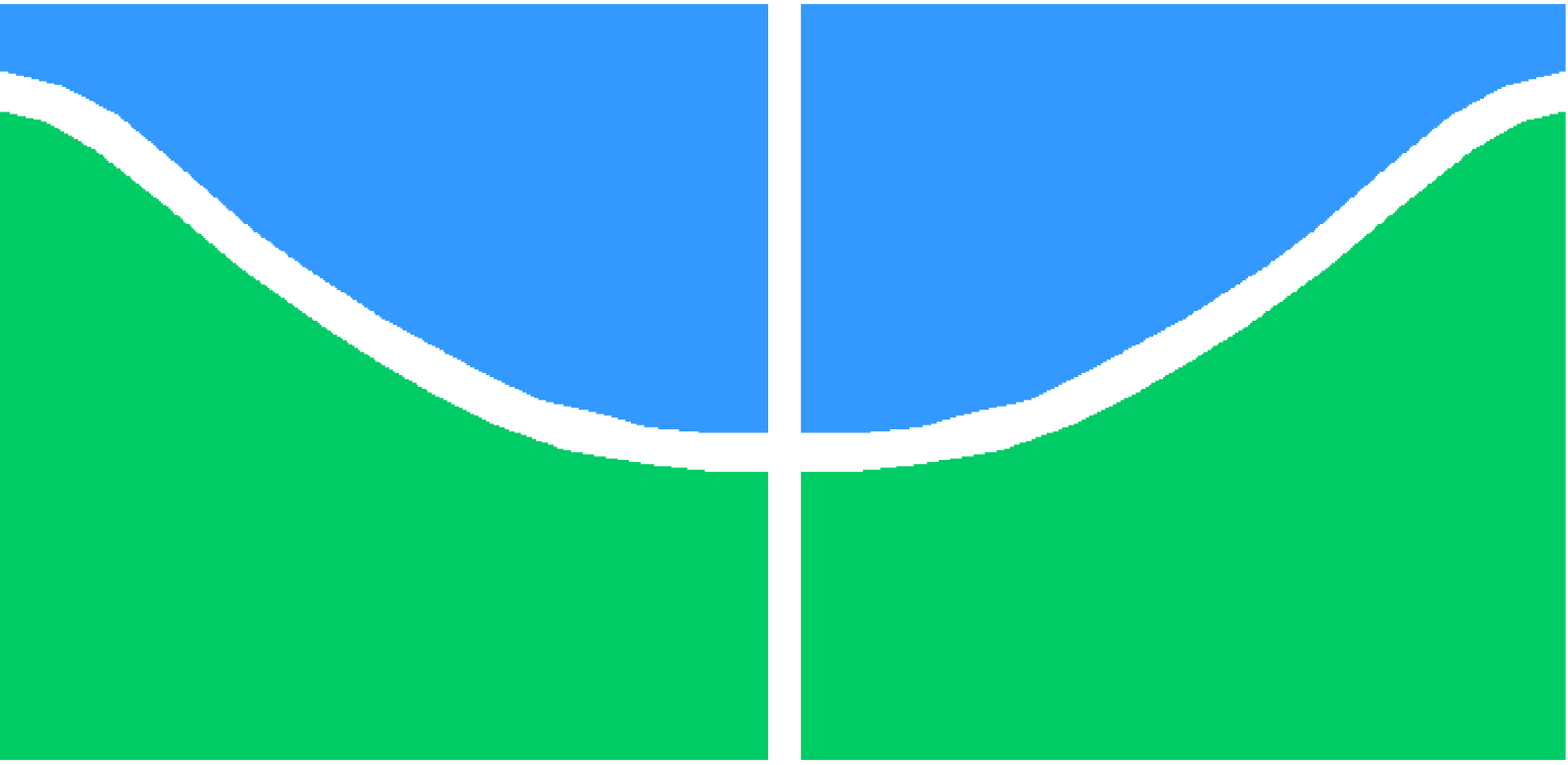

Universidade de Brasília - UnB
Faculdade UnB Gama - FGA
Engenharia de Software

O uso de testes exploratórios: uma aplicação prática

Autor: Iolane Caroline Alves de Andrade
Orientador: Prof. Msc. Ricardo Ajax Dias Kosloski

Brasília, DF
2020

Iolane Caroline Alves de Andrade

O uso de testes exploratórios: uma aplicação prática

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

Universidade de Brasília - UnB

Faculdade UnB Gama - FGA

Orientador: Prof. Msc. Ricardo Ajax Dias Kosloski

Coorientador: Prof. Msc. Rafael Fazzolino

Brasília, DF

2020

Iolane Caroline Alves de Andrade
O uso de testes exploratórios: uma aplicação prática/ Iolane Caroline Alves de Andrade. – Brasília, DF, 2020-
121 p. : il. (algumas color.) ; 30 cm.

Orientador: Prof. Msc. Ricardo Ajax Dias Kosloski

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

1. Qualidade de produto de software. 2. Teste exploratório de software. I. Prof. Msc. Ricardo Ajax Dias Kosloski. II. Universidade de Brasília. III. Faculdade UnB Gama. IV. O uso de testes exploratórios: uma aplicação prática

CDU 02:141:005.6

Iolane Caroline Alves de Andrade

O uso de testes exploratórios: uma aplicação prática

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

Trabalho aprovado. Brasília, DF, 15 de dezembro de 2020:

Prof. Msc. Ricardo Ajax Dias Kosloski
Orientador

Prof. Msc. Rafael Fazzolino
Co-Orientador

Prof.ª Msc. Cristiane Ramos
Convidada

Brasília, DF
2020

Este trabalho é dedicado especialmente aos meus pais. Minha mãe Iolanda e meu pai José Roberto, que sempre trabalharam muito e se sacrificaram diversas vezes para me proporcionar o privilégio de conquistar um diploma de Ensino Superior. Amo vocês!

Agradecimentos

Agradeço primeiramente à Deus por estar sempre cuidando de mim. Agradeço aos meus pais que sempre me apoiaram e me guiaram pelos caminhos da educação. Agraço também aos meus orientadores, que com toda empatia, zelo e amor por ensinar, me auxiliaram na construção deste Trabalho de Conclusão de Curso. Agradeço também a todos meus professores que, ao longo da vida, me auxiliaram na minha jornada pelo conhecimento. Cada um de vocês faz parte dessa conquista. Obrigada!

“ O tempo muito me ensinou: Ensinou a amar a vida, não desistir de lutar, renascer na derrota, renunciar às palavras e pensamentos negativos, acreditar nos valores humanos, e a ser otimista. Aprendi que mais vale tentar do que recuar ... Antes acreditar do que duvidar, que o que vale na vida, não é o ponto de partida e sim a nossa caminhada“.

(Cora Coralina)

Resumo

Este Trabalho de Conclusão de Curso aborda a avaliação da qualidade do produto de software do ponto de vista funcional. Visando realizar uma avaliação a nível de sistema e sob o ponto de vista funcional, serão utilizados testes exploratórios de software e testes tradicionais de software. O teste exploratório, por sua vez, é um tipo de teste vantajoso em relação a aprendizagem, pois possibilita ao testador aprender sobre o sistema enquanto realiza o design e a execução dos mesmos. Outras vantagens apontadas estão relacionadas aos contextos onde eles se encaixam. Os teste exploratórios de software são bem aceitos onde é necessário *feedback* ou aprendizagem rápida do produto, outra vantagem é a maior diversidade de testes aplicados, a construção de testes a partir da perspectiva do usuário, entre outras. Para tanto, foi selecionado para o estudo empírico o sistema PGTBL, que é uma aplicação que automatiza as atividades da metodologia TBL(do inglês *Team Based Learning*) ou Aprendizado Baseado em Equipe.

Palavras-chaves: qualidade de software, testes exploratório, qualidade, verificação, validação, medição.

Abstract

This monography addresses the evaluation of software product quality from a functional point of view. Aiming to perform an assessment at the system level and from a functional point of view, exploratory software tests and traditional software tests will be used. The exploratory test, in turn, is a type of advantageous test in relation to learning, as it allows the tester to learn about the system while carrying out its design and execution. Other advantages pointed out are related to the contexts in which they fit. Exploratory software tests are well accepted where textit feedback or quick product learning is needed, another advantage is the greater diversity of tests applied, the construction of tests from the user's perspective, among others. Therefore, the PGTBL system was selected for an empirical study, which is an application that automates the activities of the TBL methodology (from English textit Team Based Learning) or Team Based Learning.

Key-words: software quality, exploratory testing, quality, verification, validation, measurement.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

TCC	Trabalho de Conclusão de Curso
ET	Teste Exploratório
US	*User Store* ou história de usuário
TBL	*Team Based Learning* – Aprendizado Baseado em Equipe.
GQM	Goal-Question-Metric.
IHC	Interação Humano Computador
PGTBL	Programa TBL
EP	Épics
FE	Features
US	User story

Sumário

1 INTRODUÇÃO

1.1 Contextualização

Na Era Digital, a busca pela automatização de processos, outrora desempenhados manualmente, e a troca de informações em volumes cada vez maiores, demandam o desenvolvimento de softwares que supram essas necessidades da sociedade moderna.

O desenvolvimento de software geralmente pode seguir uma metodologia tradicional ou ágil. A metodologia tradicional costuma ser mais formal por ser fortemente orientada a documentações e, portanto, com um maior custo para realizar alterações. Já as metodologias ágeis se adéquam a contextos onde podem existir frequentes mudanças nos requisitos,além da participação ativa do cliente na sua validação dos requisitos (KOSCIANSKI; SOARES, 2007). De acordo com (PRESSMAN, 2011), agilidade se tornou a palavra-chave no mundo ágil.

Quando um cliente busca uma solução de software, espera receber um produto de qualidade, dentro do prazo e dos custos planejados. Porém, existem diversas formas de se avaliar a qualidade do produto.

No desenvolvimento de um software há diversas fases, onde, para se avaliar a qualidade do produto são empregadas atividades de verificação e validação e alinhadas à essas atividades, estão as atividades de teste. No nível de negócios estão os testes de aceitação, no nível de requisitos estão os testes de sistema, no nível de design estão os teste de integração e na *build* estão os testes unitários (SOUZA; GASPAROTTO, 2013).

Na atualidade, os testes de software vêm sendo adotados como uma prática importante relacionada a qualidade de software. Eles auxiliam na avaliação da qualidade do produto, identificando defeitos presentes no sistema para que possam ser corrigidos, de maneira que seja entregue ao cliente um produto mais seguro e confiável (DELAMARO; MALDONADO; JINO, 2011).

Neste Trabalho de Conclusão de Curso, abordaremos o uso de testes exploratórios em comparação ao uso de testes tradicionais para avaliar a qualidade do produto de software, realizados em nível de sistema e sob o ponto de vista funcional, seguindo métricas de esforço, prazo, quantidade de defeitos, taxa de identificação de defeitos, entre outros, abordados no capítulo 4 de Aplicação.

Este trabalho está organizado em capítulo 2 Metodologia; capítulo 3 Referencial teórico; capítulo 4 Aplicação; capítulo 5 Análise e interpretação dos dados; capítulo 7 Considerações finais.

1.2 Problema

O crescente uso de metodologias ágeis podem gerar impactos na qualidade dos produtos de software entregues e o uso da aplicação de testes é largamente utilizada para avaliar a qualidade dos produtos. O uso de técnicas de teste convencionais ou tradicionais, requer um planejamento e documentação prévia, que pode acabar não acompanhando o tempo de desenvolvimento.

Os testes exploratórios de software são vantajosos, pois ao mesmo tempo que o testador aprende sobre o sistema, ele realiza o design e a execução dos testes. Porém, (WHITTAKER, 2010) destaca que os testes exploratórios não estão isentos de documentação, todavia elas são geradas durante o processo.

Outros contextos onde o uso de testes exploratórios são interessantes, seria, onde é necessário um *feedback* ou aprendizagem rápida do produto, ou quando não há disponibilidade de tempo para aplicações sistemáticas de testes. E um ponto positivo seria a maior diversidade de testes que podem ser aplicados com a exploração.

Todavia a escolha das *tours* ainda fica a cargo da experiência do testador. Logo o número de defeitos identificados pelos testadores também é influenciada por esse fator. Os testes tradicionais, por outro lado, trazem um pré-documentação por meio de um plano de testes e casos de teste, podendo auxiliar os testadores menos experientes a realizar os testes.

Uma lacuna identificada está em entender quais as vantagens do uso de testes exploratórios e que tipo de teste pode ser mais vantajoso em questão de detecção de defeitos, esforço e dificuldades relativas a experiencia do testador.

1.3 Questão de pesquisa

Com base no contexto e no problema relatado, a seguinte questão de pesquisa levantada neste trabalho de conclusão de curso:

"Como os testes exploratórios podem contribuir na avaliação da qualidade de produtos de software, considerando o ponto de vista funcional dessa avaliação?".

1.4 Objetivos

Nesta seção estão definidos os objetivos geral e específicos desta monografia.

1.4.1 Objetivos gerais

O objetivo geral deste TCC é aplicar testes exploratórios, avaliando sua contribuição para qualidade do produto de software.

1.4.2 Objetivos específicos

Para este Trabalho de Conclusão de Curso, visa-se atingir os seguintes objetivos específicos:

- identificar na teoria as definições e características sobre qualidade de produto de software;
- identificar na teoria as definições e características sobre testes exploratórios e testes tradicionais;
- Descrever o sistema selecionado e sua utilidade no contexto para qual foi construído;
- Definir as medições que serão feitas para avaliar o sistema;
- Definir a aplicação dos testes exploratórios no sistema a ser avaliado;
- Aplicar os testes nas áreas selecionadas do sistema;
- Avaliar os resultados gerados.

2 Metodologia

2.1 Considerações iniciais

Neste capítulo será definida a metodologia de pesquisa utilizada neste trabalho de conclusão de curso. Será feita a classificação da pesquisa e o planejamento, descrevendo a coleta de dados, a análise, interpretação e a redação dos resultados.

2.2 Classificação da Metodologia de Pesquisa

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

Quanto à natureza, esta pesquisa é aplicada, pois, tem como objetivo a geração de conhecimentos dirigidos à solução de problemas específicos (GIL, 2008), definindo como avaliar a qualidade de um produto de software específico e sobre o ponto de vista funcional, por meio de testes de software.

Quanto à abordagem, é classificada como qualitativa. A pesquisa qualitativa possui a preocupação com aspectos da realidade que não podem ser quantificados. No caso deste trabalho, levando-se em consideração a questão de pesquisa definida, visa-se avaliar a qualidade de um determinado produto de software sob o ponto de vista funcional, usando para isso recursos de área de testes de software e, mais especificamente, o uso de testes exploratórios. Assim sendo, as avaliações posteriores sobre a qualidade do software se dará de forma qualitativa. Este trabalho prevê também o uso de grupo focal, que é uma técnica qualitativa, onde entrevistas guiadas gerarão dados com o objetivo de entender a experiência dos participantes no uso de testes de software, sejam exploratórios ou tradicionais e as dificuldades encontradas durante o processo.

Porém, este trabalho também prevê o uso e aplicação de um questionário para entender a experiência dos testadores e a dificuldade que eles sentiram na execução dos testes previstos. Neste sentido esta pesquisa também, tem características quantitativas pois, a pesquisa quantitativa é adequada para mensurar atitudes e preferencias de uma população, e descobrir quantas pessoas partilham de aspectos semelhantes. Além disso, segundo (MORESI et al., 2003), não requer o uso de métodos e técnicas estatísticas e o ambiente natural é a fonte direta para coleta de dados.

Em relação ao objetivo, a presente pesquisa caracteriza-se como descritiva. A pesquisa descritiva expõe características de determinada população ou de determinado fenô-

meno. Pode também estabelecer correlações entre variáveis e definir sua natureza. Desta maneira, nesta pesquisa é visado descrever a capacidade dos testes exploratórios e dos testes tradicionais em detectar defeitos do sistema e avaliar a qualidade do produto entregue, bem como, realizar a comparação do uso das técnicas.

Quanto às técnicas de coletas de dados, foram elaborados instrumentos como roteiros de entrevistas, treinamentos e análise dos resultados de testes.

Uma representação da seleção metodológica é apresentada na Figura 1.

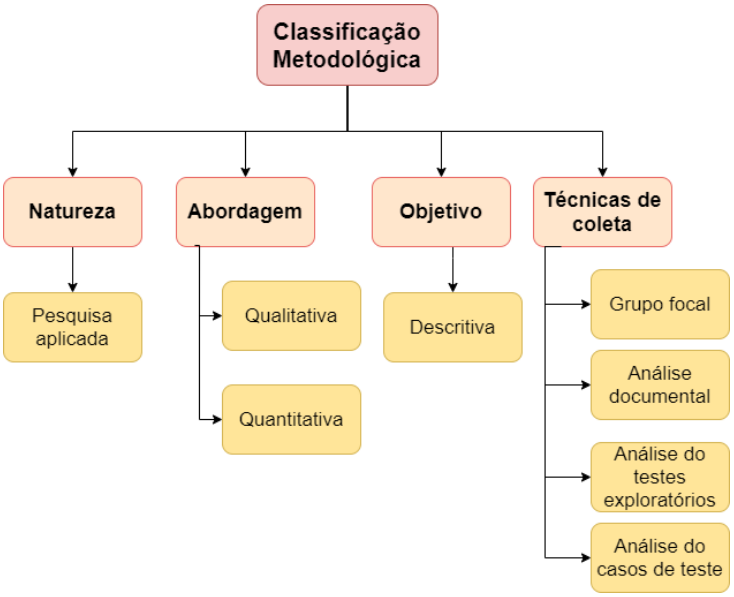

Figura 1 – Classificação metodológica. Fonte: Autor.

2.3 Plano Metodológico Adotado

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

O processo de pesquisa adotado abrange as fases: Planejamento; Coleta de Dados; Análise e Interpretação dos dados, e Redação do Resultado. A partir da metodologia de pesquisa adotada e da determinação das fases que a configuram. Sendo assim, foi elaborado um plano metodológico e foram definidas as etapas da pesquisa, nas quais são empregados os procedimentos e as técnicas de coleta de dados.

Na Figura 2 apresenta-se o esquema do plano metodológico adotado.

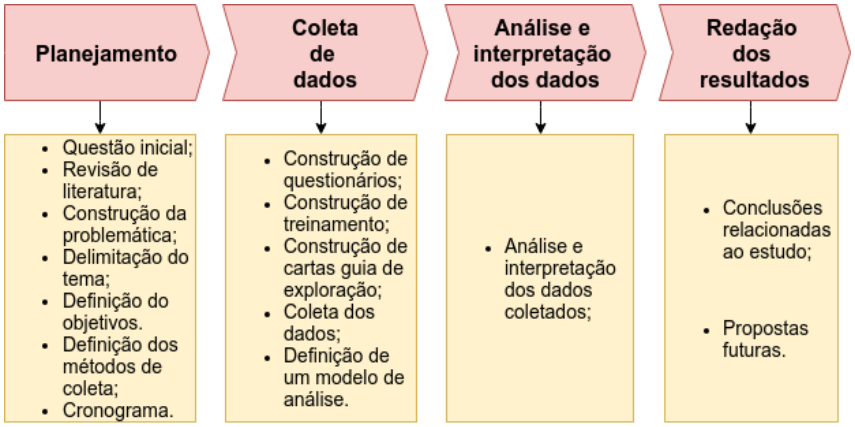

Figura 2 – Plano metodológico. Fonte: Autor.

2.3.1 Planejamento da pesquisa

Esta fase inicial compreende o estabelecimento do Problema, dos objetivos (geral e específicos), da pergunta de pesquisa, da seleção metodológica, com fases e etapas da pesquisa, e com o planejamento dos procedimentos de pesquisa e das técnicas de coletas de dados a serem executados.

2.3.2 Coleta de dados

A etapa de Pesquisa bibliográfica compreende o levantamento da literatura para a construção do referencial teórico, a fim de proporcionar o exame de um tema sob novo enfoque ou abordagem chegando a soluções inovadoras (MARCONI; LAKATOS, 2010). Como tipo de fontes bibliográficas foram buscadas publicações de livros, revistas e pesquisas que, por sua vez, retratassem a importância de avaliar a qualidade do software entregue, dos processos de verificação e validação, do uso de testes neste contexto, de metodologias de definição de métricas de software e, finalmente sobre o uso de testes exploratórios aplicados ao desenvolvimento de software. As publicações foram recuperadas a partir do portal de periódicos mantido pela CAPES.

Nesta etapa de Pesquisa documental, de acordo com (MARCONI; LAKATOS, 2010), em relação à fonte de coleta de dados, esta pesquisa é classificada como:

a) documental indireta: usa pesquisa bibliográfica;

b) documental direta: usando diretamente a documentação disponível do sistema desenvolvido a ser testado.

Para que os objetivos deste trabalho sejam alcançados serão utilizadas as seguintes técnicas de coleta

- Pesquisa bibliográfica;

- Análise documental;
- Grupo focal;
- Aplicação de treinamento;
- Elaboração da carta guia de exploração;
- Elaboração de casos de teste tradicionais.

2.3.2.1 Grupo focal

Segundo (VAUGHN, 1996 apud DIAS, 2010) a técnica de grupo focal é qualitativa e pode ser usada em conjunto com outras técnicas para aprofundar a coleta dos dados. Podem ser usadas tanto técnicas qualitativas, quanto quantitativas.

Segundo (GIL, 2008), a entrevista é uma ferramenta de coleta de dados que pode ser utilizada para investigar um tema mais a fundo. É comumente utilizada em pesquisas qualitativas como os grupos focais. Para a realização desta pesquisa foi escolhida a entrevista por pautas. De acordo com (GIL, 2008), a entrevista por pautas é guiada por um conjunto de pontos que são de interesse do investigador, e que podem ser exploradas ao longo do processo de entrevista. O entrevistador prepara algumas perguntas diretas, entretanto os entrevistados podem falar livremente a respeito da pauta.

A construção do roteiro das pautas que conduziram a entrevista nesta pesquisa, baseiam-se nas abstrações adquiridas no *GQM* e na matriz *SWOT*. De acordo com (GÜREL; TAT, 2017) A análise *SWOT* é uma ferramenta usada para dimensionar recursos de uma organização. Com a análise é possível identificar fatores ambientais internos e externos. A sigla *SWOT* vem do inglês, mas traduzindo para o português significa "forças", "fraquezas", "oportunidades"e "ameaças". Logo, os fatores internos seriam as forças e fraquezas. Já os fatores externos são as oportunidades e ameaças.

Buscou-se realizar perguntas que respondessem a dois tipos de itens. São eles:

- Itens para avaliar o conhecimento/experiência do participante;
- Itens para abstrair percepções do participante

2.3.2.1.1 Roteiro de entrevista e questionário

O Questionário é aplicado antes da realização dos testes, tendo como objetivo apurar se o participante já possui algum conhecimento prévio em relação aos testes de software.

Abaixo estão descritas os objetivos das questões utilizadas no questionário.

- Questões 1 e 2: Essas questões tem como objetivo averiguar se o participante possui alguma experiência no uso de testes de software.
- Questões 3, 4 e 5: Essas questões tem como objetivo averiguar se o participante conhece o método de testes exploratórios e se já possui alguma experiência no uso da técnica.

A entrevista é aplicada ao final da realização dos testes, e tem como objetivo apurar as dificuldades percebidas pelo participante ao longo da aplicação dos testes.

Abaixo estão descritos os objetivos das questões utilizadas na entrevista 2.

- Questão 1: Essa questão tem como objetivo avaliar de qual grupo pertence o participante.
- Questões 2, 3, 4, 5, 6, 7: Essas questões têm como objetivo averiguar as dificuldades que o participante sentiu em cada fase da aplicação dos testes.

2.3.2.2 Aplicação de Treinamento

Para a aplicação deste trabalho de conclusão de curso, optou-se pela aplicação de um treinamento prévio a aplicação dos testes.

Segundo (CHIAVENATO, 1998), "Treinamento é o ato de preparar as pessoas para o ambiente de trabalho". Para a realização de qualquer atividade, os indivíduos participantes necessitam de uma orientação para obterem um resultado de melhor qualidade e serem mais produtivos. (CHIAVENATO, 1998), complementa o conceito de treinamento afirmando que é o processo educacional, aplicado de maneira sistêmica e com objetivos específicos gerando, assim, conhecimentos, atitudes e habilidades ao indivíduo.

O objetivo da aplicação do treinamento seria então, a melhora de aspectos técnicos dos participantes, uma vez que um participante pode ter tido pouco ou nenhum contato com o conceito de teste exploratório. (CHIAVENATO, 1994) diz que treinamento é ”o ato intencional de fornecer os meios para proporcionar a aprendizagem”, ou seja, fornecer ao participante um meio para a auto-capacitação.

Existe uma variedade de tipos de treinamento. As mais usadas, segundo (MARRAS, 2011) são: aula expositiva, estudo de caso, dramatização, *workshop*, *brainstorming*, simulação, painel, simpósio, palestra e conferência. Segundo (MARRAS, 2011), a técnica escolhida varia conforme a situação, os objetivos, o público alvo, o tempo, o custo, entre outros.

Para este trabalho de conclusão de curso foi escolhida a técnica de aula expositiva, onde será fornecida uma video aula e um material de apoio ao participante.

2.3.2.3 Participantes

Para este trabalho de Conclusão de Curso, os participantes selecionados são estudantes da disciplina de Testes de Software da Universidade de Brasília, *campus* Gama.

Neste ano de 2020, devido a pandemia de covid-19, todas as fases do experimento serão realizadas de forma remota. Utilizando a ferramenta *Teams* para encontros online e a plataforma Aprender 3 para disponibilização de conteúdos de apoio e controle de prazos e entregas.

As entrevistas com os grupos tem o tempo limite de 40 minutos. E os treinamentos ocorreram no horário da aula de Testes de Software.

2.3.2.4 Carta guia de exploração

A carta guia de exploração (Apêndice C) contém uma explicação do trecho do sistema a ser testado, onde o participante poderá usar o nível médio de exploração. Entretanto, a cada *tour* realizado o participante deve preencher uma documentação com os dados do *tour* utilizado, a estratégia seguida, os *inputs*, *outputs* e principalmente, relatar se algum defeito foi apresentado.

2.3.2.5 Casos de teste

Nos casos de teste tradicionais (Apêndice D), consta uma pré documentação do que será testado no sistema. Ao longo do processo o participante preencherá uma documentação com os resultados dos testes realizados e principalmente o número de defeitos encontrados e a duração dos testes.

2.3.3 Análise e interpretação dos dados

Nesta fase, os resultados obtidos a partir de cada técnica de coletas de dados são analisados com o objetivo de refinamento e validação.

Este Trabalho de Conclusão de Curso visa a utilização de testes exploratórios para avaliar a qualidade do produto de software selecionado, o sistema PGTBL. Para tanto, será utilizado o auxílio de voluntários para a aplicação dos teste exploratórios. O processo foi dividido em algumas etapas, onde:

- A pesquisa bibliográfica faz parte da fase de compreensão tema e do contexto abordado;
- A análise documental foi utilizada para analisar os documentos existentes do sistema PGTBL, para compreensão dos requisitos.

- A aplicação do questionário, online tem o objetivo de conhecer o perfil dos participantes e suas experiências na utilização de testes de software e testes exploratórios, em específico.
- A aplicação da entrevista ao final da exploração, deve buscar responder um segundo questionamento relativo a sua percepção sobre o nível de dificuldade enfrentada durante as explorações.
- A realização de um treinamento online, onde o objetivo do treinamento é a explicação do uso de testes exploratórios, os distritos existentes e os possíveis *tours*, bem como o contexto do sistema a ser testado.
- A carta guia de exploração deve gerar uma documentação relatando cada *tour* realizada. O participante deve preencher a carta guia com os dados do *tour* utilizado, a estratégia seguida, os *inputs*, *outputs* e principalmente, relatar se algum defeito foi apresentado. A interpretação destes dados são relatados nos resultados.
- A realização dos casos de teste devem gerar uma documentação relatando os resultados dos testes realizados e principalmente o número de defeitos encontrados. A interpretação destes dados são relatados nos resultados.

2.3.4 Relato dos resultados

O relato dos resultados desta pesquisa é constituída, além das suas definições e contextualizações motivadoras, pelo Referencial Teórico deste trabalho, com a execução da pesquisa bibliográfica; o estabelecimento de uma estratégia de testes envolvendo testes exploratórios e testes tradicionais; seus resultados, considerações finais e conclusões.

Além disso, os resultados obtidos após a execução dos testes exploratórios e tradicionais e a aplicação das entrevistas nos grupos focais, serão analisados por meio dos documentos gerados e com o cruzamento dos dados obtidos. Dados como o número de defeitos encontrados por cada *tours* realizada, a dificuldade percebida pelos testadores na realização dos *tours*, o tempo e o esforço.

Ao final das análises e exposição dos resultados, serão elaboradas as considerações finais do trabalho, com uma recapitulação dos resultados obtidos e a identificação de trabalhos futuros.

3 Referencial Teórico

3.1 Qualidade de Software

As discussões sobre controle de qualidade são amplamente debatidas em todas as áreas do conhecimento. (KOSCIANSKI; SOARES, 2007) cita que nos anos de 1940 surgiram diversas organizações relacionados a esse controle, como: ASQC(*American Society for Quality Control*), a ABNT(Associação Brasileira de Normas Técnicas), e a ISO(*International Standardization Organization*).

Embora, historicamente, os primeiros relatos a respeito de controle de qualidade tenham surgido há mais de quatro mil anos, nas civilizações egípcias, onde eram estabelecidas unidades de medidas muito particulares como o cúbito, que era uma medida relacionada ao tamanho do braço do faraó. Essas medidas eram utilizadas nas construções das pirâmides e de tempos em tempos era necessário verificar se as medidas reais estavam de acordo com os padrões estabelecidos (KOSCIANSKI; SOARES, 2007).

Desde o anos 1970 tem-se reconhecido a gestão de qualidade como um instrumento conceitual de grande valor, pois, com o renascimento da indústria japonesa, que fez do controle de qualidade uma ferramenta para vantagem competitiva no mercado, seguindo as determinações de um consultor americano W.E. Deming. Já nos anos de 1980 a indústria automobilística japonesa, que era vista com desdém pelos fabricantes norte americanos, se tornou extremamente competitiva, com níveis consistentes de qualidade associados a bons preços e bons serviços pós venda (MARTINS; LAUGENI, 2005).

Os desenvolvedores de software, com o tempo, vêm reconhecendo a importância do controle de qualidade. (PRESSMAN, 2011) diz que:

> "Até mesmo os desenvolvedores de software mais experientes concordarão que software de qualidade é um objetivo importante".

No desenvolvimento de software, segundo (PRESSMAN, 2011), a qualidade de um projeto envolve o grau de conformidade em que a implementação atendeu as funções e características que foram especificadas no modelo de requisitos. Ou seja, o sistema resultante deve atender as necessidades e metas de desempenho pré-estabelecidas.

McCall apud (PRESSMAN, 2011) propõem três aspectos fundamentais que impactam a qualidade de um produto de software. São características operacionais como Usabilidade, Confiabilidade, Integridade, Correção e Eficiência, outro aspecto levantado é a habilidades de suportar mudanças, que engloba a facilidade de manutenção, Flexibilidade, Testabilidade e a Adaptabilidade a novos ambientes.

A ISO 9126 (COALLIER, 2001) foi desenvolvida para indicar atributos de qualidade de produtos de software. Tais como: Funcionalidade, Confiabilidade, Usabilidade, Eficiência, Facilidade de manutenção e Portabilidade

A ISO/IEC 25000 (STANDARD, 2005), também conhecida como norma SQuaRE, de acordo com (KOSCIANSKI; SOARES, 2007) é uma evolução da ISO/IEC 9126 e ISO/IEC 14598, ambas tratam de qualidade de produto de software. A norma SQuaRE, como um projeto que tinha como objetivo unificar, de certa forma, os modelos de qualidade, propondo um padrão internacional, unificando os vocabulários

Para a sua organização, a norma SQuaRE foi dividida em 5 partes, são elas:

- Gerenciamento de Qualidade (2500n);
- Modelo de Qualidade (2501n);
- Medição de Qualidade(2502n);
- Requisitos de Qualidade(2503n);
- Avaliação de Qualidade(2504n).

Gerenciamento de Qualidade 2500n: Este tópico fornece orientações introdutórias aos gerentes de produto de software, bem como aos desenvolvedores ou qualquer parte interessada, com especificações e avaliações. Desta forma, este documento provê maneiras de utilização do **SQuaRE**.

Modelo de Qualidade 2501n: Este tópico fornece orientações a respeito de Modelo de Qualidade, que aborda fatores de qualidade interna, externa e em uso do produto de software. O documento busca decompor esses fatores em sub-características de maneira a oferecer orientações práticas. (KOSCIANSKI; SOARES, 2007) relata que o modelo propõem uma classificação hierárquica dos aspectos de qualidade e são esmiuçados cada aspecto envolvendo cada ator envolvido na construção do produto.

Medição de Qualidade 2502n: Este tópico fornece orientações relacionadas a medições de qualidade de software nos aspectos internos, externos e em uso. São definidas maneiras de medições qualitativas e quantitativas do produto, bem como as orientações e definições de cada medida proposta. (KOSCIANSKI; SOARES, 2007) indica dois pontos em destaque neste tópico, que são: o fato do documento definir o que é medição e a descrição da realização das atividades correlatas. E o segundo tópico importante é a propostas das métricas que podem ser customizadas de acordo com o contexto do produto.

Requisitos de Qualidade 2503n: Este tópico fornece orientações que auxiliam a determinar e mapear os requisitos de qualidade do produto. (KOSCIANSKI; SOARES,

2007) 2007, destaca que esta divisão retoma princípios da ISO/IEC 9126, estabelecendo objetivos de qualidade, ou seja, é necessário uma definição prévia do foco de qualidade.

Avaliação de Qualidade 2504n: Este tópico fornece requisitos, orientações e diretrizes para a avaliação de qualidade do produto. Segundo (KOSCIANSKI; SOARES, 2007), esta divisão é onde a norma SQuaRE se concretiza, pois, após serem realizadas as medições do produto, os resultados são confrontados e avaliados. Esta divisão da norma fornece diretrizes que auxiliam diferentes tipos de avaliação para diferentes públicos alvo, como desenvolvedores ou clientes.

A norma SQuaRE busca ajudar a medir a qualidade do produto de software, e para isso a divide em três categorias: Qualidade interna, Qualidade externa e Qualidade em uso.

Qualidade interna 25022: a norma fornece apoio para medições quantitativas internas. Segundo (KOSCIANSKI; SOARES, 2007), a norma avalia o produto de forma estática, analisando a arquitetura interna do produto como: organização, complexidade do produto que servem de indicadores para custos de manutenção e velocidade de execução.

Qualidade externa 25023: a norma fornece apoio para medições quantitativas externas. Segundo (KOSCIANSKI; SOARES, 2007), a qualidade externa trata o produto como uma caixa-preta, realizando a verificação de partes de um sistema ou um sistema completo por meio de testes de funcionalidades.

Qualidade em uso 25024: este tópico da norma fornece as medidas para avaliar a qualidade em uso do produto, bem como orientações de como utilizá-las. De acordo com (KOSCIANSKI; SOARES, 2007), a qualidade em uso de um produto faz parte da visão do usuário, seja ele um funcionário da empresa ou um desenvolvedor. Quando a qualidade em uso está sendo avaliada do ponto de vista de execução, deve-se levar em consideração diversas condições, como ambientes ou qualificação dos usuários, até o hardware utilizado pode interferir na avaliação.

Segundo (KOSCIANSKI; SOARES, 2007), o modelo de qualidade SQuaRE é hierárquico e decomposta em características e subcaracterísticas e seus atributos. A qualidade em uso possui as seguintes características: eficácia, eficiência, satisfação, liberdade de risco e cobertura de contexto, e cada característica possui suas atividades definidas pelos *stakeholders*. Veja a Figura 3 .

Já a qualidade interna e externa, Segundo (KOSCIANSKI; SOARES, 2007), quase não se difere ao modelo proposto na norma ISO/IEC 9126, e é dividida em seis características que derivam subcaracterísticas: *Funcionalidade, Manutenibilidade, Usabilidade, Confiabilidade, Eficiência e Portabilidade*. A Figura 4 ilustra melhor essa hierarquia da qualidade interna e externa.

Uma observação importante que (KOSCIANSKI; SOARES, 2007) faz é que a ISO

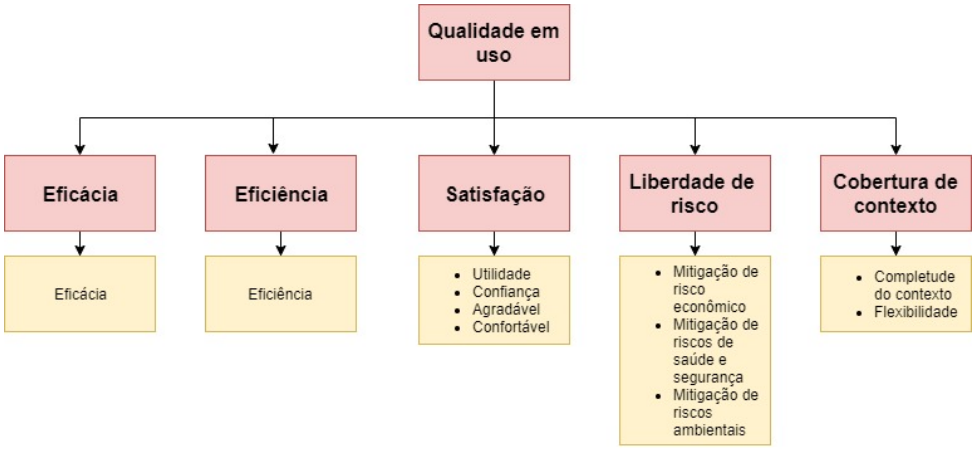

Figura 3 – Modelo de qualidade em uso na ISO/IEC 25000. Fonte: baseado em (STANDARD, 2005)

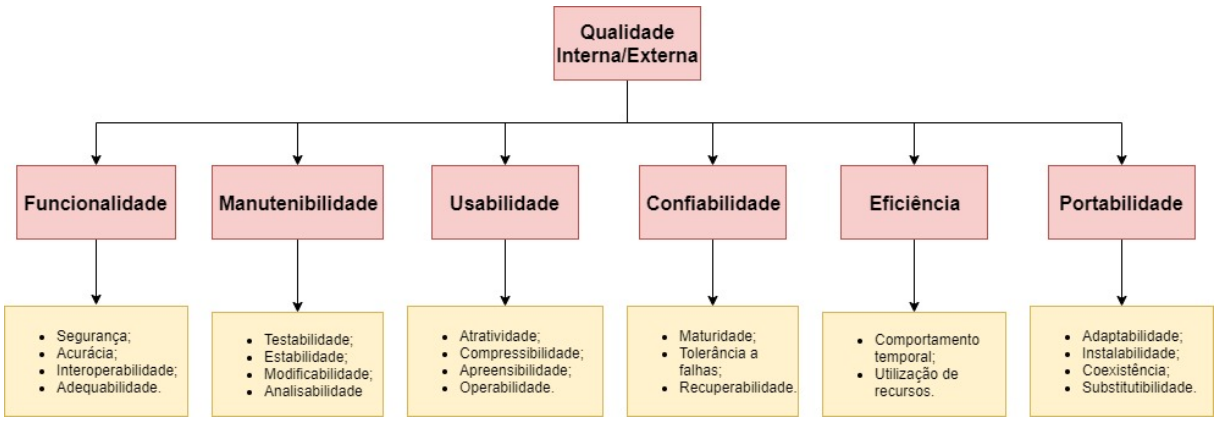

Figura 4 – Modelo de qualidade interna/externa da ISO/IEC 9126. Fonte: baseado em (COALLIER, 2001)

9126 não define os atributos, pois é de responsabilidade da equipe reconhecer os atributos em seu contexto, uma vez que cada produto possui suas particularidades. As características de qualidade interna/externa foram definidas de forma a evitar ambiguidade, desta forma as outras características não seriam sobreposta a esta. Definindo melhor cada característica, temos:

Funcionalidade: a funcionalidade é uma ação do software que é acionada pelo usuário, como apresentar dados na tela, imprimir ou salvar um dado. (KOSCIANSKI; SOARES, 2007) define como a capacidade para o cumprimento de uma tarefa, se assemelhando a definição de requisito funcional. As *subcaracterísticas* de acurácia referem-se a precisão envolvendo cálculos no programa. A *adequabilidade* seria a capacidade do sistema em realizar uma ação adequada ao contexto, como exemplo temos o *check-in* em hotel ou em aeroporto, são contextos diferentes e exigem modelagens distintas para se adequarem ao contexto. A *interoperabilidade* é a habilidade do sistema de se comunicar com outros sistemas de forma clara. A *segurança* é uma *subcaracterística* que se refere a proteção dos

dados do sistema.

Manutenibilidade: esta característica se refere a capacidade de realizar modificações no produto, seja evoluindo e adicionando novos requisitos ou apenas reparando erros do sistema. É uma característica que está diretamente ligada aos desenvolvedores do sistema. A *subcaracterística* de *modificabilidade* avalia a clareza do código, a adequação arquitetural e documental do produto. A *estabilidade do produto* avalia a capacidade do produto de sofrer modificações sem impactos negativos como mudanças de comportamento. A *analisabilidade* está ligada a facilidade de analisar o produto e identificar possíveis deficiências no sistema. A *testabilidade* está ligada a facilidade em validar um sistema após modificações.

Usabilidade: está ligada a facilidade de uso do produto. (KOSCIANSKI; SOARES, 2007) classifica esta característica como a mais difícil e subjetiva de ser verificada e validada, pois depende da interação do usuário com a interface do sistema, sendo importante reduzir ao máximo a interferência/influência do avaliador em um teste. Esta característica possui quatro *subcaracterísticas*. A primeira é a **operabilidade** que está ligada a capacidade se uma determinada ação ser controlada pelo usuário, ou seja, se o usuário desejar listar seus dados, por exemplo, e não ter a capacidade de interromper a ação antes de finalizada, isso determina falta de controle do usuário. A segunda é a *compressibilidade*, está ligada a habilidade do programa ser compreensível para o usuário, como exemplo o uso de vocabulário adequado. A terceira é a *apreensibilidade* seria a facilidade de aprender a usar o sistema, um exemplo são as interfaces intuitivas. A quarta é a *atratividade* e está ligada a beleza do sistema, a habilidade de prender a atenção do usuário, que é uma característica importante em contextos de sistemas educativos para crianças.

Confiabilidade: esta característica é definida na ISO/IEC 9126(COALLIER, 2001) como a habilidade do produto em sustentar um nível de desempenho em ambientes especificados. (KOSCIANSKI; SOARES, 2007) complementa dizendo que é esperado que o sistema seja capaz de se recuperar de falhas e continuar funcionando. Isso nos leva a uma de suas *subcaracterísticas* que é a *recuperabilidade*, ou a habilidade do sistema de se recuperar de falhas e voltar a funcionar corretamente. Outra *subcaracterística* é a *maturidade*, que remete a um sistema grande e robusto, como muitos módulos e que já é capaz de lidar com as possíveis falhas do sistema e se recuperar.

Eficiência: esta característica é considerada quantitativa e avalia a utilização dos recursos. A *subcaracterística* de *comportamento temporal* do sistema geralmente avalia quesitos do sistema em ambiente controlado, pois diversos fatores podem influenciar o resultado como: memória cache, memória RAM, velocidade de CPU, Sistema Operacional, entre outros. Uma outra *subcaracterística* é a *utilização de recursos* que abrange os recursos que não são tempo de CPU. Como exemplo de uma medição, (KOSCIANSKI; SOARES,

2007) fala da memória ocupada, que se for dependente de tráfego de rede, é importante pré-estabelecer parâmetros para avaliar, pois, medições coletadas sobre o tráfego de rede possuem variáveis ambientais difíceis de controlar para serem coletadas.

Portabilidade: é a capacidade de um sistema ser utilizado em diferentes tipos de plataformas. A *subcaracterística* de *coexistência* refere-se a sistemas que dividem recursos. A *adaptabilidade* é a habilidade do sistema em se adaptar a diferentes ambientes. A *instalabilidade* é a capacidade/facilidade de instalar o produto considerando as definições e parâmetros ambientais. A *substitutibilidade* é a compatibilidade do produto com outros produtos.

3.2 Medição de software

A medição de software é um elemento essencial para alcançar bons resultados no desenvolvimento. Segundo (FENTON; BIEMAN, 2014), bons desenvolvedores de software medem suas características para garantir que:

- Os requisitos estão completos e consistentes;
- Se o design é de alta qualidade;
- Se o código está pronto para uma *release*;

Medir os atributos do processo e do produto é uma prática para conseguir indicativos de quando o software está pronto para ser entregue e se os custos estão dentro do planejado.

O processo de medição é utilizado para analisar se o produto está em conformidade com os requisitos dos clientes e se tem qualidade suficiente para que os mantenedores consigam evoluir o produto e realizar melhorias nele.

Para (FENTON; BIEMAN, 2014), realizar medições está ligada ao sucesso, pois a partir delas temos uma base para tomarmos decisões. Exemplos de medições que realizamos no dia-a-dia para tomada de decisões são:

- Medições econômicas que nos auxiliam a determinar preços ou aumentos salariais;
- Medições em radares que nos permitem detectar aeronaves quando a visão não é clara;
- Medições atmosféricas que são a base para predições do tempo;

As medições fazem parte do dia-a-dia de todos, pois, quando calculamos qual a melhor rota ou quanto de gasolina vamos gastar em um percurso, medir nos dá a melhor perspectiva para realizarmos o melhor julgamento possível.

Então, (FENTON; BIEMAN, 2014) chegam a uma definição para medição que é: "o processo pelo qual números ou símbolos são atribuídos à atributos ou entidades do mundo real de tal maneira que descrevam de acordo e com regras claras e definidas".

(FENTON; BIEMAN, 2014) classificam a escala de medidas em cinco tipos: Nominal, Ordinal, Intervalo, Razão e Absoluto.

Nominal: é um nome ou elemento associado a uma classe, que não necessita de ordenação entre elas ao passo que não há uma grandeza numérica relacionada a ela. Exemplos de escala nominal são: sexo, nacionalidade, cor dos cabelos.

Ordinal: é composta por classes ordenadas em relação aos seus atributos. Os números atribuídos representam apenas a classificação da classe e é permitido combinações se a lógica fizer sentido. Um exemplo dessa escala seria o mapeamento da complexidade de módulos do sistema, onde a ordem importa.

Intervalo: tem como objetivo a comparação de intervalos entre classes. Nesta classe a ordem é importante e é aceitável soma e subtração para calcular o intervalo, mas não é permitido divisões ou multiplicações. Um exemplo é o grau de satisfação, intervalo de tempo, entre outras.

Razão: esta escala permite realizar análises mais avançadas como a quantidade de vezes que um atributo é maior que outro. É necessário que se mantenha a ordem entre as medidas e ela pode crescer de forma proporcional a partir de uma escala iniciada em zero. Um exemplo dessa escala é o calculo para saber quanto tempo a mais um projeto pode ter consumido a mais que outro.

Absoluto: a escala absoluta é uma atribuição numérica a um atributo e é realizada apenas contando o número de ocorrências. Exemplos de medidas absolutas são: número de falhas observadas ou número de funcionários de um projeto.

3.2.0.1 Método GQM

De acordo com (SOLINGEN; BERGHOUT, 1999) o método GQM (do inglês *Goal-Question-Metric*) foi desenvolvido por V. Basili and D. Weiss e evoluido por D. Rombach, sendo resultado de muitos anos de aplicações práticas. O GQM é uma abordagem sistemática utilizado para auxiliar na organização e planejamento de medições de software, onde as medições são feitas apenas com objetivos específicos. O método GQM está dividido em quatro fases, evidenciados na Figura 5:

- **Planejamento:** nesta fase é desenvolvido um projeto para aplicação das medições;
- **Definição:** nesta fase são definidas e documentadas as metas, perguntas, métricas e hipóteses;

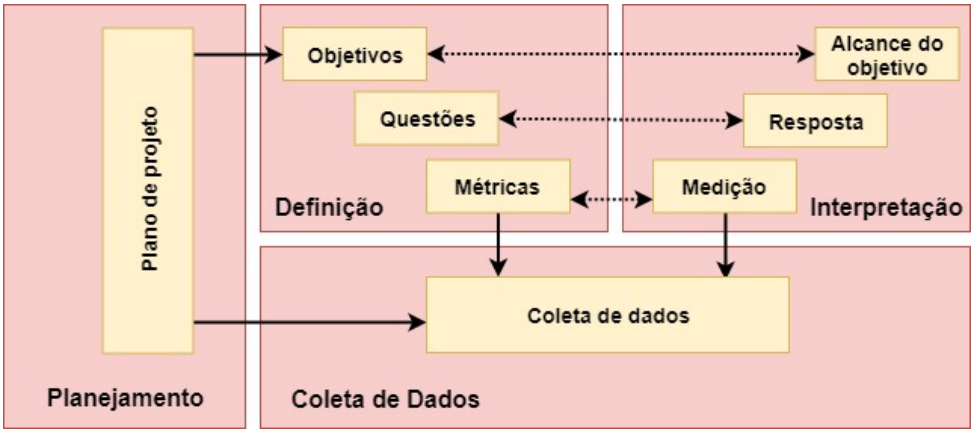

Figura 5 – Fases do GQM. Adaptado de: (SOLINGEN; BERGHOUT, 1999)

- **Coleta de dados:** nesta fase ocorre a coleta de dados de fato;
- **Interpretação:** nesta fase os dados coletados são analisados com base nas métricas pré definidas, resultando na respostas para as perguntas.

Segundo (SOLINGEN; BERGHOUT, 1999)

> "O GQM busca ser um método orientado a objetivos. O GQM define um determinado objetivo, refina esse objetivo em perguntas e define métricas que devem fornecer as informações para responder a essas perguntas. Ao responder às perguntas, os dados medidos definem as metas operacionalmente e podem ser analisados para identificar se os objetivos foram alcançados. Assim, o GQM define métricas de uma perspectiva de cima para baixo e analisa e interpreta os dados de medição de baixo para cima."

Figura 6.

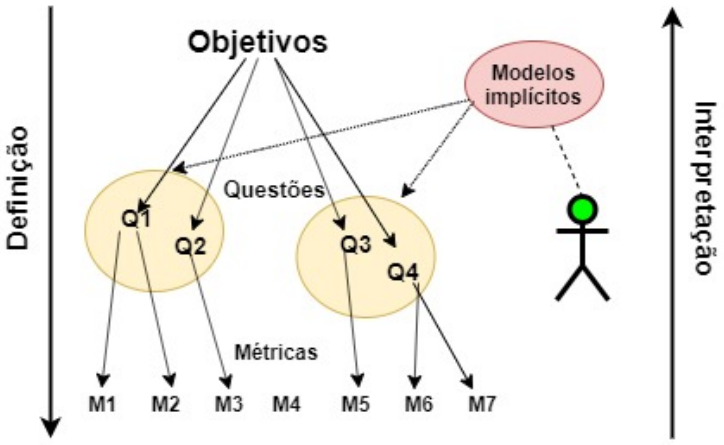

Figura 6 – GQM. Adaptado de:(SOLINGEN; BERGHOUT, 1999)

Para cada etapa é necessário definir um elemento especificado abaixo:

Objetivos: são selecionados de acordo com as necessidades dos *stakeholders*. São definidas medições de acordo com os objetivos e de acordo com os requisitos e seu grau de importância, desta maneira é possível gerenciar os recursos de acordo com o as necessidades do produto.

Questões: são perguntas feitas para guiar a medição e espera-se que sejam respondidas após a atividade de medição.

Categorias: como as questões levantadas podem trazer informações distintas, é importante que se categorize os resultados para uma melhor avaliação.

Formulários: é um elemento importante para auxiliar os avaliadores no processo de registro da coleta dos dados. A padronização da documentação de coleta de dados traz a vantagem de amenizar possíveis erros de coleta.

3.3 Verificação e Validação(V&V)

Segundo (SOMMERVILLE, 2005), As atividades que envolvem a verificação e análise das etapas de construção de um software, garantindo que as especificações atendam as necessidades do cliente são chamados de Verificação e Validação (V&V). Essa atividade faz parte de todo o ciclo de vida do produto, desde a elicitação dos requisitos até a fase de testes.

Na prática, os termos Verificação e Validação possuem significados diferentes. Segundo Boehm (1979, *apud* (SOMMERVILLE, 2005)), o termo Verificação questiona se estamos construindo o produto certo, já a Validação questiona se estamos construindo certo o produto. Essas duas definições elucidam que a verificação é uma checagem do software de acordo com as especificações dos requisitos funcionais e não funcionais. Já a validação é um processo que envolve o cliente, assegurando que o produto entregue está de acordo com as suas expectativas e necessidades.

Segundo (PRESSMAN, 2011), a verificação e validação do produto envolvem diversas atividades de garantia de qualidade do software, SQA(*software quality assurance*): revisões técnicas; auditorias de qualidade; monitoramento de desempenho; simulação; revisão de documentação e base de dados; teste de usabilidade, aceitação, qualificação e instalação.

(DELAMARO; MALDONADO; JINO, 2011) diz que usualmente as atividades de Verificação e Validação são divididas em estática e dinâmica. A atividade estática é aquela em que não há a necessidade de execução ou existência de um modelo executável para ser realizada. Já a dinâmica, por sua vez, é baseada justamente na execução do programa.

(SOMMERVILLE, 2005) destaca duas técnicas de checagem e análise que são

usadas em operações de V&V, que são: Inspeção de Software e Testes de Software.

As inspeções de software envolvem a checagem documental do produto, ou seja, requisitos, diagramas de projeto e código-fonte. Esta é uma técnica estática que pode ser aplicada em todas as fases do projeto.

Os testes de software abrangem a execução do software para verificar seu comportamento operacional. Esta é uma técnica dinâmica que trabalha com formato executável do produto.

Existem alguns termos muito utilizados na área de Verificação, Validação e Testes. São eles: defeito, engano, falha, erro, dado de teste e caso de teste.

Defeito: (do inglês, *fault*) é a definição de dados incorretos.

Engano: (do inglês, *mistake*) é uma operação humana que gera um defeito.

Erro: devido a presença de um defeito, o programa pode apresentar um erro ao ser executado, caracterizando um estado inesperado. Este resultado pode ocasionar a falha.

Falha: quando os resultados são diferentes do esperado, é denominada falha.

A Figura 7 representa as atividade de verificação e validação, alinhando as atividade de teste às fases de desenvolvimento de um sistema, com o objetivo de identificar os defeitos gerados durante o processo. Na Figura 7 as atividades de teste estão à direita e as atividades de desenvolvimento estão à esquerda(SOUZA; GASPAROTTO, 2013).

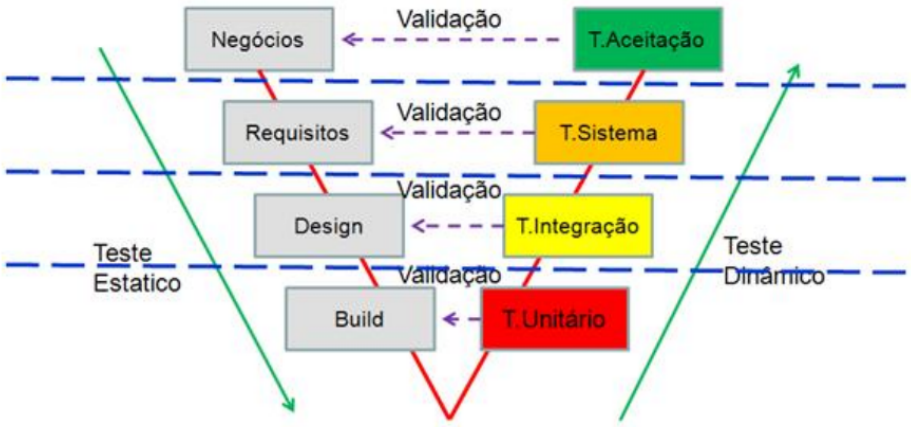

Figura 7 – Atividades de VV&T, SILVA,2013 apud (SOUZA; GASPAROTTO, 2013).

(PRESSMAN, 2011) destaca que embora a atividade de teste seja de extrema importância em V&V, muitas outras atividades também são. Os testes auxiliam a estimar a qualidade de um produto, porém eles não garante a qualidade por si só. A qualidade é incorporada ao longo do processo de desenvolvimento do software. (Miller *apud* (PRESSMAN, 2011)) complementa identificando a aplicação de testes como um meio ao qual obtemos uma confirmação da qualidade do software.

3.4 Teste de software

Segundo (GALIN, 2004), teste de software é um processo descrito e executado por uma equipe de teste, na qual uma ou várias unidades de software, ou até um software inteiro são examinadas e executados em um computador. Todos os testes associados são realizados de acordo com procedimentos de teste aprovados em casos de teste aprovados.

As fases de teste são divididos em fases com objetivos diferentes. São estabelecidas em:

- Teste de unidade;
- Teste de integração;
- Teste de sistema;
- Teste de regressão.

Teste de unidade: o objetivo desta fase é testar pequenas unidade do software, como: funções, procedimentos, métodos ou classes. O teste de unidade pode ser implementado em todo o processo de construção de cada unidade do software, sem a necessidade do sistema está completo. Nesta fase é esperado a identificação de erros de algoritmos ou estruturas de dados incorretos ou mal implementados;

Teste de integração: Esta fase ocorre após todas as unidades serem testadas individualmente. O foco está na estrutura

Teste de sistema: esta fase tem como objetivo verificar se todas as funcionalidades estão em conformidade com os requisitos. Ela é realizada quando o sistema estiver completo e com todas as partes integradas. Os testes de sistema buscam explorar elementos de:

- Correção;
- Completude;
- Coerência;
- Requisitos não funcionais.

Teste de regressão: esta fase ocorre durante manutenções de software. Durante a manutenção de sistemas, geralmente, qualquer modificação pode ocasionar na inserção de novos defeitos. Tendo em vista este fato, os testes de regressão se tornam importantes à medida que precisamos garantir que as novas modificações efetuadas estão corretas, ou seja, que essas novas adições funcionem como o esperado e que os requisitos testados anteriormente continuem funcionando.

As fases para a execução das atividades de teste são:

- Planejamento;
- Projeto de Casos de teste;
- Execução;
- Análise.

3.4.1 Técnicas e critérios de testes

Segundo (DELAMARO; MALDONADO; JINO, 2011), é importante buscarmos maneira de utilizar apenas subconjuntos reduzidos do domínio de testes. Porém, estes devem conter uma grande probabilidade apresentar a existência de defeitos. Seria a ideia de subdomínio de teste.

O **subdomínio de teste** é um subconjunto do domínio de entrada, engloba dados similares.

Há duas maneiras para escolher os elementos de cada subdomínio de teste, que são: teste aleatório e teste de partição.

Teste aleatório: é um teste feito a partir da escolha aleatória de um grande volume de casos de teste, onde seja provável que todos os subdomínios sejam representados.

Teste de partição: os subdomínios que serão testados são definidos para que possa ser escolhidos os casos de testes pertencentes às partes.

É importante explicitar os critérios de teste para auxiliar na definição dos subconjuntos, divididos em três critérios: Funcionais, estruturais e baseados em defeitos.

O uso de testes possui limitações, pois, segundo (DELAMARO; MALDONADO; JINO, 2011) os testes não servem para mostrar que o software está correto, mas apenas indicar a presença defeitos, se existirem. Dificilmente é feita a afirmação de um software estar correto, pois diversos casos de testes podem ser produzidos e mesmo assim algum defeito ser revelado em um caso de teste que ficou de fora. Logo, com a atividade de teste é possível alcançar apenas a confiança de um produto que se comporta de maneira correta.

3.4.2 Teste Funcional

Segundo (DELAMARO; MALDONADO; JINO, 2011) o teste funcional, é um método usado para projetar casos de teste onde o sistema é considerado caixa preta. Neste tipo de teste são fornecidas entradas e é verificada se as saídas estão conforme o foi especificado.

A priori, o teste funcional tem a capacidade de detectar todos os defeitos, se for realizado um teste exaustivo onde são testadas todas as entradas do sistema, entretanto, como foi dito anteriormente, o domínio de entrada dos testes pode ser muito grande, tendendo ao infinito, tornando inviável a realização completa da atividade. Tendo em vista esta limitação, algumas técnicas de teste foram propostas para tornar o processo mais formal e sistemático, auxiliando assim na análise dos resultados.

Em relação a técnica de teste funcional, (DELAMARO; MALDONADO; JINO, 2011) aponta alguns critérios, são eles:

- Particionamento em classes de equivalência;
- Análise do valor limite;
- Teste funcional sistemático;
- Grafo causa-efeito;
- Error Guessing.

Particionamento em classes de equivalência: tendo em vista que o teste exaustivo é inviável de ser aplicado nas atividades de teste, o particionamento em classes de equivalência visa auxiliar na determinação do subdomínio de teste, reduzindo o número de elementos a serem testados. Separando o sistema em classe equivalentes, em relação às especificações dos requisitos, ou seja, elementos que possuem comportamentos semelhantes, é possível assegurar que aquela classe é um representante e que se ela apresentar defeito, as outras também apresentariam.

Segundo (DELAMARO; MALDONADO; JINO, 2011), uma classe de equivalência aponta estados válidos ou inválidos. Então, é necessário definir casos de teste que cubram o maior número possível de elementos de classes válidas, e para as classes inválidas é necessário construir novos casos de testes específicos, evitando mascaramento da validação do elemento.

Análise do valor limite: é um critério utilizado em conjunto ao particionamento em classes de equivalência, entretanto existem alguns critérios ao selecionar os dados de teste. Segundo Myers (apud (DELAMARO; MALDONADO; JINO, 2011)) é feita uma escolha mais criteriosa dos dados de teste, levando em consideração a limitante de cada classe de equivalência investigada.

Teste funcional sistemático: é um critério que engloba tanto o particionamento em classes de equivalência quanto a análise do valor limite. Este critério necessita de no mínimo dois casos de teste analisando o limite de cada partição, desta maneira o critério visa minimizar os erros e evitar que os erros sejam mascarados.

Grafo causa-efeito: é um critério que, segundo Myers (apud (DELAMARO; MALDONADO; JINO, 2011))., busca combinar os dados de entrada, desta maneira, os casos de teste selecionados exploram incompletudes e duplicidades no sistema. Os desafios deste critério estão complexidade de construir e converter em tabelas os grafos booleanos, principalmente quando muitas causas e efeitos são identificadas ao longo do processo.

Error Guessing: é uma abordagem ad-hoc. Os casos de teste são derivados de forma intuitiva ou da experiência do testador. O objetivo deste critério é computar os possíveis erros ou fatores que gerem erros no sistema.

3.4.3 Testes exploratórios

(BACH, 2001) define testes exploratório (ET) como uma abordagem de teste poderosa de aprendizagem simultânea. (BACH, 2001) acredita que todo testador já realizou teste exploratório, mesmo que inconscientemente. O ET é o pensamento científico em tempo real. Já para (Tinkham e Kaner apud (ITKONEN; RAUTIAINEN, 2005)) o teste exploratório está ligado ato da própria atividade de testes gerar informações para que novos testes possam ser gerados.

Para (Kaner, Bach, *et.* Pettichord *apud* (ITKONEN; RAUTIAINEN, 2005)), explorar é vagar de propósito: em um determinado espaço com uma missão geral, mas sem uma rota pré-estabelecida. Explorar tem a ver com a aprendizagem e a experimentação contínua.

No SWEBOK (BOURQUE; FAIRLEY et al., 2014) conceitua teste exploratório como:

> "O teste exploratório é definido como simultâneo aprendizado, design e execução de testes; isto é, os testes não são definidos previamente em um plano de teste estabelecido, mas são dinamicamente projetado, executado e modificado. A eficácia dos testes exploratórios depende do software conhecimento do engenheiro, que pode ser derivado de várias fontes: comportamento observado do produto durante o teste, familiaridade com o aplicativo, a plataforma, o processo de falha, o tipo de possíveis falhas e falhas, o risco associado a uma determinado produto e assim por diante".

James Bach apud (ITKONEN; RAUTIAINEN, 2005) aponta seis contextos em que os testes exploratórios podem ser aplicados, são elas:

- Onde é necessário um *feedback* ou aprendizagem rápida do produto;
- Quando não há disponibilidade de tempo para aplicações sistemáticas de testes;
- Auxiliar na investigação de fatores de risco específicos;

- Maior diversidade de testes;
- Na realização de teste de regressão baseados em relatórios de defeitos;
- Na construção de testes a partir da perspectiva do usuário, originando-se de um manual de usuário, por exemplo.

(WHITTAKER, 2010) destaca que o testadores exploratórios não estão livres de documentações, a diferença é que na abordagem exploratória a documentação é gerada ao longo do processo à medida que os testes são executados. (WHITTAKER, 2010) destaca também que as ferramentas de captura de tela e gravação de teclas são ideais para gravação resultado do teste exploratório.

Um ponto importante é apontado por (WHITTAKER, 2010) a respeito de testes exploratórios. Ele afirma que a realização de ET sem uma boa orientação ou estratégia pode se tornar um risco e uma perda de tempo, pois os testadores podem ficar vagando pelo sistema procurando coisas a serem testados e não obter sucesso.

3.4.3.1 Metáfora do turista

A palavra metáfora se origina do grego e significa “transferência”. Sendo assim, (MARCUSCHI, 2000) diz que a metáfora é um figura de linguagem, caracterizado como um recurso semântico explicável para transferir o significado de algo, realizando comparações abreviadas. Ela apoia o ponto de vista criativo do indivíduo da óptica operacional.

A metáfora do turista apresentada por (WHITTAKER, 2010), faz uma alusão a um turista que visita um grande cidade como Nova Iorque, ou Londres. Sem a preparação de um guia, é basicamente impossível conhecer a cidade em pouco tempo, pois são diversos bares, restaurantes, parques, galerias, entre outros lugares a serem conhecidos. Ou seja, é extremamente importante bolar uma estratégia para que o tempo seja bem aproveitado e guiar as decisões. O mesmo acontece com um testador bem preparado que deseja explorar software complexos, pois a escolha da estratégia e a definição das metas.

3.4.3.2 Planejamento do teste do turista

Segundo (WHITTAKER, 2010), o primeiro passo antes de iniciar o planejamento de teste de um sistema é decompô-lo em partes menores para facilitar a gerência do recurso. Ao distribuindo os recursos, o esforço de teste também é dividido e auxilia no acompanhamento do progresso.

A metáfora do turista sugere uma decomposição baseada na intenção e não em estrutura correlacionadas do aplicativo em teste. Isso faz alusão a turista que a medida que as férias se aproximam, se organiza com a intenção de ver o maior número de lugares

no menor período de tempo possível. Assim como um turista, o testador também deve organizar suas excursões.

Um bom turista, ao planejar sua viagem, divide o destino em distritos, como distrito comercial, de entretenimento, teatro, entre outros. Desta mesma maneira, um testador de software pode dividir um sistema em distritos, seguindo um critério lógico.

Assim, (WHITTAKER, 2010) propõem que as funcionalidade sejam separados em distritos, podendo ser: distrito comercial, o distrito histórico, o distrito turístico, o distrito de entretenimento, o distrito do hotel e o distrito decadente. Cada distrito apresentado subdivide-se em tipos de *tours*, como está representado na Figura 8 .

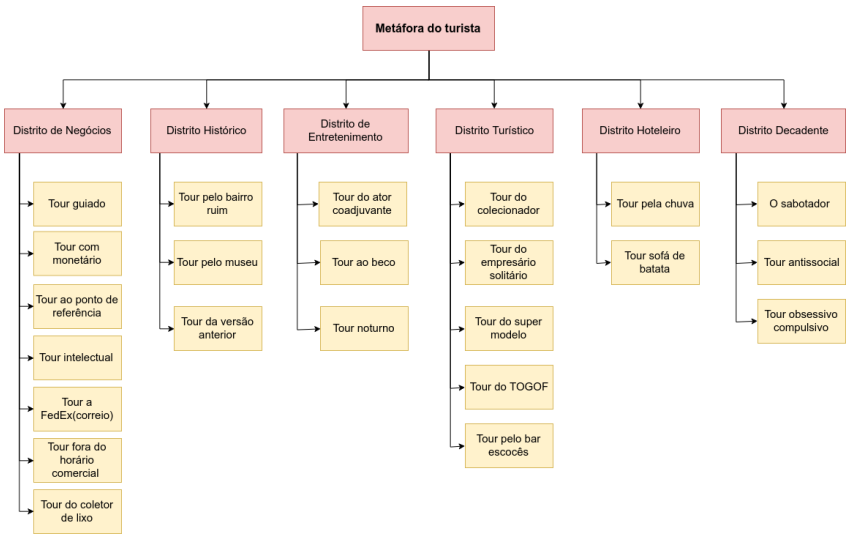

Figura 8 – Metáfora do turista, divisões de distritos e *tours* Fonte: Autor

3.4.3.2.1 Distrito de negócios *(Business district)*

É conhecido como "o local onde os negócios são realizados, onde existe hora do rush em determinados horários, assim como horário comercial e grandes centros comerciais. Para software seriam os sistemas de dependem de inicialização e desligamento, prontos para uso. (WHITTAKER, 2010) propõem sete tipos de *tours* para este distrito (Tabela 1):

3.4.3.2.2 Distrito histórico *(Historical District)*

Este distrito, no contexto de softwares, tem como objetivo testar software legado. Assim como existem lugares que foram cenários de eventos históricos e despertam o interesse de muitos turistas sobre o que aquele cenário tem a nos contar, os software também possuem uma história a ser contada, em suas inúmeras versão, modificações e defeitos antigos. (WHITTAKER, 2010) propõem três tipos de *tours* para este distrito (Tabela 2):

Tabela 1 – Tours ao distrito de negócios

Nome do tour	**Objetivo**
our guiado *(Guidebook tours)*	Este tour se propõem a seguir o manual de usuário.
Tour com monetário *(money tour)*	Este *tour* busca verificar a coerência das funcionalidades de uma versão demo do produto com as especificações do sistema.
Tour ao ponto de referência *(Landmark tour)*	Este *tour* visa selecionar as principais características de um sistema e ordenar a ordem de visita
Tour intelectual *(Tour intelectual)*	Visa realizar perguntas difíceis para o software, que demandem mais esforço na resposta.
Tour a FedEx (correio) *(The FedEx Tour)*	Este *tour* visa analisar o que está acontecendo com os dados desde o momento de envio até a entrega.
Tour fora do horário comercial *(The After-Hours Tour)*	Este *tour* visa verificar as atividades de backup e armazenamento de dados de um sistema.
Tour do coletor de lixo *(The Garbage Collector's Tour)*	Assim como coletores de lixo conhecem bem a cidade e bolam estratégias para visitar cada setor, este *tour* visa bolar uma estratégia para visitar funcionalidades de forma metódica.

Tabela 2 – Tours ao distrito histórico

Nome do tour	**Objetivo**
Tour pelo bairro ruim *(The Bad-Neighborhood Tour)*	Este *tour* visa concentrar os testes em regiões com maior número defeitos.
Tour pelo museu *(The Museum Tour)*	Este tour visa testar modificações recentes de sistemas legado.
Tour da versão anterior *(The Prior Version Tour)*	Este *tour* visa testar um software após a realização de modificações para garantir que o que estava funcionando continue funcionando.

3.4.3.2.3 Distrito de entretenimento *(Entertainment District)*

Às vezes o testador necessita de testes mais "tranquilos" de serem executados. Os testes que envolvem características que não são as principais do sistema. Assim como, diversas vezes, um turista necessita fazer passeios mais relaxantes, que não envolvam tanta reflexão. (WHITTAKER, 2010) propõem três tipos de *tours* para este distrito (Tabela 3):

3.4.3.2.4 Distrito turístico (Tourist District)

Este distrito refere-se ao ato de visitar as funcionalidades ainda não visitadas. Assim como existem turistas que gostam de visitar pontos turísticos em cidades que

Tabela 3 – Tours ao distrito de entretenimento

Nome do tour	**Objetivo**
Tour do ator coadjuvante *(The Supporting Actor Tour)*	Este tour visa dar atenção às características periféricas de um sistema.
Tour ao beco *(The Back Alley Tour)*	Este tour visa testar recursos do sistema que não costumam ser muito utilizados pelos usuários.
Tour noturno *(The All-Nighter Tour)*	Este tour visa forçar entradas consecutivas para testar os valores das variáveis de forma incansável.

ainda não conhece. (WHITTAKER, 2010) propõem cinco tipos de *tours* para este distrito (Tabela 4):

Tabela 4 – Tours ao distrito turístico

Nome do *tours*	**Objetivo**
***Tours* do colecionador** *(The Collector's Tour)*	Este *tours* visa o maior número de recursos do software e documentar essas saídas.
***Tours* do super modelo** *(The Supermodel Tour)*	Este *tours* visa identificar defeitos superficiais, gerados em primeiras impressões de um sistema, por meio da própria interface, testando seus elementos.
***Tours* do TOGOF** *(The TOGOF Tour)*	Este *tours* visa avaliar um sistema executando várias cópias simultaneamente, analisando características relacionados a concorrência e compartilhamento.
***Tours* pelo bar escocês** *(The Scottish Pub Tour)*	Este *tours* visa identificar características que geralmente devem está citadas em um guia. Geralmente é necessário listá-las e realizar testes para executá-las.

3.4.3.2.5 Distrito hoteleiro *(Hotel District)*

Os destinos turísticos precisam oferecer bons locais de descanso para os turistas. Assim como os software precisam testar recursos secundários do sistema, e deixar o recursos principais em repouso algumas vezes. (WHITTAKER, 2010) propõem dois tipos de *tours* para este distrito (Tabela 2):

3.4.3.2.6 Distrito decadente *(Seedy District)*

São lugares desagradáveis e cheios de coisas ruim para os turistas, mas mesmo assim alguns optam por visitar. Para os testadores de software, são teste que precisam ser feitos para evitar que os usuários passem por momentos incômodos. (WHITTAKER, 2010) propõem três tipos de *tours* para este distrito (Tabela 6):

Tabela 5 – Tours ao distrito hoteleiro

Nome do *tours*	**Objetivo**
***Tours* pela chuva** *(The Rained-Out Tour)*	Assim como um dia chuvoso pode atrapalhar o roteiro de atividades de um turista, este *tours* visa testar o comportamento do sistema ao iniciar operações e interrompê-las em seguida.
***Tours* sofá de batata** *(The Couch Potato Tour)*	Este *tour* visa testar os recursos do sistema, como formulários, com o mínimo de dados possível para verificar se o sistema está aceitando quantidades menores de dados.

Tabela 6 – Tours ao distrito decadente

Nome do *tour*	**Objetivo**
O sabotador *(The Saboteur)*	Este *tour* visa "sabotar" o sistema, aplicando todo tipo de teste e de maneiras diferentes.
Tour antissocial *(The Antisocial Tour)*	Este *tour* subdivide-se em *tour* oposto, *tour* arrastão e *tour* do rumo errado, fornecendo dados pouco prováveis e que não deveriam ser executados além de não obedecer uma ordem pré-definida
***Tour* obsessivo compulsivo** *(The Obsessive Compulsive Tour)*	Este *tour* visa testar determinados recursos de forma repetida, "compulsivamente".

3.4.3.3 Níveis de exploração em testes exploratórios

(GHAZI et al., 2018) veem o teste exploratório como uma maneira poderosa e eficiente de testar software, pois é possível integrar projeto, execução e análise de testes durante o processo. Os autores identificam também que é possível que existam diferentes níveis de teste, desde uma exploração livre à uma exploração baseada em *script* e desta maneira, propõem cinco níveis exploração. São eles:

- Estilo livre *(Freestyle)*;
- Alto grau de exploração *(High degree of exploration)*;
- Grau médio de exploração *(Medium degree of exploration)*;
- Baixo grau de exploração *(Low degree of exploration)*;
- Totalmente com *script (Fully scripted)*.

Estilo livre *(Freestyle)*: o objeto de teste é fornecido ao testador para que ele possa explorar livremente o sistema.

Alto grau de exploração *(High degree of exploration)*: um ou mais grandes objetivos são fornecidos ao testador, porém o testador pode testar livremente também.

Grau médio de exploração *(Medium degree of exploration)*: São fornecidos um ou mais grandes objetivos, entretanto são fornecidas restrições. Podem ser fornecidas fronteiras para limitar o trabalho do testador como objetivos, prioridades, riscos, ferramentas ou funcionalidade que o testador precisa se concentrar e que precisam ser cobertas.

Baixo grau de exploração *(Low degree of exploration)*: Neste nível de exploração são fornecidas informações e é necessário que alguns passos de teste sejam seguidos e o testador é incentivado a escolher os dados de teste a serem usados nas etapas de teste.

Totalmente com *script (Fully scripted)*: é um teste onde são fornecidos os passos de teste e os dados de teste ao testador, desta maneira não sobra espaço para exploração.

(GHAZI et al., 2018) destaca que os níveis podem ser usados para auxiliar na estruturação das cartas/guias de teste. No estilo livre o testador recebe apenas o objeto de teste e é instigado a explorar livremente. A cada nível que se segue é adicionado um elemento que define o escopo do espaço de exploração para o testador e as informações são adicionadas a carta de teste, desta maneira, nos níveis seguintes, a liberdade de exploração vai sendo reduzida.

(GHAZI et al., 2018) conclui, que os testes exploratórios podem levar a diferentes resultados e que a questão não é se devemos ou não aplicá-los, mas sim quando devemos aplicá-los e qual nível de teste exploratório devemos aplicar para alcançar o resultado almejados. Conclui também, através de um estudo com grupos focais que o estilo livre obteve uma melhor detecção de defeitos, economia de tempo e esforço. Também foi apontada uma facilidade de gerenciar mudanças e relativo a aprendizagem e motivação.

4 Aplicação

4.1 Proposta

Este Trabalho de Conclusão de Curso se propõem a aplicar Teste Exploratórios em um produto de software existente, a fim de avaliar a sua qualidade sob o ponto de vista das suas funcionalidades (foco funcional de avaliação). Neste sentido, será usada a teoria de testes exploratórios para identificar vantagens do seu uso. Além disso o trabalho inclui a elaboração de métricas para analisar não somente os defeitos encontrados, mas também o prazo, a taxa de identificação de defeitos, a dificuldade e o esforço gasto para a execução dos *tours* e distritos como previsto na abordagem de Whitakker para testes exploratórios. O produto de software usado como estudo de caso para a aplicação dos testes exploratórios e métricas associadas é descrito na sequência deste capítulo e constitui o objeto que será usado no estudo empírico da aplicação do trabalho.

(BACH, 2001), (ITKONEN; RAUTIAINEN, 2005), (GHAZI et al., 2018) e (WHITTAKER, 2010) entram em um consenso em relação ao fato da experiência do testador influenciar na execução dos teste exploratórios e nos resultados obtidos por eles, pois testadores mais experientes possuem uma certa facilidade de identificar possíveis regiões propensas a apresentarem defeitos, desta maneira os autores indicam a elaboração de uma carta de teste, esta carta seria como carta de navegação para um marinheiro ou um guia turístico para um turista, fazendo uma analogia a metáfora do turista proposta por (WHITTAKER, 2010).

4.2 PGTBL

O sistema PGTBL é um projeto *open source* desenvolvido por um discente da Universidade de Brasília. O objetivo do sistema é auxiliar nas atividades relativas a aplicação do TBL, tornando o processo mais fácil, constante e automatizado. O processo do TBL será abordado mais a frente.

O projeto é uma plataforma web e foi construído em uma abordagem ágil de desenvolvimento utilizando o *framework Django* e algumas métricas foram coletadas durante esse processo, são elas:

- Cobertura de Testes;
- Índice de qualidade (Certification);
- Quantidade de *issues* geradas pela ferramenta de análise estática;

Figura 9 – Logo marca do sistema PGTBL. Fonte: (PGTBL, 2019)

- *Churn*;
- Alterações de arquivo;
- Complexidade ciclomática;
- Quadro de conhecimento.

A arquitetura escolhida para a construção do sistema PGTBL foi o MVC (Model-View-Controller), entretanto esta arquitetura foi adaptada à tecnologia escolhida, *Django*. O *Django* possui uma arquitetura MVT(Model-View-Template) onde a *view* e a *controller* do MVC equivalem ao *template* e *view* do MVT, respectivamente. A figura 10 representa o esquemático da arquitetura do PGTBL.

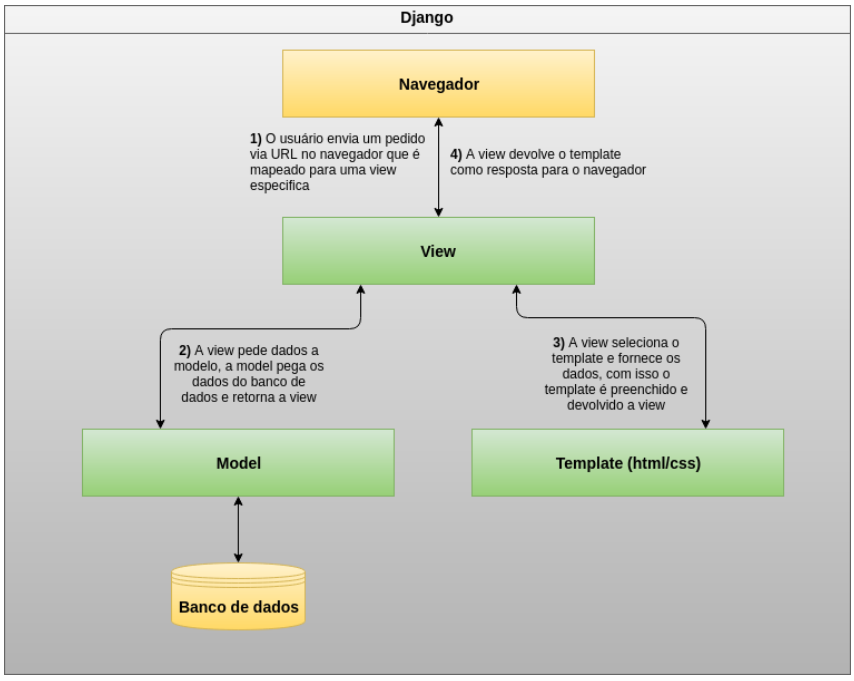

Figura 10 – Representação da arquitetura do sistema PGTBL. Fonte: (PGTBL, 2019)

O PGTBL buscou seguir alguns princípios e diretrizes de design de IHC, de forma a atender as necessidades dos usuários. Estes princípios foram:

- Correspondência com as expectativas dos usuários;
- Simplicidade nas estruturas das tarefas;
- Equilíbrio entre controle e liberdade do usuário;
- Consistência e Padronização;
- Promover a eficiência do usuário;
- Antecipação;
- Visibilidade e Reconhecimento;
- Conteúdo relevante e expressão adequadas;
- Projetos para erros e Mensagens de erro.

Nas figuras 11 e 12 estão representadas algumas interfaces do sistema PGTBL, onde estão ilustrados a página de perfil e a página de *rank* dos grupos.

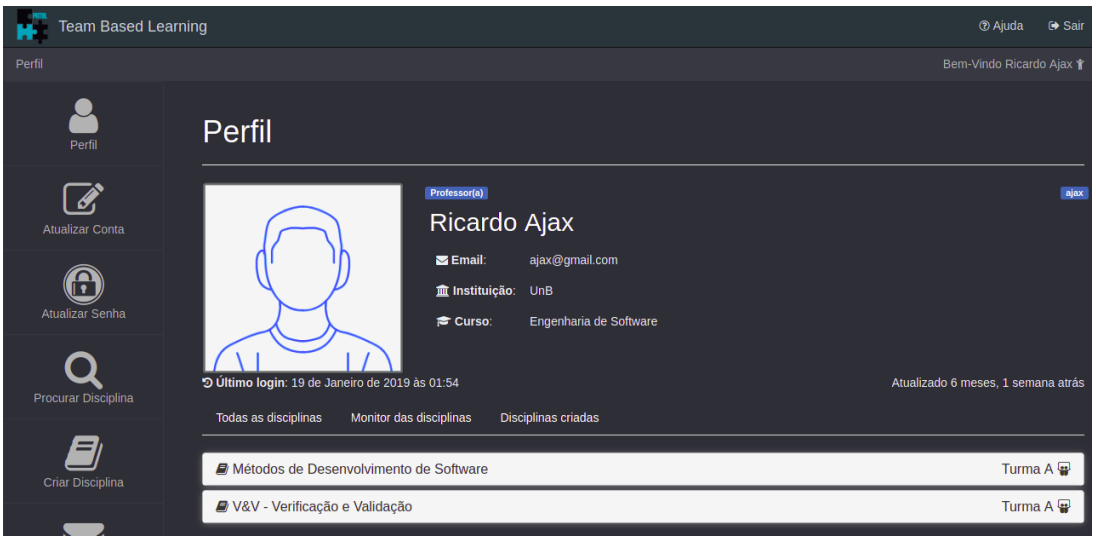

Figura 11 – Perfil. Fonte: (PGTBL, 2019)

O projeto foi construído como parte de um trabalho de conclusão de curso que se iniciou em 2018 e foi concluído no primeiro semestre de 2019. O processo utilizado foi baseado no *SAFe*, possuindo três níveis: Portfólio, Programa e Time. No nível de portfólio foi coletado o tema de investimento, épicos e *enables* e as *features*, utilizando a técnica de elicitação *brainstorm*. No nível de programa foram criadas as histórias de usuário e seus critérios de aceitação. Já no nível de time foram planejadas as *sprints*. Durante esse processo foi gerado um *backlog* do produto PGTBL.

Figura 12 – PGTBL Rank. Fonte: (PGTBL, 2019)

4.2.1 *Backlog* PGTBL

Esta seção é destina a descrição do *backlog* do produto. As siglas de identificação são: EP referente ao termo "épico"; FE referente ao termo *"feature"* e US referente ao termo "user story".

4.2.1.1 **EP01: Home Page.**

FE01: O sistema deve ter uma página principal para quem não está logado com informações e notícias importantes do software.

[US02]: Eu como usuário, desejo ter uma página com as informações do software antes de eu me cadastrar.

4.2.1.2 **EP02: Autenticação e autorização.**

FE02: O sistema deve permitir que usuário mantenha uma conta.

[US06] - Eu como usuário, desejo me cadastrar no sistema TBL.

[US07] - Eu como usuário, desejo com as minhas credenciais de cadastro, logar no sistema TBL

[US08] - Eu como usuário, desejo me deslogar do sistema TBL

[US10] - Eu como usuário desejo visualizar as minhas informações de perfil

[US11] - Eu como usuário desejo editar minhas informações pessoais e senha

[US12] - Eu como usuário desejo deletar minha conta

FE03: O sistema deve identificar e dar suas respectivas permissões a usuários cadastrados como professor ou aluno.

[US09] - Eu como administrador, desejo saber se o usuário é professor ou aluno

4.2.1.3 **EP03: Administração de Disciplinas e turmas.**

FE04: O sistema deve permitir que o professor mantenha a disciplina.

[US13] - Eu como professor, desejo deletar turmas de uma determinada disciplina que irei ministrar.

[US14] - Eu como usuário, desejo visualizar informações da disciplina que faço parte ou criei.

[US15] - Eu como professor, desejo editar turmas de uma determinada disciplina que irei ministrar.

[US16] - Eu como professor, desejo criar turmas de uma determinada disciplina que irei ministrar.

[US19] - Eu como professor deseja ter a opção de fechar a turma sem ela ter sido completada.

[US20] - Eu como professor, desejo deletar alunos de uma turma de uma disciplinas que criei.

[US21] - Eu como professor, desejo inserir alunos de uma turma de uma disciplinas que criei.

[US22] - Eu como usuário, desejo visualizar alunos de uma turma de uma disciplina que faço parte ou criei.

[US24] - Eu como aluno, desejo procurar disciplinas disponibilizadas pelo professor.

FE05: O sistema deve permitir que o professor mantenha a turma.

[US13] - Eu como professor, desejo deletar turmas de uma determinada disciplina que irei ministrar.

[US14] - Eu como usuário, desejo visualizar informações da disciplina que faço parte ou criei.

[US15] - Eu como professor, desejo editar turmas de uma determinada disciplina que irei ministrar.

[US16] - Eu como professor, desejo criar turmas de uma determinada disciplina que irei ministrar.

[US19] - Eu como professor deseja ter a opção de fechar a turma sem ela ter sido completada.

[US20] - Eu como professor, desejo deletar alunos de uma turma de uma disciplinas que criei.

[US21] - Eu como professor, desejo inserir alunos de uma turma de uma disciplinas que criei.

[US22] - Eu como usuário, desejo visualizar alunos de uma turma de uma disciplina que faço parte ou criei.

[US24] - Eu como aluno, desejo procurar disciplinas disponibilizadas pelo professor.

FE06: O sistema deve permitir que o aluno entre em uma turma de uma disciplina disponibilizado pelo professor.

[US20] - Eu como professor, desejo deletar alunos de uma turma de uma disciplinas que criei.

[US21] - Eu como professor, desejo inserir alunos de uma turma de uma disciplinas que criei.

[US22] - Eu como usuário, desejo visualizar alunos de uma turma de uma disciplina que faço parte ou criei.

[US23] - Eu como usuário, desejo entrar em uma turma de uma disciplinas disponibilizadas pelo professor, passando a senha que o professor disponibiliza.

[US24] - Eu como aluno, desejo procurar disciplinas disponibilizadas pelo professor.

FE09: O sistema deve ter um local para manter arquivos para estudo da disciplina e uma lista de exercício.

[US27] - Eu como professor desejo inserir arquivos da disciplina para download

[US28] - Eu como professor desejo editar algum arquivo.

[US29] - Eu como professor desejo deletar um arquivo.

[US30] - Eu como aluno desejo visualizar todos os arquivos e baixa-los,

[US35] - Eu como professor desejo criar alternativas para as questões

[US36] - Eu como professor desejo editar as alternativas

[US37] - Eu como professor desejo visualizar as alternativas

[US38] - Eu como professor desejo criar questões para as avaliações e lista de exercícios.

[US39] - Eu como professor desejo editar questões.

[US40] - Eu como professor desejo deletar uma questão.

[US41] - Eu como usuário desejo visualizar e realizar a lista de exercício.

[US42] - Eu como professor desejo visualizar a lista de questões para inserção nas avaliações.

4.2.1.4 **EP04: Administração de alunos e grupos.**

[FE06] - O sistema deve permitir que o aluno entre em uma turma de uma disciplina disponibilizado pelo professor.

[US20] - Eu como professor, desejo deletar alunos de uma turma de uma disciplinas que criei.

[US21] - Eu como professor, desejo inserir alunos de uma turma de uma disciplinas que criei.

[US22] - Eu como usuário, desejo visualizar alunos de uma turma de uma disciplina que faço parte ou criei.

[US23] - Eu como usuário, desejo entrar em uma turma de uma disciplinas disponibilizadas pelo professor, passando a senha que o professor disponibiliza.

[US24] - Eu como aluno, desejo procurar disciplinas disponibilizadas pelo professor.

FE07: O sistema deve criar os grupos e permitir que o professor mantenha os grupos.

[US19] - Eu como professor desejo que os grupos sejam feitos assim que a disciplina for fechada.

[US20] - Eu como professor desejo criar novos grupos.

[US22] - Eu como professor desejo editar os grupo de alunos da disciplina.

[US23] - Eu como professor desejo retirar aluno de algum grupo.

[US24] - Eu como professor deseja deletar um grupo.

[US25] - Eu como professor desejo inserir alunos em um grupo.

[US26] - Eu como professor desejo disponibilizar os grupos para os alunos

4.2.1.5 **EP05: Administração de sessões do TBL.**

[FE08] - O sistema deve permitir que o professor mantenha sessões do TBL.

[US31] - Eu como estudante e professor desejo visualizar as sessões de TBL

[US32] - Eu como professor desejo criar novas sessões de TBL.

[US33] - Eu como professor desejo editar as sessões de TBL.

[US34] - Eu como professor desejo deletar uma sessão de TBL

4.2.1.6 **EP06: Preparação.**

[FE09] - O sistema deve ter um local para manter arquivos para estudo da disciplina e uma lista de exercício.

[US27] - Eu como professor desejo inserir arquivos da disciplina para download.

[US28] - Eu como professor desejo editar algum arquivo.

[US29] - Eu como professor desejo deletar um arquivo.

[US30] - Eu como aluno desejo visualizar todos os arquivos e baixa-los.

[US35] - Eu como professor desejo criar alternativas para as questões.

[US36] - Eu como professor desejo editar as alternativas.

[US37] - Eu como professor desejo visualizar as alternativas.

[US38] - Eu como professor desejo criar questões para as avaliações e lista de exercícios.

[US39] - Eu como professor desejo editar questões.

[US40] - Eu como professor desejo deletar uma questão.

[US41] - Eu como usuário desejo visualizar e realizar a lista de exercício.

[US42] - Eu como professor desejo visualizar a lista de questões para inserção nas avaliações.

4.2.1.7 **EP07: Garantia de preparo individual ou iRAT.**

[FE10] Avaliação iRAT.

[US43] - Eu como professor desejo criar a avaliação iRAT.

[US38] - Eu como professor desejo criar questões para as avaliações e lista de exercícios.

[US39] - Eu como professor desejo editar questões.

[US40] - Eu como professor desejo deletar uma questão.

[US42] - Eu como professor desejo visualizar a lista de questões para inserção nas avaliações.

[US35] - Eu como professor desejo criar alternativas para as questões.

[US36] - Eu como professor desejo editar as alternativas.

[US37] - Eu como professor desejo visualizar as alternativas.

[US44] - Eu como professor desejo editar a avaliação iRAT.

[US46] - Eu como aluno desejo ver a nota da avaliação que realizei.

[US45] - Eu como usuário desejo visualizar a prova.

[US47] - Eu como aluno desejo realizar a avaliação iRAT.

[US47] Eu como professor desejo editar as notas das avaliações dos alunos.

[US48] Eu como professor desejo visualizar e editar as questões da avaliação gRAT.

4.2.1.8 **EP08: Garantia de preparo em grupo ou gRAT.**

FE11: Avaliação gRAT.

[US43] - Eu como professor desejo criar a avaliação iRAT.

[US38] - Eu como professor desejo criar questões para as avaliações e lista de exercícios.

[US39] - Eu como professor desejo editar questões.

[US40] - Eu como professor desejo deletar uma questão.

[US42] - Eu como professor desejo visualizar a lista de questões para inserção nas avaliações.

[US35] - Eu como professor desejo criar alternativas para as questões.

[US36] - Eu como professor desejo editar as alternativas.

[US37] - Eu como professor desejo visualizar as alternativas.

[US44] - Eu como professor desejo editar a avaliação iRAT.

[US45] - Eu como usuário desejo visualizar a prova.

[US46] - Eu como aluno desejo ver a nota da avaliação que realizei.

[US47] - Eu como aluno desejo realizar a avaliação iRAT.

[US47] - Eu como professor desejo editar as notas das avaliações dos alunos.

[US48] - Eu como professor desejo visualizar e editar as questões da avaliação gRAT.

[US49] - Eu como aluno desejo responder a avaliação gRAT.

4.2.1.9 **EP09: Aplicação de conceitos.**

[FE12] - Avaliação prática e avaliação em pares.

[US47] Eu como professor desejo editar as notas das avaliações dos alunos.

[US48] Eu como professor desejo visualizar e editar as questões da avaliação gRAT.

[US50] Eu como professor/aluno desejo criar/editar/visualizar a avaliação prática.

[US51] Eu como professor desejo criar e editar a avaliação em pares.

[US52] - Eu como aluno de um grupo desejo avaliar meus colegas.

4.2.1.10 **EP10: Avaliação em pares.**

[FE12] - Avaliação prática e avaliação em pares.

[US47] Eu como professor desejo editar as notas das avaliações dos alunos.

[US48] Eu como professor desejo visualizar e editar as questões da avaliação gRAT.

[US50] Eu como professor/aluno desejo criar/editar/visualizar a avaliação prática.

[US51] Eu como professor desejo criar e editar a avaliação em pares.

[US52] - Eu como aluno de um grupo desejo avaliar meus colegas.

4.2.1.11 **EP11: Apelação e Notificação.**

FE15: Apelação.

[US56] - Lista de apelações.

[US57] - Criando uma apelação.

[US58] - Detalhes da apelação.

[US59] - Discussão da apelação.

FE16: Mensagens e Notificações.

[US60] - Criando Mensagens (Fórum).

[US61] - Lista de notificações.

[US62] - Observadores de notificações.

4.2.1.12 **EP12: Relatório**

FE13: Relatório.

[US55] - Relatórios em forma de gráfico.

4.2.1.13 **EP13: Gameficação.**

FE14: Gameficação.

[US53] - Finalizar sessão de TBL.

[US54] - Resetando uma disciplina.

4.3 TBL - Aprendizado Baseado em Equipe

O TBL (do inglês *Team Based Learning*) surgiu nos anos 70 e foi desenvolvido por Larry Michaelsen, no curso de Administração. O TBL constitui uma metodologia de aprendizagem em equipe, voltada para grandes turmas, onde o professor coordena atividades de capacitação dos alunos (RAMOS C. S. ; KOSLOSKI, 2018). Essa abordagem é dividida em três principais fases:

1. Preparação pré-classe;
2. Avaliação de prontidão (ou garantia de preparo);
3. Aplicação de conceitos;

Essas fases do ciclo estão ilustradas na figura 11:

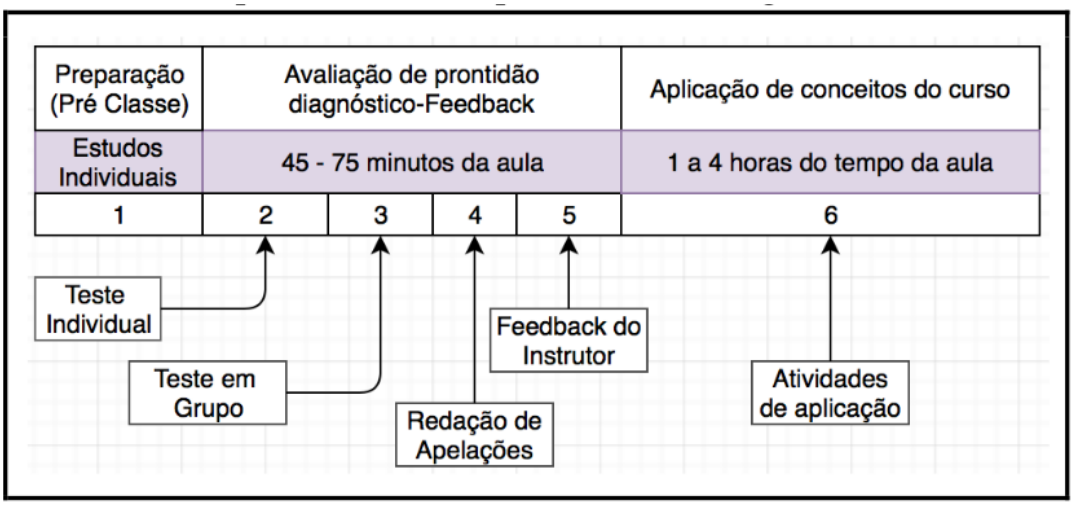

Figura 13 – Processo TBL. Fonte: (RAMOS C. S. ; KOSLOSKI, 2018)

Essa metodologia visa incentivar o pensamento crítico do aluno e fomentar debates relacionados à disciplina estudada. Visa também a socialização, trabalho em grupo e a responsabilização do aluno pela própria aprendizagem, além de disponibilizar *feedbacks* constantes aos participantes (RAMOS C. S. ; KOSLOSKI, 2018).

A metodologia TBL foi aplicada na Universidade de Brasília campus Gama em diversas disciplinas, como: IE - Introdução à Engenharias; VVS - Verificação e validação de Software; REQ - Requisitos de Software; MED - Medição e Análise e MPS - Melhoria de Processo de Software, obtendo resultados interessantes, entretanto o processo era realizado de forma não automatizada. O PGTBL foi desenvolvido justamente para auxiliar e facilitar esse processo, automatizando-o. (RAMOS C. S. ; KOSLOSKI, 2018)

4.4 Divisões do Sistema PGTBL

O sistema PGTBL possui seis principais atividades: Cadastro de usuário; Criação de Turma; Preparação; Garantia de preparo ou RAT; Aplicação de conceitos e avaliações em pares; e Apelação. Estas atividades estam ilustradas na Figura 14.

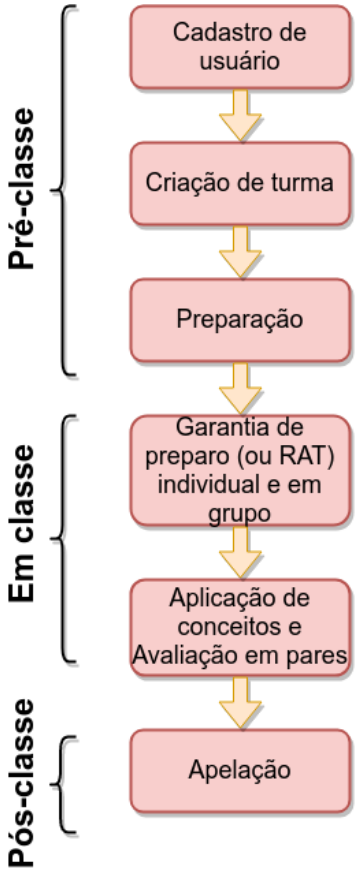

Figura 14 – PGTBL - Atividades. Fonte: Autor

Essas atividades estão muito bem divididas em relação ao processo de aplicação de um *TBL* convencional com etapas pré-classe, em classe e pós-classe, e torna-se necessário que sua sequência seja respeitada, visto que cada atividade que se segue tem como pré-requisito o cumprimento atividade anterior.

4.4.1 Cadastro de usuário

Esta atividade é composta por funcionalidades relativas ao épico EP02 de autenticação e autorização e as *features* FE02 onde o sistema deve permitir que usuário mantenha uma conta e FE03 onde o sistema deve identificar e dar suas respectivas permissões a usuários cadastrados como professor ou aluno.

A Tabela 7 está dividida em três colunas. Uma coluna com o identificador de épico, uma coluna com identificador de *features*, e uma coluna dedicadas às histórias de usuário. Nesta atividade ainda não existe uma distinção de usuários entre professor ou

aluno. Entretanto a *features* FE03 é voltada a um usuário administrador do sistema. Tais características estão detalhadas no *backlog* **??** do produto.

Tabela 7 – Cadastro de usuário

ID Épico	**ID Feature**	**ID user store**
EP02	FE02	[US06] [US07] [US08] [US10] [US11] [US12]
EP02	FE03	[US09]

4.4.2 Criação de turma

Esta atividade é composta por funcionalidades relativas ao épico EP03 relativa à administração de disciplinas e turmas, e as *features* FE04 onde O sistema deve permitir que o professor mantenha a disciplina. A *features* FE05 onde o sistema deve permitir que o professor mantenha a turma. Também é composta pelo épico EP04 que é sobre a Administração de alunos e grupos. Dela fazem parte a *features* FE06 onde o sistema deve permitir que o aluno entre em uma turma de uma disciplina disponibilizado pelo professor. A *feature* FE07 onde o sistema deve criar os grupos e permitir que o professor mantenha os grupos. A *features* FE09 onde o sistema deve ter um local para manter arquivos para estudo da disciplina e uma lista de exercício. Tais características estão detalhadas no *backlog* **??** do produto.

A Tabela 8 está dividida em quatro colunas. Uma coluna com o identificador de épico, uma coluna com identificador de *features*, e duas colunas dedicadas às histórias de usuário, onde possuem US voltada a professor e outra para alunos.

As *features* FE04 e FE05 possuem as mesmas histórias de usuário, logo, aparecem na tabela na mesma linha. A *features* FE06 possui todas a histórias de usuário que FE04 e FE05, entretanto possui uma história de usuário a mais.

As histórias de usuário que aparecem tanto na coluna de professor como na coluna de alunos, significa que foi indicada na documentação do PGTBL como um usuário genérico.

A *features* FE09 pertencente ao épico EP03 também pertence ao épico EP06 relativa a atividade de preparação, logo, por questão de organização, será abordada da tabela da atividade de Preparação.

4.4.3 Preparação

Esta atividade é composta por funcionalidades relativas ao épico EP05 que trata da Administração de sessões do TBL, com a *feature* FE08 relativo ao sistema permitir que o professor mantenha sessões TBL. E do épico EP06 relativo a Preparação, com a *feature*

Tabela 8 – Criação de turma

ID Épico	**ID *Feature***	**ID US professor**	**ID US aluno**
EP03	FE04 / FE05	[US13] [US14] [US15] [US16] [US19] [US20] [US21] [US22]	[US14] [US24]
EP04	FE06	[US23]	[US23]
EP04	FE07	[US20] [US22] [US23] [US24] [US25] [US26]	-

FE09 onde o sistema deve ter um local para manter arquivos para estudo da disciplina e uma lista de exercício. Tais características estão detalhadas no *backlog* **??** do produto.

A Tabela 9 está dividida em quatro colunas. Uma coluna com o identificador de épico, uma coluna com identificador de *feature*, e duas colunas dedicadas às histórias de usuário, onde possuem US voltada a professor e outra para alunos.

Tabela 9 – Preparação

ID Épico	**ID Feature**	**ID US professor**	**ID US aluno**
EP05	FE08	[US31] [US32] [US33] [US34]	-
EP06	FE09	[US27] [US28] [US29] [US35] [US36] [US37] [US38] [US39] [US40] [US41] [US42]	[US30] [US41]

4.4.4 Garantia de preparo (ou RAT) individual e em grupo

Esta atividade está dividida em dois épicos do sistema PGTBL a serem testados. O épico EP07 relativo à garantia de preparo individual ou iRAT e o épico EP08 relativo à garantia de preparo em grupo ou gRAT. Esses épicos abrangem as *features* FE10 relativa a avaliação iRAT e a *features* FE11 relativa a avaliação gRAT. Tais características estão detalhadas no *backlog* **??** do produto.

A Tabela 10 está dividida em quatro colunas. Uma coluna com o identificador de épico, uma coluna com identificador de *features*, e duas colunas dedicadas às histórias de usuário, onde possuem US voltada a professor e outra para alunos.

As histórias de usuário (US) que compreendem as *features* FE10 e FE11 são as mesmas, por esse motivo estão apresentadas na mesma linha. Uma peculiaridade a ser destacada na documentação do PGTBL é a duplicação do identificador US47, onde duas histórias de usuário distintas estão documentadas com o mesmo identificador. Logo foi necessário utilizar o acréscimo do identificador gerado pelo *github*, entre parênteses, para distinguí-las na tabela.

Tabela 10 – Garantia de preparo (ou RAT) individual e em grupo

ID Épico	**ID *Feature***	**ID US professor**	**ID US aluno**
EP07 / EP08	FE10 / FE11	[US35] [US36] [US37] [US38] [US39] [US40] [US42] [US43] [US44] [US45] [US48] [US47(TBL#94)]	[US46] [US45] [US47(TBL#91)]

4.4.5 Aplicação de conceitos e Avaliação em pares

Esta atividade é composta por funcionalidades relativas ao épico EP09 relativa a aplicação de conceitos, ao épico EP10 referente a avaliação em pares e a *feature* FE12 referente a avaliação prática e avaliação em pares, que abrange os dois épicos. Tais características estão detalhadas no *backlog* **??** do produto.

A Tabela 11 está dividida em quatro colunas. Uma coluna com o identificador de épico, uma coluna com identificador de *feature*, e duas colunas dedicadas às histórias de usuário, onde possuem US voltada a professor e outra para alunos.

Tabela 11 – Aplicação de conceitos e Avaliação em pares

ID Épico	**ID *Feature***	**ID US professor**	**ID US aluno**
EP09 / EP10	FE12	[US47] [US48] [US50] [US51]	[US50] [US52]

4.4.6 Apelação

Esta atividade é composta pelo épico EP11 referente à apelação e notificação, entretanto apenas a *feature* FE15 de Apelação será abrangida. Tais características estão detalhadas no *backlog* **??** do produto.

A Tabela 12 está dividida em quatro colunas. Uma coluna com o identificador de épico, uma coluna com identificador de *feature*, e duas colunas dedicadas às histórias de usuário, onde possuem US voltada a professor e outra para alunos.

Tabela 12 – Apelação

ID Épico	**ID *Feature***	**ID US professor**	**ID US aluno**
EP11	FE15	[US56] [US57] [US58] [US59]	[US56] [US57] [US58] [US59]

4.5 O que será testado?

Para este Trabalho de conclusão de curso foram selecionadas três atividades do sistema *PGTBL* para que possam ser testadas.

Foram selecionadas as três primeiras atividades do sistema *PGTBL*, Cadastro de usuário e Criação de turma, e Preparação, fechando a fase pré-classe do sistema, destacadas na Figura 15, tendo em vista que as atividades subsequentes do sistema possuem a pré-condição da atividade anterior ser satisfeita e que o *PGTBL* ainda não possui nenhum teste realizado.

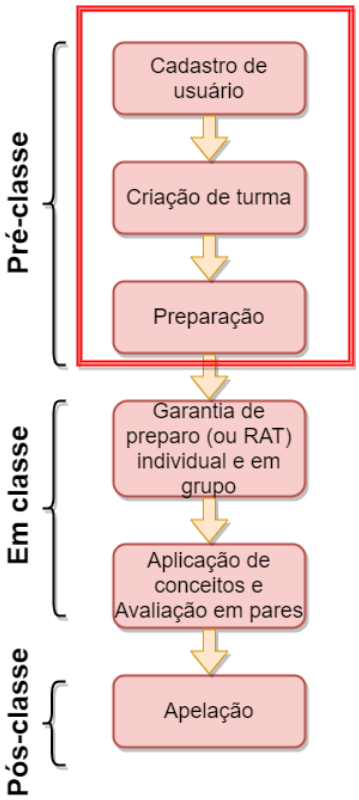

Figura 15 – PGTBL - Seleção das atividades a serem testadas Fonte: Autor

Selecionar apenas três partes do sistema possui o intuito de gerar mais empenho por parte dos testadores ao explorarem o software, obtendo assim melhores resultados.

4.6 Objetivos de medição

Nesta seção estão descritos os objetivos de medição, bem como as métricas escolhidas a partir do uso da técnica *GQM*.

Nas Tabelas 13 e 14 estão descritos os objetivos de medição, o que será analisado, o propósito, o ponto de vista e o contexto, para que possamos derivar questões e métricas direcionadas para o sistema selecionado.

Tabela 13 – Definição dos objetivos de medição

#	**Objetivo de medição**	**Definição**
1	Conhecer a qualidade do produto de software.	Avaliar se o sistema atende aos requisitos funcionais do sistema após uma avaliação usando testes de software.
2	Conhecer o tempo empregado pelos testadores.	Conhecer o esforço e o prazo empregado pelos testadores ao aplicar os testes.
3	Conhecer a dificuldade percebida pelos participantes ao aplicar os testes.	Corresponde a dificuldade que um testador percebe ao aplicar testes.

Tabela 14 – Objetivos de medição

Analisar:	O produto de software
Com o propósito de:	conhecer
Objeto(s):	1. - Qualidade do produto; 2. - Tempo empregado; 3. - Dificuldade percebida.
Ponto de Vista da:	funcional
No contexto de:	Sistema PGTBL

Objetivo 1 - Conhecer a qualidade do produto de software.

A figura 16, ilustra o objetivo 1 suas questões e métricas.

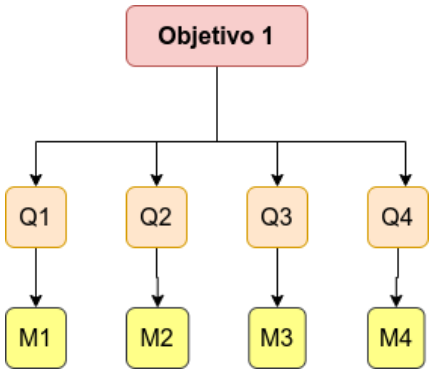

Figura 16 – Objetivo 1, questões e métricas Fonte: Autor

Questões

Q1. Qual a quantidade de defeitos identificados utilizando testes exploratórios de software?

Q2. Qual a quantidade de defeitos identificados utilizando teste clássico/tradicional?

Q3. Qual a taxa de defeitos é identificada por *tour*?

Q4. Qual a taxa de identificação de defeitos em um teste clássico/tradicional?

Métricas

M1. Quantidade de defeitos de um *tour* (Tabela 15)

M2. Quantidade de defeitos (Tabela 16)

M3. Taxa de identificação de defeitos por *tour* (Tabela 17)

M4. Taxa de identificação de defeitos (Tabela 18)

Objetivo 2 - Conhecer o tempo empregado pelos testadores.

A figura 17, ilustra o objetivo 2 suas questões e métricas.

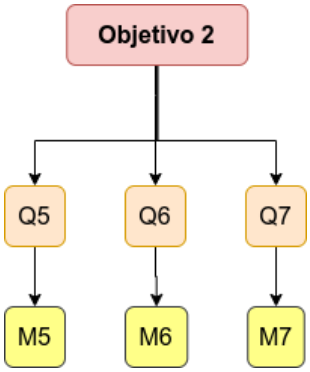

Figura 17 – Objetivo 2, questões e métricas Fonte: Autor

Questões

Q5. Qual é o prazo gasto para realizar os testes?

Q6. Qual o esforço gasto na execução de uma *tour*?

Q7. Qual o esforço gasto na execução de teste clássico/tradicional?

Métricas

M5. Prazo (Tabela 19)

M6. Esforço por *tour*(Tabela 20)

M7. Esforço por teste(Tabela 21)

Objetivo 3 - Conhecer a dificuldade percebida pelos participantes ao aplicar os testes.

A figura 18 ilustra o objetivo 3 suas questões e métricas.

Q8. Qual a dificuldade percebida pelos testadores ao realizar os testes?

Métrica

M8. Dificuldade (Tabela 22)

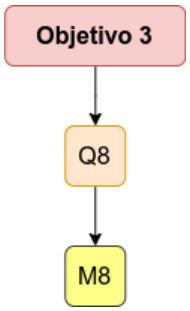

Figura 18 – Objetivo 3, questões e métricas Fonte: Autor

Tabela 15 – M1. Quantidade de defeitos de um *tour*

Objetivo de Medição	Conhecer a qualidade do produto de software
Fórmula	$DefTour = \sum_{i=1}^{n} Defeito_i$ **DefTour** = Corresponde ao somatório dos defeitos encontrados em *tours*. **n**: número de defeitos identificados. **i**: é uma observação individual.
Escala de Medição	Escala absoluta representada por um número inteiro.
Coleta	A coleta dos defeitos é feita pelos testadores participantes, de acordo com a *tour* aplicada.
Análise	A análise é feita a partir da exposição do valor absoluto encontrado na coleta.

Tabela 16 – M2. Quantidade de defeitos

Objetivo de Medição	Conhecer a qualidade do produto de software
Fórmula	$DefTsT = \sum_{i=1}^{n} Defeito_i$ **DefTsT** = Corresponde ao somatório dos defeitos encontrados na aplicação dos testes. **n**: número de defeitos identificados. **i**: é uma observação individual.
Escala de Medição	Escala absoluta.
Coleta	A coleta dos defeitos é feita pelos testadores participantes.
Análise	A análise é feita a partir da exposição do valor absoluto encontrado na coleta.

Tabela 17 – M3. Taxa de Identificação de defeitos por tour

Objetivo de Medição	Conhecer a qualidade do produto de software *tour*.
Fórmula	$TaxDef = \frac{QtdDef}{QtdTour}$ **TaxDef**: taxa de identificação defeitos; **QtdDef**: quantidade de defeitos; **QtdTour**: quantidade de tours.
Escala de Medição	Escala de medição racional.
Coleta	A coleta da quantidade de *tour* e da quantidade de defeitos é feita pelos testadores após as explorações.
Análise	A análise é feita a partir do resultado obtido na coleta, identificando qual é o *tour* que está identificando mais defeitos na exploração.

Tabela 18 – M4. Taxa de Identificação de defeitos

Objetivo de Medição	Conhecer a qualidade do produto de software *tour*.
Fórmula	$TaxDef = \frac{QtdDef}{QtdTest}$ **TaxDef**: taxa de identificação defeitos; **QtdDef**: quantidade de defeitos; **QtdTest**: quantidade de testes.
Escala de Medição	Escala de medição racional.
Coleta	A coleta da quantidade de testes e da quantidade de defeitos é feita pelos testadores após as explorações.
Análise	A análise é feita a partir do resultado obtido na coleta, identificando qual a taxa de defeitos por teste.

Tabela 19 – M5. Prazo

Objetivo de Medição	Conhecer o tempo empregado pelos testadores.
Fórmula	$PrzTst = \sum_{i=1}^{n} \Delta TstExc_i$ **PrzTst** : é o Prazo resultante da execução dos testes. **TstExc**: dias de execução de testes. **n**: número de testadores participantes. **i**: é uma observação individual.
Escala de Medição	absoluta
Coleta	A coleta será feita pelos testadores participantes, durante a execução dos testes.
Análise	A análise será feita a partir da exposição dos dados resultantes da coleta.

Tabela 20 – M6. Esforço por tour

Objetivo de Medição	Conhecer o tempo empregado pelos testadores
Fórmula	$EsfTour = \sum_{i=1}^{n} \frac{\Delta tempoExec_i}{tour}$ **EsfTour** : é o somatório da variação do tempo para a execução de *tours*. **tempoExec**: tempo de execução da tour. **tourEsp**: é uma tour especifica. **n**: número de testadores participantes. **i**: é uma observação individual.
Escala de Medição	intervalo
Coleta	A coleta é feita pelo testador durante a realização da exploração.
Análise	A análise será feita por meio da construção de um gráfico categorizando os dados encontrados em cada *tour*.

Tabela 21 – M7. Esforço por teste

Objetivo de Medição	Conhecer o tempo empregado pelos testadores
Fórmula	$EsfTest = \sum_{i=1}^{n} \frac{\Delta tempoExec_i}{teste}$ **EsfTest** : é o somatório da variação do tempo para a execução de um teste. **tempoExec**: tempo de execução de um teste. **n**: número de testadores participantes. **i**: é uma observação individual.
Escala de Medição	Intervalo
Coleta	A coleta é feita pelo testador durante a realização dos testes.
Análise	A análise será feita por meio da construção de um gráfico categorizando os dados encontrados em cada teste.

Tabela 22 – M8. Dificuldade

Objetivo de Medição	Conhecer a dificuldade percebida pelos participantes ao aplicar os testes.
Fórmula	Classificação: 1. Baixa 2. Regular 3. Média 4. Alta 5. Extrema
Escala de Medição	ordinal
Coleta	A coleta será feita através de uma entrevista ao final dos testes.
Análise	A análise será feita a partir da exposição dos dados resultantes da coleta.

5 Análise e Interpretação dos Dados

Este estudo empírico foi realizado com os alunos da disciplina de Testes de Software, ministrado na Universidade de Brasília. Este capítulo tem como objetivo realizar a análise e interpretação dos resultados obtidos na aplicação do Treinamento; Questionário sobre experiência; Aplicação dos testes; e das Entrevistas pós aplicação dos testes, que estão detalhados no capítulo 2 de Metodologia.

5.1 Treinamento

O treinamento foi dividido em 3 atividades. Na primeira atividade foi disponibilizada uma aula assíncrona sobre testes exploratórios. A segunda atividade foi constituída de uma aula síncrona, com o objetivo de tirar dúvidas dos alunos a cerca da aula gravada. A terceira etapa foi a disponibilização de uma lista de exercícios, individual, contendo dez questões para a consolidação do estudo.

O treinamento foi realizado no final de outubro de 2020. Como estamos em tempos de pandemia de *covid-19*, todas as etapas do treinamento foram realizadas de forma remota. A turma da disciplina de Testes de Software é constituída de cinquenta e seis alunos. Cerca de 48 alunos participaram das três etapas do treinamento com uma média geral de acertos de 9,5 na lista de exercícios.

A fase de treinamento foi considerada importante, pois buscou nivelar o conhecimento dos participantes a respeito de testes exploratórios. Embora a maioria dos participantes tenham atuado nesta fase, o desejável seria uma adesão integral dos participantes, visando amenizar os impactos causados pela falta de experiência.

5.2 Questionário de experiência

No mês de outubro foi realizada um encontro síncrono com os participantes, onde foi explicado como aconteceria a parte prática do estudo. Neste encontro foram passadas orientações acrescidas de documentações explicativas de todos as etapas da parte prática do estudo.

A fase de questionário foi realizada antes da aplicação prática dos testes pelos participantes e tinha como objetivo entender a experiência deles. O questionário foi elaborado utilizando a ferramenta *Forms* do *Google* e cerca de 48 alunos participaram dessa fase.

Ao avaliarmos o resultado dos questionários observamos que, metade dos alunos, que participaram dessa fase, não possuíam experiência alguma em testes de software. Dos

alunos que possuíam alguma experiência, um aluno relatou possuir mais de quatro anos de experiência, cinco alunos possuíam mais de um ano de experiência e cerca de dezenove alunos possuíam até um ano de experiência.

Em relação à experiência em testes exploratórios, especificamente, os resultados foram que, cerca de trinta e três alunos afirmaram que já conheciam a técnica de testes exploratórios, entretanto, apenas um participante afirmou já ter utilizado a técnica e possuir experiência de até um ano.

A aplicação de questionários teve uma boa aderência da turma. A participação nos questionários é importante para entender o impacto da experiência nas dificuldades percebidas pelos participantes.

5.3 Aplicação dos Testes

A aplicação dos testes teve como objetivo coletar dados que pudessem ser utilizados para identificar vantagens do uso de testes exploratórios. Para tanto, os alunos da disciplina de Testes de Software foram divididos em dois temas. Testes exploratórios e testes tradicionais. Esta divisão foi feita para que houvesse um parâmetro a respeito das dificuldades relatadas. Sete grupos participaram desta fase, contendo de sete à dez integrantes em cada grupo. Foi preparada um plano de testes para os grupos, abordando o funcionamento do sistema a ser testado, o PGTBL, as instruções para instalar o ambiente e foi apresentado o foco de testes do estudo. Um diferencial entre as documentações de cada tema eram, a presença de casos de teste, para os grupos que ficaram com testes tradicionais, e a presença de uma carta guia de exploração, para os grupos que ficaram com testes exploratórios. Os grupos também receberam uma planilha com um modelo para documentar os resultados obtidos.

A aplicação dos testes teve início logo após a aplicação dos questionários, onde os grupos tinham o período de uma semana para aprender mais a fundo sobre o sistema PGTBL, instalar o ambiente, realizar os testes e documentar os resultados. Na tabela 23, está a divisão dos grupos, o número de participantes e o tema de cada grupo. Notem que não existe o grupo 4, pois houve a dissolução do grupo, quando a maior parte dos alunos deste grupo desistiu da disciplina. Para não gerar um conflito de informações entre os grupos informado nas gravações das entrevistas e uma possível reorganização dos grupos, foi necessário mantê-los com suas numerações originais.

Na planilha de documentação dos testes os participantes deveriam preencher dados específicos, como: identificador do teste (ID), um nome para o teste que resumisse o objetivo do teste, as entradas utilizadas no teste, os procedimentos utilizados, as saídas retornadas, o número de defeitos encontrados, a descrição do defeito, a data de realização do teste, a hora de início e final do teste. Estes dados eram em comum aos dois temas de

Tabela 23 – Grupos e temas

Grupo	**Nº de participantes**	**Tipo de teste**
1	7	Tradicional
2	8	Tradicional
5	8	Tradicional
3	10	Exploratório
6	7	Exploratório
7	8	Exploratório
8	8	Exploratório

teste, entretanto aos grupos que realizaram testes exploratórios haviam mais dois itens a reportar, que eram o tipo de *tour* e o distrito dos testes aplicados.

5.3.1 Testes clássicos/tradicionais

A Tabela 24 apresenta os resultados obtidos com os grupos que trabalharam com testes tradicionais. Baseado nas métricas definidas no capítulo 4 de Aplicação, podemos analisar a quantidade de defeitos, taxa de defeitos, o esforço e o prazo.

Tabela 24 – Dados resultantes dos testes tradicionais

Grupo	quantidade de testes aplicados	total defeitos	taxa defeitos	media de esforço por testes (em horas)	Esforço total (em horas)	Prazo (em dias)
1	48	14	0,29	0:02:09	1:43:00	2 dias (22 à 23 out)
2	48	15	0,31	0:01:00	1:23:00	4 dias (19 à 22 out)
5	42	15	0,36	0:03:23	2:22:00	5 dias (21 à 25 out)

Os grupos que trabalharam com testes tradicionais receberam 48 casos de teste. Como é possível observar na tabela 24, apenas o grupo 5 realizou menos testes que o previsto, todavia, a quantidade de defeitos relatados pelos grupos foi bem equilibrado, variando entre 14 e 15 defeitos. Logo, resultou em valores de taxa de defeitos bem similares, variando entre 0,29 e 0,35. Os grupos receberam o período de uma semana para realizarem a atividade de teste, dos quais usaram ao menos dois dias para concluírem a atividade. Entretanto, os grupos deste tema necessitaram de uma média de 3,6 dias para realizarem os teste. Os grupos necessitaram de uma média de uma hora e quarenta e nove minutos para concluírem essa fase.

5.3.1.1 Q2. Qual a quantidade de defeitos identificados utilizando teste clássico/tradicional?

5.3.1.1.1 M2. Quantidade de defeitos

Ao analisarmos a tabela 24, observar que média de defeitos encontradas pelos grupos foi de 14,6 defeitos.

5.3.1.2 Q4. Qual a taxa de identificação de defeitos em um teste clássico/tradicional?

5.3.1.2.1 M4. Taxa de identificação de defeitos

Ao analisarmos a tabela 24, observamos que a média da taxa de identificação de defeitos encontradas pelos grupos foi de 0,32 defeitos.

5.3.1.3 Q5. Qual é o prazo gasto para realizar os testes?

5.3.1.3.1 M5. Prazo

Ao analisarmos a tabela 24, observamos que o Prazo média conclusão dos testes pelos grupos foi de 3,6 dias.

5.3.1.4 Q7. Qual o esforço gasto na execução de teste clássico/tradicional?

5.3.1.4.1 M7. Esforço por teste

Ao analisarmos a tabela 24, observamos que o Esforço médio para realizar um teste foi de dois minutos e onze segundos. Para a realização total dos testes o esforço foi em média de uma hora e quarenta e nove minutos.

5.3.2 Testes exploratórios

Os grupos que trabalharam com testes exploratórios receberam uma carta de exploração definindo partes do sistema que seriam obrigatórios à exploração, os deixando com um grau médio de exploração. Quatro grupos foram selecionados para trabalhar com testes exploratórios. O que podemos nos atentar é que existiu uma variação notável entre os dados relatados pelos grupos. Inicialmente observamos que o número de testes aplicados pelos grupos variou entre dezenove e oitenta e seis explorações.

A tabela 25 traz mais detalhes dos resultados gerais apresentados pelos grupos que trabalharam com testes exploratórios.

Dos possíveis tipos de *tours* propostos por Whittaker, os participantes conseguiram aplicar dezesseis tipos diferentes, foram eles: *tour* antissocial; *tour* ao Beco; *tour* pela chuva; *tour* coletor de Lixo; *tour* guiado; *tour* obsessivo compulsivo; *tour* preguiçoso;

Tabela 25 – Dados resultantes dos testes exploratórios

Grupo	número de testes aplicados	Total Defeitos	Taxa de Defeitos	Quantidade de *tours* diferentes	Esforço total	Esforço por teste	Prazo
3	43	12	0,28	10	8:55:00	0:13:03	5 dias (22 à 26 de out)
6	86	44	0,51	16	17:47:00	0:12:16	4 dias (21 à 24 out)
7	19	38	2	7	5:58:00	0:18:51	3 dias (24 à 26 de out)
8	76	13	0,17	8	5:25:00	0:04:17	5 dias (22 à 26 de out)

tour do sabotador; *tour* do super modelo; *tour* TOGOF; *tour* ao bar escocês; *tour* do colecionador; *tour* do empresário solitário; *tour* FedEx; *tour* intelectual e *tour* ao ponto de referência.

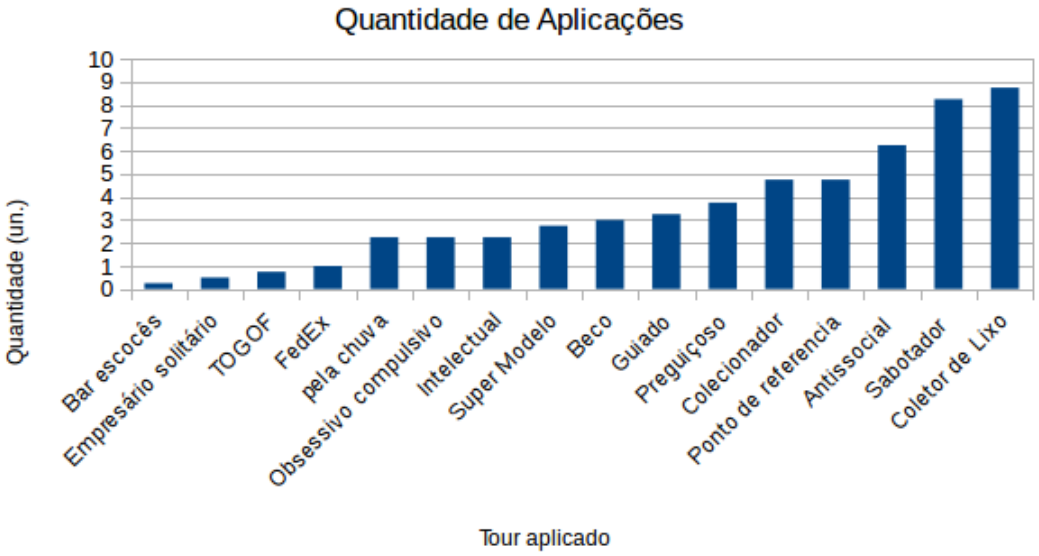

Figura 19 – Quantidade de aplicações por tour. Fonte: Autor

Na Figura 19 podemos avaliar a quantidade média de defeitos que cada *tour* identificou no sistema.Os valores no gráfico da imagem foram obtidos a partir da média de defeitos relatada pelos grupos para cada *tour*. O que podemos observar é que o *Tour* do Coletor de Lixo, com uma média de 8,75 vezes aplicado e o *Tour* do Sabotador, com uma

média de 8,5 vezes aplicado foram os mais *tours* de mais foram escolhidos entre os grupos.

5.3.2.1 Q1. Qual a quantidade de defeitos identificados utilizando testes exploratórios de software?

Analisando de modo geral, observando os dados da tabela 25, podemos observar que a média da quantidade de defeitos encontrada por grupo foi de 26,75 defeitos. Para gerarmos uma análise mais detalhada em relação a qual *tour* conseguiu melhores resultados, vamos utilizar a Métrica M1 "Quantidade de defeitos de um *tour*".

5.3.2.1.1 M1. Quantidade de defeitos de um tour

Na Figura 20 podemos avaliar a quantidade de defeitos identificados por *tour* aplicado. Os valores no gráfico da imagem foram obtidos a partir da média de defeitos relatada pelos grupos para cada *tour*. Podemos observar que os *tours* que apresentaram a maior quantidade de defeito foram: O *tours* do sabotador, com uma média de 6,5 defeitos, o *tours* antissocial, com uma média de 4,75 defeitos e o *tours* do colecionador, com uma média de 3,5 defeitos.

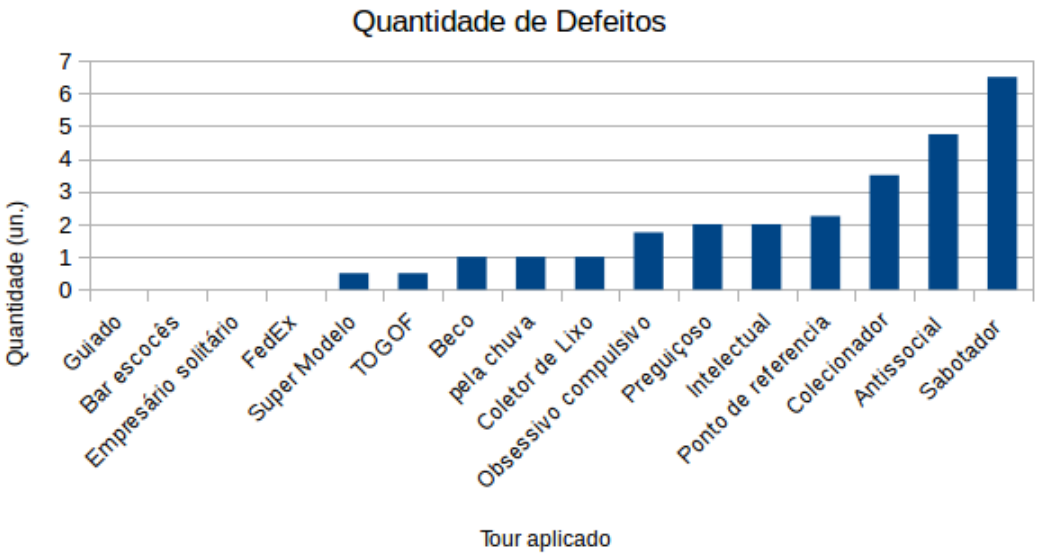

Figura 20 – Quantidade de defeitos por tour. Fonte: Autor

5.3.2.2 Q3. Qual a taxa de defeitos é identificada por tour?

Analisando de modo geral, observando os dados da tabela 25, podemos observar que a média da Taxa Defeitos encontrada por grupo foi de 0,74 defeitos por teste. Para gerarmos uma análise mais detalhada em relação a qual *tour* conseguiu melhores resultados, vamos utilizar a Métrica M2 "Taxa de Identificação de defeitos por *tour*".

5.3.2.2.1 M3. Taxa de identificação de defeitos por tour

Na Figura 21 podemos observar a taxa de identificação de defeitos por *tour*, desta maneira buscamos entender a efetividade dos *tours* aplicados. Neste contexto, os *tour* que obtiveram os melhores resultados foram o *tour* pela chuva, o *tour* do sabotador e o *tour* antissocial.

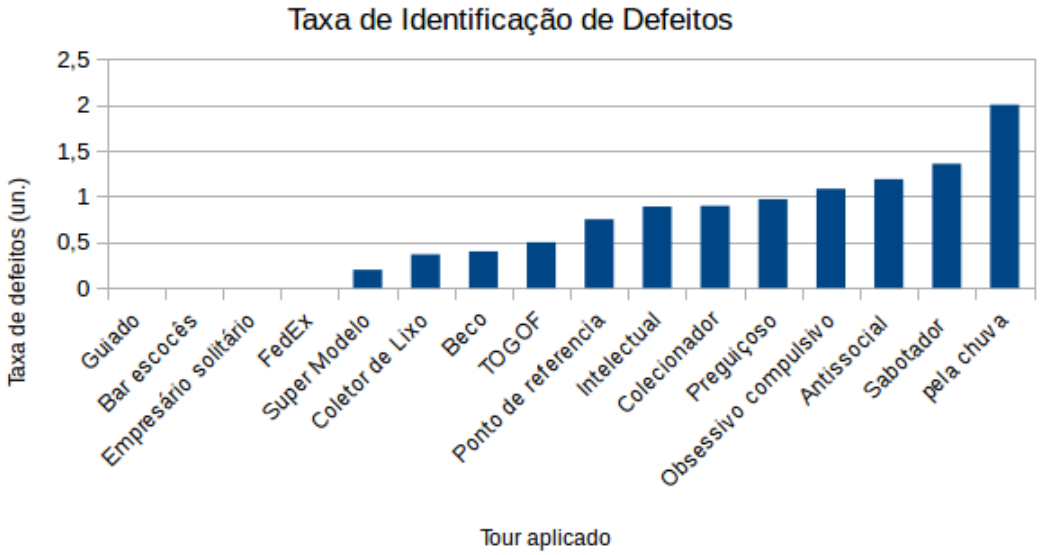

Figura 21 – Taxa de Identificação de Defeitos por tour. Fonte: Autor

5.3.2.3 Q5. Qual é o prazo gasto para realizar os testes?

5.3.2.3.1 M5. Prazo

Ao analisarmos a tabela 25, observamos que o Prazo média conclusão dos testes exploratórios pelos grupos foi de 4,25 dias.

5.3.2.4 Q6. Qual o esforço gasto na execução de uma tour?

Analisando de modo geral, observando os dados da tabela 25, podemos observar que a média de esforço por grupo foi de nove horas e trinta e um minutos para concluir a exploração, tendo uma média de doze minutos e sete segundos para concluir um *tour*. Para gerarmos uma análise mais detalhada em relação a qual *tour* conseguiu melhores resultados, vamos utilizar a Métrica M6. "Esforço por *tour*".

5.3.2.4.1 M6. Esforço por tour

Na Figura 22 ilustra a média do esforço em horas gasta para realizar um tipo de *tour*. Observamos, então, que os *tour* que apresentaram maior esforço durante o estudo

foram o *tour* obsessivo compulsivo, o *tour* do preguiçoso e o *tour* do sabotador.

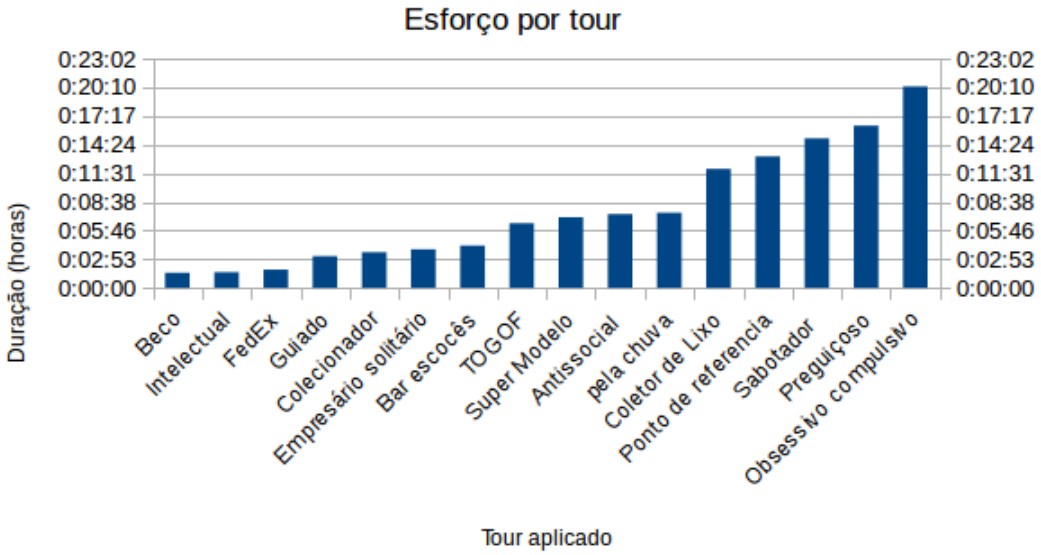

Figura 22 – Esforço por tours. Fonte: Autor

De modo geral, o esforço gasto entre os grupos de testes tradicionais e exploratório foram significativamente distintos, uma vez que os grupos que trabalharam com testes tradicionais tiveram um esforço médio de uma hora e quarenta e nove minutos para concluir suas atividades. Já os grupos que trabalharam com testes exploratórios apresentaram um esforço superior, com uma média de nove horas e trinta e um minutos de trabalho.

A análise dos resultados demonstrou indícios de vantagens na aplicação de testes exploratórios em relação a quantidade de defeitos relatados e a taxa de identificação de defeitos.

5.4 Entrevistas

As entrevistas ocorreram logo após a aplicação dos testes e ocorreram de forma separada com cada grupo e tiveram uma duração média de 30 minutos. As entrevistas foram semi-estruturadas, contando com perguntas guias e novas perguntas podiam surgir durante o processo. As entrevistas foram essenciais para a análise dos dados fornecidos pelos alunos após a aplicação dos testes, pois visava entender as dificuldades sentidas pelos participantes durante todo o estudo, obedecendo a questão **Q8. Qual a dificuldade percebida pelos testadores ao realizar os testes?** e a métrica **M8. Dificuldade** descritos no capitulo 4.

A primeira questão realizada nas entrevistas foi a respeito da dificuldade em compreender o sistema. A maior parte dos entrevistados respondeu que sentiu uma dificuldade baixa ou regular. Os entrevistados relataram que a presença da documentação a respeito

do sistema entregue aos grupos, facilitou bastante a compreensão do funcionamento do *PGTBL*.

A segunda questão foi relacionada à dificuldade percebida em relação à instalação do ambiente de testes. A maior parte dos entrevistados considerou a dificuldade baixa ou regular, devido a presença de uma documentação guia para a instalação do ambiente de testes. Entretanto, houveram alguns relatos de dificuldade extrema, em razão do uso de sistemas operacionais *Windows* e *macOS*. Para a instalação do ambiente de testes, os participantes receberam um manual de instruções com passos para instalar o ambiente e os pré-requisitos do sistema, como o *Docker* e do *Git*, por exemplo. O manual foi construído visando o uso de um sistema operacional baseado no Linux. Entre os participantes que utilizaram *macOS*, o relato foi a necessidade da adição de algumas dependências que sanassem as incompatibilidades. Já entre os alunos que utilizaram *Windows*, a dificuldade relatada foi uma questão técnica, referente ao uso de comandos *docker*. Um participante, que utilizou *Windows*, relatou que sentiu a necessidade de criar uma máquina virtual *linux* para contornar as dificuldades. Também houve o relato de alunos que decidiram realizar os testes de forma pareada para contornar essa dificuldade.

A terceira pergunta foi relacionada a dificuldade em aplicar os testes tradicionais. Esta pergunta foi voltada para os grupos deste tema específico. A maior parte dos alunos relatou uma dificuldade baixa a regular. Uma participante relatou que o fato deles receberem casos de testes definidos fez com que ela se sentisse mais confiante em realizar a aplicação dos testes, em razão da sua falta de experiência prévia, os casos de teste proporcionaram um apoio significativo.

Para os grupos que trabalharam com testes exploratórios, a questão levantada foi relacionada a dificuldade percebida em relação a identificação dos *tour* e distritos, ao realizarem os testes. A maior parte dos entrevistados indicou sentir uma dificuldade regular à alta. Os participantes relataram que a disponibilização da aula, na fase de treinamento, foi importante, pois, trouxe os principais conceitos necessários para compreensão dos *tours* e escolhas durante a aplicação dos testes. Os participantes que apontaram sentir uma dificuldade alta, justificaram a resposta em razão da grande quantidade de conceitos e da necessidade de revisar o material diversas vezes, durante o processo. Houve o caso de um participante, sentir dificuldade ao realizar um determinado teste e relatar que quebrou a regra dos testes funcionais/manuais e manipulou o código localmente para que o sistema aceitasse os dados inseridos.

A próxima questão foi relacionada a dificuldade percebida em relação a documentação dos testes. A dificuldade percebida pelos grupos que realizaram testes tradicionais foi relatada como sendo de baixa a regular. Já os grupos que realizaram testes exploratórios, em sua maioria, relataram dificuldade regular à média. Os entrevistados que realizaram testes exploratórios justificaram suas respostas em razão da dificuldade de compreender

os campos a serem preenchidos na planilha. Alguns grupos tiveram o cuidado de padronizar suas respostas, entretanto, houveram grupos onde não existiu nenhuma padronização entre os membros do grupo.

A última questão realizada na entrevista buscou entender se os participantes sentiram a dificuldade diminuir à medida que foi aplicando os testes. A grande maioria dos participantes indicou concordar totalmente ou concordar parcialmente. Por mais que, ao longo do processo, os alunos ganhassem mais experiência e conhecimento do sistema, ainda sentiram que novos desafios eram apresentados durante processo.

Ao final da entrevista foi aberto um espaço para os participantes falarem se gostaram da experiência e darem uma visão geral. Em relação aos testes tradicionais, destacarei dois comentários de participantes. O primeiro participante disse:

> "Sim, foi uma experiência interessante. Fazer os testes tradicionais foi quase como seguir uma manual de instruções, então não chegou a ser trabalhoso e nem desafiador, mas foi bom ver que atividades assim não são um "bicho de sete cabeças", como eu imaginava".

O segundo participante relatou:

> "Gostei sim, eu nunca tinha feito nenhum tipo de teste ou visto uma documentação para isso. Então foi bem legal ter esse primeiro contato, ainda mais com teste tradicional, porque facilita a ambientação. Talvez se depois eu tivesse que fazer testes exploratórios nessa plataforma, seria mais fácil por ter tido a chance de fazer os tradicionais primeiro."

Em relação aos testes exploratórios, destacarei dois comentários. O terceiro participante disse:

> "Sim, no inicio foi um pouco complicado iniciar os testes, pois não conhecia o sistema testado e não sabia por onde começar e como começar, por qual tour utilizar, etc. Para eu começar a ter um entendimento melhor de como aplicar os tours dos testes exploratórios foi necessário eu me basear primordialmente ao backlog com as features e user cases apresentados no pdf de instruções da atividade. No final foi uma experiência enriquecedora do ponto de vista de maneiras alternativas de se testar um sistema."

O quarto participante disse:

> "Sim, gostei do testes exploratórios. Principalmente porque os *tours* te dão um foco a uma perspectiva de teste. Acho que com mais exercícios práticos

> seria mais fácil de identificar numa aplicação prática a escolha de um *tour*, às vezes ficou confuso."

A fase de entrevistas foi muito enriquecedora e promoveu diversas percepções mais pessoais em relação a todos os dados gerados. É possível perceber pelos relatos que os alunos gostaram de participar do estudo. Embora dificuldades tenham sido encontradas devido ao contexto, a experiência adquirida pelos participantes, possivelmente melhoraria os resultados de uma segunda rodada de aplicação de testes.

6 Cronograma

Nesta capitulo serão descritas as macro atividades pertencentes ao desenvolvimento deste trabalho de conclusão de curso.

6.1 Cronograma TCC 1

Nesta seção, a Tabela 26 contém as macro atividades pertencentes ao TCC 1.

Tabela 26 – Cronograma TCC 1

Atividades	**Setembro**	**Outubro**	**Novembro**	**Dezembro**
Proposta inicial de tema	x			
Aprovar tema	x			
Definir objetivo da pesquisa	x			
Definir questão de pesquisa	x			
Revisão de literatura	x	x	x	
Referencial teórico	x	x	x	
Descrição de Qualidade de Software	x	x		
Descrição de Testes de software	x	x	x	
Ajuste do tema			x	
Descrição de V&V	x	x		
Descrição de testes exploratórios		x	x	
Descrição de métricas de Software		x	x	
Descrever Problema		x	x	
Descrever Metodologia			x	
Descrever aplicação			x	
Descrever o produto utilizado no estudo de caso			x	
Selecionar o que será testado			x	
Desenvolver um plano de medição		x	x	
Ajustes para entrega do TCC 1			x	x
Entrega do TCC 1				x

6.2 Cronograma TCC 2

Nesta seção, a Tabela 27 contém as macro atividades pertencentes ao TCC 2.

Tabela 27 – Cronograma TCC 2

Atividades	**Ago.**	**Set.**	**Out.**	**Nov.**	**Dez.**
Ajustes e correções oriundas do TCC1	x	x			
Revisão de literatura	x	x			
Elaborar um plano de testes	x	x			
Elaborar questionários		x			
Elaborar treinamento		x			
Elaborar carta guia de exploração		x			
Ajustar ambiente de teste		x			
Aplicar questionário			x		
Aplicar treinamento			x		
Realizar aplicação dos testes			x		
Analisar os resultados dos questionários				x	
Analisar os resultados das aplicações				x	
Descrever os resultados				x	x
Conclusões e considerações finais					x
Entrega do TCC 2					x

7 Considerações Finais

Este Trabalho de Conclusão de Curso apresentou um estudo empírico (*TCC*), que visou abordar questões relacionadas a qualidade de produtos. Para realizar essa avaliação foi selecionado um sistema, chamado PGTBL. O sistema foi avaliado a nível de sistema sob o ponto de vista funcional. Para esta avaliação foram realizados testes exploratórios e testes tradicionais. com o intuito de identificar vantagens do uso de testes exploratórios. O estudo foi dividido em quatro etapas e a aplicação das etapas foi explicada em detalhes no capítulo 5 nos relatos de cada fase.

Neste TCC foram realizadas pesquisas no intuito de identificar características de qualidade de produto, pois é desejável que os produtos de software sejam entregues de forma a suprir as necessidades e orientações do cliente. Atividades de verificação, validação e testes são importantíssimas para avaliar e acompanhar a qualidade dos produtos entregues. As atividades de teste, principalmente, possuem níveis alinhados as fases de desenvolvimento, onde o teste de aceitação está alinhado a validação de negócios, o teste de sistema alinhado a validação dos requisitos, o teste de integração está alinhado a validação de design e os testes unitários estão alinhados a validação da *build*.

Os testes exploratórios, por sua vez, é um tipo de teste vantajoso em relação a aprendizagem, pois possibilita ao testador aprender sobre o sistema enquanto realiza o design e a execução dos mesmos. Outras vantagens apontadas estão relacionadas aos contextos onde eles se encaixam. Os teste exploratórios de software são bem aceitos onde é necessário *feedback* ou aprendizagem rápida do produto, outra vantagem é a maior diversidade de testes aplicados, a construção de testes a partir da perspectiva do usuário, entre outras.

A metáfora do turista, proposta por Whitakker, trouxe uma analogia as atividades de exploração de um testador e as estratégias que o turista utiliza para aproveitar melhor uma viagem, dividindo o sistema em distritos e *tour*, analogamente a divisão de uma cidade turística que será visitada.

Para a estratégia de aplicação deste estudo de empírico foi realizada uma análise documental do sistema PGTBL para que fossem compreendidos os requisitos do sistema, suas divisões e pontos principais. Foram selecionados épicos, *features* e *user stores* das principais atividades do sistema PGTBL. Logo após foi realizado um GQM para compreender os objetivos de avaliação, as questões derivadas desses objetivos e por fim as métricas que serão coletadas durante a avaliação do sistema.

Para a construção da metodologia e escolha das técnicas, foi realizado um estudo estratégico utilizando uma matriz SWOT. A parte experimental do estudo seguiu o uso

de grupos focais, dividindo o protocolo em quatro fases, todas aplicadas remotamente. As fases foram: treinamento, questionário, aplicação dos testes e entrevistas.

A partir dos relatos dos resultados de cada etapa, podemos observar que algumas etapas obtiveram bons resultados, entretanto ainda necessitam de melhorias. Ao analisarmos a entrevistas e os dados gerados na documentação dos testes, foram identificadas ameaças a validade interna e externa. As ameaças a validade interna identificadas foram: Aplicação do estudo inteiramente remoto; Falta de instruções de preenchimento da planilha. As ameaças a validade externa identificadas foram: Pandemia de *covid-19*; Falta de experiência dos participantes; Falta de comunicação com os grupos; Dificuldade dos participantes em documentar os resultados obtidos.

A aplicação do estudo inteiramente remoto, tinha o intuito de proporcionar aos integrantes dos grupos a oportunidade de alinharem seus horários da forma mais conveniente possível, e em razão da pandemia de *covid-19*, não existiu alternativa a mais além da aplicação remota do estudo.

Embora existisse documentações sobre o sistema e a instalação do ambiente, a ausência de uma documentação com instruções de como preencher a planilha, acabou por chocar com a falta de experiência dos participantes e a falta de comunicação com alguns grupos. Foi disponibilizado todos os contatos disponíveis para que os participantes pudessem sanar suas dúvidas durante o período das atividades. Entretanto, poucos grupos utilizaram este recurso e boa parte das documentações foram geradas de forma pouco padronizada entre as equipes.

Tendo em vista que a fase mais prejudicada no estudo foi a fase de Aplicação dos testes, as recomendações de melhoria se concentrarão nesta etapa. Logo, é recomendado que seja adicionado um novo documento junto aos documentos entregues na fase de aplicação dos testes. Este documentos teria o objetivo de gerar instruções do preenchimento da planilha e um exemplo de boa prática de documentação de testes.

As outras fases do estudo mostraram bons resultados sendo aplicados de forma online. Entretanto, a fase de aplicação dos testes não se mostrou eficiente em um contexto onde os participantes tem pouca experiência. Logo, a recomendação é que esta fase seja feita presencialmente, em um ambiente controlado e com o auxilio de tutores/monitores para que cada grupo possa tirar suas dúvidas e receberem *feedbacks* construtivos. Observando que a parte do sistema que foi separada para ser trabalhada com os grupos é constituída de três atividades do *PGTBL*, a proposta seria realizar a aplicação dos testes em horário de aula, sendo necessárias duas aulas para esta fase.

Analisando de forma macro, de modo geral, três das quatro fases obtiveram bons resultados, cabendo também a possibilidade de aplicar alguns poucos ajustes, como a disponibilização de mais questões para serem sortidas na lista de exercícios e também a

vedação da participação dos integrantes em etapas subsequentes sem o cumprimento das etapas anteriores. A aplicação dos testes, embora tenha sido realizada em nível de estudo empírico, teve seu andamento prejudicado por causa do contexto pandêmico ocorrido durante a elaboração desse TCC. No entanto, foram identificados alguns indícios como: a quantidade de defeitos identificados, a taxa de identificação dos defeitos e as principais dificuldades percebidas pelos participantes, em relação ao andamento do estudo. Além disso, as dificuldades encontradas na realização do estudo empírico conduziu a raciocínios de melhoria no protocolo do próprio estudo tais como: a adição de um novo documento explicativo sobre como preencher a planilha dos testes e boas práticas de documentação.

O estudo realizado e seus resultados demonstram indícios de vantagens no uso de testes exploratórios como uma maior habilidade de encontrar defeitos. Porém, com as dificuldades encontradas devido ao contexto vivenciado na execução deste estudo, ressaltam-se as oportunidades de melhorias no protocolo do estudo que pode ser aplicado em casos futuros a título de comparar testes clássicos com testes exploratórios. Desta forma a sugestão é aplicar as melhorias no protocolo e realiza-lo em outros estudos empíricos para confirmar os benefícios com o uso de testes exploratórios.

Referências

BACH, J. What is exploratory testing? and how it differs from scripted testing. *StickyMinds, Enero*, 2001. Citado 2 vezes nas páginas 40 e 47.

BOURQUE, P.; FAIRLEY, R. E. et al. *Guide to the software engineering body of knowledge (SWEBOK (R)): Version 3.0*. [S.l.]: IEEE Computer Society Press, 2014. Citado na página 40.

CHIAVENATO, I. Gerenciando pessoas: O passo para a administração participativa. *P. imprenta: Sao Paulo: Makron Books*, 1994. Citado na página 24.

CHIAVENATO, I. Recursos humanos-edição compacta.-5ª edição, são paulo. *Editora Atlas SA*, 1998. Citado na página 24.

COALLIER, F. Software engineering–product quality–part 1: Quality model. *International Organization for Standardization: Geneva, Switzerland*, 2001. Citado 4 vezes nas páginas 9, 28, 30 e 31.

DELAMARO, M. E.; MALDONADO, J. C.; JINO, M. *Introdução ao Teste de Software*. Rio de Janeiro: Elsevier Editora Ltda, 2011. 394 p. (1a. edição). ISBN 978-85-352-2634-8. Citado 5 vezes nas páginas 17, 35, 38, 39 e 40.

DIAS, C. A. Grupo focal: técnica de coleta de dados em pesquisas qualitativas. 2010. Citado na página 23.

FENTON, N.; BIEMAN, J. *Software metrics: a rigorous and practical approach*. [S.l.]: CRC press, 2014. Citado 2 vezes nas páginas 32 e 33.

GALIN, D. *Software quality assurance: from theory to implementation*. [S.l.]: Pearson Education India, 2004. Citado na página 37.

GHAZI, A. N. et al. Levels of exploration in exploratory testing: From freestyle to fully scripted. *IEEE Access*, IEEE, v. 6, p. 26416–26423, 2018. Citado 3 vezes nas páginas 45, 46 e 47.

GIL, A. C. *Métodos e técnicas de pesquisa social*. [S.l.]: 6. ed. Editora Atlas SA, 2008. Citado 3 vezes nas páginas 20, 21 e 23.

GÜREL, E.; TAT, M. Swot analysis: a theoretical review. *Journal of International Social Research*, v. 10, n. 51, 2017. Citado na página 23.

ITKONEN, J.; RAUTIAINEN, K. Exploratory testing: a multiple case study. In: IEEE. *2005 International Symposium on Empirical Software Engineering, 2005.* [S.l.], 2005. p. 10–pp. Citado 2 vezes nas páginas 40 e 47.

KOSCIANSKI, A.; SOARES, M. S. *Qualidade de Software: Aprenda as metodologias e técnicas mais modernas para o desenvolvimento de software*. São Paulo: NovatecEditora Ltda, 2007. 395 p. (2a. edição). ISBN 978-85-7522-112-9. Citado 7 vezes nas páginas 17, 27, 28, 29, 30, 31 e 32.

MARCONI, M. d. A.; LAKATOS, E. Fundamentos da metodologia cientifica. 7ª edição-são paulo: Atlas. 2010. Citado na página 22.

MARCUSCHI, L. A. A propósito da metáfora. *Revista de Estudos da Linguagem*, v. 9, n. 1, p. 31–70, 2000. Citado na página 41.

MARRAS, J. P. Administração de recursos humanos: Do operacional ao estratégico (14a). *Edição. São Paulo: Saraiva*, 2011. Citado na página 24.

MARTINS, P. G.; LAUGENI, F. P. *Administração da produção*. [S.l.]: 2. ed. Editora Saraiva, 2005. Citado na página 27.

MORESI, E. et al. Metodologia da pesquisa. *Brasília: Universidade Católica de Brasília*, v. 108, p. 24, 2003. Citado na página 20.

PGTBL. 2019. Disponível em: <https://github.com/VictorDeon/PGTBL/wiki>. Acesso em: 10 nov. 2019. Citado 4 vezes nas páginas 9, 48, 49 e 50.

PRESSMAN, R. S. *Engenharia de Software: Uma Abordagem Profissional*. São Paulo: AMGH Editora Ltda., 2011. 780 p. (7a. edição). ISBN 978-85-63308-33-7. Citado 4 vezes nas páginas 17, 27, 35 e 36.

RAMOS C. S. ; KOSLOSKI, R. A. D. . V. E. . F. R. M. C. . D. V. H. A. Tbl como metodologia ativa para o ensino/aprendizagem em disciplinas de engenharia de software. *Simpósio Brasileiro de Engenharia de Software (SBES)*, São Paulo, 2018. Disponível em: <http://itrac.unb.br/?p=404>. Acesso em: 10 nov. 2019. Citado 2 vezes nas páginas 9 e 57.

SOLINGEN, D. R. van; BERGHOUT, E. W. *The Goal/Question/Metric Method: a practical guide for quality improvement of software development*. [S.l.]: McGraw-Hill, 1999. Citado 3 vezes nas páginas 9, 33 e 34.

SOMMERVILLE, I. *Engenharia de Software*. São Paulo: Pearson Education, 2005. 592 p. (6a. edição). ISBN 85-88639-07-6. Citado na página 35.

SOUZA, K. P.; GASPAROTTO, A. M. S. A importância da atividade de teste no desenvolvimento de softeare. *XXXIII Encontro Nacional de Engenharia de Produção, Enegep*, 2013. Citado 3 vezes nas páginas 9, 17 e 36.

STANDARD, I. Software engineering–software product quality requirements and evaluation (square)–guide to square. *ISO Standard*, v. 25000, p. 2005, 2005. Citado 3 vezes nas páginas 9, 28 e 30.

VAUGHN, S. e. a. *Focus group interviews in education and psychology*. [S.l.]: Thousand Oaks, CA: Sage Publications, 1996. Citado na página 23.

WHITTAKER, J. A. *Exploratory Software Testing: Tips, Tricks, Tours and Technics to Guide Test Design*. São Paulo: Pearson Education,, 2010. 224 p. (2a. edição). ISBN 978-0-321-63641-6. Citado 6 vezes nas páginas 18, 41, 42, 43, 44 e 47.

Apêndices

APÊNDICE A – Autorização de participação na pesquisa

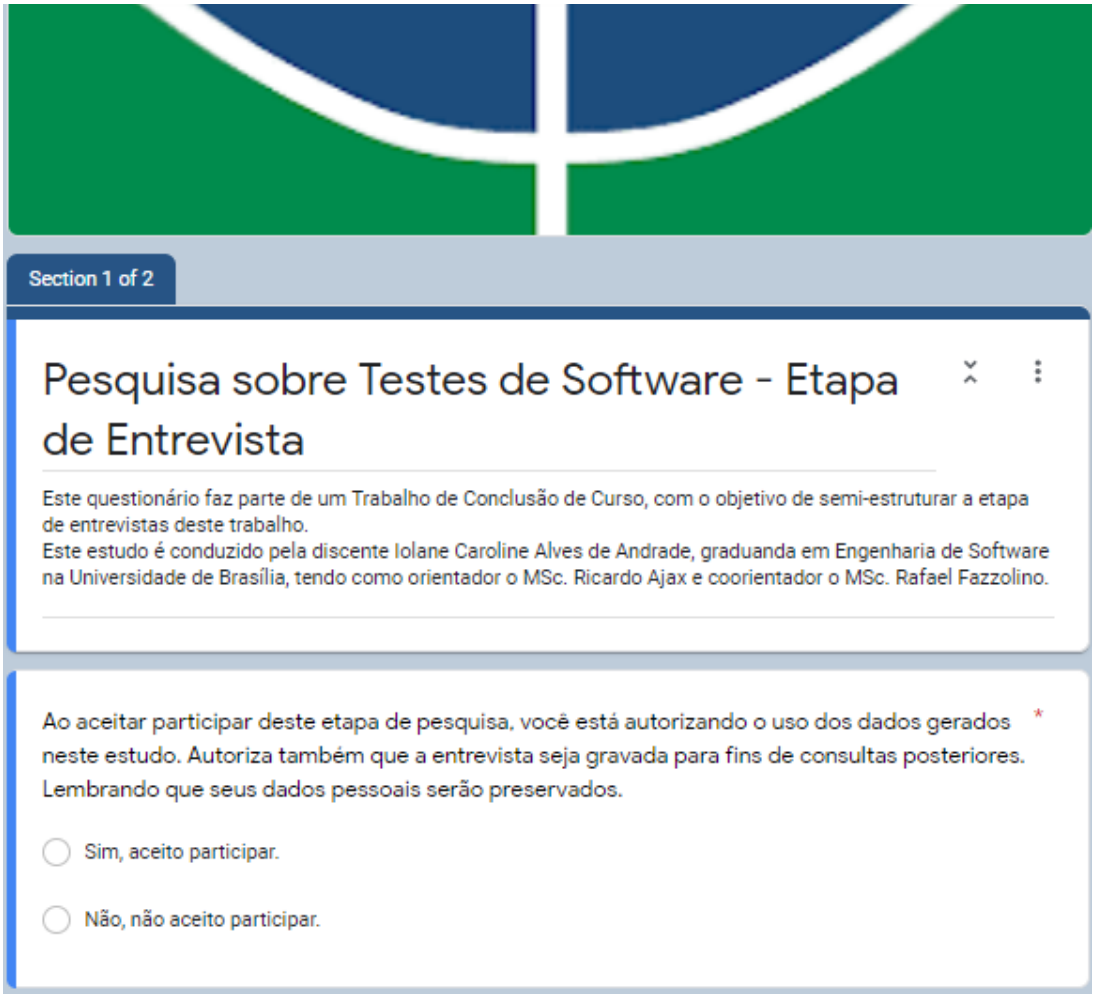

Figura 23 – Autorização de participação do estudo

APÊNDICE B – Análise estratégica utilizando matriz SWOT

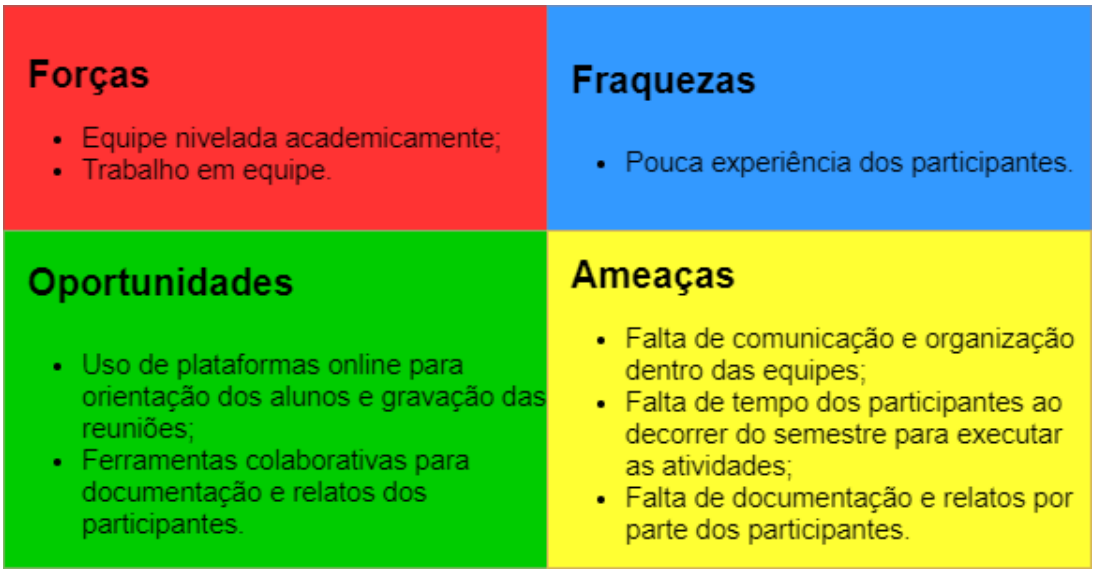

Figura 24 – Análise estratégica utilizando matriz SWOT

APÊNDICE C – Fase de Questionário

C.1 Questionário

1 - Você possui experiência em aplicação de testes de software?

[ ] Sim

[ ] Não

2 - Se sim, indique a sua faixa de experiência

[ ] até 1 ano

[ ] 1 à 2 anos

[ ] 2 à 4 anos

[ ] 4 à 5 anos

[ ] mais de 5 anos

3 - Você conhece a técnica de testes exploratórios?

[ ] Sim

[ ] Não

4 - Você já usou essa técnica?

[ ] Sim

[ ] Não

5 -Se sim, indique a sua faixa de experiência

[ ] até 1 ano

[ ] 1 à 2 anos

[ ] 2 à 4 anos

[ ] 4 à 5 anos

[ ] mais de 5 anos

APÊNDICE D – Fase de Entrevistas

D.1 Guia da Entrevista

1 - Qual tipo de teste você realizou?

[ ] Teste tradicional

[ ] Teste exploratório

2 - Qual o nível de dificuldade você sentiu em compreender o funcionamento do sistema a ser testado? (PGTBL)

[ ] Baixa

[ ] Regular

[ ] Média

[ ] Alta

[ ] Extrema

3 - Qual o nível de dificuldade você sentiu em instalar o ambiente do PGTBL?

[ ] Baixa

[ ] Regular

[ ] Média

[ ] Alta

[ ] Extrema

4 - Qual o nível de dificuldade que você sentiu para identificar os tipos de *tour* ao aplicar os testes exploratórios?

[ ] Baixa

[ ] Regular

[ ] Média

[ ] Alta

[ ] Extrema

5 - Qual o nível de dificuldade que você sentiu para identificar os tipos de Distrito ao aplicar os testes exploratórios?

[ ] Baixa

[ ] Regular

[ ] Média

[ ] Alta

[ ] Extrema

6 - Qual o nível de dificuldade você sentiu em documentar os testes?

[ ] Baixa

[ ] Regular

[ ] Média

[ ] Alta

[ ] Extrema

7 - A medida que você foi aplicando os testes a dificuldade foi diminuindo?

[ ] Baixa

[ ] Regular

[ ] Média

[ ] Alta

[ ] Extrema

8 - De modo geral, você gostou de participar desta atividade de testes e gostou da experiência? Comente::

APÊNDICE E – Carta guia de exploração

E.1 Teste exploratório do sistema PGTBL

Anote tudo relacionado aos testes realizados, podendo ser uma divergência ou uma anomalia que possa ser considerada um defeito no sistema. Lembre-se que os requisitos do sistema, muitas vezes, não estarão detalhadas. Então utilize os seus instintos exploradores e o bom senso para decidir o que deve ser reportado. A documentação da exploração é muito importante. Tente identificar o tipo de *tour* utilizado e documentar suas estratégias.

E.1.1 Informações gerais do PGTBL

O PGTBL é uma sistema de aprendizagem em equipe, baseado no TBL. O TBL se divide em um três fases: pré-classe, em classe e pós-classe. Desta maneira o sistema se dividiu em seis principais atividades que contemplam essas fases. São elas: Cadastro de usuário; Criação de Turma; Preparação; Garantia de preparo ou RAT; Aplicação de conceitos e avaliações em pares; e Apelação. Para este experimento de teste, exploraremos as atividades de Cadastro de usuário e Criação de turmas.

O sistema permite cadastrar dois tipos de perfis, sendo:

- Professor;
- Aluno.

Ambos os perfis permitem ao usuário:

- Visualizar seus dados pessoais;
- Editar seus dados pessoais;
- Inserir uma foto em seu perfil;
- Editar sua senha pessoal;
- Buscar por disciplinas cadastradas no sistema;
- Visualizar os dados de uma disciplina;
- Visualizar os participantes de uma turma.

O professor, por sua vez, possui alguns privilégios como:

- Cadastrar disciplinas;
- Criar turmas com senhas de cadastro;
- Definir o número de monitores e alunos em cada turma, sendo que o número mínimo de alunos de uma turma é cinco e a turma pode ou não ter monitores;
- Buscar por usuários cadastrados no sistema;
- Matricular usuários em suas turma como alunos ou monitores;
- Desmatricular um usuário de uma turma cadastrada;
- Editar os dados de uma turma;
- Fechar uma turma;
- Deletar uma turma.

A turma permite ao aluno:

- Matricular-se em uma disciplina através de uma senha de acesso;
- Desmatricular-se de uma turma;

Abaixo estão descritos os principais itens a serem testados:

E.1.1.1 Register

Descrição: Todos os itens abaixo devem ser exibidos na página "register". Em caso de sucesso do cadastro, uma mensagem de confirmação.

- Nome;
- User name;
- Email;
- Type of user;
- Senha;
- Confirmação de senha;

E.1.1.2 Login

Descrição: Todos os itens abaixo devem ser exibidos na página "login". Em caso de sucesso o sistema será redirecionado para o profile do usuário, junto a uma mensagem de sucesso.

- User name;
- Senha.

E.1.1.3 Profile Aluno

Descrição: Todos os itens abaixo devem ser exibidos na página "profile".

- Profile;
- Update account;
- Update password;
- Search discipline;
- Notifications;
- Delete account;
- Last login;
- User name;
- Email;
- Institution;
- Course;
- All disciplines;
- Student of discipline;
- Discipline monitor;
- Help;
- Logout.

E.1.1.4 Profile Professor

Descrição: Todos os itens abaixo devem ser exibidos na página “profile”.

- Profile;
- Update account;
- Update password;
- Search discipline;
- Create discipline;
- Notifications;
- Delete account;
- Last login;
- User name;
- Email;
- Institution;
- Course;
- All disciplines;
- Student of discipline;
- Discipline monitor;
- Help;
- Logout;

E.1.1.5 Update account

Descrição: Na página profile encontra-se o item “update account”.Todos os itens abaixo devem ser exibidos na página “update account”. Todos os campos podem ser editados e uma mensagem de sucesso será exibida ao concluir uma edição.

- Escolher arquivo;
- User name;
- Name;

- Email;
- Institution;
- Course;
- Edit;

E.1.1.6 Update password

Descrição: Na página profile encontra-se o item "update password". Todos os itens abaixo devem ser exibidos na página "update password". Todos os campos podem ser editados e uma mensagem de sucesso será exibida ao concluir uma alteração.

- Senha antiga;
- Nova senha;
- Confirmação da nova senha;
- Edit.

E.1.1.7 Create discipline

Descrição: Na página profile encontra-se o item "create discipline". Todos os itens abaixo devem ser exibidos na página "create discipline". Uma mensagem de sucesso deverá ser exibida ao concluir a operação.

- Discipline title;
- Discipline course;
- Discipline description;
- Preview;
- Classroom;
- Username;
- Password;
- Students limit in class;
- Monitor limit;
- Send.

E.1.1.8 Discipline

Descrição: Na página profile são listadas as disciplinas cadastradas no sistema. Ao selecionar visualizar uma disciplina serão exibidos os detalhes da mesma. Todos os itens abaixo devem ser exibidos em "discipline details".

- Discipline details;
- Student list;
- Final grades;
- Groups;
- Groups rank;
- TBL session;
- Discipline files;
- Forum;
- Reset discipline (apenas para professor);
- Close discipline(apenas para professor);
- Discipline title;
- Discipline course;
- Teacher' name;
- Teacher' email;
- Discipline description.

E.1.1.9 Student list

Descrição: Na página profile são listadas as disciplinas cadastradas no sistema. Ao selecionar visualizar uma disciplina serão exibidos os detalhes da mesma. Todos os itens abaixo devem ser exibidos em "student list".

- Titulo <Student list>;
- Nome da disciplina;
- Filtro;
- Todos;

- Students;
- Monitors;
- Lista de usuários matriculados na disciplina;
- paginação.

E.1.1.10 Search discipline

Descrição: Na página de profile, existe a opção search discipline. Todos os itens abaixo devem ser exibidos na página "search discipline". Todas as disciplinas buscadas devem ser listadas na página.

- Persquisar;
- Ordem (course, discipline, teacher).

E.1.1.11 Discipline Files/Session Files

Descrição: Existe a possibilidade de armazenar arquivos dentro de uma disciplina ou uma sessão de TBL. Estes arquivos podem ser baixados pelos alunos matriculados em uma disciplina. Caso estes arquivos pertençam a uma sessão de TBL, os alunos só poderão acessá-los quando o TBL estiver aberto.

- Paginação;
- Lista de arquivos;
- Botão de adição
- File title;
- File extension;
- File description;
- Escolher arquivo;

E.1.1.12 TBL Session

Descrição: Na página de TBL Session é possível visualizar a lista de sessões de TBL e adicionar novas sessões. É possível criá-la com dois possíveis status. O status *"close session"* onde o TBL é visível apenas para o professor que a criou. A outra opção é desmarcando o *"close session"*, desta maneira o TBL será visível para todos os matriculados na disciplina também.

-
- Paginação;
- Lista de TBL Sessions;
- Botão de adição
- Session title;
- Session description;
- Close session checkbox;

E.1.1.13 Session Exercises

Descrição: Esta sessão encontra-se dentro de uma TBL *Session* e permite que o professor crie uma banco de questões com quatro alternativas, obrigatoriamente e uma única alternativa correta. Se o TBL estiver aberto para os alunos, eles podem visualizar e acessar as questões disponíveis no banco de questões.

- Lista de questões;
- Adicionar nova questão;
- Question title;
- Question topic;
- Nivel;
- Basic;
- Intermediary;
- Advanced;
- Checkbox “is exercise”;
- Alternatives

APÊNDICE F – Casos de teste

F.1 Casos de teste do sistema PGTBL

Anote tudo relacionado aos testes realizados, podendo ser uma divergência ou uma anomalia que possa ser considerada um defeito no sistema. Lembre-se que os requisitos do sistema, muitas vezes, não estarão detalhadas. Então utilize os seus o bom senso para decidir o que deve ser reportado. A documentação dos testes é muito importante, então, caso encontre algum defeito no sistema, tente detalhar ao máximo.

Tabela 28 – Cadastro de usuário #1

ID	TC01-EP02FE02US06
Nome	Cadastro com sucesso no sistema
Ator	Professor
Pré-condição	É necessário criar uma senha com no mínimo 8 dígitos, devendo conter ao menos números e letras.
Procedimentos	1. Na pagina inicial selecione a opção “Register”; 2. Preencha o nome; 3. Preencha o usuário desejado; 4. Preencha o email corretamente; 5. Selecione a opção de perfil; 6. Preencha sua senha; 7. Confirme a senha; 8. Selecione “send”.
Pós-condição	O perfil será criado com sucesso e será redirecionado para a página do perfil.

Tabela 29 – Cadastro de usuário #2

ID	TC02-EP02FE02US08
Nome	Logout no sistema
Ator	Professor
Pré-condição	É necessário possuir uma conta no sistema.
Procedimentos	1. Na página de perfil selecione a opção “Logout”;
Pós-condição	O sistema será redirecionado para a página *home*.

Tabela 30 – Cadastro de usuário #3

ID	TC03-EP02FE02US07
Nome	Login no sistema
Ator	Professor
Pré-condição	É necessário possuir uma conta no sistema
Procedimentos	1. Na página inicial selecione a opção “Login”; 2. Preencha o usuário; 3. Preencha sua senha; 4. Selecione “enter”.
Pós-condição	A página será redirecionada para o perfil de usuário.

Tabela 31 – Cadastro de usuário #4

ID	TC04-EP02FE02US07
Nome	Login no sistema com senha incorreta
Ator	Professor
Pré-condição	É necessário possuir uma conta no sistema.
Procedimentos	1. Na página inicial selecione a opção “Login”; 2. Preencha o usuário; 3. Preencha a senha de forma diferente a cadastrada anteriormente; 4. Selecione “enter”.
Pós-condição	Uma mensagem de erro deve ser exibida.

Tabela 32 – Cadastro de usuário #5

ID	TC05-EP02FE02US07
Nome	Login no sistema com senha incorreta
Ator	Professor
Pré-condição	É necessário possuir uma conta no sistema.
Procedimentos	1. Na página inicial selecione a opção “Login”; 2. Preencha o usuário; 3. Preencha a senha de forma diferente a cadastrada anteriormente; 4. Selecione “enter”.
Pós-condição	Uma mensagem de erro deve ser exibida.

Tabela 33 – Cadastro de usuário #6

ID	TC06-EP02FE02US06
Nome	Cadastro com usuário já existente
Ator	Professor
Pré-condição	É necessário criar uma senha com no mínimo 8 dígitos, devendo conter ao menos números e letras.
Procedimentos	1. Na pagina inicial selecione a opção “Register”; 2. Preencha o nome; 3. Preencha o campo de usuário com o nome de um usuário que você já cadastrou no sistema; 4. Preencha o email corretamente; 5. Selecione a opção de perfil; 6. Preencha sua senha; 7. Confirme a senha; 8. Selecione “send”.
Pós-condição	Uma mensagem de erro referente ao nome de usuário deverá ser exibida.

Tabela 34 – Cadastro de usuário #7

ID	TC07-EP02FE02US06
Nome	Cadastro com email já existente
Ator	Professor
Pré-condição	É necessário criar uma senha com no mínimo 8 dígitos, devendo conter ao menos números e letras.
Procedimentos	1. Na pagina inicial selecione a opção “Register”; 2. Preencha o nome; 3. Preencha o campo de usuário; 4. Preencha o email com um email que já foi registrado no sistema; 5. Selecione a opção de perfil; 6. Preencha sua senha; 7. Confirme a senha; 8. Selecione “send”.
Pós-condição	Uma mensagem de erro referente ao email de usuário deverá ser exibida.

Tabela 35 – Cadastro de usuário #8

ID	TC08-EP02FE02US11
Nome	Editar informações pessoais
Ator	Professor
Pré-condição	É necessário estar logado em uma conta de usuário.
Procedimentos	1. Acesse a opção “update account” 2. Preencha o campo de instituição; 3. Preencha o campo de curso. 4. Selecione a opção “edit”
Pós-condição	A página será redirecionado para o perfil e uma mensagem de sucesso será exibida na tela.

Tabela 36 – Cadastro de usuário #9

ID	TC09-EP02FE02US11
Nome	Editar imagem do perfil
Ator	Professor
Pré-condição	É necessário estar logado em uma conta de usuário.
Procedimentos	1. Acesse a opção “update account” 2. Adicione uma imagem ao perfil; 3. Selecione a opção “edit”
Pós-condição	A página será redirecionado para o perfil e uma nova imagem de perfil será exibida.

Tabela 37 – Cadastro de usuário #10

ID	TC10-EP02FE02US11
Nome	Editar senha da conta
Ator	Professor
Pré-condição	• É necessário estar logado em uma conta de usuário; • É necessário criar uma senha com no mínimo 8 dígitos, devendo conter ao menos números e letras.
Procedimentos	1. Acesse a opção “update password” 2. Preencha o campo “senha antiga”; 3. Preencha o campo “nova senha”; 4. Preencha a confirmação da senha; 5. Selecione a opção “edit”.
Pós-condição	A página será redirecionado para a página de login onde deverá realizar um novo login com a nova senha.

Tabela 38 – Cadastro de usuário #11

ID	TC11-EP02FE02US12
Nome	Deletar conta
Ator	Professor
Pré-condição	É necessário estar logado em uma conta de usuário
Procedimentos	1. Selecione a opção “delete account”; 2. Uma mensagem de confirmação será exibida, então selecione “Apagar” 3. A página será redirecionada para a home, junto a uma mensagem de sucesso; 4. Faça o login com os dados da conta apagada para confirmar que o procedimento obteve sucesso;
Pós-condição	O login com os dados da conta apagada não será completado, pois as credenciais estarão inválidas.

Tabela 39 – Criação de turma #1

ID	TC12-EP03FE04US16
Nome	Criar turma com sucesso
Ator	Professor
Pré-condição	• Possuir uma conta cadastrada com o perfil de professor; • Estar logado no sistema. • O titulo da classe deve conter “Class A-Z” • Uma turma deve conter ao menos 5 alunos; Todos os campos de disciplina são obrigatórios.
Procedimentos	1. Selecione a opção “create discipline”; 2. Insira um titulo para a disciplina; 3. Insira o curso a qual a disciplina faz parte; 4. Insira descrição da disciplina; 5. Insira o titulo da classe; 6. Insira uma senha; 7. Selecione o limite de alunos; 8. Selecione o limite de monitores 9. Selecione a opção “send”
Pós-condição	A turma será cadastrada e uma mensagem de sucesso será exibida.

Tabela 40 – Criação de turma #2

ID	TC13-EP03FE04US16
Nome	Criar turma com o limite de alunos inferior
Ator	Professor
Pré-condição	• Possuir uma conta cadastrada com o perfil de professor; • Estar logado no sistema.
Procedimentos	1. Selecione a opção “create discipline”; 2. Insira um titulo para a disciplina; 3. Insira o curso a qual a disciplina faz parte; 4. Insira descrição da disciplina; 5. Insira o titulo da classe; 6. Insira uma senha; 7. Selecione o limite de alunos inferior a 5; 8. Selecione o limite de monitores 9. Selecione a opção “send”
Pós-condição	A turma não será cadastrada e uma mensagem de erro será exibida no limite de alunos da turma.

Tabela 41 – Criação de turma #3

ID	TC14-EP03FE04US16
Nome	Criar turma sem título
Ator	Professor
Pré-condição	• Possuir uma conta cadastrada com o perfil de professor; • Estar logado no sistema.
Procedimentos	1. Selecione a opção “create discipline”; 2. Deixe título da disciplina em branco; 3. Insira o curso a qual a disciplina faz parte; 4. Insira descrição da disciplina; 5. Insira o titulo da classe; 6. Insira uma senha; 7. Selecione o limite de alunos (ao menos 5); 8. Selecione o limite de monitores 9. Selecione a opção “send”
Pós-condição	A turma não será cadastrada e uma mensagem de erro será exibida no título da turma.

Tabela 42 – Criação de turma #4

ID	TC15-EP03FE04US16
Nome	Criar turma sem curso definido
Ator	Professor
Pré-condição	• Possuir uma conta cadastrada com o perfil de professor; • Estar logado no sistema.
Procedimentos	1. Selecione a opção “create discipline”; 2. Defina um título para disciplina; 3. Deixe o curso a qual a disciplina faz parte em branco; 4. Insira descrição da disciplina; 5. Insira o titulo da classe; 6. Insira uma senha; 7. Selecione o limite de alunos (ao menos 5); 8. Selecione o limite de monitores; 9. Selecione a opção “send”.
Pós-condição	A turma não será cadastrada e uma mensagem de erro será exibida em relação ao curso.

Tabela 43 – Criação de turma #5

ID	TC16-EP03FE04US16
Nome	Criar turma sem curso definido
Ator	Professor
Pré-condição	• Possuir uma conta cadastrada com o perfil de professor; • Estar logado no sistema.
Procedimentos	1. Selecione a opção “create discipline”; 2. Defina um título para disciplina; 3. Insira o curso a qual a disciplina faz parte; 4. Deixe a descrição da disciplina em branco; 5. Insira o titulo da classe; 6. Insira uma senha; 7. Selecione o limite de alunos (ao menos 5); 8. Selecione o limite de monitores 9. Selecione a opção “send”
Pós-condição	A turma não será cadastrada e uma mensagem de erro será exibida em relação a descrição da disciplina.

Tabela 44 – Criação de turma #6

ID	TC17-EP03FE04US16
Nome	Criar turma sem curso definido
Ator	Professor
Pré-condição	• Possuir uma conta cadastrada com o perfil de professor; • Estar logado no sistema.
Procedimentos	1. Selecione a opção “create discipline”; 2. Defina um título para disciplina; 3. Insira o curso a qual a disciplina faz parte; 4. Insira a descrição da disciplina; 5. Insira o título da classe (modelo “Class A-Z”); 6. Deixa a senha em branco; 7. Selecione o limite de alunos (ao menos 5); 8. Selecione o limite de monitores; 9. Selecione a opção “send”.
Pós-condição	A turma não será cadastrada e uma mensagem de erro será exibida em relação a senha da disciplina.

Tabela 45 – Criação de turma #7

ID	TC18-EP03FE04US15
Nome	editar turma
Ator	Professor
Pré-condição	• Possuir uma conta cadastrada com o perfil de professor; • É necessário que exista uma turma cadastrada previamente; • Estar logado no sistema.
Procedimentos	1. Selecione uma turma cadastrada; 2. Selecione a opção “editar”; 3. Altere dados da turma; 4. Selecione a opção “send”.
Pós-condição	As informações da turma serão editadas e uma mensagem de sucesso será exibida.

Tabela 46 – Criação de turma #8

ID	TC19-EP03FE04US14
Nome	Visualizar turma
Ator	Professor
Pré-condição	• Possuir uma conta cadastrada com o perfil de professor; • É necessário que exista uma turma cadastrada previamente; • Estar logado no sistema.
Procedimentos	1. Selecione uma turma cadastrada; 2. Selecione a opção “visualizar”;
Pós-condição	As informações da turma serão exibidas na tela.

Tabela 47 – Criação de turma #9

ID	TC20-EP03FE04US19
Nome	Fechar turma
Ator	Professor
Pré-condição	• Possuir uma conta cadastrada com o perfil de professor; • É necessário que exista uma turma cadastrada previamente; • Estar logado no sistema.
Procedimentos	1. Selecione uma turma cadastrada; 2. Selecione a opção “visualizar” 3. Selecione a opção “close discipline”
Pós-condição	A turma será fechada e uma nova opção “open discipline” será exibida na tela

Tabela 48 – Criação de turma #10

ID	TC21-EP03FE04US24
Nome	Procurar disciplina
Ator	Aluno
Pré-condição	• É necessário usar um cadastro de aluno; • É necessário que existam disciplinas registradas pelo professor previamente; • É necessário estar logado no sistema.
Procedimentos	1. Selecione a opção “search discipline”;
Pós-condição	Uma nova página será exibida e nela haverá uma lista de disciplinas.

Tabela 49 – Criação de turma #11

ID	TC22-EP03FE04US21
Nome	Inserir alunos na disciplina
Ator	Professor
Pré-condição	• É necessário estar logado em uma conta como professor;
Procedimentos	1. Selecione uma turma; 2. Selecione a opção visualizar; 3. No canto inferior direito da tela selecione o ícone “+”; 4. Uma lista de usuários serão exibidos na tela; 5. Adicione o usuário desejado ao selecionar o ícone “+”
Pós-condição	Uma mensagem de sucesso será exibida na parte superior da tela com o nome do usuário e o nome da disciplina.

Tabela 50 – Criação de turma #12

ID	TC23-EP03FE04US21
Nome	Inserir mais alunos que o limite estipulado.
Ator	Professor
Pré-condição	• Estipular um limite mínimo em uma turma; • Possuir um número de cadastros de alunos acima do número limite da turma;
Procedimentos	1. Selecione uma turma; 2. Selecione a opção visualizar; 3. No canto inferior direito da tela selecione o ícone “+”; 4. Uma lista de usuários serão exibidos na tela; 5. Adicione o usuário desejado ao selecionar o ícone “+” 6. Adicione mais usuário do que o limite estipulado na turma;
Pós-condição	Uma mensagem de erro será exibida em relação ao limite de alunos na turma.

Tabela 51 – Criação de turma #13

ID	TC24-EP03FE04US22
Nome	Visualizar alunos de uma disciplina que faço parte.
Ator	Aluno
Pré-condição	• Estar matriculado em uma disciplina;
Procedimentos	1. No perfil do aluno haverá uma lista com todas as disciplinas a qual o aluno está matriculado; 2. Selecione uma disciplina; 3. Selecione a opção visualizar; 4. Será exibida a página da disciplina com informações gerais sobre ela; 5. Selecione a opção “student list”
Pós-condição	Serão exibidos todos os alunos que fazem parte daquela disciplina.

Tabela 52 – Criação de turma #14

ID	TC25-EP03FE04US22
Nome	Visualizar alunos de uma disciplina que criei.
Ator	Professor
Pré-condição	• Ter cadastrado uma disciplina; • Ter alunos matriculados nesta disciplina;
Procedimentos	1. No perfil do professor haverá uma lista com todas as disciplinas cadastradas; 2. Selecione uma disciplina; 3. Selecione a opção visualizar; 4. Será exibida a página da disciplina com informações gerais sobre ela; 5. Selecione a opção “student list”.
Pós-condição	Serão exibidos todos os alunos que fazem parte daquela disciplina;

Tabela 53 – Criação de turma #15

ID	TC26-EP03FE04US20
Nome	Deletar alunos de uma disciplina que criei
Ator	Professor
Pré-condição	• Ter cadastrado uma disciplina; • Ter alunos matriculados nesta disciplina;
Procedimentos	1. No perfil do professor haverá uma lista com todas as disciplinas cadastradas; 2. Selecione uma disciplina; 3. Selecione a opção visualizar; 4. Será exibida a página da disciplina com informações gerais sobre ela; 5. Selecione a opção “student list” 6. Será exibida uma lista com todos os alunos cadastrados na disciplina; 7. Escolha uma aluno que será deletado da disciplina e selecione o ícone de “x”; 8. Uma mensagem de confirmação será exibida na tela, então selecione “remover” para confirmar;
Pós-condição	Será exibida uma mensagem de sucesso com o nome do aluno e o nome da disciplina a qual ele foi removido.

Tabela 54 – Criação de turma #16

ID	TC27-EP03FE04US23
Nome	Entrar em uma turma usando senha correta
Ator	Aluno
Pré-condição	• O aluno deve estar logado no sistema; • Um professor ter cadastrada uma disciplina; • O professor necessita disponibilizar a senha da turma para que seus alunos possam se matricular na disciplina;
Procedimentos	1. No perfil do aluno selecione a opção “search discipline”; 2. Selecione a turma desejada; 3. Preencha o campo “access” com a senha passada pelo professor; 4. Selecione o ícone ao lado do campo para autenticar e efetivar a matrícula;
Pós-condição	Será exibida uma mensagem de sucesso com o nome do aluno e o nome da disciplina a qual ele foi matriculado.

Tabela 55 – Criação de turma #17

ID	TC28-EP03FE04US23
Nome	Entrar em uma turma usando senha incorreta.
Ator	Aluno
Pré-condição	• O aluno deve estar logado no sistema; • Um professor ter cadastrada uma disciplina;
Procedimentos	1. No perfil do aluno selecione a opção “search discipline”; 2. Selecione a turma desejada; 3. Preencha o campo “access” com a senha aleatória; 4. Selecione o ícone ao lado do campo para autenticar e efetivar a matrícula;
Pós-condição	Será exibida uma mensagem de erro e o aluno não será matriculado na disciplina.

APÊNDICE G – Instruções de Instalação do Ambiente de Teste

G.1 Instalação do ambiente de teste

Para navegar pelo sistema PGTBL e realizar os testes é necessário seguir alguns passos e instalação e observar se os pré-requisitos estão sanados na sua máquina. A seguir, algumas breves instruções.

G.1.1 Pré-Instalação

Para utilizar o sistema PGTBL na sua máquina é necessário ter o docker e o compose instalados. Para a instalação do docker acesse o link abaixo e siga as instruções:

<https://docs.docker.com/engine/install/ubuntu/>

Para a instalação do docker-compose acesse o link abaixo e siga as instruções:

<https://docs.docker.com/compose/install/>

Também é necessário sua máquina possua a ferramenta Git instalada. Caso sua máquina não possua o Git, acesse o link abaixo e siga as intruções de instalação:

<https://git-scm.com/download/linux>

G.1.2 Instalação Ambiente

Passo 1:

Clone o repositório do sistema PGTBL na sua máquina. No terminal digite:

git clone https://github.com/VictorDeon/PGTBL.git

Passo 2:

Ao finalizar a clonagem, entre na pasta do repositório:

cd PGTBL

Passo 3:

Para iniciar o container, digite no terminal:

sudo docker-compose up –build

sudo docker-compose up

Passo 4:

Abra o navegador e acesse o localhost para usar o sistema:

<http://127.0.0.1/>

G.1.3 Erro comum no ambiente

Algumas vezes o ambiente pode apresentar um erro no banco de dados ao tentar realizar os primeiros cadastros. Em caso de erro no banco de dados, siga as próximas instruções para sanar o problema.

Passo 1:

apague a pasta .ignore

Passo 2:

no terminal, digite:

docker-compose down

Passo 3:

Crie os container novamente:

docker-compose up –build

Passo 4:

Vamos parar a execução dos containers com o atalho de teclado:

ctrl c

Passo 5:

Suba os containers em modo dettach:

docker-compose up -d

Passo 6:

Por fim executar os comandos para criar as migrations e o usuário:

docker-compose exec pgtbl python pgtbl/manage.py makemigrations

docker-compose exec pgtbl python pgtbl/manage.py migrate

docker-compose exec pgtbl python pgtbl/manage.py createsuperuser

APÊNDICE H – Acesso externo

No link H encontra-se toda documentação auxiliar que foi gerada neste TCC, bem como, aula, material de aula, as gravações das entrevista, etc.

<https://unbbr-my.sharepoint.com/:f:/g/personal/130028355_aluno_unb_br/EjKjhFp1B75e=pFy2Af>