmo, não foi necessário a aplicação dos aspectos de qualidade. No caso da Overmedia, os recursos financeiros disponíveis e a dificuldade de configuração de um ambiente específico para homologação, devido a grande dependência da infraestrutura da empresa, impossibilitaram a execução do processo de teste por completo, sendo realizados somente testes caixa preta no ambiente de produção, sem um prévio planejamento. Para os três projetos que utilizaram do QA, foram feitos tanto testes manuais, quanto automatizados, com tipos e ferramentas utilizadas que variaram de acordo com cada projeto.

No PHI foram aplicados todos os passos propostos pelo núcleo de qualidade, desde a criação de um plano de testes, passando pelo mapeamento e execução, até a inclusão de um pipeline CI/CD para a execução automatizada dos testes desenvolvidos. Essa aplicação forneceu uma cobertura completa do aplicativo móvel e trouxe resultados muito importantes para a qualidade do produto desenvolvido.

Para o projeto V4H, devido a dimensão do software em desenvolvimento e a inclusão do processo de qualidade ter iniciado após o início do projeto, não foi possível incluir CI/CD até a data final desta pesquisa. Todavia, todos os outros passos apresentados na seção 5.2.2 deste trabalho foram cumpridos com sucesso e apresentaram resultados bastante satisfatórios. Um participante do questionário que é também integrante do V4H relatou a importância da equipe de QA para projetos em geral.

> "[...] A equipe de QA é na teoria e ainda mais na prática muito importante para um bom desenvolvimento e qualidade de produto, testes automatizados e CI/CD são uma peça essencial para os projetos se tornarem cada vez mais eficientes." (P17, sobre a pergunta "Você gostaria de dizer mais alguma coisa?")

O projeto Hope teve incluído em sua rotina a prática de escrever testes automatizados no decorrer do desenvolvimento, tanto unitários quanto de navegação. Para este caso em questão, foi incluída uma rotina de integração contínua com o apoio da plataforma Heroku. A execução dos testes manuais e cadastro/controle de issues foi feita como proposto neste trabalho.

De modo geral, em todos os projetos foi possível assegurar uma melhor qualidade do artefato entregue, devido a grande cobertura que os testes proporcionam. Nenhum produto desenvolvido com o apoio de um time de qualidade foi liberado para a produção com uma taxa de menos que 95% de aceitação nos testes realizados. Esses fatos proporcionaram aos times de desenvolvimento a diminuição de retrabalho e também uma diminuição significativa de feedbacks negativos vindos dos clientes finais.

7. CONCLUSÕES E TRABALHOS FUTUROS

A proposta desenvolvida neste trabalho consiste em estruturar e padronizar um ambiente controlado para o cadastro e acompanhamento de atividades, de modo que seja proporcionado às equipes dos projetos o acesso a demandas claras e diretas. Também prevê a padronização de uma ferramenta de comunicação, para tornar mais fácil, rápido e impessoal. Por fim, sugere a inclusão de práticas necessárias para garantir uma maior qualidade de software.

Considerando a aplicação prática da proposta no estudo de caso, pode-se identificar que houveram ganhos na execução dos projetos. Dentre estes ganhos, podem ser citados a maior agilidade na execução das demandas, melhorias na comunicação e integração de equipes interdisciplinares, que passaram a ter um contato mais fluido e independente. Além disso, o processo de qualidade aprimorou o desenvolvimento de software, considerando o melhor desempenho e a maior segurança das equipes ao entregar os artefatos desenvolvidos após uma rotina de testes e inspeções.

Tais recomendações sistematizadas no capítulo 5 e aplicadas no capítulo 6 não são restritas ao contexto do LAVID-UFPB. A aplicação da proposta nos múltiplos projetos é uma evidência inicial de que as recomendações podem ser aplicadas tanto em projetos vinculados à pesquisa e extensão (Vlibras, PHI) ou mesmo ao atendimento de necessidades de produção do mercado (Overmedia, V4H e Hope). Em contextos de produção, semelhantes ao do estudo de caso, acreditamos que os benefícios também podem ser aplicados.

Para expandir e compreender melhor a validade das recomendações, será necessário a aplicação de tais itens em outros laboratórios, bem como, em novos contextos de desenvolvimento de software, como por exemplo: startups, pequenas empresas e ONG 'S. Além disso, a verificação pode estar atrelada a métricas de gestão e negócios como o tempo de produção, retorno financeiro ou mesmo estudos qualitativos para compreender o nível de satisfação da equipe. Tais impactos ficam a cargo de estudos futuros na continuação desta pesquisa.

REFERÊNCIAS

[1] FOWLER, Martin. The new methodology. **Wuhan University Journal of Natural Sciences**, v. 6, n. 1-2, p. 12-24, 2001.

[2] DOS SANTOS SOARES, Michel. Comparação entre metodologias Ágeis e tradicionais para o desenvolvimento de software. **INFOCOMP Journal of Computer Science**, v. 3, n. 2, p. 8-13, 2004.

[3] DIAS, Ricardo. O Modelo em Cascata. Medium, 2019. Disponível em:<https://medium.com/contexto-delimitado/o-modelo-em-cascata-f2418addaf36>. Acesso em: 17 jul. 2020.

[4] SOMMERVILLE, Ian. **Engenharia de Software**, 9ª Edição. Pearson. São Paulo, Brasil, 2011.

[5] JACOBSON, Ivar et al. **The Essentials of Modern Software Engineering**. ACM, New York, v. 54, 2019.

[6] ROCHA, Carla. RUP (Rational Unified Process). GitHub, 2017. Disponível em:<https://github.com/fga-eps-mds/A-Disciplina-MDS-EPS/wiki/RUP-(Rational-Unified-Process)>. Acesso em: 11 jun. 2020.

[7] BECK, Kent. et. al. Manifesto for Agile Software Development. Agile Manifesto ORG, 2001. Disponível em:<http://agilemanifesto.org/>. Acesso em: 11 jun. 2020.

[8] DRUMOND, Claire. Scrum, saiba como usar o scrum da melhor forma. Atlassian Agile Coach. Disponível em:<https://www.atlassian.com/br/agile/scrum>. Acesso em: 11 jun. 2020.

[9] KERN, Matthew. Agile is Dead. LinkedIn. 2016. Disponível em: <https://www.linkedin.com/pulse/agile-dead-matthew-kern/>. Acesso em: 16 maio 2020.

[10] WILSENACH, Rouan. DevOpsCulture. 2015. Disponível em: <http://martinfowler. com/bliki/DevOpsCulture>. Acesso em: 17 jun. 2020.

[11] PÍCOLO, Mariana. Integração Contínua Deploy Contínuo. GitHub, 2017. Disponível em:<https://github.com/fga-eps-mds/A-Disciplina-MDS-EPS/wiki/Integra%C3%A7%C3%A3o-Cont%C3%ADnua---Deploy-Cont%C3%ADnuo>. Acesso em: 3 jun. 2020.

[12] FOWLER, Martin; FOEMMEL, Matthew. Continuous integration. Thought-Works) http://www. thoughtworks. com/Continuous Integration. pdf, v. 122, n. 14, p. 1-7, 2006.

[13] FOWLER, Martin. Continuous delivery. martinfowler.com, p. 17, 2013.

[14] UDALOVA, Nadia. How to get UX and DEV teams in sync?. Ux Collective, 2018. Disponível em: <https://uxdesign.cc/want-ux-and-dev-building-killer-features-here-is-how-to-infuse-ux-into-dev-part-1-c5f3d651b9c1>. Acesso em: 4 jun. 2020.

[15] ARMITAGE, John. Are agile methods good for design?. interactions, v. 11, n. 1, p. 14-23, 2004.

[16] PREECE, Jenny et al. Human-computer interaction. Addison-Wesley Longman Ltd., 1994.

[17] AMAZON. AWS DevOps - O que é DevOps? - Amazon Web Services. Disponível em: <https://aws.amazon.com/pt/devops/what-is-devops/>. Acesso em: 30 jun. 2021.

[18] DRESCH, Aline; LACERDA, Daniel Pacheco; JÚNIOR, José Antonio Valle Antunes. Design science research: método de pesquisa para avanço da ciência e tecnologia. Bookman Editora, 2015.