UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS DE RUSSAS

CURSO DE ENGENHARIA DE SOFTWARE

ANNA VICTÓRIA FERREIRA DA SILVA

ESTIMATIVA DE TESTE DE SOFTWARE COM O PROCESSO QESTIMATION: UM ESTUDO DE CASO NO DESENVOLVIMENTO DA FERRAMENTA USINN MODELER

RUSSAS

2022

ANNA VICTÓRIA FERREIRA DA SILVA

ESTIMATIVA DE TESTE DE SOFTWARE COM O PROCESSO QESTIMATION: UM ESTUDO DE CASO NO DESENVOLVIMENTO DE UMA FERRAMENTA DE MODELAGEM

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

Orientador(a):
Profa. Dra. Anna Beatriz dos Santos Marques

+

RUSSAS

2022

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

S578e Silva, Anna Victória Ferreira da.
Estimativa de teste de software com o processo qEstimation: um estudo de caso no desenvolvimento da ferramenta USINN Modeler / Anna Victória Ferreira da Silva. – 2022.
79 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Russas, Curso de Engenharia de Software, Russas, 2022.
Orientação: Prof. Anna Beatriz dos Santos Marques.

1. Estimativa de esforço de teste. 2. Teste de Software. 3. USINN Modeler. I. Título.

CDD 005.1

ANNA VICTÓRIA FERREIRA DA SILVA

ESTIMATIVA DE TESTE DE SOFTWARE COM O PROCESSO QESTIMATION: UM ESTUDO DE CASO NO DESENVOLVIMENTO DE UMA FERRAMENTA DE MODELAGEM

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

Aprovada em: 24 de Junho de 2022

BANCA EXAMINADORA

Profa. Dra. Anna Beatriz dos Santos Marques (Orientadora)

Universidade Federal do Ceará (UFC)

Profa. Dra. Adriana Lopes Damian

Instituto Eldorado

Prof. Ms. José Osvaldo Mesquita Chaves

Universidade Federal do Ceará (UFC)

À Deus, por ser meu alicerce. E a minha mãe, que nunca mediu esforços, sempre me apoiou e incentivou nesta caminhada.

AGRADECIMENTOS

Inicialmente, gostaria de agradecer a Deus por ter me sustentado até aqui, com sua presença e seu poder. Obrigada Senhor por me livrar de todo mal, por me dar discernimento, por ser a calmaria em meio às aflições, a Ele toda honra e glória.

À minha mãe Ivone, meu maior exemplo e inspiração. Sem dúvidas a pessoa que mais esteve comigo em todos os momentos, essa vitória é nossa, e a minha maior satisfação é poder compartilhar ela com você. Nenhuma palavra vai definir toda a gratidão que tenho a ti, por tudo o que você é pra mim e todo esforço que fez para permitir que eu pudesse seguir meus sonhos. Você é a pessoa mais importante da minha vida, obrigada por ser minha paz em todos os momentos, Te amo!

À minha irmã Lusiane que sempre se fez presente nessa jornada, foi meu maior apoio e encorajamento no início. Obrigada por sonhar junto comigo e me ajudar a dar os passos mais importantes da minha vida, estamos juntas para tudo o que vier.

Ao meu irmão Lucas que também foi meu apoio e com seu bom coração sempre esteve disposto a me ajudar. Você é especial e também faz parte dessa vitória.

Ao meu cunhado Kleber que sempre foi um dos meus incentivadores, me ajudou em todos os momentos que precisei, e sempre me aplaudiu de pé. Obrigada por tudo, especialmente aos momentos vividos.

Ao meu pai Adilson e a toda minha família que sempre me incentivaram, agradeço por toda a torcida e boas vibrações, vocês são essenciais.

À minha madrinha Vanessa que sempre esteve presente, me apoiou e é uma das pessoas mais importantes pra mim. Obrigada por ser luz na minha vida.

Aos meus avós, Francisco, Valentim (in memoriam), Maria das Graças e Josefa por serem meus anjinhos, de coração honesto e puro, obrigada por todas as orações e incentivos durante a caminhada.

À minha amiga Mavi por estar presente durante toda essa jornada, por todos os perrengues, com toda certeza, seu apoio e amizade me fortaleceu muito até aqui, obrigada por fazer parte da minha vida, e estar presente nos piores e melhores momentos dessa jornada acadêmica.

Ao meu amigo Moab que sempre esteve junto comigo, enfrentando todos os obstáculos dessa trajetória, e que se tornou alguém muito importante pra mim, Obrigada por tudo amigo, você é um presente que a UFC me deu.

Aos meus amigos, Isaac, Hismael, Ívina, Matheus, Milleny, Junior, Rafa, Rodrigo e Jonathan por tornar meus dias mais alegres, pela parceria, e por tudo o que vivemos. Vocês foram essenciais, e são especiais pra mim.

Aos amigos Nidson, Israely e Vinny que são parceiros desde o início da faculdade, sou grata por tudo o que vivemos e pela parceria durante a faculdade.

As amigas Gleicinha, Vitoria, Iasmim por tornarem os dias em Russas mais leves, pelas gargalhadas, pelos momentos de conselhos e pelo ombro amigo, quero a amizade de vocês pra sempre.

A todos os meus amigos e primos do Piauí que mesmo longe, sempre vibraram, e estavam na torcida. Vocês também fazem parte dessa conquista.

À minha querida orientadora Dra. Anna Beatriz, que é inspiração, tornou todo este trabalho mais leve, obrigada por todos os ensinamentos e por acreditar em mim. Você é uma das pessoas que tornou essa jornada melhor.

A toda equipe do projeto Usinn por aceitar participar deste estudo, e dispor do seu tempo e conhecimento para contribuir, sem vocês não teria sido possível.

A todos os professores e servidores da UFC que diretamente ou indiretamente contribuíram pra minha formação. Vocês são peças fundamentais.

Agradeço por todos os momentos vividos, todo o conhecimento adquirido, todas as lições aprendidas, por toda a minha evolução, nunca imaginei viver experiências tão incríveis. Como disse Martin Luther King, “Talvez não tenha conseguido fazer o melhor, mas lutei para que o melhor fosse feito. Não sou o que deveria ser, mas Graças a Deus, não sou o que era antes”.

Obrigada por tudo o que me proporcionou Deus!

“Os que confiam no Senhor são como o monte de Sião, que não se abala, firme para sempre. Como em redor de Jerusalém estão os montes, assim o Senhor, em derredor do seu povo, desde agora e para sempre.”

(Salmo 125: 01-03)

RESUMO

Aplicar métodos para estimular a qualidade de software é fundamental durante o ciclo de vida do software, pois ela se trata de um conjunto de atributos que devem atender às necessidades do usuário. Para executar um programa utilizando algumas entradas em particular e verificar se seu comportamento está de acordo com o esperado, são utilizadas técnicas de teste de software, que aplicadas a um projeto de software envolvem: estudo, modelagem, documentação apropriada e uma parcela significativa de tempo para a equipe de desenvolvimento, que na maioria das vezes possui prazo limitado. Assim, estimar o esforço destas atividades torna-se crucial para o gerenciamento efetivo do projeto. Apesar de muitas pesquisas sobre métricas de teste de software terem sido realizadas, a maioria das propostas não apresenta estimativa de esforço para testes. Este trabalho apresenta o uso de uma abordagem de estimativas de esforço de teste existente: o *qEstimation*, que foi adotado no desenvolvimento de uma ferramenta de modelagem, a *USINN Modeler*. Um estudo de caso foi conduzido com o objetivo de explorar a estimativa de esforço de teste. A estimativa gerada ultrapassou o esforço real gasto, logo na tentativa de tornar essa estimativa mais próxima do real, é necessário aplicar a abordagem novamente melhorando os parâmetros utilizados a partir de ciclos de testes anteriores. O processo de planejamento proposto pela abordagem possibilitou ao projeto Usinn uma equipe de testadores capacitadas, projeto de casos de testes, dados de geração de estimativa e o primeiro ciclo de testes de regressão realizado.

Palavras-chave: Estimativa de esforço de teste. Teste de Software. USINN Modeler.

ABSTRACT

The application of methods to stimulate software quality is critical during the software life cycle because it is a set of attributes that should satisfy the user's needs. To run a program using some particular inputs and verify that its behavior is as expected, software testing techniques are used, which apply to a software project: study, modeling, proper documentation, and a significant amount of time for the development team, which most of the time has limited time. Thus, estimating the effort of these activities becomes crucial for effective project management. Although much research has been done on software testing metrics, most proposals do not present effort estimates for testing. This paper presents the use of an existing test effort estimation approach: *qEstimation*, which was adopted in the development of a modeling tool, *USINN Modeler*. A case study was conducted to explore test effort estimation. The estimate generated exceeded the actual effort expended, so in an attempt to bring this estimate closer to the actual effort, it is necessary to reapply the approach, improving the parameters used from previous test cycles. The planning process proposed by the approach allowed the project Usinn a team of qualified testers, test case design, data estimate generation, and the first cycle of regression testing performed.

Keywords: Test effort estimation. Software Testing. USINN Modeler.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

API	Application Programming Interface
CT	Caso de Teste
GREat	Fábrica de Testes do Grupo de Redes, Engenharia de Software e Sistemas
TCPA	Test Case Point Analysis
UCP	Use Case Points
UI	User Interface
USINN	*USability-oriented INteraction and Navigation Model*

SUMÁRIO

1 INTRODUÇÃO

A Qualidade de Software está presente durante o ciclo de vida do software. Um dos métodos mais utilizados para garantir a qualidade é a Verificação e Validação, em que a equipe de desenvolvimento executa atividades baseadas neste método para minimizar custos, riscos e estimular a qualidade durante o desenvolvimento (EGE, 2017).

A Verificação está relacionada ao conjunto de tarefas que garantem que o software funcione corretamente em determinada função. A Validação se refere ao conjunto de tarefas que certificam a criação do software e o seu rastreamento de acordo com os requisitos. A definição de Verificação e Validação engloba muitas atividades de garantia da qualidade de software, tais como, revisão técnica, monitoramento de desempenho, estudo da viabilidade, análise de algoritmo e diferentes técnicas de teste (PRESSMAN, 2006).

Andrade (2015) afirma que "Qualidade de Software pode ser definida como um conjunto de atributos de software que devem ser satisfeitos de modo que o software atenda às necessidades do usuário (seja ele um usuário final, um desenvolvedor ou uma organização), onde a determinação dos atributos relevantes para cada software varia em função do domínio da aplicação, das tecnologias utilizadas, das características específicas do projeto e das necessidades do usuário e da organização".

A qualidade também é consequência da visão de quem avalia, podendo ter necessidades diferentes, como é o caso do usuário, que explora o software sem conhecer seus aspectos internos, considerando apenas aspectos como facilidade de uso, confiança nos resultados, performance e preço. Já a equipe de desenvolvimento, além de explorar aspectos internos, também avalia a concordância com os requisitos (ANDRADE, 2015).

Os produtos de software têm muitas partes envolvidas, como o próprio time de desenvolvimento, os que adquirem e usam, diretamente ou indiretamente. Para garantir uma boa experiência às partes interessadas, a especificação e avaliação extensiva da qualidade de software é um fator fundamental. É relevante que as condições relacionadas à qualidade sejam medidas e avaliadas, principalmente levando em conta o impacto que o sistema tem sobre suas partes interessadas (ISO/IEC-25010, 2011).

As técnicas de teste de software aplicadas em um projeto de software envolvem estudo, modelagem e documentação apropriada para a geração de casos de teste. Delamaro *et*

al. (2007) define teste de software como “uma atividade dinâmica, seu intuito é executar o programa ou modelo utilizando algumas entradas em particular e verificar se seu comportamento está de acordo com o esperado. Caso a execução apresente resultados não especificados, dizemos que um erro ou defeito foi identificado. Além disso, os dados de tal execução podem servir como fonte de informação para a localização e a correção de tais defeitos.”

Adotar técnicas de testes de software envolve uma parcela significativa de tempo para a equipe de desenvolvimento, que na maioria das vezes já tem um prazo limitado. Assim, estimar o esforço destas atividades torna-se crucial para o gerenciamento efetivo do projeto. Apesar de muitas pesquisas sobre métricas de teste de software terem sido realizadas, a maioria das propostas não apresenta estimativa de esforço para teste. Este tipo de estimativa é extremamente importante, pois reflete na qualidade da equipe de teste, como também nos casos de teste, além de ser necessária para derivar o cronograma e calcular o custo do projeto (KUSHWAHA; MISRA, 2008).

De acordo com a definição da norma ISO/IEC 25010 (2011 apud Hanolu e Tarhan, 2019, p. 689) “testabilidade é o grau de eficácia e eficiência do esforço de teste despendido para verificar se um produto desempenha sua função. De acordo com essa definição, um dos fatores que afetam a testabilidade e a qualidade do software é o esforço de teste.”

Mesmo que o fato da estimativa de esforço para teste de software seja uma tarefa importante durante o desenvolvimento de software, existe um número reduzido de trabalhos científicos para resolver esse problema (BLUEMKE; MALANOWSKA, 2021).

Esta pesquisa investigou o uso de uma abordagem para estimativas de teste que foi adotada no desenvolvimento de uma ferramenta de modelagem, a *USINN Modeler*. Um estudo de caso foi conduzido com o objetivo de explorar a estimativa de esforço de teste aplicando o processo *qEstimation*.

A ferramenta *USINN Modeler* nasce com a finalidade de tornar mais prático e sintaticamente correto a criação e edição de diagramas USINN (COSTA; MARQUES, 2019). E a notação USINN surge com o intuito de facilitar a representação de mecanismos de usabilidade em atividades de modelagem. Com o USINN é possível ampliar elementos de navegação e interação que estejam correlacionados com a usabilidade (MARQUES; BARBOSA; CONTE, 2017).

O restante deste documento está organizado da seguinte maneira: na Seção 2 são apresentados os objetivos de pesquisa. A Seção 3 descreve a fundamentação teórica para o entendimento e realização deste trabalho. A Seção 4 apresenta os trabalhos relacionados. Na Seção 5 são apresentados os procedimentos metodológicos adotados para a execução deste trabalho. A Seção 6 fala sobre o estudo de caso. A Seção 7 apresenta a análise dos resultados. Por fim, a Seção 8 aborda conclusões e trabalhos futuros.

2 OBJETIVOS

Nesta seção, são descritos o objetivo geral e os objetivos específicos desta pesquisa.

2.1 Objetivo geral

O objetivo geral deste trabalho é investigar a utilidade da abordagem de testes de software *qEstimation* no desenvolvimento de uma ferramenta de modelagem denominada *USINN Modeler*, com foco na estimativa de esforço para testes.

2.1 Objetivos específicos

- Relatar o planejamento e a execução do processo de testes na ferramenta *USINN Modeler*.
- Aplicar o processo *qEstimation*, gerando uma estimativa de esforço para teste de software.
- Comparar a estimativa de esforço realizada com o esforço real gasto.
- Investigar a percepção do time sobre a utilidade da abordagem de teste de software adotada no projeto.

3 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta conceitos importantes para fundamento teórico deste trabalho, visando auxiliar no entendimento da pesquisa. No tópico 3.1 são apresentados métodos de Teste de Software, já no tópico 3.2 são apresentados técnicas de Casos de Teste, no tópico 3.3 é citado as principais ideias sobre Estimativa de Esforço de Teste, no tópico 3.4 é apresentado a abordagem de estimativa de teste escolhida, por fim, o tópico 3.5 fala sobre o software utilizado, *USINN Modeler*.

3.1 Teste de Software

Bartié (2002) fala sobre não mentalizar que teste de software se resume a "evidenciar que algo não funciona", pois já é um progresso no entendimento sobre o processo de qualidade, sendo teste definido como um processo planejado e sistemático que tem como finalidade única a identificação de erros.

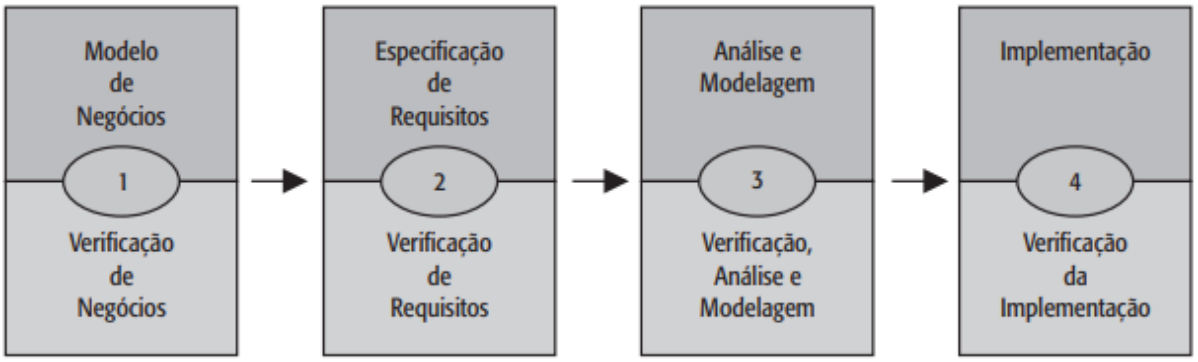
Myers *et al.* (2004) define teste como um processo de execução de um programa que tem a intenção de encontrar erros. Compreender o real sentido da definição de teste, provoca uma diferença expressiva no sucesso dos seus esforços. A partir da definição de teste citada, é aceitável o pensamento de que é viável encontrar todos os erros de um software durante a execução de testes. Myers chama de "The Economics of Testing" a prática em que menciona que é impraticável, ou até mesmo impossível encontrar todos os erros de um programa, e que este pensamento pode ter consequências para a economia de testes.

Para garantir que os defeitos não se mantenham no software e que ele esteja de acordo com a documentação de especificação de requisitos. Para isso, existem atividades como "Validação, Verificação e Teste", que podem ser dinâmicas ou estáticas. Sendo as dinâmicas que se embasam na execução de um modelo ou software, já as estáticas não necessitam da existência de um executável para serem realizadas. Como o teste de software é definido com uma atividade dinâmica que verifica durante uma execução o comportamento do software, os resultados colhidos servem de base para correção de defeitos (DELAMARO *et al.*, 2007).

Sommerville (2011) afirma que a principal função do teste é evidenciar que um software realmente faz o que lhe foi proposto, como também desvendar os defeitos que existem, antes do software ser liberado para utilização. Esse processo pode ser dividido em dois escopos, um primeiro é que a partir dos testes é validado que o programa atende seus requisitos. A segunda situação é quando há comportamentos inesperados no software, o teste de defeito é focado na extinção de qualquer comportamento indesejável do sistema, como por exemplo, interações desnecessárias, bugs, processamentos incorretos, entre outros.

Um bom processo de qualidade de software deve potencializar as duas formas de testes, chamadas de verificação e validação, de modo que os esforços sejam reduzidos e os resultados positivos. Não envolver o processamento de softwares, é a principal característica dos testes de verificação, pois essas atividades são antecedentes a criação do software, que podem entendidos como um processo de acompanhamento de atividades e avaliação de documentos gerados durante todas as fases do processo de engenharia de software (BARTIÉ, 2002). A Figura 1 ilustra as etapas dos testes de verificação, que são aplicados em quatro estágios. Em cada etapa são realizados testes de verificação para garantir que todas as definições estabelecidas foram entendidas e aceitas pelas partes interessadas que formam o desenvolvimento do software.

Figura 1 - Etapas dos testes de verificação

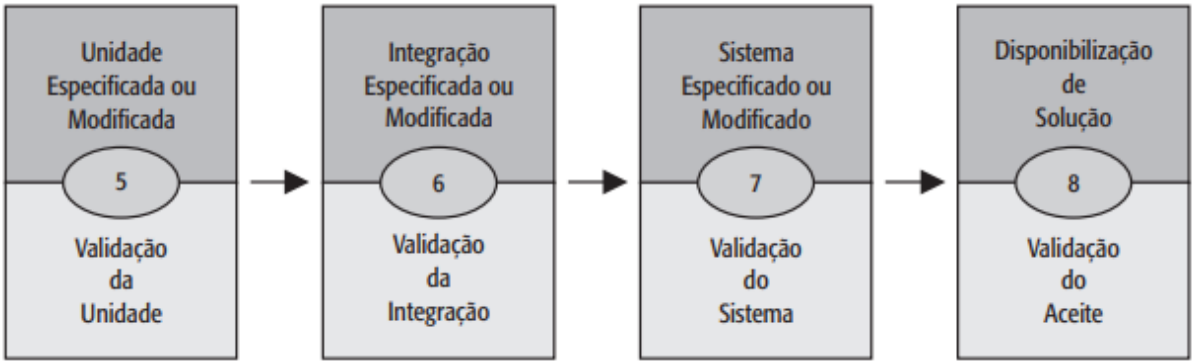


Fonte: (BARTIÉ, 2002)

Por sua vez, os testes de validação são fundamentados no comportamento do software, através das condições simuladas, para que sejam comparados com as especificações desenvolvidas. O objetivo destes testes é avaliar a semelhança do software com o documento de requisitos, eles podem ser entendidos como um processo de avaliação de softwares, cuja

avaliação pode seguir em componentes isolados, módulos, ou até mesmo no sistema por inteiro (BARTIÉ, 2002). A Figura 2 apresenta as etapas do teste de validação, que são aplicadas respeitando os estágios de desenvolvimento. Cada teste aplicado em sua determinada categoria do processo, traz resultados do processamento do software, que é o principal ponto de validação.

Figura 2 - Etapas dos testes de validação



Fonte: (BARTIÉ, 2002)

3.2 Caso de Teste

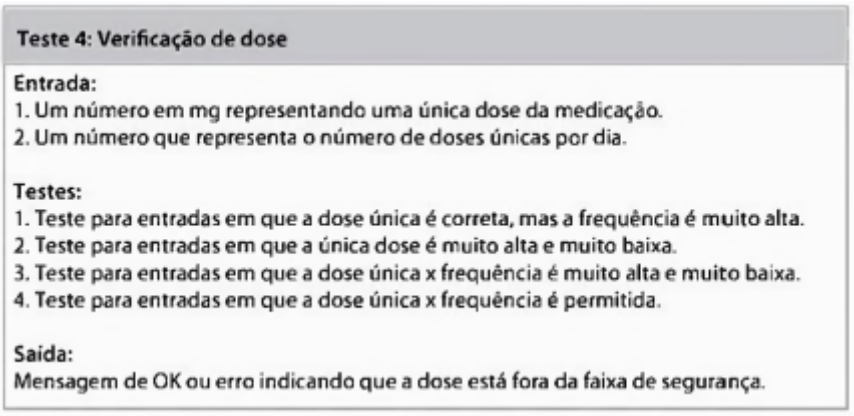
Myers *et al.* (2004) relata que o objetivo da atividade de teste é revelar defeitos, uma vez que, por meio da execução do programa, não se pode provar sua correção. Já que o teste bem-sucedido é o que revela a presença de um defeito, um bom caso de teste é o que tem alta probabilidade de revelar um defeito ainda não descoberto. Pois um caso de teste é provado ser investimento valioso para o projeto, quando encontra um novo erro. O teste de software é o método mais apropriado para encontrar erros e assumi-los. O caso de teste é exatamente a ferramenta que irá guiar, promovendo o progresso, e evidenciando as falhas do software. Myers também expõe que o teste também pode ter o propósito de demonstrar confiança no software, evidenciando que ele faz o que realmente é proposto, porém a melhor forma de atingir esse nível de qualidade é atigindo uma busca dedicada pelos erros.

Um teste inteligente deve possuir duas características: não ser muito simples e nem um tanto complexo. Apesar de ser viável combinar uma série de testes em um caso de teste, seus possíveis efeitos podem mascarar erros. Além disso, ao projetar uma iniciativa de

casos de teste deve ser considerado diversos fatores que podem causar falhas no sistema, como manipulação de processamento, paralelismo de tarefas e temporização de dados (PRESSMAN, 2006).

Sommerville (2011) define que os "casos de teste são especificações das entradas para o teste e da saída esperada do sistema (os resultados do teste), além de uma declaração do que está sendo testado. Os dados de teste são as entradas criadas para testar um sistema. Às vezes, os dados de teste podem ser gerados automaticamente, mas a geração automática de casos de teste é impossível, pois as pessoas que entendem o propósito do sistema devem ser envolvidas para especificar os resultados esperados. No entanto, a execução do teste pode ser automatizada. Os resultados esperados são automaticamente comparados aos resultados previstos, por isso não há necessidade de uma pessoa procurar erros e anomalias na execução dos testes". A Figura 3 apresenta um exemplo figurativo de caso de teste com descrição resumida, seu objetivo é verificar se a dose prescrita de uma medicação não fica fora dos limites de segurança conhecidos.

Figura 3 - Descrição do caso para verificação da dose



Fonte: (SOMMERVILLE; 2011)

De início, é improvável revelar a probabilidade de um caso de teste descobrir ou não um defeito, para isso existem as técnicas de teste, que auxiliam na probabilidade de revelar defeitos por meio da escolha sistemática de um conjunto de casos de teste. Com os critérios das técnicas, como a funcional ou estrutural, subdividem o domínio em subdomínios. Citando como exemplo, ao definir que um certo caminho do programa deve ser praticado, é estabelecido um subconjunto do domínio como entrada, formado por dados de teste que são

capazes de executar tal caminho e do qual um caso de teste deve ser selecionado. A partir desses critérios, torna-se maior a possibilidade de que o conjunto construído vá revelar possíveis defeitos no software (DELAMARO *et al.*, 2007).

Myers *et al.* (2004) fala sobre definir o que é a saída ou resultado esperado de um caso de teste, pois se o resultado esperado não foi predefinido, há chances de que o resultado seja interpretado incorretamente. Outro ponto necessário é sobre o design de caso de teste, que é tão considerável porque o teste completo é impossível, porém esse mesmo teste deve ser o mais completo possível. A principal estratégia é, de acordo com as limitações de tempo e custo, verificar qual subconjunto de todos os casos de testes possíveis tem a maior probabilidade de detectar a maioria dos erros.

3.3 Estimativa de Esforço de Teste

Sommerville (2011) define dez princípios sobre planejamento, e em um deles fala sobre estimativas, cujo objetivo é oferecer indicadores de custo, esforço e prazo, baseado no que se entende sobre o trabalho a realizar. As estimativas devem ser baseadas no que se conhece, caso haja poucas informações ou inseguras, elas não serão confiáveis. Para Myers (MYERS *et al*., 2004) vários fatores podem afetar a probabilidade de uma determinada estimativa de esforço de teste, como o tamanho e complexidade da aplicação, a quantidade de integrantes na equipe, o cronograma para desenvolvimento, como também, a cultura e a formação da equipe de desenvolvimento.

Bartié (2002) conceitua como o esforço de garantir qualidade todo o investimento feito com o propósito de um software atingir a qualidade desejada. Esses investimentos alocam todos os esforços aos custos das conformidades, como também aos das não-conformidades. Além das características citadas, Bartié (2002) apresenta outros conceitos, tais como:

- **Custos da Conformidade:** tudo que é realizado com a intenção de melhorar e garantir o processo de desenvolvimento deve ser considerado custo da conformidade, ou seja, o custo para se obter e garantir qualidade. Seu principal objetivo é a prevenção e detecção de erros do processo.

- **Custos da Não-conformidade:** tudo que é realizado ou gerado em função de defeitos produzidos durante os projetos de software, ou seja, são os custos provenientes da falta de qualidade. Até mesmo custos financeiros causados por esses defeitos, são englobados como não-conformidade.

Atributos de teste são reunidos durante o esforço de teste, como funcionalidade, UI, desempenho, usabilidade, segurança, acessibilidade entre outros. Todos esses atributos determinam a qualidade do produto, no entanto, as necessidades de testar esses componentes podem variar ou sofrer mudanças devido a razões externas, usuários finais ou requisitos da aplicação (SHARMA, 2016).

Para evitar a dispersão natural no esforço de criação dos testes em uma aplicação, é necessário realizar a categorização de testes durante o planejamento, para serem aplicadas ao processo de validação do software. Sua finalidade é aumentar a eficiência da detecção do maior número possível de cenários de testes (BARTIÉ, 2002).

Cada categoria tem um determinado objetivo que define a intenção da realização dos testes. Entender a categoria de testes e aplicar como serão organizados e estruturados os cenários que irão ser executados, proporciona a equipe focada em testes a melhor conduzir seus trabalhos (BARTIÉ, 2002).

3.4 qEstimation

A estimativa em teste de software ainda é uma área com vários desafios. Ao pensar em estimar esforço de teste, duas abordagens geralmente são seguidas, tradicionalmente, Método Development Ratio e técnicas de estimativa independentes dos requisitos de teste. Ainda assim, abordagens antigas caracterizavam-se pela ausência de correlação direta entre o esforço estimado e a criticidade do sistema em teste em seu contexto operacional (GHANIM, 2017).

A importância de uma estimativa adequada se deve ao fato que é fundamental saber o esforço necessário da realização das atividades de testes a fim de determinar a quantidade de recursos alocados e prazos de entrega dos testes a serem executados. Visto que, estimativas realizadas apenas baseadas na experiência da equipe tendem a ser imprecisas,

podendo causar uma alocação de recursos inadequada ou problemas para cumprir os prazos de execução dos testes (SOUSA *et al.,* 2018).

O *qEstimation* propõe medir o tamanho do caso de teste com base em seus pontos de verificação, que são os dados de teste, pré-condições, como também o tipo de teste. A abordagem divide o processo em duas etapas principais, sendo a primeira estimar o teste utilizando Test Case Point Analysis (TCPA), que consiste em medir o tamanho do caso de teste avaliando os elementos de complexidade do caso de teste (Nguyen *et al.,* 2013).

Na Tabela 1 é possível visualizar os pesos definidos para as pré-condições de um caso de teste, segundo a abordagem *qEstimation*.

Tabela 1 - Pesos definidos para pré-condições

Nº de pré-condições	**Nível de complexidade**	**Peso atribuído**	**Descrição**
0	Nulo	0	A pré-condição não é necessária para execução do caso de teste.
1 a 3	Baixo	1	Alguns parâmetros ou critérios são necessários para a execução do caso de teste.
4 a 6	Médio	3	A preparação é necessária para execução do caso de teste, alguns critérios são necessários.
Maior que 6	Alto	5	Configurações em um nível mais alto são necessárias para execução do caso de teste.

Fonte: adaptado de Nguyen *et al.* (2013) e Sousa *et al.* (2018)

A Tabela 2 apresenta os pesos definidos para dados de testes de um caso de teste, seguindo a abordagem *qEstimation*.

Tabela 2 - Pesos definidos para dados de testes

Nº de dados de teste	**Nível de complexidade**	**Peso atribuído**	**Descrição**
0	Nulo	0	Não é necessário nenhuma preparação para dados de teste.

1 a 5	Baixo	1	Um dado de teste simples é necessário e pode ser criado durante o tempo de execução do caso de teste.
6 a 8	Médio	3	O dado de teste é pensadamente preparado, necessitando de esforço extra para criá-lo.
Maior que 8	Alto	6	O dado de teste é pensadamente preparado, necessitando de esforço considerável para criá-lo.

Fonte: adaptado de Nguyen *et al.* (2013) e Sousa *et al.* (2018)

Como mencionado anteriormente, seguindo a abordagem, um dos pontos de parâmetros para medição do caso de teste são os pesos para cada tipo de teste. O *qEstimation* define os tipos de testes que podem ser utilizados para realizar a estimativa de testes. Na Tabela 3 são descritos os pesos para cada tipo de teste definido por esta abordagem, como também a descrição de cada um deles.

Tabela 3 - Pesos definidos para tipos de testes

Tipos de teste	**Peso**	**Descrição**
Funcional	1.0	Considerado linha de base para início dos testes.
API	1.22	Verifica a precisão das interfaces no fornecimento de serviços.
Base de dados	1.36	Testa a precisão do banco de dados scripts, integridade de dados e/ou dados migração.
Segurança	1.39	Verifica o quão bem o sistema lida com ataques de hackers, usuário não autorizado e acessos não autenticados.
Instalação	1.09	Essenciais em cenários atualizar, instalar ou desinstalar processos de software.
Rede	1.27	Testa a comunicação entre entidades via redes.
Algoritmo	1.38	Verifica algoritmos e projetos computacionais que foram implementados no sistema.
Usabilidade	1.12	Testa a facilidade de uso e outros atributos de usabilidade do sistema.
Performance	1.33	Verifica se o sistema atende aos requisitos de

		desempenho.
Recuperação	1.07	Confere a precisão do processo que o sistema leva ao se recuperar de falhas e/ou erros.
Compatibilidade	1.01	Testa a compatibilidade do sistema com outros elementos em que ele deve operar.

Fonte: adaptado de Nguyen *et al.* (2013)

A segunda etapa consiste em medir o esforço de teste usando o índice de produtividade ou o método de regressão linear simples. O índice de produtividade pode ser calculado usando dados históricos de teste ou até por meio de um experimento que colete os pontos de caso de teste e o esforço gerado, caso os dados históricos ainda não existam no momento da estimativa. Outra forma de medir o esforço de teste é usando o método de regressão linear simples, que calcula os intervalos da estimativa e o nível de confiança da estimativa, o método necessita de pelo menos cinco ciclos de teste ou projetos semelhantes para fornecer estimativas precisas (Nguyen *et al.,* 2013).

Segundo SOUSA *et al.* (2008), para calcular utilizando o índice de produtividade é necessário calcular o número de Pontos de Casos de Testes (TCP) executado por pessoa em um período de tempo e atribuir o valor do peso ao determinado tipo de teste (TP).

$$\text{Esforço} = \Sigma(\text{TCP} \cdot \text{TP}) \div \text{produtividade}$$

A segunda forma de calcular o esforço é por meio da regressão linear. Dado N projetos completos em que o i-ésimo projeto tem o número de TCP_i e esforço (E_i), a regressão linear é apresentada por:

$$E_i = A + B \cdot TCP_i + e_i$$

Onde A e B são coeficientes e e_i é o fator de erro do modelo. Os valores A e B são então usados no modelo para estimar o esforço de novos projetos como segue:

$$E = A + B \cdot TCP$$

3.5 USINN Modeler

Permanecer com modelos de interação e navegação consistentes não é uma tarefa fácil, visto que, os modelos de interação utilizam diferentes elementos como base e os modelos de navegação geralmente se baseiam em unidades de apresentação da interface, sendo uma possível solução, utilizar elementos comuns em ambos (MARQUES *et al*., 2016).

Com o objetivo de representar graficamente a interação e navegação de sistemas interativos, considerando aspectos de usabilidade, foi proposto o modelo denominado de USINN (*USability-oriented INteraction and Navigation Model*) para proporcionar melhoria na usabilidade e qualidade de uso no produto final (MARQUES *et al*., 2016).

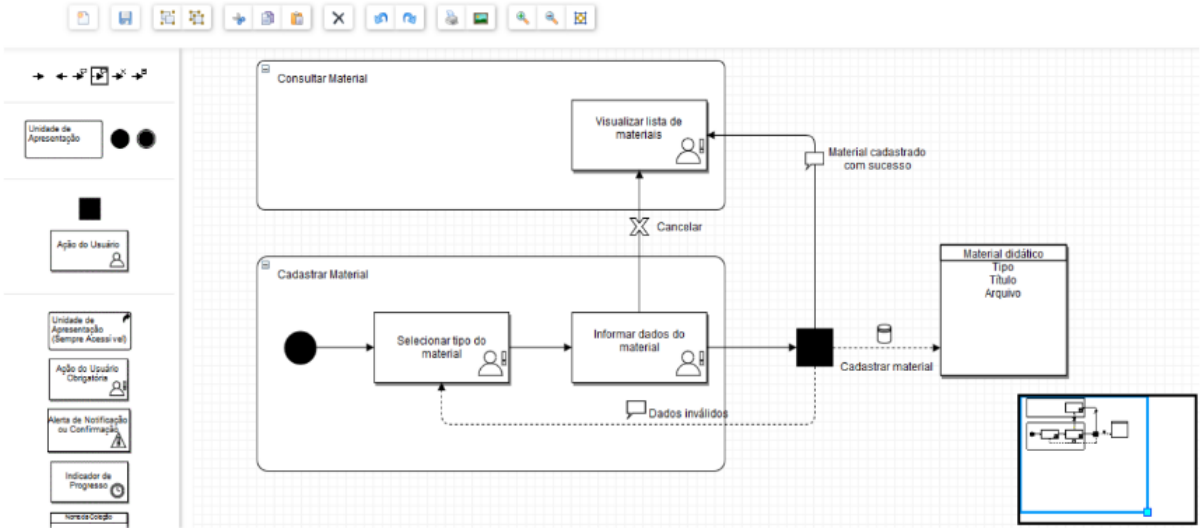
Porém, elaborar diagramas USINN ainda não era algo prático, pela falta de uma ferramenta para elaborar diagramas com esta notação, tornando os diagramas passíveis de erro de sintaxe. Com a finalidade de criar e editar modelos desta notação, foi desenvolvida uma ferramenta web para elaboração desses modelos (COSTA; MARQUES, 2019).

A ferramenta denominada de *USINN Modeler* torna a elaboração de modelos USINN mais prática e sintaticamente correta. Sua primeira versão possui alguns dos requisitos essenciais para a notação USINN. Costa e Marques (2019) apresentam as principais funcionalidades do *USINN Modeler*:

- **Gerenciar graficamente os elementos da notação:** através da técnica "arrastar e soltar" é possível inserir elementos na notação.
- **Conectar os elementos da notação:** a partir dos relacionamentos permitidos entre os elementos, é possível inserir conectores entre eles.
- **Impedir conexões inválidas:** há uma verificação para identificar se a conexão é válida, ao conectar um elemento a outro.
- **Validar sintaxe do diagrama:** cada ação é avaliada dentro das regras do USINN, caso não seja permitida, é exibida uma mensagem de alerta.

A Figura 4 ilustra a tela do software *USINN Modeler*, em que está sendo elaborado um diagrama USINN.

Figura 4 - Exemplo de diagrama USINN sendo elaborado com o USINN Modeler



Fonte: (COSTA; MARQUES, 2019)

4 TRABALHOS RELACIONADOS

Os trabalhos apresentados nesta seção estão relacionados à Estimativa de Esforço de Teste, Teste de Software, Casos de Teste, como também ferramentas e processos criados e/ou utilizados para colaborar com o estudo de Estimativas de Teste. O capítulo é dividido em três seções, em que a primeira seção fala do Estado da Arte sobre Estimativa de Esforço de Teste, a segunda seção sobre Estudo de Caso Estimativa de Esforço de Teste, e a terceira seção apresenta sobre Software de código aberto e Esforço de Teste.

4.1 Estado da Arte sobre Estimativa de Esforço de Teste

Adali *et al.* (2017) apresentam uma revisão sistemática da literatura e uma pesquisa industrial, que foi realizada com o intuito de investigar o estado da arte em estimativa de esforço de teste e a prática atual da indústria de software na Turquia. De início, foi realizada uma análise sobre as entidades de software utilizadas como entrada principal e medidas de base usadas na estimativa, cujos resultados mostraram que casos de teste são utilizados como entrada principal, seguido de, casos de uso, código, especificação de requisitos e testadores. Sobre as métricas mais utilizadas incluem o número de casos de teste, número de testadores, Use Case Points (UCP), e várias outras métricas definidas. Para trazer a prática da indústria, os resultados da revisão de literatura foram usados para desenvolver uma pesquisa e distribuí-la para profissionais do mercado. Foram coletadas um total de 99 respostas de diversos domínios de negócios e diferentes proporções de empresas. Sobre as pesquisas realizadas na Indústria de Software Turca, foi observado que, a maioria das empresas usam um método para estimar o esforço de teste, e que apenas alguns métodos da literatura são utilizados pela indústria, revelando uma lacuna de conhecimentos que existe entre os dois ambientes estudados, e por fim, um outro ponto identificado, foi a sugestão da maioria dos participantes em adicionar métricas para melhorar os métodos atuais usados.

Bluemke e Malanowska (2021) apresentam uma revisão sistemática da literatura sobre estimativa de esforço de teste, tal qual tem como objetivo identificar métodos e ferramentas existentes de estimativa de esforço de teste de software e sua utilização na academia e na indústria. Foram selecionados 309 artigos e 173 aceitos ao final, a partir disso,

foi realizada uma análise profunda dos resultados e síntese descritivas destes trabalhos. Além de identificar os métodos existentes de estimativa de esforço de teste de software, os autores também estavam interessados em métodos que atendessem os critérios de: i) aplicável na fase inicial do ciclo de vida de desenvolvimento de software; ii) possível de ser automatizados e iii) independente do paradigma de programação utilizado. Duas decisões cruciais foram tomadas durante o processo de pesquisa, sendo a primeira decisão estender o escopo para tópicos relacionados (identificados na pesquisa de banco de dados) e a definição de categorias e classificação para os artigos. Como resultado, apenas 84 do total de 173 descrevem métodos de estimativa de esforço de teste, ilustrando a dificuldade de encontrar fontes relevantes. O *qEstimation* foi classificado pelos autores como um dos trabalhos que propõe novas métricas e descreve soluções inovadoras. Por fim, foi relatado que apesar do fato de que a estimativa do esforço de teste de software se apresenta como uma tarefa muito importante, existe um número extremamente limitado de abordagens científicas para resolver esse problema. Além de tudo, algumas soluções que pretendem resolver o problema de estimativa de teste revelam-se de qualidade muito duvidosa.

4.2 Estudo de Caso sobre Estimativa de Esforço de Teste

Nguyen *et al.* (2013) apresentam uma proposta de um processo para estimar o tamanho e o esforço de teste de software: *qEstimation*. A abordagem propõe medir o tamanho do caso de teste com base em seus pontos de verificação, dados de teste, pré-condições, como também o tipo de teste. O processo é dividido em duas etapas principais, estimar o teste utilizando *Test Case Point Analysis* (TCPA) e o esforço de teste usando o índice de produtividade ou a regressão linear simples. Os autores aplicaram o processo em dois projetos de teste de software em uma empresa de desenvolvimento global de software. No primeiro estudo foi utilizado para estimar o esforço de executar 1.631 testes manuais, no segundo estudo, estimou-se o esforço para escrever scripts de testes automatizados para 160 casos de teste. Em ambos os projetos, as estimativas foram consideradas confiáveis, superando a estimativa baseada na experiência feita por testadores individuais. Entretanto, o processo ainda tem algumas limitações, os casos de teste devem estar disponíveis para realizar

tal estimativa, dados de testes reutilizados em vários casos de teste não tão extensos, poderá ter uma contagem desses dados várias vezes.

Sousa *et al.* (2018) apresentam um estudo de caso em um ambiente real de fábrica de teste, cujo objetivo principal era identificar prováveis técnicas de estimativa de teste, através de uma pesquisa na literatura. O objetivo consistiu em selecionar a melhor abordagem para ser aplicada na Fábrica de Testes do Grupo de Redes, Engenharia de Software e Sistemas (GREat) da Universidade Federal do Ceará. Dentre as abordagens encontradas na busca, a selecionada como adequada para o contexto foi a *qEstimation*. Foram realizadas algumas alterações na abordagem, para adaptá-la ao contexto mencionado, como a forma do cálculo da estimativa, e a computação da complexidade de casos de testes de forma automática, através de uma ferramenta. O processo foi dividido em duas grandes etapas, cálculo da complexidade dos casos de teste e cálculo do esforço para realizar as atividades de testes, sendo realizado em três sistemas web diferentes utilizados por empresas parceiras do Grupo. Após a execução, os resultados obtidos com a aplicação do processo se evidenciaram próximos dos esforços realmente gastos, como também ficaram aproximados dos resultados obtidos pelos analistas de testes. A diferença média entre o esforço estimado e o real gasto ficou em 34,18%, com uma diferença de 5,31% comparando as estimativas feitas pelos analistas de testes e o real gasto, que é de 28,87%. Os resultados mostram que a estimativa realizada pelo *qEstimation* se aproxima mais da realidade que a feita pelos analistas de testes. Apesar de se aproximar da realidade, a abordagem ainda possui algumas limitações, pois ela não é capaz de estimar outras atividades de teste, como documentações e o esforço gasto para reportar inconsistências (*bugs*). Sendo capaz apenas de estimar o esforço para execução dos testes com base na especificação dos casos de teste. Por fim, foi percebido que os resultados se mostraram promissores, e que adaptar e automatizar alguns pontos da abordagem, tornaria esta abordagem ainda mais viável para ser aplicada na indústria.

Santos *et al.* (2019) apresentam um relato de experiência sobre automação das atividades de geração de procedimentos de testes em uma Fábrica de Testes. De início, foi realizada uma busca na literatura sobre soluções que realizem a automação de atividades semelhantes, em seguida, foram selecionadas as soluções que mais se adaptam a um projeto da indústria, que serviram de conhecimento para a criação de uma ferramenta personalizada. Na execução desta experiência, foi utilizada uma metodologia formada por quatro etapas: Identificar potenciais áreas de melhoria baseadas em necessidades da indústria, que consistiu

em observar atividades relacionadas aos testes da aplicação, na etapa 2 foi elaborada uma solução, seguindo com a etapa 3 que foi realizada avaliação sob a ferramenta por um pesquisador e um analista, e por fim, a última etapa foi realizada com a implantação da solução. Durante a execução, foram selecionados sete casos que somavam 41 fluxos e 186 passos para utilizar na ferramenta. Para cada um desses casos, o analista de testes especificou uma suíte de casos de teste manual e depois uma utilizando a ferramenta, em que foi possível observar a redução de esforço na geração de testes a partir do uso da ferramenta, especialmente em casos de uso com grande número de passos. Contudo, também foi possível observar que a automação está ligada ao seguimento do padrão de escrita, e que apenas os casos de uso textuais não trazem todas as informações necessárias.

4.3 Software de código aberto e Esforço de Teste

Hanoglu e Tarhan (2019) apresentam um método para a análise de relação entre esforço e sucesso de teste, que necessita apenas de código-fonte e dados que possam ser acessados por todos no ambiente em que o projeto é publicado. Os métodos propostos são examinados empiricamente por meio da análise de 17 projetos de software livre. As métricas do conjunto de testes que indicam a medição de esforço de teste foram selecionadas e indicadores de tendência centrados no desenvolvedor foram utilizados na avaliação do sucesso do software de código-fonte aberto. Após examinar os métodos de medição, chegou-se à conclusão de que não existe uma métrica final para a medição do sucesso, pois cada métrica oferece uma perspectiva diferente para a medição do sucesso. Com o resultado do método de classificação aplicado, 17 softwares foram classificados de forma compatível e integrada. Chegou-se à conclusão de que o sucesso do software não está totalmente relacionado ao grande esforço de teste, porém existe uma correlação significativa entre o esforço de teste gasto em projetos de software de código aberto, que pode estar relacionado ao sucesso do software.

4.4 Tabela Comparativa

Na Tabela 4 é apresentado a correlação entre os trabalhos relacionados e esta pesquisa.

Tabela 4 - Comparação entre os trabalhos relacionados

Trabalhos	**Metodologia**	**Contexto**	**Aplica estimativa de esforço de testes**	**Avalia a percepção do time**
Adali *et al.* (2017)	Revisão sistemática da literatura e Pesquisa na Indústria	Indústria	Não	Sim
Bluemke e Malanowska (2021)	Revisão sistemática da literatura	Academia e Indústria	Não	Não
Nguyen et al. (2013)	Proposta de processo e Estudo de Caso	Indústria	Sim	Não
Sousa et al. (2018)	Estudo de Caso	Indústria	Sim	Não
Santos et al. (2019)	Proposta de processo e Estudo de Caso	Indústria	Sim	Não
Hanoglu e Tarhan (2019)	Proposta de processo e Estudo de Caso	Indústria	Sim	Não
Este trabalho	Estudo de Caso	Academia	Sim	Sim

Fonte: Elaborado pela autora (2022)

Diferente dos trabalhos mencionados, este trabalho tem a finalidade de realizar um estudo de caso, iniciando uma abordagem de testes no software *USINN Modeler* e aplicando uma técnica de estimativa de esforço de teste no mesmo.

Com base nos trabalhos relacionados a esta pesquisa, apenas o trabalho de Nguyen *et al*. atendeu aos critérios levantados nesta pesquisa. Com isso, optou-se por investigar em mais detalhes sobre o trabalho de Nguyen *et al*. aplicar o processo *qEstimation* e avaliar o seu uso para identificar os ganhos na sua utilização.

5. PROCEDIMENTOS METODOLÓGICOS

Com base nas definições de estudos experimentais em Engenharia de Software (WOHLIN et al., 2000), o estudo caracteriza-se como uma pesquisa exploratória. Wohlin *et al*. (2000) relata que em pesquisas exploratórias "as informações são coletadas e analisadas, e os resultados são usados para melhorar a investigação completa. Em outras palavras, a pesquisa exploratória não responde à questão de pesquisa, mas pode fornecer novas possibilidades que podem ser analisadas e devem, portanto, ser acompanhadas na pesquisa mais focada ou completa".

O trabalho que está sendo realizado para esta pesquisa, possui parceria com toda a equipe de manutenção e desenvolvimento da ferramenta *USINN Modeler*, visando um software mais funcional, com uma melhor performance e experiência de usuário. Os processos sugeridos por Wohlin *et al*. (2000) foram seguidos nesta pesquisa para melhor condução e avaliação da mesma.

Este capítulo apresenta as atividades que foram realizadas para o alcance dos objetivos deste trabalho. A Figura 5 apresenta as etapas do processo metodológico definido para o desenvolvimento deste trabalho.

Figura 5 - Etapas da metodologia adotada

Fonte: Elaborado pela autora (2022)

5.1 Revisão da literatura

Esta etapa teve a finalidade de realizar um estudo de temas que estão relacionados a este trabalho. Para embasar esta pesquisa, foram exploradas algumas fontes, como livros, teses, revistas e artigos que apontam estudos, conceitos, objetivos e aplicações práticas. Dentre os acervos explorados, optou-se principalmente por fazer o uso das bibliotecas digitais ACM (ACM Digital Library) e IEEE (IEEE Xplore), pelo fato de terem uma quantidade considerável de estudos publicados.

A string de busca utilizada como artefato de entrada para busca nas bases de dados foi elaborada através da união de termos que devem conter nos resultados pretendidos pelo mapeamento. A string de busca definida para este trabalho é apresentada abaixo.

("systematic literature review" AND "software test")

OR

("systematic literature review" AND "v&v")

OR

(“software testing process” AND “experience report”)

OR

(“test effort estimate”AND “software test”)

Foram lidos cerca de 15 artigos e selecionados 6 desses, pois os trabalhos que foram descartados abordavam estimativas de esforço não relacionado a teste de software.

As buscas realizadas na etapa de revisão de literatura servem como referência essencial para esta pesquisa, principalmente para a parte dos trabalhos relacionados, que pode ser lida na Seção 4.

5.2 Seleção da abordagem de estimativa de esforço de testes

Esta etapa visou a seleção da abordagem de estimativa de esforço de testes para ser aplicada neste trabalho. Ao realizar o levantamento das ferramentas/métodos que foram

candidatas a serem utilizadas, foi realizado um filtro para que a abordagem escolhida seguisse os critérios definidos.

A partir das abordagens de estimativa de esforço de teste encontradas, o *qEstimation* foi a mais apropriada para este contexto. Visto que, a análise para escolha da abordagem levou em consideração quatro critérios: (i) A abordagem deve ter sido executada e avaliada em um ambiente real; (ii) Tempo e esforço devem ser adequados; (iii) Aplicação da abordagem em outros softwares forneceu estimativas precisas; (iv) Os pré-requisitos necessários para utilização da ferramenta são viáveis e (v) Não demandar esforço excessivo do time.

5.3 Planejamento do estudo de caso

Nesta etapa foi planejado como será conduzido o estudo de caso, seguindo as recomendações construídas por Wohlin *et al*. (2000).

Na etapa inicial foi realizada uma coleta de informações sobre *USINN Modeler*, desde a especificação inicial do sistema, entendendo sobre a abordagem USINN, conversa com o time de desenvolvimento e acesso a plataforma. Foi observado que não existe um mapeamento para inconsistências (bugs) ou para melhorias em funcionalidades no sistema, como também não há esforço de teste atualmente realizado na plataforma.

A partir desse gargalo surgiu a proposta do presente trabalho, tendo como motivação principal iniciar um processo de testes e garantir a qualidade do software.

O estudo de caso será conduzido com o intuito de responder às seguintes questões de pesquisa:

Q1. “Quais as funcionalidades essenciais do sistema?”;

Q2. “Quais são as funcionalidades mais difíceis de executar? (Necessitam de maior conhecimento de regras de negócio)”;

Q3. “A iniciativa de abordagem de testes no *USINN Modeler* trouxe vantagens para o mapeamento de inconsistências e melhorias para a ferramenta?”;

Q4. “A estimativa gerada trouxe dados semelhantes ao executado no ambiente real?”;

Q5. “O processo de estimativa de esforço de teste contribuiu para melhoria/evolução da ferramenta?”;

Para responder às questões de pesquisa, foi conduzido um estudo de caso no projeto Usinn onde os dados foram coletados e analisados a partir das experiências realizadas durante o processo de testes, a estimativa gerada, como também o *feedback* obtido com a equipe de testes.

5.4 Condução do estudo de caso

A partir do planejamento do estudo de caso que orientou a condução do mesmo, a pesquisa foi seguida com o intuito de investigação e aplicação do processo escolhido para colher e comparar os resultados finais. Esta pesquisa foi realizada em um projeto de pesquisa e desenvolvimento tecnológico coordenado pela orientadora deste trabalho, do qual participam atualmente 10 estudantes de graduação em Engenharia de Software e Ciência da Computação desempenhando atividades de requisitos, desenvolvimento e design.

Diante deste contexto, a condução desse estudo de caso deve não só responder às questões de pesquisa, como também trazer resultados que irão contribuir para a manutenção e evolução do *USINN Modeler*. Nesta fase foram realizadas sete etapas, sendo elas:

1. Workshop sobre elaboração de Casos de Teste.
2. Definição dos tipos de testes a serem escritos.
3. Criação dos Casos de Teste.
4. Aplicação do processo de estimativas de esforço de testes.
5. Execução de testes de regressão na ferramenta *USINN Modeler*.
6. Comparação entre as estimativas geradas e o que foi realizado no ambiente real de testes.
7. *Feedback* com a equipe de testadores.

Cada uma das etapas são descritas detalhadamente no Capítulo 6 e 7.

5.5 Análise dos resultados

Nesta etapa foi realizada a análise dos dados obtidos a partir da execução do estudo de caso, que foi conduzida na etapa anterior. Na análise dos resultados foram investigados três principais pontos que trouxessem evidências para discussão das questões definidas nesta pesquisa, que são:

1. Avaliação da experiência adquirida pela equipe de testadores: foi aplicada a técnica de grupo focal para colher as respostas dos testadores.
2. Avaliação do projeto de casos de teste: durante a execução dos testes de regressão foi possível avaliar a qualidade do projeto de casos de teste.
3. Avaliação da utilização do processo de estimativa de esforço de teste: a partir da análise dos dados estimados e dos números obtidos no ambiente real tornou-se possível avaliar a utilidade da abordagem escolhida.

6. ESTUDO DE CASO

Propondo-se a investigar as questões de pesquisa definidas, o estudo de caso foi conduzido seguindo o trabalho de Wohlin *et al*. (2000) como orientação.

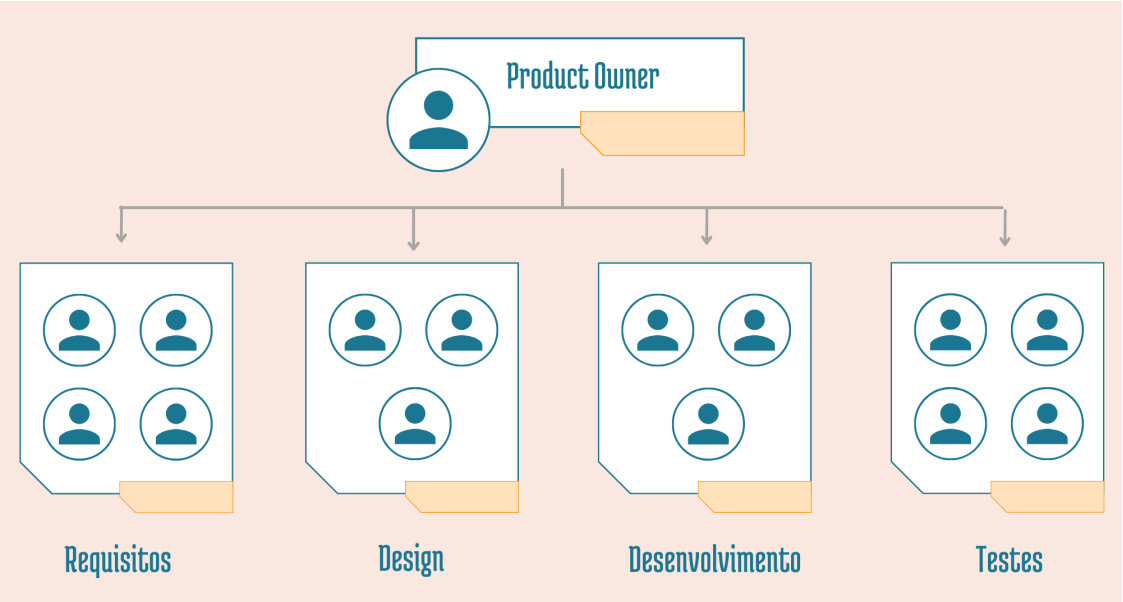
6.1 Contexto e participantes

O estudo de caso foi realizado juntamente com o projeto Usinn, projeto este conduzido pela professora e orientadora deste trabalho Anna Beatriz dos Santos Marques. De início, foram disponibilizadas as documentações mais atuais sobre software e também o seu acesso, para que fosse possível entender o seu nível de maturidade em relação ao desenvolvimento.

O estudo de caso foi conduzido após o recrutamento de 4 membros para o projeto Usinn para compor a equipe de testes de software. Assim, foi possível adotar o processo definido pela abordagem *qEstimation*, uma vez que a equipe estava em fase de organização para iniciar suas atividades.

O projeto tem uma configuração dividida em setores, que são: requisitos, design, desenvolvimento e testes, alguns membros atuaram em mais de um setor, como é o caso do setor de requisitos que numa fase inicial do projeto, quase todos os membros atuaram nessa categoria. A Figura 6 ilustra como funciona a estrutura do projeto Usinn atualmente.

Figura 6 - Estrutura por setores do projeto Usinn



Fonte: Elaborado pela autora (2022)

Os encontros das equipes acontecem uma vez a cada semana, conduzidos pela *Product Owner* e orientadora do projeto. Neles são revisados as atividades da semana anterior, os gargalos impeditivos e as próximas atividades.

Além da reunião com todos os membros, a equipe de testes também se reúne individualmente uma vez por semana, os encontros são conduzidos pela Analista de testes da equipe. Neles são analisadas as atividades realizadas pelos membros e o progresso das mesmas. O objetivo dessa reunião é alinhar toda a equipe em relação aos prazos e metas, como também tirar dúvidas e realizar treinamentos, quando necessário.

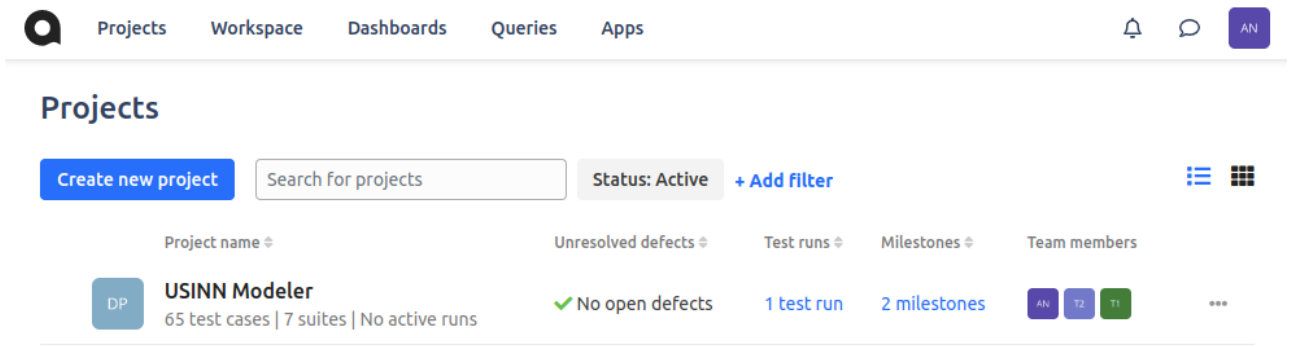
6.2 Preparação

Visando contextualizá-los sobre a importância desta pesquisa e a sua contribuição para o projeto Usinn, foi realizado um workshop sobre elaboração de casos de teste para todos os membros do projeto. Nesta apresentação foram abordados conceitos fundamentais para a condução desse estudo: teste de software, teste funcional, teste unitário, teste não funcional, teste de regressão, caso de teste, especificações de caso de teste.

Durante o workshop realizado presencialmente, os membros do projeto levantaram dúvidas sobre o tema e também praticaram a criação de casos de teste. Eles também conheceram a ferramenta escolhida para criação e execução dos testes: o *Qase*. No planejamento do estudo de caso, a proposta de ferramenta a ser utilizada seria o *Testlink,* porém o *Qase* facilitou alguns gargalos que o *Testlink* possui, como por exemplo, a facilidade da plataforma ser armazenada em nuvem, evitando instalações, como também a possibilidade de organizar os casos de teste em grupos lógicos, definindo sua gravidade, prioridade e pré-condições.

A Figura 7 ilustra a tela inicial da ferramenta *Qase* após a inserção das credenciais de acesso a conta, nela é possível visualizar os projetos que o respectivo usuário está inserido, como também algumas informações sobre o mesmo: nome do projeto, defeitos não resolvidos, testes executados, principais entregas e membros do projeto.

Figura 7 - Tela inicial da ferramenta *Qase*



Fonte: Qase (2022)

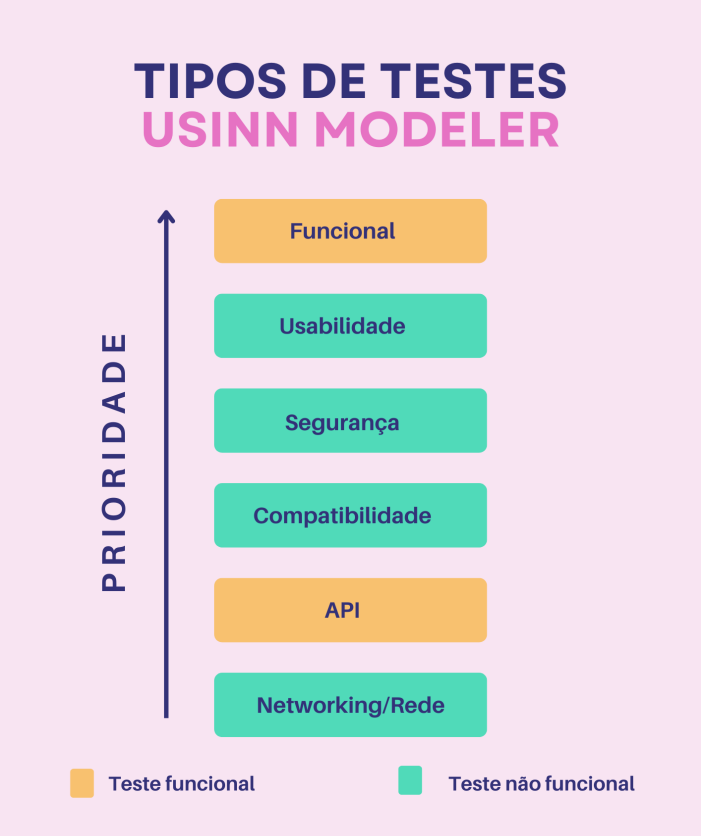
6.3 Definição dos tipos de testes

Em outra reunião, também de forma presencial, junto com os integrantes do projeto foram definidos os tipos de testes a serem realizados no software *USINN Modeler*. A definição dos tipos de testes também interfere na decisão dos tipos de casos de teste a serem criados, pois a maior referência para condução de todo e qualquer teste no *USINN Modeler* será o projeto de casos de teste.

Dessa maneira, a seleção dos tipos de testes foi realizada nesta reunião, considerando a opinião da *Product Owner* do projeto, Anna Beatriz, da equipe de Requisitos e Documentação, como também da equipe de Desenvolvimento e *User Experience.* Aleḿ de participar da definição dos tipos testes, a equipe de testes tomou a decisão final acerca dos tipos de testes a serem realizados no software.

A seleção dos tipos de testes levou em conta os tipos de testes que a abordagem *qEstimation* engloba na sua estrutura, como pode ser visto na Figura abaixo. A Figura 8 ilustra o que foi definido para organização dos casos de teste de acordo com o tipo, levando em conta a prioridade dos testes a serem realizados.

Figura 8 - Tipos de testes definidos para o USINN Modeler



Fonte: Elaborado pela autora (2022)

6.4 Criação dos Casos de Teste

Nesta etapa foi realizada a criação do projeto de casos de teste, sendo conduzida pela Analista de Testes e autora desta pesquisa, e desenvolvida pelos testadores do projeto Usinn. Pelo fato de o estudo ser focado em testes, foi necessário desenvolver um cronograma de encontros semanais entre a equipe de testadores e analista.

Dado que a equipe é formada por cinco pessoas, os integrantes foram divididos em duas duplas e a analista que permaneceu no gerenciamento do projeto. Houve uma média de duas a três reuniões por semana, dependendo da ocorrência em que os testadores necessitavam de suporte, podendo ser: esclarecimento de regras de negócio ou requisitos, vocabulário e termos técnicos na escrita de casos de teste, entre outras.

Os testadores se reuniam também com suas duplas com intuito de tornar mais produtiva a escrita dos cenários de teste. Para reunir conhecimento sobre o software e tornar mais eficiente a escrita, os testadores tiveram acesso a todas as documentações disponíveis sobre o sistema, sendo elas: protótipos de interface, documento de requisitos, histórias de usuários, documento de personas, diagramas e artigos.

Entre as mais utilizadas é a documentação sobre histórias de usuários, pela razão de ser a documentação mais atualizada sobre o software até o momento, sendo benéfica até mesmo pela forma que é escrita, em *Behavior Driven Development (BDD)*. Ela também descreve cenários de usabilidade, regras de negócio, modelos de interação, protótipos e fluxos principais e alternativos de determinadas funcionalidades.

Na Figura 9 é possível visualizar um exemplo de história de usuário que consta na documentação do *USINN Modeler*.

Figura 9 - Exemplo de história de usuário

<table>
<tr><td>US:</td><td>04</td></tr>
<tr><td>CARTÃO:</td><td>Como Usuário, gostaria de desfazer as alterações feitas em um diagrama para que eu possa garantir que erros despercebidos não foram largamente propagados</td></tr>
<tr><td>MECANISMO DE USABILIDADE:</td><td>Mecanismos de usabilidade:

Desfazer (M5):
R11- As alterações feitas no arquivo poderão ser desfeitas/refeitas, portanto, certifique-se de que uma opção de desfazer/refazer seja fornecida.</td></tr>
<tr><td>CONFIRMAÇÃO:</td><td>Critério: desfazer erros feitos no diagrama.
Dado que estou na área de criação de diagramas
Quando começo a criar um diagrama

<table><tr><td>Fluxo principal</td><td>Fluxo alternativo</td></tr><tr><td>E insiro um elemento no diagrama
E apago esse elemento
E aperto Ctrl + Z
Ou clico na função desfazer
Então o sistema refaz as últimas alterações</td><td>E insiro um elemento no diagrama
E apago esse elemento
E aperto Ctrl + Z
Ou clico na função desfazer
E o sistema não refaz minha última ação
Ou o sistema atingiu o limite de ações registradas no buff
Então o sistema retorna feedback sobre o limite ou que ocorreu algum problema.</td></tr></table></td></tr>
</table>

Fonte: Documentação do projeto Usinn (2022)

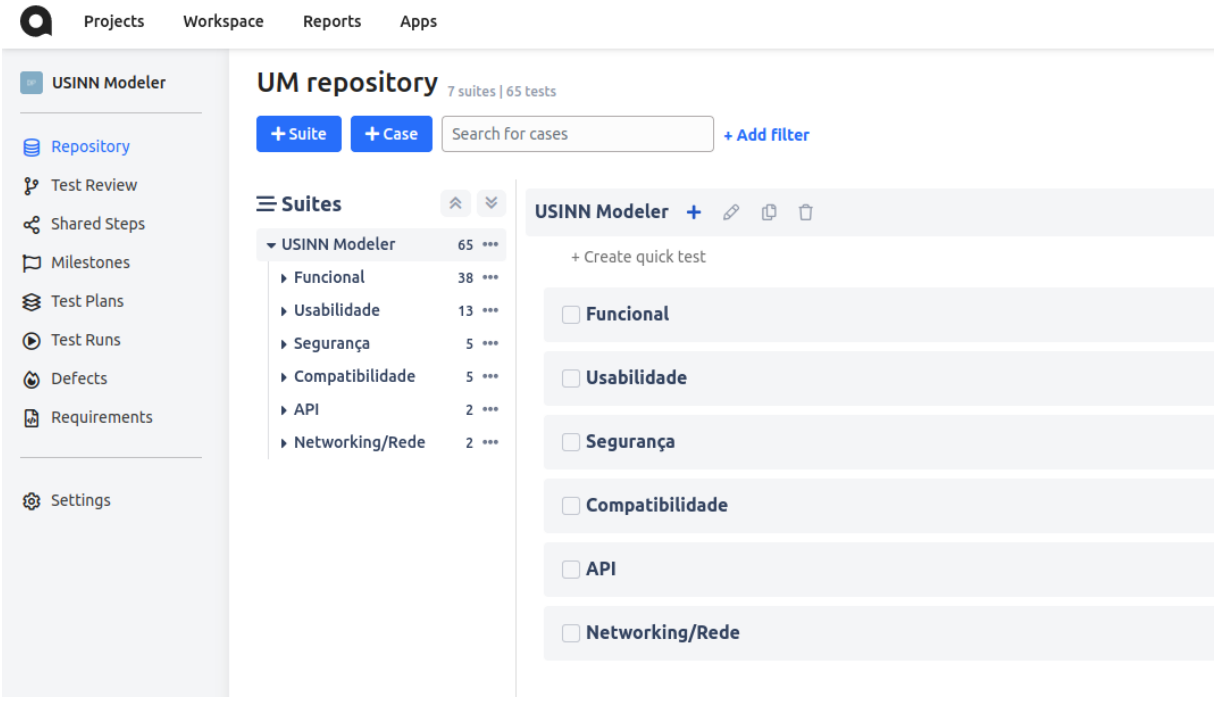
Foi também a partir das documentações e dos encontros com toda a equipe do projeto que foi observada a necessidade de elaborar alguns cenários de casos de teste com comportamento negativo. Casos de teste assim tem como característica a execução de vários passos que leva a uma resposta de impedimento do sistema sob tal comportamento, assim é possível visualizar o comportamento do sistema em situações de erro, por exemplo, quando o software deve exibir um aviso de alerta.

De acordo com Desikan, (2008 apud Medeiros, 2008, p. 29) ao selecionar casos de testes para o ciclo de regressão, não se deve enfatizar os testes que tenham nenhuma (ou pouca) relevância no reparo dos erros. Desikan [5] afirma que se deve dar preferência aos testes positivos, já que uma falha neste tipo de teste pode criar alguma confusão. É recomendável, porém, que os ciclos regulares de teste, antes do teste de regressão, devam ter uma mistura de casos de teste positivos e negativos.

A etapa de escrita superou as expectativas, pois o prazo dado a esta etapa foi concluído em período menor que o planejado, utilizando 83% do tempo estipulado para a criação de casos de testes, permitindo assim que a equipe evoluísse para a criação de casos de testes para melhorias do sistema. Ou seja, funcionalidades ainda não implementadas no software, apenas planejadas e/ou prototipadas.

A Figura 10 exibe o repositório de casos de teste do *USINN Modeler*. O repositório foi nomeado de *USINN Modeler* e contém 6 suítes de testes, que armazenam os casos de testes de acordo com o tipo de teste a ser realizado.

Figura 10 - Tela do repositório USINN Modeler



Fonte: Qase (2022)

Além de contribuir para iniciativa do projeto de testes, os casos de testes escritos para melhorias no sistema também colaboram na fase de desenvolvimento, pois é uma documentação segura a ser seguida que contém parâmetros que evidenciam a forma que tal funcionalidade deve se comportar no software.

A correção dos casos de testes é uma tarefa contínua em um projeto de testes, visto que um software sempre necessita de correções e melhorias. A correção teve início nesta

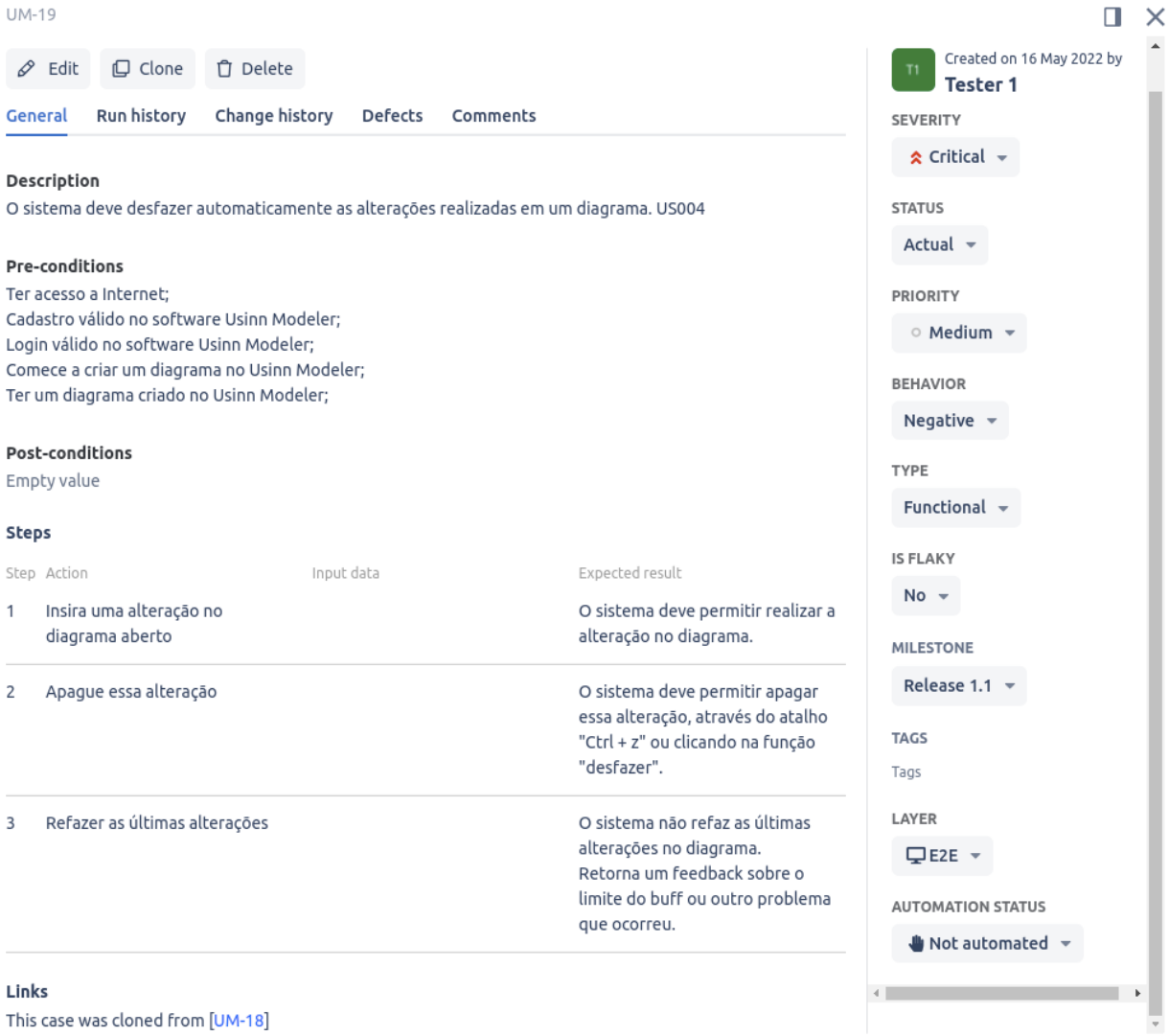
pesquisa logo após a escrita, em que um testador fazia revisão de um caso de teste criado por outro testador.

Além de possuírem pontos de verificação, dados de teste, pré-condições, e tipo de teste. Os casos de teste também contam com outros parâmetros, que são:

- *Severity*: indica a gravidade do caso de teste, podendo ser bloqueadora, crítica, maior, normal, menor ou trivial.
- *Status*: informa a situação real da escrita do caso de teste, podendo ser atual, rascunho ou descontinuado.
- *Priority*: relata a prioridade do caso de teste, tendo a possibilidade de ser alta, média ou baixa.
- *Behavior*: indica o comportamento esperado no caso de teste, capaz de ser positivo, negativo ou destrutivo.
- *Milestone*: marco de versionamento em que o caso de teste está incluído, como exemplo, uma nova versão do software.
- *Layer*: camada que será testada, podendo ser teste de ponta a ponta, *API* (Application Programming Interface), teste unitário.
- *Automation status*: indica se o caso de teste é automatizado ou não.

Na Figura 11 é possível observar esses parâmetros sendo utilizados na configuração de cada caso de teste.

Figura 11 - Tela de visualização dos detalhes de um caso de teste



Fonte: Qase (2022)

6.5 Aplicação do Processo de Estimativa para Testes

Nesta etapa iniciou-se a aplicação da abordagem selecionada para estimativas: *qEstimation*. Após a conclusão da escrita dos casos de testes, por responsabilidade da analista de testes, que aplicou o *qEstimation* no projeto de testes. Realizando a contagem de todos os parâmetros necessários para aplicar a abordagem, medindo o tamanho do caso de teste avaliando seus elementos de complexidade.

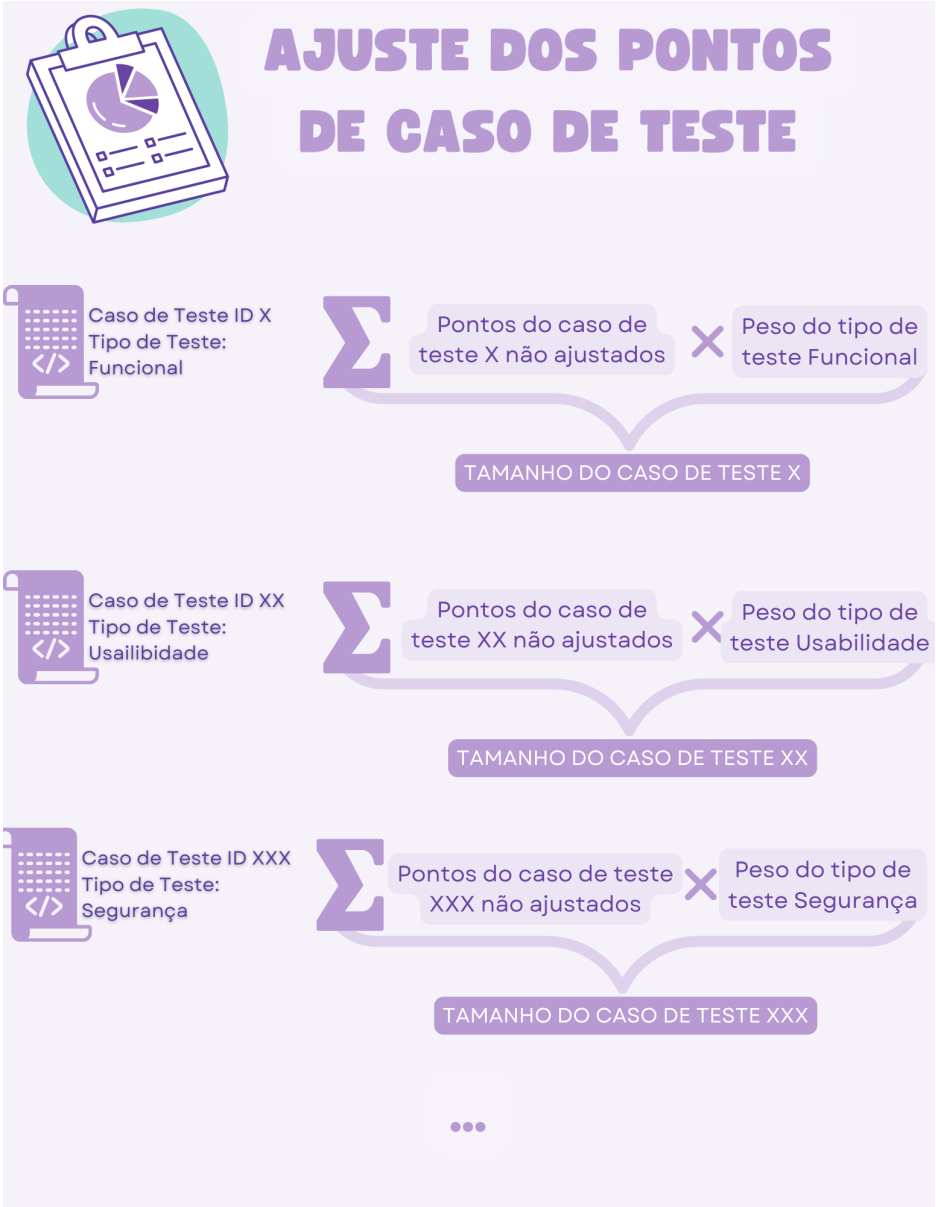
Durante a criação dos casos de testes, houve a escrita de CTs (Caso de Teste) que ainda não possuíam a funcionalidade já desenvolvida, contendo atualmente uma documentação para auxiliar nas regras de negócios e critérios de aceitação para o

desenvolvimento. Dessa maneira, durante a escrita foram sinalizados os casos de testes que ainda não eram possíveis de serem testados. E assim, eles também foram ocultados da aplicação da abordagem de estimativas.

Nesta primeira etapa foi avaliado cada caso de teste criado, levando em conta todos os parâmetros necessários para realizar a contagem utilizando a abordagem *qEstimation*. Para facilitar o cálculo, os casos de testes eram contados de acordo com o peso atribuído ao tipo de teste e a quantidade de pontos contidos no caso de teste. Na Seção 3.4 é descrita em detalhes como a operação dos dados é realizada.

Na Figura 12 é possível observar a forma em que a primeira etapa de aplicação do *qEstimation* é realizada, coletando todos os pontos de caso de teste não ajustados e o peso do tipo de teste, o resultado dado é fundamental para realização da segunda etapa.

Figura 12 - Ajuste dos pontos de caso de teste



Fonte: Elaborado pela autora (2022)

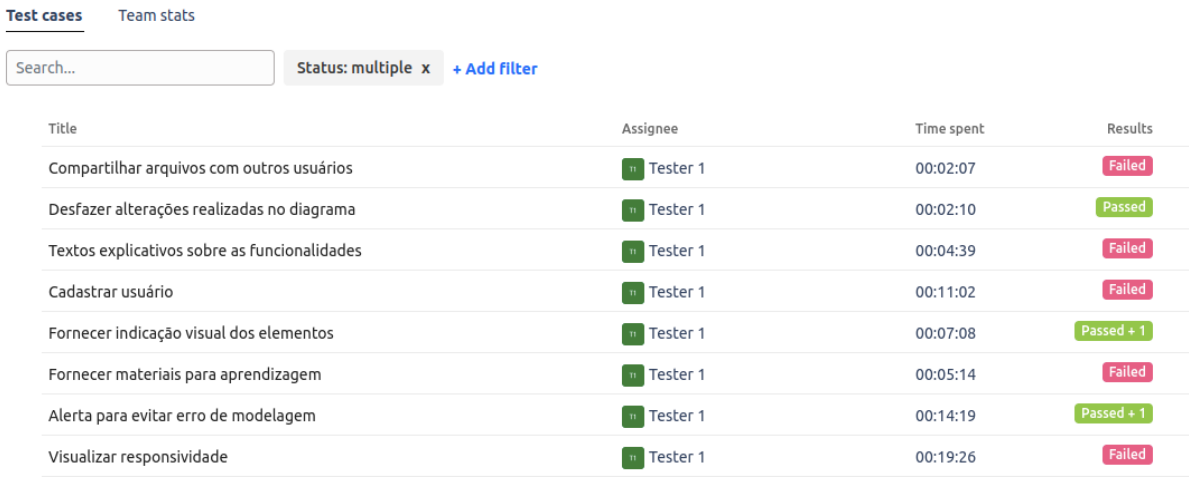
Na segunda etapa para medir o esforço, foi escolhido o índice de produtividade, já que o método de regressão linear necessitava de dados de ciclos de testes, e a iniciativa de testes é recente no projeto Usinn.

O experimento feito para coletar o índice de produtividade foi escolhido com base nas sugestões do trabalho de Nguyen *et al.* (2013), que consiste em escolher alguns testadores e alguns casos de teste (de diferentes níveis de complexidade e familiaridade) para serem executados. No projeto Usinn, toda a equipe de testadores executou casos de teste de

diferentes níveis, e com o auxílio da ferramenta *Qase*, foi possível coletar o tempo gasto em cada execução.

A Figura 13 abaixo mostra algumas informações que são possíveis de visualizar durante a execução de casos de testes na *Qase*, uma delas é o tempo gasto, que auxiliou a tornar realista o índice de produtividade.

Figura 13 - Tela de detalhes da execução de testes



Fonte: Qase (2022)

Para geração da estimativa, o experimento foi realizado antecipadamente a execução dos testes de regressão. Após a criação dos casos de testes, com o objetivo de obter o índice de produtividade, foram selecionados pela analista alguns casos de teste de diferentes níveis e executados pelos testadores. Desta forma, foi verificado a média de tempo que os testadores usaram para executar cada ponto de caso de teste, chegando a 11,88 minutos por TCP.

Por fim, para concluir a geração da estimativa de esforço de testes, o esforço é medido através de cada caso de teste, coletando os pontos de casos de teste ajustados (realizado na primeira etapa) e o índice de produtividade (realizado na segunda etapa). E assim, tem-se a estimativa total dos testes gerada pelo *qEstimation*.

6.6 Execução dos Testes de Regressão

Após o cálculo de geração da estimativa, era necessário iniciar os testes de regressão no software *USINN Modeler*, sendo uma fase muito importante deste estudo, pois além de aprimorar as experiências e conhecimentos em testes, foi possível conhecer mais sobre o sistema, visualizar os pontos mais críticos, inconsistências em funcionalidades, como também a inaugural e real prática de testes no *USINN Modeler*.

O teste de regressão é realizado após uma melhoria funcional ou reparo no programa. Seu objetivo é determinar se a mudança regrediu outros aspectos do programa. Geralmente é realizado por algum subconjunto dos casos de teste do programa (MYERS *et al*., 2004). Sommerville (2011) afirma que o teste de regressão envolve a execução de conjuntos de testes que tenham sido executados com sucesso, após as alterações serem feitas em um sistema. O teste de regressão verifica se essas mudanças não introduziram novos bugs no sistema e se o novo código interage com o código existente conforme o esperado. Como atualmente o time de desenvolvimento está desenvolvendo novas funcionalidades da *USINN Modeler*, os testes executados são classificados como testes de regressão.

Os testes de regressão foram guiados pelos casos de testes criados na etapa anterior. Elaborados e armazenados em um repositório na ferramenta de gerenciamento de testes: *Qase*, possibilitando neste mesmo ambiente criar o plano de testes para regressão e posteriormente executá-lo.

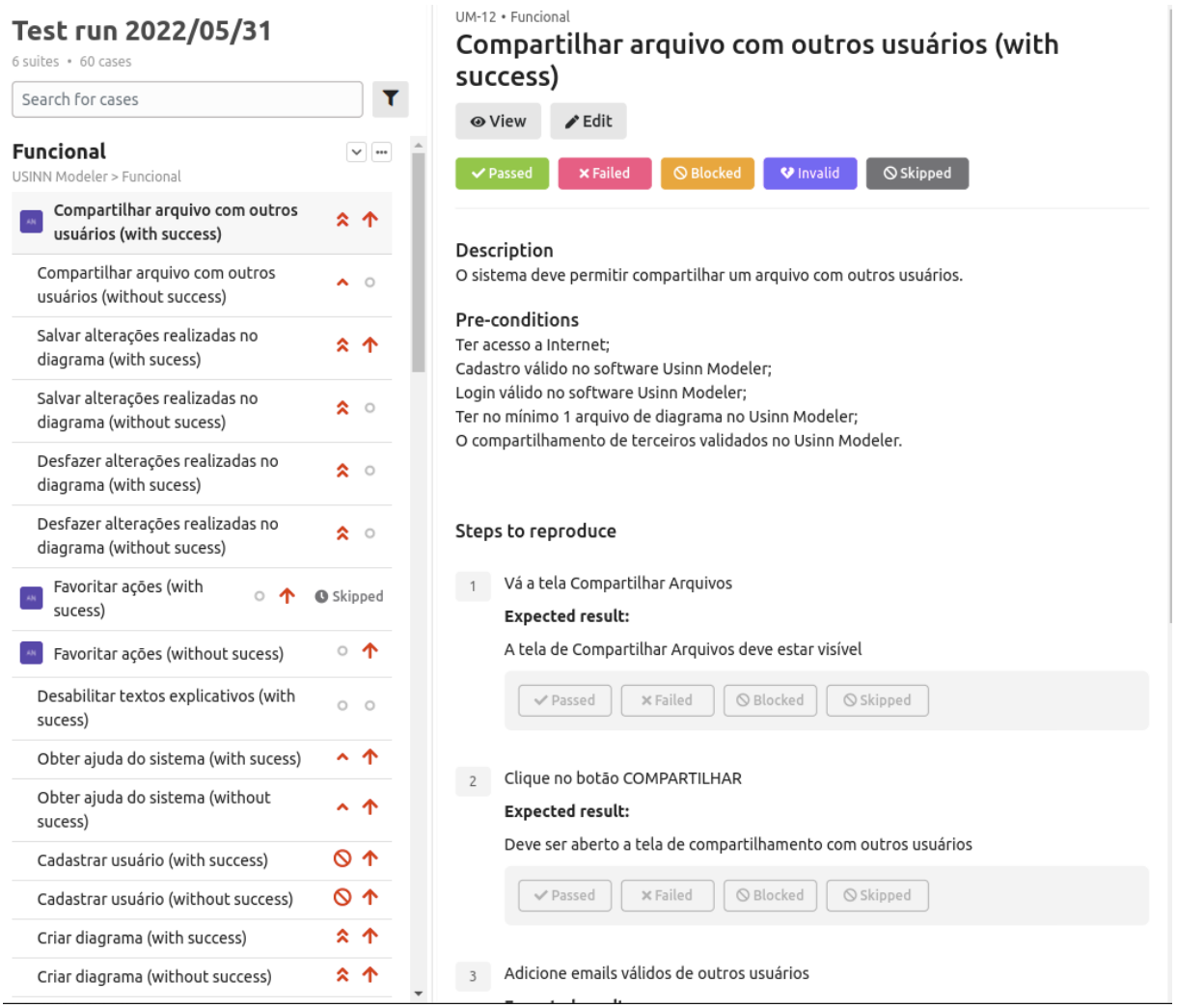
Apesar de utilizar a versão gratuita do *Qase* com algumas limitações, manifestou-se uma ferramenta eficiente para gerenciamento dos testes. Para execução dos testes de regressão, a ferramenta possibilitou a validação de cada *step* (passo a realizar) concluindo com a validação do caso de teste, no fim é possível visualizar graficamente os resultados da execução. A correção também é inserida nesta fase, visto que nesta etapa ocorre o melhor contexto para correção da escrita.

Como dito anteriormente, a versão gratuita da ferramenta possui algumas restrições, sendo uma delas a quantidade de usuários inseridos em um mesmo projeto, com limite de três usuários. Dessa maneira, foi dividido entre a equipe uma conta de acesso para cada dupla de testador, totalizando em duas contas, e a última conta de nível administrador do projeto de testes com a analista de testes e autora desta pesquisa.

Apesar da quantidade reduzida de contas, o trabalho e a produtividade dos testadores não foram prejudicadas, pois o *Qase* permite que uma mesma conta tenha acesso simultâneo em dispositivos diferentes. Sendo assim, foi possível realizar atividades com toda a equipe conjunta acessando a ferramenta.

Na Figura 14 a seguir é possível visualizar a etapa de execução dos testes de regressão seguindo casos de teste. A plataforma *Qase* traz recursos de avaliação para o teste, e também a opção de edição do próprio caso de teste, caso haja necessidade de correção.

Figura 14 - Tela de execução dos testes de regressão do USINN Modeler



Fonte: Qase (2022)

7. ANÁLISE DOS RESULTADOS

Este capítulo apresenta os resultados sobre as estimativas de esforço de testes gerados, resultados quantitativos da execução dos testes e *feedback* coletado com a equipe de testadores. Na análise dos resultados, foi investigado pontos que trouxessem respostas para as questões definidas por este trabalho.

7.1 Estimativa de esforço de testes gerada pelo qEstimation

Após a criação dos casos de testes, que é uma documentação necessária para utilizar a abordagem *qEstimation*, os dados foram coletados para gerar a estimativa. Mais detalhes sobre como a abordagem calcula a estimativa de testes, podem ser vistos na Seção 3.4.

Para gerar a estimativa pelo *qEstimation*, foram ignorados os casos de testes que ainda não são possíveis de serem testados, considerando somente as funcionalidades existentes no *USINN Modeler*.

A abordagem não leva em conta a quantidade de pessoas envolvidas, já que o cálculo de esforço corresponde à quantidade de homem-hora (Hh) para realizar atividades de testes. Sendo assim, a estimativa gerada corresponde ao valor de um testador, e para adaptar a realidade do projeto e dos integrantes que participam das atividades de teste, é necessário fragmentar a estimativa gerada pelo número de testadores.

Na Tabela 5 abaixo é possível visualizar os dados gerados pela estimativa e o esforço real gasto na primeira execução de testes do *USINN Modeler*.

Tabela 5 - Resultados da estimativa para o software USINN Modeler

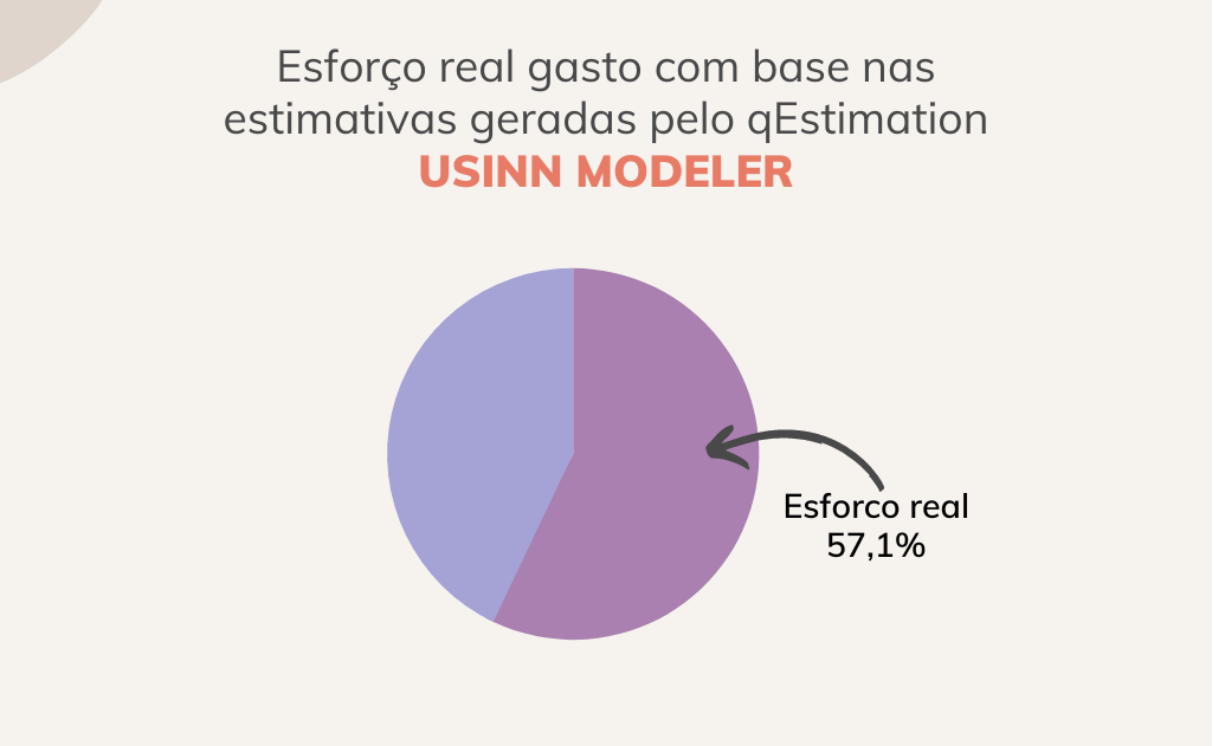
ID	Casos de testes	Pontos de Casos de Testes (TCP)	Produtividade Média (TCP/min)	Estimativa da abordagem (Hh)	Esforço gasto (Hh)	Desvio da abordagem (%)
EST01	41	140	11,88	12,78	7,29	28,05

Fonte: Elaborado pela autora (2022)

A abordagem indica que a produtividade sendo determinada por dados históricos ou ciclos de testes anteriores auxilia a favor de uma estimativa mais próxima do real, no contexto do *USINN Modeler* não era possível utilizar dados anteriores de testes para calibrar as estimativas, dessa forma foi-se utilizado um experimento para obter a produtividade média dos testadores.

Como visto anteriormente na Tabela 5, a estimativa fornecida pela abordagem ultrapassa o esforço realmente gasto na primeira estimativa gerada no *USINN Modeler*. A Figura 15 exibe o esforço real gasto com base nas estimativas geradas pelo processo *qEstimation*.

Figura 15 - Esforço gasto com base nas estimativas geradas



Fonte: Elaborado pela autora (2022)

A analista de testes e escritora deste estudo analisou o contexto do projeto em relação à abordagem escolhida e executada. Foi identificado alguns pontos que poderiam melhorar a calibragem da estimativa de esforço de testes em outras rodagens. Com base nos estudos de Nguyen *et al.* (2013) e Sousa et al. (2018) que também aplicaram o *qEstimation*,

foram selecionados pontos que influenciaram e podem ser melhorados em outras execuções da estimativa:

- Índice de produtividade sendo medido através do experimento: apesar de ser uma alternativa indicada pela abordagem quando não há dados históricos de testes sobre o software. Não é o mais favorável para resultar em uma estimativa próxima do real, diferente de ciclos de testes anteriores que trazem dados vigentes.
- Criação dos casos de testes e execução dos testes de regressão realizado pela mesma equipe de testadores: para realizar a estimativa com o *qEstimation*, foi necessário anteriormente iniciar um processo de testes, formado por várias etapas, sendo uma delas a escrita dos casos de testes, em que os testadores obtiveram conhecimento sobre a documentação do projeto e detalharam em cada CT os passos a realizar tal teste. Conclui-se que a criação dos CTs também foi um ponto que facilitou a execução dos testes, otimizando o tempo gasto durante os testes.
- Estimativa tendenciosa a ultrapassar o esforço realmente gasto: por meio de uma visão geral, como também observado no estudo de Sousa et al. (2018), notou-se que a estimativa tende a apresentar resultados que ultrapassem o esforço realmente gasto, mesmo que seja em desvios menores.

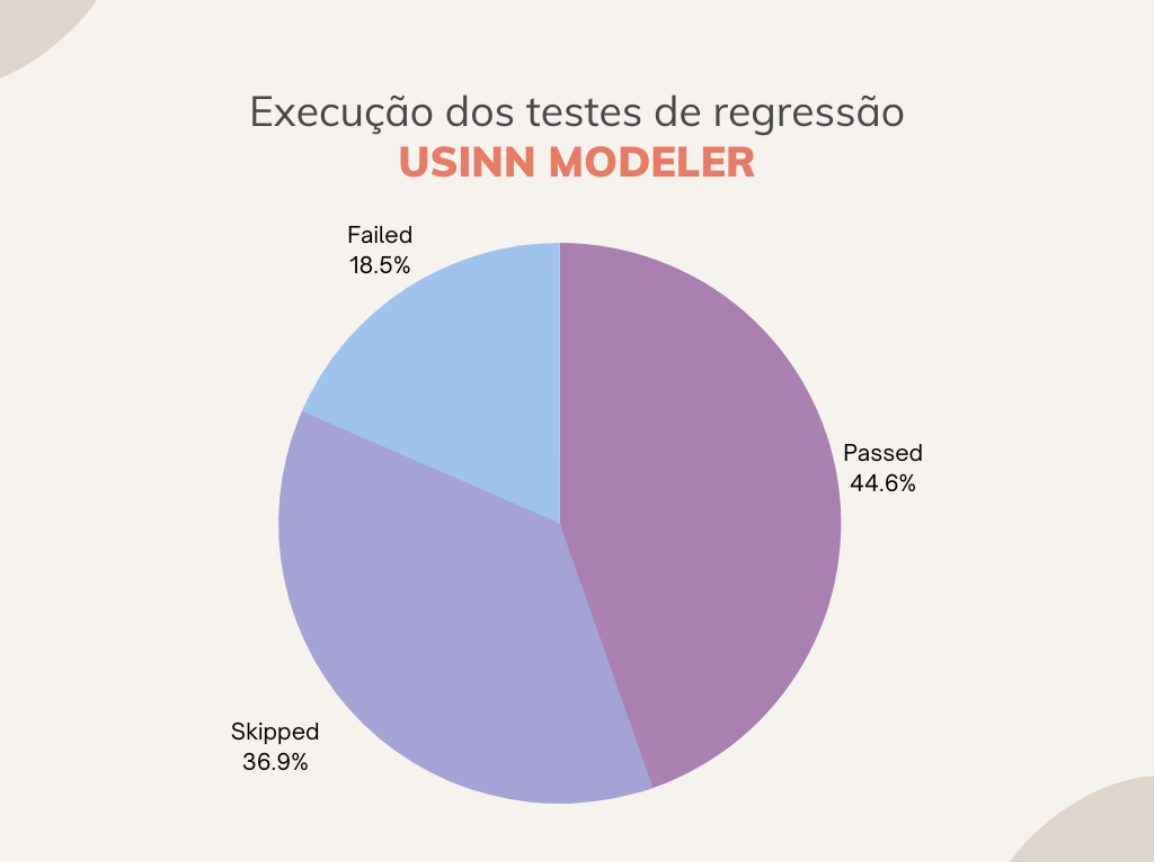
A partir dos dados e informações citados anteriormente, foi possível responder a questão de pesquisa Q4 (A estimativa gerada trouxe dados semelhantes ao executado no ambiente real?). Foi observado que a estimativa gerada pela abordagem não trouxe dados próximos aos executados no ambiente real, como é possível visualizar na Tabela 5. Dessa forma, após a análise do contexto do projeto em relação ao *qEstimation*, foram citados três pontos que influenciaram no resultado da estimativa: índice de produtividade sendo medido através do experimento, criação dos casos de testes e execução dos testes de regressão realizado pela mesma equipe de testadores e estimativa tendenciosa a ultrapassar o esforço realmente gasto.

7.2 Resultados quantitativos da execução dos testes de regressão

Foram criados 65 casos de testes, de diferentes tipos de testes: funcional, usabilidade, segurança, compatibilidade, API e rede. Na Figura 16 é possível visualizar toda a execução dos casos de testes. Foi obtido três situações de resultados da regressão:

- *Passed:* testes realizados com sucesso no *USINN Modeler* seguindo os passos dos casos de testes.
- *Skipped:* testes ignorados, pelo fato de não terem funcionalidades desenvolvidas dos respectivos casos de testes.
- *Failed:* testes realizados sem sucesso no *USINN Modeler* seguindo os passos dos casos de testes.

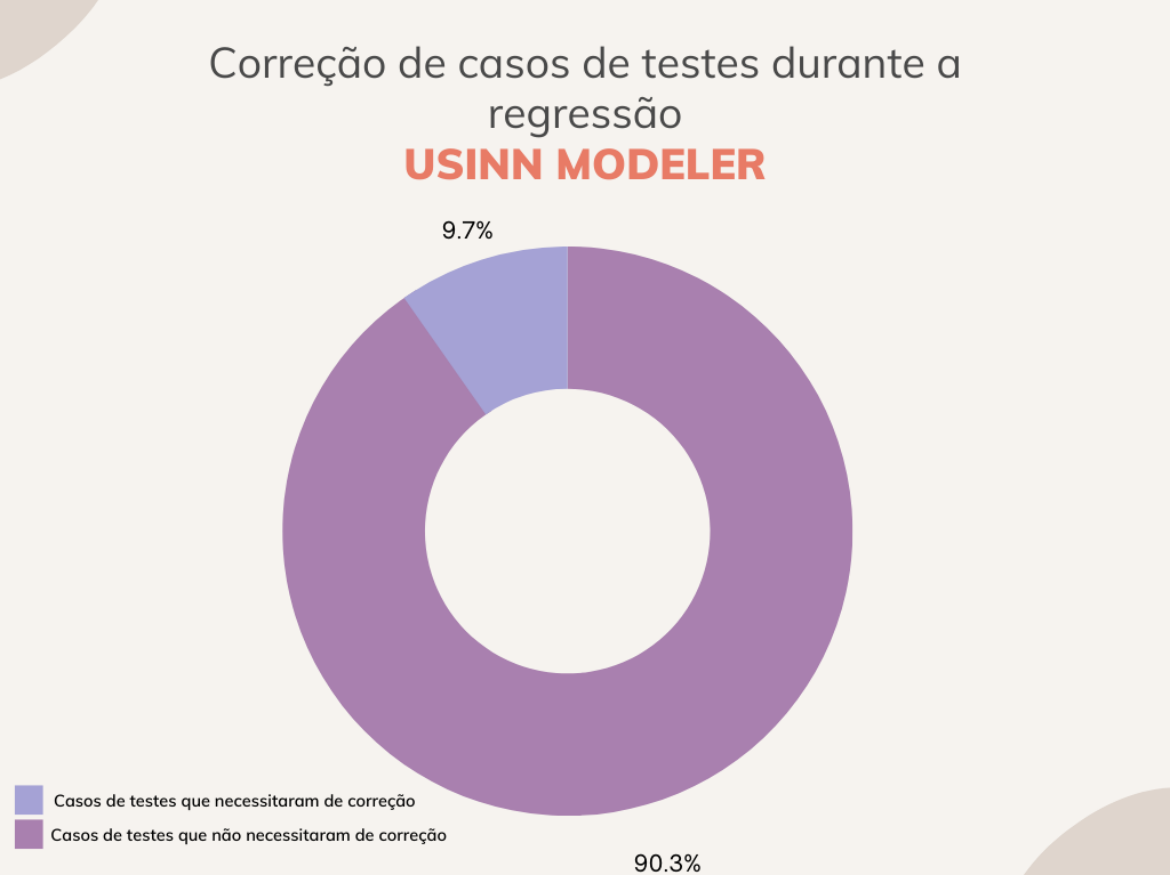
Figura 16 - Resultado dos testes de regressão do USINN Modeler



Fonte: Elaborado pela autora (2022)

Durante a regressão também houve a necessidade de correção de CTs, o percentual foi baixo, indicando que a escrita realizada anteriormente foi assertiva. Abaixo é possível ver a Figura 17 que mostra o percentual de correção necessário durante os testes.

Figura 17 - Percentual de correção de casos de testes durante a regressão



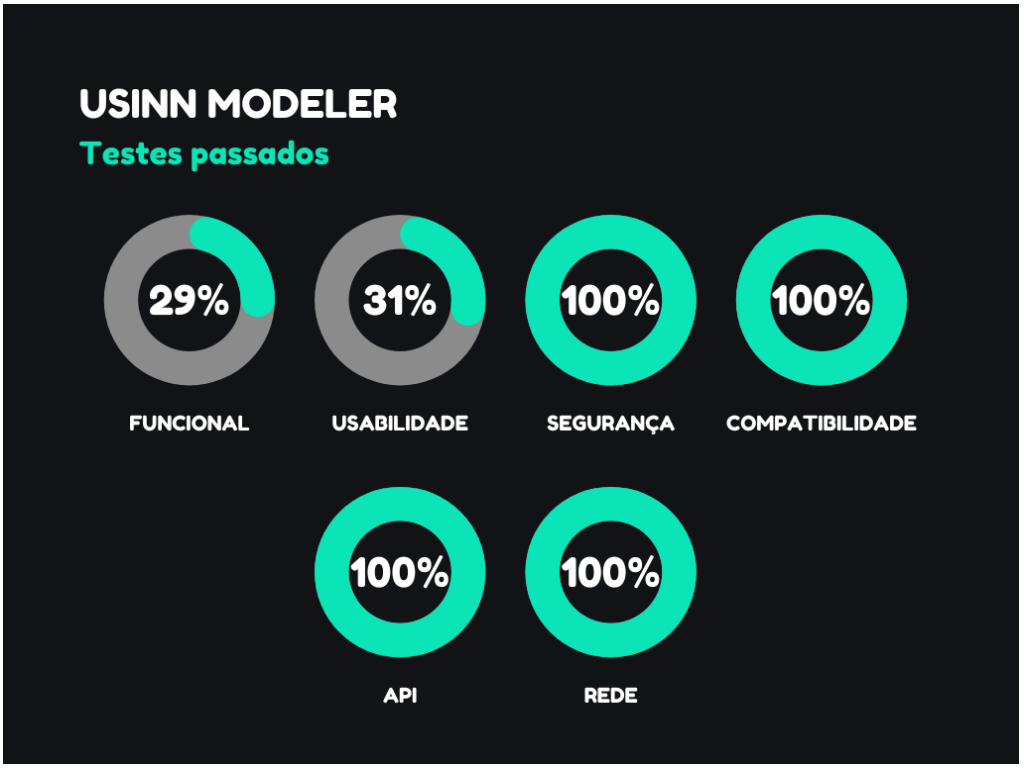
Fonte: Elaborado pela autora (2022)

Sempre que necessário corrigir algum caso de teste, seja por motivo de incoerência, passo incompleto, entre outras situações, o testador já tinha a possibilidade de realizar a correção na mesma tela de execução no *Qase*. Após as alterações é possível voltar ao respectivo caso de teste e continuar a execução. Os resultados obtidos pelo índice de correção são otimistas, já que durante a execução dos testes foi configurado para que um mesmo testador não executasse os mesmos CTs criados por ele anteriormente. Evitando vícios durante a rodagem dos testes e que erros despercebidos passassem.

Dentre os dados coletados, foram filtrados os percentuais de testes passados, falhos e ignorados durante a abordagem.

A Figura 18 mostra o percentual de testes passados durante os testes de regressão no software *USINN Modeler*, sendo os testes do tipo: segurança, compatibilidade, API e rede concluídos com o índice total de aceitação.

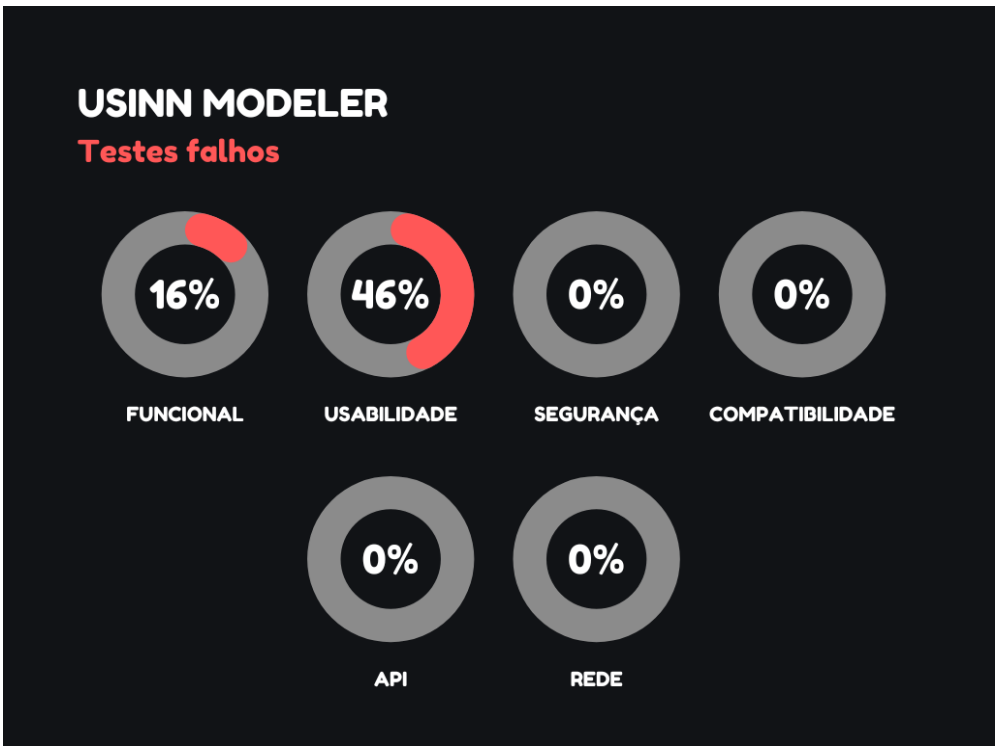
Figura 18 - Percentual de testes passados durante a regressão



Fonte: Elaborado pela autora (2022)

A Figura 19 exibe a porcentagem de testes falhos durante a regressão, que foram encontrados apenas nos tipos de testes funcional e de usabilidade.

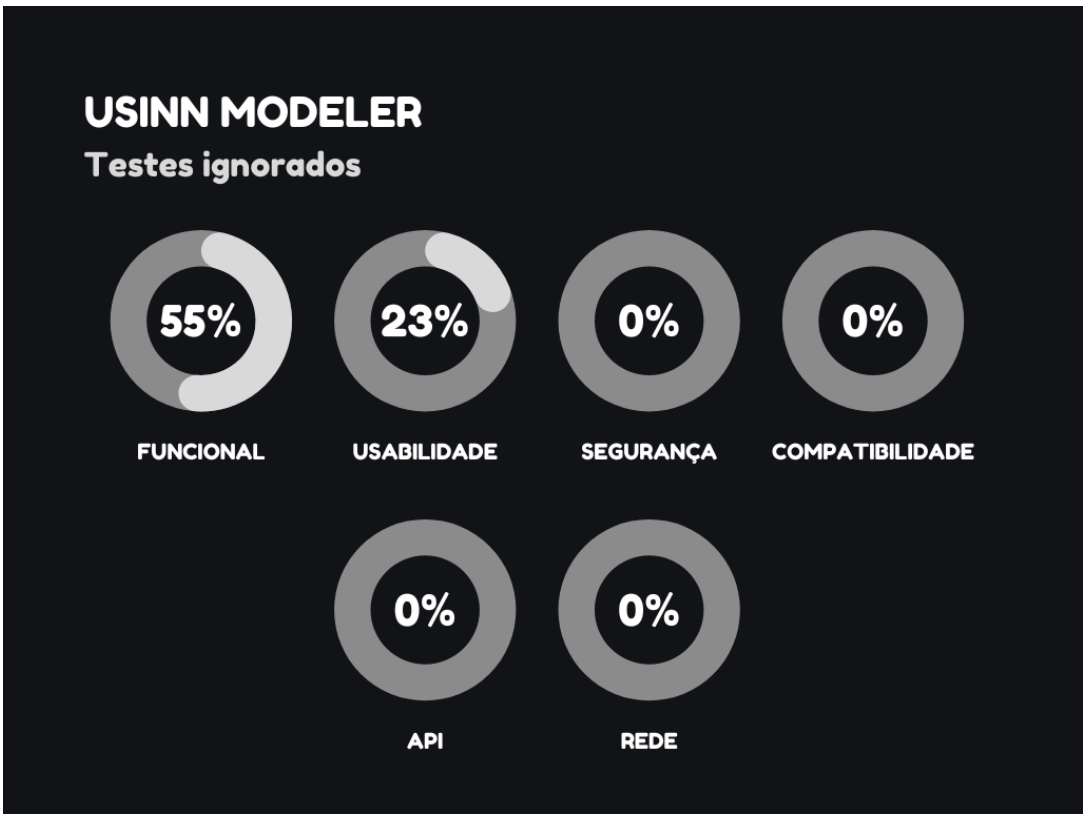
Figura 19 - Percentual de testes falhos durante a regressão



Fonte: Elaborado pela autora (2022)

A Figura 20 apresenta a margem de testes ignorada, levando em conta a documentação de casos de teste criada, porém sem funcionalidade desenvolvida no software. As únicas categorias que tiveram testes ignorados foram a funcional e a de usabilidade.

Figura 20 - Percentual de testes ignorados durante a regressão

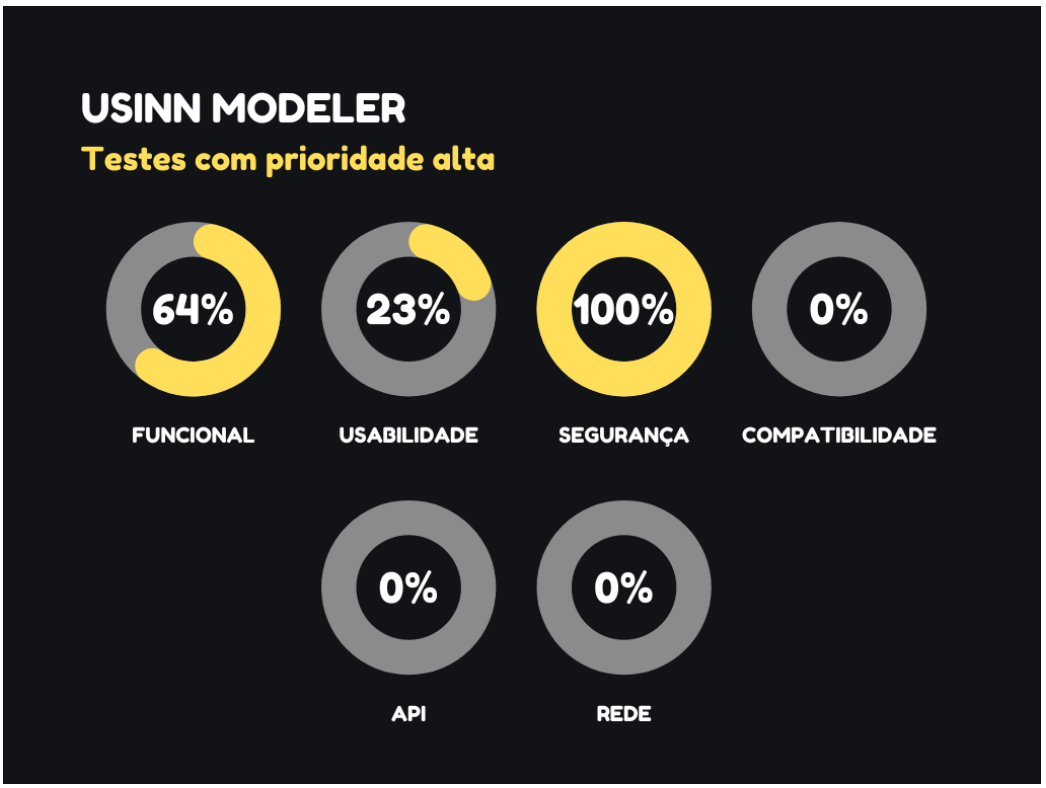


Fonte: Elaborado pela autora (2022)

Por fim, serão apresentadas a quantidade de testes que possuem prioridade alta, média e baixa de acordo com tipo de teste. Os percentuais mostrados abaixo, levam em conta todo o repositório de casos de teste do USINN Modeler, considerando até os casos de testes que foram ignorados durante a regressão.

Na Figura 21 é apresentado o nível percentual de testes com prioridade alta, ou seja, os que possuem maior necessidade de serem testados. Sendo os testes do tipo segurança com maior taxa de prioridade.

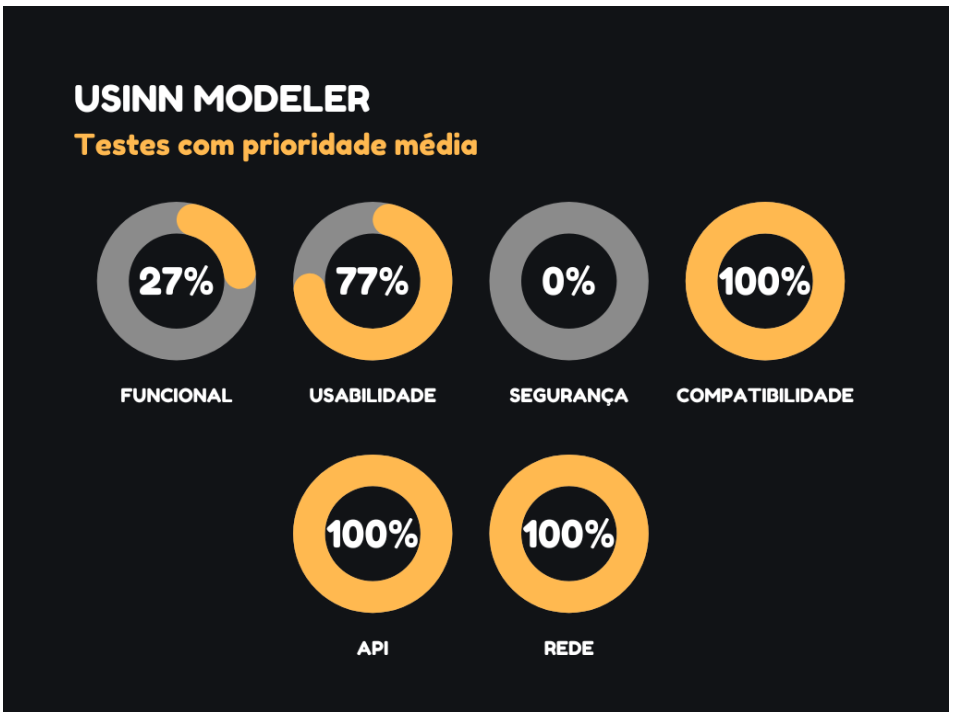
Figura 21 - Percentual de testes com prioridade alta



Fonte: Elaborado pela autora (2022)

Na Figura 22 é possível visualizar a porcentagem de testes com prioridade média, que são testes importantes de serem executados, mas não sendo os mais prioritários. No nível médio, os tipos de teste de compatibilidade, API e rede tem índice total de prioridade média.

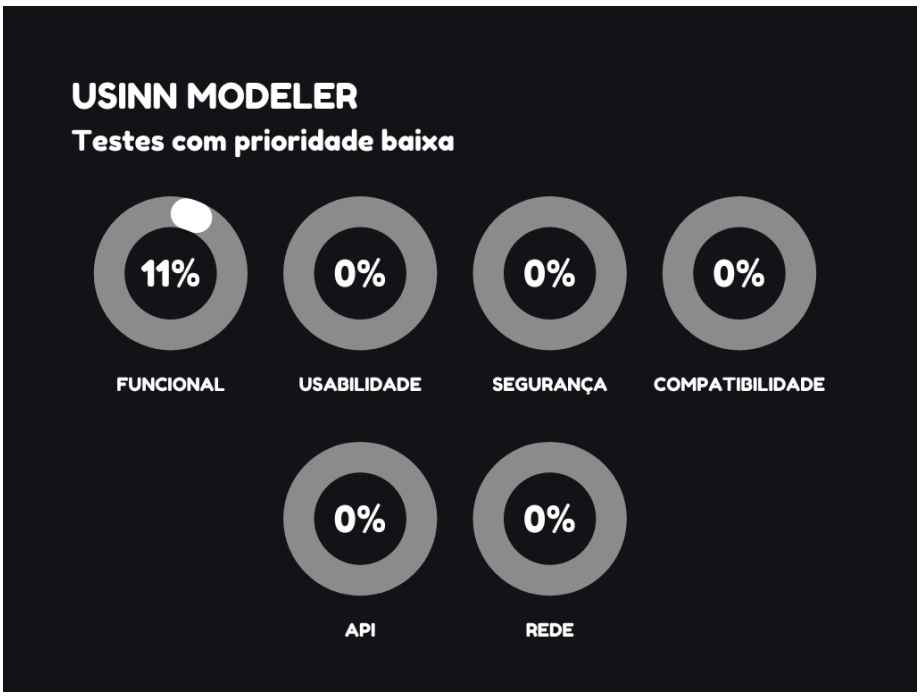
Figura 22 - Percentual de testes com prioridade média



Fonte: Elaborado pela autora (2022)

A Figura 23 ilustra os testes com menor prioridade na fila de execução, que são os testes de prioridade baixa, no *USINN Modeler* apenas um pequeno percentual de testes do tipo funcional estão inseridos nesta categoria.

Figura 23 - Percentual de testes com prioridade baixa



Fonte: Elaborado pela autora (2022)

Com base no que foi apresentado nesta subseção, foi possível responder a questão de pesquisa Q3 (A iniciativa de abordagem de testes no *USINN Modeler* trouxe vantagens para o mapeamento de inconsistências e melhorias para a ferramenta?), que através do processo de testes foi possível mapear inconsistências, que são os testes falhos na execução, como mostra a Figura 19. E como melhoria, tem-se o levantamento da prioridade de cada caso de teste, como mostra as Figuras 21, 22, e 23, além dos casos de testes criados até mesmo sem funcionalidades desenvolvidas, sendo exibidos na Figura 16.

Para responder a questão de pesquisa Q5 (O processo de estimativa de esforço de teste contribuiu para melhoria/evolução da ferramenta?) é possível observar nesta subseção a contribuição que o processo de estimativa de esforço de teste proporcionou para o *USINN Modeler*, uma vez que, a partir da abordagem escolhida, foi necessário iniciar um processo de testes voltado para a ferramenta (*USINN Modeler)* que trouxe além da execução de testes, informações sobre a qualidade do software, como: testes com prioridade alta, média e baixa (Figuras 21, 22 e 23), porcentagem de testes passados, falhos e ignorados (Figuras 18, 19 e 20) e o percentual de correção dos CTs durante regressão (Figura 17).

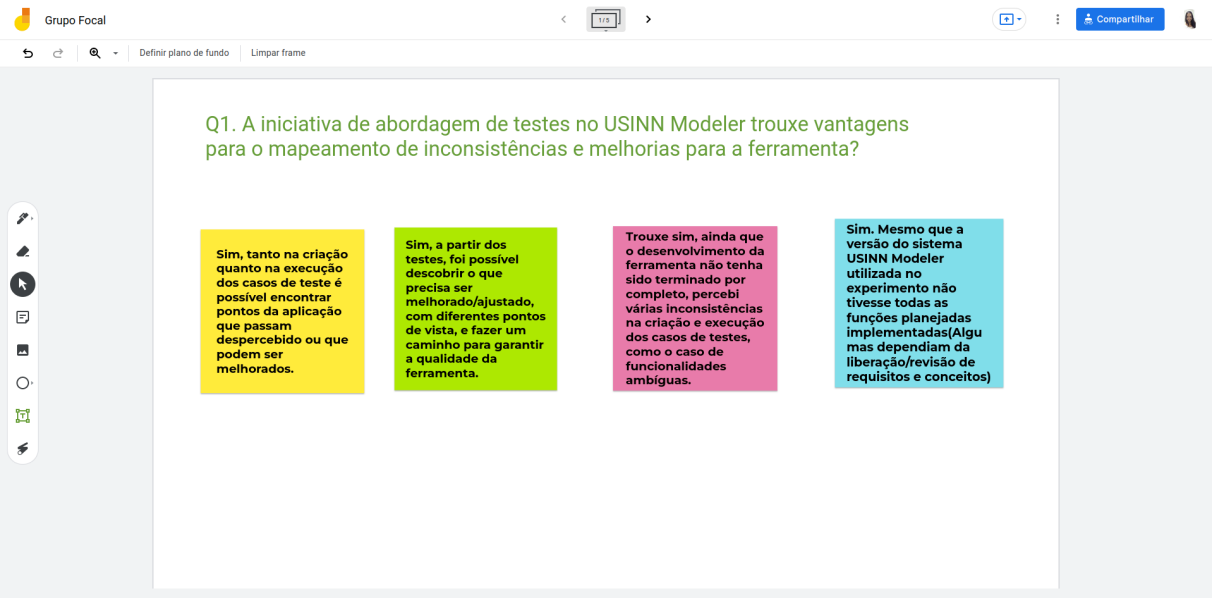
A questão de pesquisa Q3 terá sua resposta complementada a partir do *feedback* coletado com os testadores da equipe na subseção 7.3.

7.3 Feedback coletado com a equipe de testadores

Entre todas as etapas executadas, o *feedback* é fundamental para reunir as opiniões dos testadores, pessoas essas que estavam presentes executando atividades em quase todas as etapas deste estudo. O *feedback* foi obtido por meio da condução de um grupo focal após a execução dos testes, no formato remoto. A ferramenta utilizada para apresentação de perguntas e respostas foi o *Jamboard*.

A Figura 24 exibe a ferramenta, nela é possível criar cartões de respostas, compartilhar com vários usuários e criar quadros. Para o *feedback* foi utilizado um quadro para cada pergunta. Os questionamentos podem ser visualizados no Apêndice A e as respostas de cada participante podem ser vistas no Apêndice B

Figura 24 - Feedback com a equipe de testadores



Fonte: Jamboard (2022)

A Figura 25 abaixo apresenta o percentual levantado de acordo com o retorno coletado dos testadores. Toda equipe afirma ter a primeira experiência com testes após o ingresso no projeto Usinn, experiência essa que iniciou-se a partir da execução desta pesquisa. Todos os integrantes são estudantes de graduação da Universidade Federal do Ceará, Campus

de Russas, sendo metade estudantes do curso de Engenharia de Software e a outra metade estudantes de Ciência da Computação.

Figura 25 - Perfil dos testadores do projeto Usinn

Fonte: Elaborado pela autora (2022)

Outra informação coletada foi a quantidade de disciplinas cursadas que são relacionadas a testes ou qualidade de software, todos os participantes afirmam ter cursado pelo menos uma disciplina relacionada aos temas. Sendo mencionadas as disciplinas de Qualidade de Software, Verificação e Validação e Manutenção de Software, que são componentes curriculares obrigatórios e optativos dos cursos de Engenharia de Software e Ciência da Computação, respectivamente.

A partir dos dados citados anteriormente na Figura 24, percebe-se uma certa similaridade entre os perfis dos testadores, já que eles possuem níveis semelhantes de conhecimento técnico sobre teste de software, o que resulta numa estimativa justa gerada pelo *qEstimation*, que não leva em consideração a quantidade de pessoas envolvidas e o nível

técnico dos testadores. Sendo assim, as características da equipe foram benéficas para abordagem escolhida, evitando vícios relacionados ao desequilíbrio de nível técnico da equipe.

Tendo como base os estudos de Costa (2005) foi realizada uma entrevista de grupo focal com a equipe de testadores do projeto Usinn. Para Costa (2005) "O objetivo central do grupo focal é identificar percepções, sentimentos, atitudes e idéias dos participantes a respeito de um determinado assunto, produto ou atividade. Seus objetivos específicos variam de acordo com a abordagem de pesquisa".

A discussão durou cerca de uma hora e trinta minutos, tendo como moderadora a autora desta pesquisa, os participantes da entrevista foram informados sobre o objetivo geral da pesquisa e também sobre a não obrigatoriedade de responder todas as perguntas. A moderadora conduziu a reunião para promover a discussão entre os participantes, porém sem interferir na opinião dos participantes.

A entrevista de grupo focal foi composta por cinco perguntas relacionadas a experiência dos testadores com o processo de testes, após coletadas, as respostas comuns foram unidas, para melhor compreensão.

A Tabela 6 é preenchida pela pergunta de número um, que questiona os participantes da entrevista se há vantagens sobre a iniciativa da abordagem de testes no *USINN Modeler*, na opinião de todos os participantes, houve vantagens, que são citadas na Tabela abaixo. A partir desta, foi possível complementar a resposta da questão de pesquisa Q3 (A iniciativa de abordagem de testes no *USINN Modeler* trouxe vantagens para o mapeamento de inconsistências e melhorias para a ferramenta?) com a visão dos testadores.

Tabela 6 - Pergunta 01 da entrevista de grupo focal

Pergunta 01
A iniciativa de abordagem de testes no USINN Modeler trouxe vantagens para o mapeamento de inconsistências e melhorias para a ferramenta?
Participante 01, Participante 02 - Foi possível captar pontos da aplicação que passam despercebidos ou que podem ser melhorados. Participante 03 - Melhor visualização de funcionalidades ambíguas no software. Participante 04 - Notou-se a necessidade de melhoria da documentação existente, pois algumas funcionalidades planejadas ou implementadas dependem da revisão de requisitos.

Fonte: Elaborado pela autora (2022)

Na Tabela 7 é discutido sobre as funcionalidades mais difíceis de serem executadas, sendo respondida a questão de pesquisa Q2 (Quais são as funcionalidades mais difíceis de executar?), em que os testadores citam as funcionalidades mais difíceis que executaram, levando em conta a experiência durante os testes de regressão.

Tabela 7 - Pergunta 02 da entrevista de grupo focal

Pergunta 02
Quais são as funcionalidades mais difíceis de executar? (Necessitam de maior conhecimento de regras de negócio)
Participante 01 - Função de alerta para evitar erros de modelagem. Participante 02, Participante 04 - Ajuda ao usuário. Participante 03 - Tutorial.

Fonte: Elaborado pela autora (2022)

Já a Tabela 8 com a pergunta de número três, questiona se foi perceptível para os testadores a importância de um processo de testes, na percepção de todos, teve importância, onde são destacadas algumas etapas abaixo.

Tabela 8 - Pergunta 03 da entrevista de grupo focal

Pergunta 03
Foi perceptível a importância de um processo de testes em um software? Se sim, em qual das etapas?
Participante 01 - Durante todo o processo de desenvolvimento de software, para entrega de uma versão estável do sistema. Participante 02, Participante 03, Participante 04 - Durante os testes de regressão, na execução e análise do resultado do teste.

Fonte: Elaborado pela autora (2022)

Na Tabela 9 é discutido sobre as funcionalidades essenciais do sistema, sendo respondida a questão de pesquisa Q1 (Quais as funcionalidades essenciais do sistema?), em que os testadores citam as funcionalidades essenciais com base no projeto de testes criado.

Tabela 9 - Pergunta 04 da entrevista de grupo focal

Pergunta 04
Durante o processo de testes, foi possível identificar as funcionalidades essenciais do sistema?
Participante 01, Participante 02, Participante 03, Participante 04 - Cadastro e login do usuário. Participante 01, Participante 02, Participante 03 - Criar, editar e desfazer diagramas. Participante 03, Participante 04 - Ajuda ao usuário. Participante 03, Participante 04 - Tutorial.

Fonte: Elaborado pela autora (2022)

E a Tabela 10 com a pergunta cinco, discute sobre o que pode ser melhorado no processo de testes, no ponto de vista dos participantes.

Tabela 10 - Pergunta 05 da entrevista de grupo focal

Pergunta 05
O que poderia ser melhorado no processo de testes?
Participante 01 - Definição de um padrão de escrita mais completo para os casos de testes. Participante 02 - Processo automatizado durante a execução dos testes. Participante 03 - Documentação mais completa sobre o USINN Modeler. Participante 04 - Maior quantidade de integrantes na equipe de testes do projeto Usinn.

Fonte: Elaborado pela autora (2022)

Por fim, através do *feedback* foi possível entender a importância e a contribuição que a estimativa de esforço de testes e o processo de testes tiveram para o software *USINN Modeler* e para as pessoas envolvidas.

Para Costa (2005) o intuito do grupo focal é gerar novas ideias e estimular o pensamento do pesquisador. Dessa maneira, a subseção abaixo apresentará as lições aprendidas desta pesquisa, levando em conta também as perspectivas do time aqui coletadas.

7.4 Lições aprendidas

Nesta subseção serão apresentadas as contribuições desta pesquisa, como também as melhorias identificadas pela autora deste trabalho.

De início mencionando o planejamento, durante a condução deste estudo não houve alterações, seguindo precisamente o planejamento elaborado anteriormente, evidenciando que o mesmo foi assertivo para guiar a condução do estudo de caso.

As contribuições não se limitaram apenas para o *USINN Modeler*, como também englobou o grupo de pessoas envolvidos, que de início obtiveram capacitações que seriam necessárias para construir o estudo, realizaram a condução e no fim, visualizaram e identificaram pontos benéficos e a melhorar durante a execução do processo por inteiro.

Ainda que, o intuito principal seja investigar a utilidade de uma abordagem de testes de software aplicada ao *USINN Modeler*, o projeto Usinn adquiriu um projeto de casos de testes (que é uma documentação que auxilia não apenas para os testes, mas para o desenvolvimento como um todo), uma equipe de testadores capacitadas, dados de geração de estimativa, como também o primeiro ciclo de testes de regressão realizado. Contribuindo assim, para a evolução do projeto.

Falando sobre as melhorias identificadas, foi percebido pontos em que o *qEstimation* pode avançar, como é o caso da coleta e cálculo de dados realizada, que poderiam ser automatizados para torna-se mais viável utilizar a abordagem, por conta da grandiosidade geração de dados no decorrer da aplicação da abordagem.

Além disso, é necessário estudar a possibilidade de ajustar ou alterar os parâmetros que formam o experimento resultando no índice de produtividade, já que esse é um fator determinante na abordagem, e que afeta a estimativa gerada.

Outro ponto, é a ausência de geração de estimativas por testadores, apesar de não ter sido um gargalo neste trabalho, é importante pensar nessa questão, já que existem situações em que a equipe de testes possui níveis técnicos diferentes.

Também é necessário maior clareza sobre o processo de testes que deve ser criado para estar apto a realizar a estimativa de um software, pois demanda tempo da equipe, e também é importante a participação de um analista de testes para conduzi-lo.

A abordagem também se limita ao não ser capaz de estimar outras atividades de teste, como por exemplo, esforço gasto para reportar inconsistências do software.

Já falando sobre o processo de testes, também foram identificados alguns quesitos que podem ser melhorados: como a documentação própria do projeto, que possui frases em duplo sentido e termos não explicados. Outro quesito é a melhoria no padrão de escrita dos casos de testes, para reduzir ainda mais o nível de correção de CTs.

8. CONCLUSÕES E TRABALHOS FUTUROS

Este trabalho teve como objetivo principal investigar a utilidade de uma abordagem de testes de software no desenvolvimento de uma ferramenta de modelagem denominada *USINN Modeler*, com foco na estimativa de esforço para testes aplicando o processo *qEstimation*. Para condução e avaliação da pesquisa, foram seguidos os processos sugeridos no trabalho de Wohlin *et al*. (2000),

A execução foi realizada no projeto Usinn, especificamente com o time de testes, que participou de quase toda a execução da pesquisa, exceto durante a geração da estimativa, que ficou por conta da analista de testes responsável e autora deste trabalho. Após a execução, foi possível coletar e analisar os dados gerados através da estimativa e o esforço real gasto.

Nos resultados, a estimativa gerada pelo *qEstimation* ultrapassou 42,9% do esforço real utilizado, e assim foi possível identificar alguns pontos que podem melhorar a calibragem do *qEstimation* para gerar estimativas mais precisas.

Já nos testes de regressão, parte deles foram concluídos com sucesso, a segunda maior porção foram de testes ignorados e a menor porção, de testes falhos. O nível de correção foi mínimo, evidenciando a qualidade do projeto de casos de testes.

Como última etapa foi coletado o *feedback* da equipe de testes sobre o processo de testes realizados. Após esta última fase, foi possível identificar aspectos que necessitavam de melhorias, como os parâmetros que abordagem utiliza e que formam o experimento e resultam no índice de produtividade e melhorias que devem ser feitas na documentação do projeto.

Para trabalhos futuros pretende-se aplicar o *qEstimation* novamente seguido dos testes de regressão, e utilizando como índice de produtividade os dados de testes da rodagem efetuada neste trabalho, na tentativa de calibrar as estimativas geradas. Além disso, elaborar relatório de testes ao final da execução, para melhor visualização dos resultados do processo por inteiro. Incluir também testes com o usuário final, que são os testes de aceitação, para verificar a atuação do software. E por fim, iniciar um projeto de testes automatizados no *USINN Modeler,* utilizando também a abordagem *qEstimation* para estimar o esforço gasto.

REFERÊNCIAS

ANDRADE, Mayb. **Qualidade de software**. 2015, 1 ed. Rio de Janeiro, RJ: SESES, 2015.

ARRUDA DE MEDEIROS, A. C. **Método de Seleção de Casos de Teste de Regressão Baseado em Risco.** Trabalho de Conclusão de Curso — UNIVERSIDADE FEDERAL DE PERNAMBUCO: [s.n.].

BARTIÉ, ALEXANDRE. **Garantia da qualidade de software**. Rio de Janeiro: Elsevier, 2002.

BLUEMKE, I.; MALANOWSKA, A. 2021. **Software testing effort estimation and related issues: a systematic review of the literature**. ACM Computing Surveillance 54, 3, Article 53 (April 2022), 38 pages.

COSTA, A. F. F.; MARQUES, A. B. d. S. 2019. **USINN modeler: a web support tool for creating interaction and navigation models with USINN**. In Proceedings of the 18th Brazilian Symposium on Human Factors in Computing Systems. Association for Computing Machinery, New York, NY, USA, Article 64, 1–4.

COSTA, MARIA EUGÊNIA BELCZAK. **Grupo focal. Métodos e técnicas de pesquisa em comunicação.** São Paulo: Atlas, p. 180-192, 2005.

DHARMENDER SINGH KUSHWAHA; A. K. MISRA. 2008. **Software test effort estimation.** SIGSOFT Softw. Eng. Notes 33, 3, Article 6 (May 2008), 5 pages.

E. HANOĞLU; A. TARHAN, **An Empirical Study on the Relationship between Open Source Software Success and Test Effort**, 2019 4th International Conference on Computer Science and Engineering (UBMK), 2019, pp. 688-692.

EGE ADALI, O., ALPAY KARAGOZ, N., GUREL, Z., TAHIR, T., & GENCEL, C. (2017). **Software test effort estimation: State of the art in turkish software industry.** Paper presented at the Proceedings - 43rd Euromicro Conference on Software Engineering and Advanced Applications, SEAA 2017, 412-420.

G. J. MYERS, C. SANDLER, T. BADGETT E T. M. THOMAS. **The Art of Software Testing**. John Wiley & Sons, Nova York, NY, EUA, 2. ed., 2004.

IN DELAMARO, M. E., IN MALDONADO, J. C., & IN JINO, M. (2007). **Introdução ao teste de software.**

ISO / IEC 25010. **ISO / IEC 25010 : 2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models**. (2013).

MARQUES, A. B. d. S.; BARBOSA, S. D. J.; CONTE, T. 2017. **Evaluating the usability expressiveness of a USability-oriented INteraction and Navigation model**. In Proceedings

of the XVI Brazilian Symposium on Human Factors in Computing Systems (IHC 2017). Association for Computing Machinery, New York, NY, USA, Article 24, 1–10.

MARQUES, A. B.; CONTE. T.; DINIZ, S. 2016. **Representing the interaction and navigation of interactive systems through a usability-oriented model: A feasibility study.** In Proceedings of the 15th Brazilian Symposium on Human Factors in Computing Systems. Association for Computing Machinery, New York, NY, USA, Article 15, 1–10.

NGUYEN, V.; PHAM, V.; LAM, V. 2013. **QEstimation: a process for estimating size and effort of software testing**. In Proceedings of the 2013 International Conference on Software and System Process. Association for Computing Machinery, New York, NY, USA, 20–28.

PRESSMAN, R. **Engenharia de software-uma abordagem profissional**. 7ª edição. São Paulo, 2006.

RUNESON, PER; HÖST, MARTIN. **Guidelines for conducting and reporting case study research in software engineering.** Empirical software engineering, v. 14, n. 2, p. 131-164, 2009.

SANTOS, E. B. d.; COSTA, L S. d.; et al. 2019. **Extraction of test cases procedures from textual use cases to reduce test effort: Test Factory Experience Report.** In Proceedings of the XVIII Brazilian Symposium on Software Quality. Association for Computing Machinery, New York, NY, USA, 266–275.

SHARMA, Mukesh. **Software Testing 2020: Preparing for New Roles**. CRC Press, 2016.

SOMMERVILLE, IAN. **Engenharia de Software**. 7. ed. São Paulo: Pearson Education, 2011.

SOUSA, A. O. d.; SANTOS, I. d. S.; ARAGÃO, B. S.; ANDRADE, R. M. d. C. 2018. **Towards an automatic approach to estimating test effort: An Experience Report**. In Proceedings of the 17th Brazilian Symposium on Software Quality (SBQS). Association for Computing Machinery, New York, NY, USA, 305–314.

WIERINGA, ROEL. **Design science as nested problem solving.** Proceedings of the 4th international conference on design science research in information systems and technology. 2009.

WOHLIN, C., RUNESON, P., HOST, M., OHLSSON, M., REGNELL, B., WESSLÉN, A., **Experimentation in Software Engineering: an introduction**, Kluwer Academic Publishers, USA, 2000.

APÊNDICE A - QUESTIONAMENTOS UTILIZADOS

Os questionamentos apresentados no Apêndice A foram produzidos com base no trabalho de Costa (2005).

	Perguntas
Pergunta 01	A iniciativa de abordagem de testes no USINN Modeler trouxe vantagens para o mapeamento de inconsistências e melhorias para a ferramenta?
Pergunta 02	Quais são as funcionalidades mais difíceis de executar? (Necessitam de maior conhecimento de regras de negócio)
Pergunta 03	Foi perceptível a importância de um processo de testes em um software? Se sim, em qual das etapas?
Pergunta 04	Durante o processo de testes, foi possível identificar as funcionalidades essenciais do sistema?
Pergunta 05	O que poderia ser melhorado no processo de testes?

APÊNDICE B - DADOS COLETADOS

	Pergunta 01
Participante 01	Sim, tanto na criação quanto na execução dos casos de teste é possível encontrar pontos da aplicação que passam despercebidos ou que podem ser melhorados.
Participante 02	Sim, a partir dos testes, foi possível descobrir o que precisa ser melhorado/ajustado, com diferentes pontos de vista, e fazer um caminho para garantir a qualidade da ferramenta.
Participante 03	Trouxe sim, ainda que o desenvolvimento da ferramenta não tenha sido terminado por completo, percebi várias inconsistências na criação e execução dos casos de testes, como o caso de funcionalidades ambíguas.
Participante 04	Sim. Mesmo que a versão do sistema USINN Modeler utilizada no experimento não tivesse todas as funções planejadas implementadas(Algumas dependiam da liberação/revisão de requisitos e conceitos) o método usado refletiu sobre a equipe uma revisão de ideias para a execução do mesmo.

	Pergunta 02
Participante 01	A função de alerta para evitar erros de modelagem, uma vez que deve ter conhecimento pleno do modelo USINN para se obter um teste efetivo.
Participante 02	A parte de Ajuda ao usuário, pois ou não possuía um botão para que a ação fosse realizada ou o botão não funcionava
Participante 03	A funcionalidade de ajuda e tutorial.
Participante 04	As funcionalidades que remetiam a própria experiência do usuário. A respeito de suporte, orientação e acesso inicial.

Pergunta 03	
Participante 01	Sim, no desenvolvimento com a primeira versão estável do sistema, se torna importante os testes para entregar um bom produto e com o mínimo de defeitos.
Participante 02	Sim, principalmente na etapa de executar e analisar o resultado do teste.
Participante 03	Sim, em todo o processo do desenvolvimento de software, pois é fundamental para garantir a qualidade do software. Pois todos os testes são essenciais para garantir isso.
Participante 04	Sim. A etapa de testes de regressão, no qual os testadores realizam a verificação completa e direcionada de cada teste. Especificando com atenção qual case deve estar sendo seguida corretamente seguindo suas particularidades.

Pergunta 04	
Participante 01	Sim, ao testar com a essência do usuário é possível pensar e identificar essas funções. Para tal ferramenta, são elas: Criar um diagrama, Salvar um diagrama e Obter ajuda/manual.
Participante 02	Sim, cadastro/login de usuário, criação/edição de diagramas, o processo, foco na usabilidade...
Participante 03	Sim, cadastro de usuário, compartilhar diagrama, criar diagrama, alterar diagrama, desfazer alteração do diagrama, Login, tutorial e ajuda.
Participante 04	Sim, ademais as funcionalidades 'padrão' de cada sistema(Login, CRUD 's Básicos e entre outros) as funcionalidades que dão suporte

	ao usuário também são de alta importância dado que o sistema USINN Modeler é precisamente interativo.

Pergunta 05	
Participante 01	A definição e aplicação de um padrão de escrita dos casos de teste melhoraria sua manutenção e visualização. Como Título, tempo verbal, etc.
Participante 02	A parte de execução dos testes, acho que o processo automatizado seria um ganho, pois seria consumido menos tempo para realizar as execuções.
Participante 03	Para melhorar o processo, para mim seria necessário uma documentação mais completa do sistema.
Participante 04	O processo de teste possivelmente obteria um aceleramento na produção dos cases caso fosse realizado numa equipe de grande porte, pois, mesmo com conhecimento nulo sobre a plataforma não houveram empecilhos no planejamento e especificação de cada case.