INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DA PARAÍBA

TESTES DE SISTEMAS DE TRANSPARÊNCIA UTILIZANDO RELAÇÕES METAMÓRFICAS

ROMULO PEREIRA DANTAS FERREIRA

Cajazeiras, Junho de 2021

ROMULO PEREIRA DANTAS FERREIRA

TESTES DE SISTEMAS DE TRANSPARÊNCIA UTILIZANDO RELAÇÕES METAMÓRFICAS

Trabalho de conclusão de curso apresentado junto ao curso de **Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas** do **Instituto Federal de Educação, Ciência e Tecnologia da Paraíba**, como requisito parcial à obtenção do título de **Tecnólogo em Análise e Desenvolvimento de Sistemas**.

Orientador:

Prof. MSc. Diogo Dantas Moreira.

Cajazeiras, Junho de 2021

Campus Cajazeiras
Coordenação de Biblioteca
Biblioteca Prof. Ribamar da Silva
Catalogação na fonte: Daniel Andrade CRB-15/593

F383t

Ferreira, Romulo Pereira Dantas

Testes de sistemas de transparência utilizando relações metamórficas / Romulo Pereira Dantas Ferreira; orientador Diogo Dantas Moreira.- Cajazeiras, 2021.
56 f.: il.

Orientador: Diogo Dantas Moreira.
TCC (Tecnólogo em Análise e Desenvolvimento de Sistemas) – Instituto Federal de Educação, Ciência e Tecnologia da Paraíba, Cajazeiras, 2021.

1. Testes de software 2. Lei de Acesso à Informação 3. Testes Metamórficos I. Título.

004.05(0.067)

SERVIÇO PÚBLICO FEDERAL
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DA PARAÍBA
COORDENAÇÃO DE CURSOS - CAMPUS CAJAZEIRAS

Às **17:00** horas do dia **18** do mês de **junho** do ano de **2021**, via Google Meet, compareceu para defesa pública do **Trabalho de Conclusão de Curso**, requisito obrigatório para a obtenção do título de Tecnólogo em Análise e Desenvolvimento de Sistemas, o(a) aluno(a) **RÔMULO PEREIRA DANTAS FERREIRA**, matrícula **201622010060**, tendo como Título do Trabalho **TESTES DE SISTEMAS DE TRANSPARÊNCIA UTILIZANDO RELAÇÕES METAMÓRFICAS**. Constituíram a Banca Examinadora os professores **DIOGO DANTAS MOREIRA** (orientador), **FÁBIO ABRANTES DINIZ** (examinador) e **JANDERSON FERREIRA DUTRA** (examinador).

Após a apresentação e as observações dos membros da Banca Examinadora, ficou definido que o trabalho foi considerado **APROVADO** com nota **95,** com a condição de que o (a) aluno (a) entregue, no prazo máximo de 30 dias, a versão final do trabalho, via processo eletrônico à coordenação de curso. A versão deve conter a ficha catalográfica e atender às sugestões feitas pelos membros da banca. O código fonte desenvolvido no trabalho (caso haja) deve ser enviado para o e-mail da coordenação do curso (cads.cz@ifpb.edu.br).

Cajazeiras-PB, 22 de junho de 2021.

Documento assinado eletronicamente por:

- **Romulo Pereira Dantas Ferreira**, **ALUNO (201622010060) DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS - CAJAZEIRAS**, em 23/06/2021 18:27:51.
- **Fabio Abrantes Diniz**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 23/06/2021 14:14:07.
- **Janderson Ferreira Dutra**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 22/06/2021 22:40:04.
- **Diogo Dantas Moreira**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 22/06/2021 21:17:36.

Este documento foi emitido pelo SUAP em 18/06/2021. Para comprovar sua autenticidade, faça a leitura do QRCode ao lado ou acesse https://suap.ifpb.edu.br/autenticar-documento/ e forneça os dados abaixo:

Código Verificador: 199010
Código de Autenticação: ccaff6c396

AGRADECIMENTOS

A toda a **instituição** e a todos com quem tive contato. Grande parte do meu crescimento pessoal e acadêmico veio graças a todos com quem convivi durante a graduação. Ao meu orientador **Diogo Dantas Moreira** pela oportunidade de ser seu orientando e também por todo apoio, orientação e ensinamentos que me permitiram ir tão longe. Ao meu colega e amigo **Raul Coêlho Cabral**, que foi uma pessoa imprescindível durante toda a graduação e que sempre esteve ao meu lado. Aos meus amigos **Allyson Oliveira de Abreu** e **José Dijailson Dias Júnior** por todo o apoio que me ofereceram. A minha namorada **Débora Mabel de Souza Rolim Bernardo** que foi quem me encorajou e me motivou durante todo o trabalho. Aos meus irmãos **Bianca Pereira Dantas Ferreira** e **Renan Pereira Dantas Ferreira** que sempre me apoiaram e estiveram ao meu lado. Aos meus pais, **Maria Josimar Pereira** e **José Dantas Ferreira**, que são as pessoas a quem eu retribuo todos os meus esforços.

"Ninguém vai bater tão duro como a vida, mas não se trata de bater duro. Se trata de quanto você aguenta apanhar e seguir em frente, o quanto você é capaz de aguentar e continuar tentando. É assim que se consegue vencer."

Rocky Balboa, Rocky IV (1985)

RESUMO

Com a criação da Lei nº 12.527/2011, conhecida como Lei de Acesso à Informação, que regula o acesso aos dados e informações de propriedade do governo, surgiu uma importante iniciativa para a democratização dos dados públicos. Entretanto, a disponibilização desses dados em sistemas governamentais de transparência nem sempre ocorre de forma padronizada e de fácil acesso aos cidadãos. Nesse contexto, os testes se tornam imprescindíveis, pois, é de suma importância que estes sistemas ofereçam informações com um alto nível de qualidade, para que os resultados de todas as consultas oferecidas possam retornar valores consistentes, às assegurando como fontes de informações confiáveis para os cidadãos. Porém, uma dificuldade que está associada a testes desses sistemas é que seu código fonte não é disponibilizado, assim como as documentações dos requisitos funcionais e de qualidade, o que impossibilita a realização de vários tipos de testes por *stakeholders* externos aos envolvidos diretamente no desenvolvimento. O Teste de Caixa Preta é uma técnica de teste que se adequa a esse tipo de situação, visto que para utilizar essa abordagem, não é necessário ter o acesso ao código fonte, testando o sistema por meio de sua interface com o usuário. Já a abordagem de Testes Metamórficos verifica sistemas não pelo resultado de uma única saída, mas sim em observação das Relações Metamórficas, que são propriedades de um sistema ao longo de várias execuções de casos de testes. Portanto, o objetivo deste trabalho é investigar e avaliar a qualidade dos sistemas de transparência utilizados pelo governo, usando a técnica de Testes Metamórficos por meio de Testes de Caixa Preta. No único ciclo de *design* produzido neste trabalho, foi proposto o uso de cinco Relações Metamórficas identificadas a partir das consultas que são disponibilizadas nos 15 sistemas de transparência testados, onde foi gerado um total de 47 avaliações de Padrões de Saída de Relação Metamórfica. No final do ciclo de design produzido, a avaliação de todos os resultados obtidos nos sistemas de transparência testados, indicaram ausência de violações de Relações Metamórficas.

Palavras-chave: Lei de Acesso à Informação, Testes de Software, Testes Metamórficos.

ABSTRACT

With the creation of Law No. 12,527/2011, known as the Access to Information Law, which regulates access to data and information owned by the government, an important initiative for the democratization of public data emerged. However, the availability of these data in government transparency systems is not always standardized and easily accessible to citizens. In this context, the tests become essential, as it is of paramount importance that these systems provide information with a high level of quality so that the results of all the queries offered can return consistent values, ensuring them as reliable sources of information for the citizens. However, a difficulty associated with testing these systems is that their source code is not made available, as well as the documentation of functional and quality requirements, which makes it impossible to carry out various types of tests by stakeholders external to those directly involved in the development. Black Box Testing is a testing technique that suits this type of situation since to use this approach, it is not necessary to have access to the source code, testing the system through its user interface. The Metamorphic Tests approach, on the other hand, verifies systems not by the result of a single output, but by observing the Metamorphic Relationships, which are properties of a system over several test case executions. Therefore, the objective of this work is to investigate and evaluate the quality of the transparency systems used by the government, using the technique of Metamorphic Tests through Black Box Tests. In the only design cycle produced in this work, it was proposed the use of five Metamorphic Relationships identified from the queries that are made available in the 15 tested transparency systems, where a total of 47 Metamorphic Relationship Output Patterns evaluations were generated. At the end of the design cycle produced, the evaluation of all results obtained in the tested transparency systems indicated the absence of violations of -Metamorphic Relationships.

Keywords: Access to Information Law, Software Testing, Metamorphic Testing.

LISTA DE FIGURAS

Figura 2.1 – Passos do teste de *software* 22
Figura 2.2 – Teste de Caixa Preta 23
Figura 2.3 – Teste Metamórfico no motor de busca *online Google* verificando a relação Contagem (Q) $\geq$ Contagem (Q + K) 26
Figura 2.4 – Processo de Testes Metamórficos 27
Figura 3.1 – Ciclo de *design* 31
Figura 4.1 – Problema encontrado no sistema de transparência de Poço Dantas . . . 45
Figura 4.2 – Plataforma da *Publicsoft* 46

LISTA DE TABELAS

Tabela 4.1 – Resultados dos testes executados 41

LISTA DE QUADROS

Quadro A.1 – Sistemas levantados para a proposta 53

LISTA DE ABREVIATURAS E SIGLAS

TIC	Tecnologias da Informação e Comunicação(s)
LAI	Lei de Acesso à Informação
TM	Teste(s) Metamórfico(s)
RM	Relação Metamórfica(s)
PSRM	Padrão de Saída de Relação Metamórfica
DSR	*Design Science Research*
CTO	Caso de Teste Original
CTC	Caso de Teste Complementar
VI	Valor Inserido
N/A	Não Aplicável

LISTA DE SÍMBOLOS

§	Parágrafo
$\geq$	Maior ou igual
$\leq$	Menor ou igual
$+$	Soma
$=$	Igualdade
$\neq$	Diferença

SUMÁRIO

1 INTRODUÇÃO 14

1.1 Contexto 14

1.2 Motivação 15

1.3 Problemática 16

1.4 Objetivos 17

1.5 Contribuições 18

1.6 Estrutura do trabalho 18

2 REFERENCIAL TEÓRICO 19

2.1 Lei de Acesso à Informação 19

2.2 Testes de Software 20

2.2.1 Tipos e níveis de teste 21

2.2.2 Testes Funcionais (Caixa Preta) 23

2.3 Testes Metamórficos 23

2.3.1 Relações metamórficas 25

2.3.2 Processo de Testes Metamórficos 26

2.3.3 Padrões de saída de Relação Metamórfica 27

2.4 Considerações finais 28

3 METODOLOGIA 30

3.1 *Design Science* 30

3.2 Atividades 32

4 CICLOS DE DESIGN 33

4.1 Investigação do problema 33

4.2 Design de tratamento 34

4.2.1 Relações Metamórficas propostas 34

4.2.2 Ferramentas 35
4.2.3 Planejamento do processo de testes 36
4.3 Validação do tratamento 38
4.3.1 Testes realizados 42
4.4 Ciclos executados 43
4.5 Análise dos Resultados 44
4.6 Dificuldades encontradas 44

5 CONCLUSÕES 47
5.1 Considerações finais 47
5.2 Trabalhos futuros 48

REFERÊNCIAS 50

APÊNDICE A – SISTEMAS LEVANTADOS PARA A PROPOSTA . . 53

1 INTRODUÇÃO

Neste capítulo, será apresentado o contexto em que essa pesquisa foi desenvolvida, apresentando a princípio a **Lei de Acesso à Informação**, conceitos sobre testes de *software*, **Testes Metamórficos**, qualidade de *software* e a justificativa de sua importância para as atividades de testes. Na sequência, será apresentada a motivação para avaliar problemas relacionados à qualidade dos dados públicos que são disponibilizados por órgãos governamentais.

Dentro desse cenário, de acordo com a Lei de acesso à informação, que regulamenta o direito do cidadão de receber informações relativas ao governo, a problemática deste trabalho é demonstrada apresentando a falta da qualidade em como são disponibilizadas as informações aos cidadãos em sistemas de transparência. Em seguida, objetivos e contribuições deste trabalho serão enumerados. E ao final, será detalhada a organização textual do trabalho.

1.1 CONTEXTO

Com o crescimento do acesso às **Tecnologias da Informação e Comunicação** (TICs), surge a possibilidade de viabilização da disponibilização dos dados referentes aos órgãos públicos, de forma aberta e gratuita.

Em 2011, foi aprovada a Lei Federal de nº. 12.572/2011 (BRASIL, 2011), tradicionalmente conhecida como **Lei de Acesso à Informação** (LAI), trazendo em seu texto legal que as informações correlatas às atividades típicas de Estado são públicas. Esta regulamenta o direito à obtenção de informação garantida pela Constituição da República Federativa do Brasil nos artigos 5, inciso XXXIII, 37, § 3, inciso II e 216, § 2. A tutela legal, deste dispositivo, abrange todos os componentes da Administração Pública Direta, quais sejam a União, Estados, Distrito Federal e Municípios, quanto da Administração Pública Indireta, tais como autarquias, sociedades de economia mista, fundações públicas ou empresas públicas, abrangendo, ainda, empresas privadas que recebam verbas públicas para a execução de serviços públicos.

A LAI entrou em vigor dia 16 de maio de 2012 e muitos foram os benefícios, sendo destacada a formação de um novo instrumento de cidadania, sendo que essa obrigação de informar acerca dos gastos públicos, incumbe a todos os poderes e órgãos de todos os níveis e de quem mantém relacionamento com estes envolvendo dinheiro público. A lei em síntese, ganha destaque no ordenamento jurídico, pois tem como objetivo primordial garantir o direito fundamental de acesso à informação.

Segundo Brito et al. (2014), tecnologias *Web* podem ser consideradas um fator promissor para o desenvolvimento da transparência, pois fornece uma forma rápida, interativa, de fácil acesso e reutilização das informações. Porém, para que as informações sejam disponibilizadas corretamente, a qualidade dos sistemas que as proveem deve ser garantida, uma vez que a propagação de informações de má qualidade impossibilita a realização do acompanhamento dos dados públicos ou de qualquer tipo de informação.

Segundo Bourque e Fairley (2014) no SWEBOK *(Software Engineering Book of Knowledge)*, qualidade de *software* é "A capacidade de um produto de *software* satisfazer necessidades explícitas e implícitas sob circunstâncias especificadas", ou ainda, "o grau em que um produto de *software* atende aos requisitos estabelecidos [...]". Independente da definição, ambas contemplam a premissa de atendimento aos requisitos.

O Teste de *Software* é uma das atividades que se concentra na gestão da qualidade de *software* (HORCH, 2003) e pode ser definido como um conjunto de tarefas, planejadas e executadas sistematicamente com o propósito de descobrir erros cometidos durante a implementação de um software (PRESSMAN, 2015). Ao realizar testes, é possível mensurar a qualidade do produto que será disponibilizado.

Existem diversas abordagens para conseguir esse propósito de qualidade do produto ou serviço prestado, tal como **Teste Metamórfico**, que é uma abordagem de teste de *software* proposto por (CHEN et al., 1998). Essa técnica se baseia não apenas nas saídas dos casos de teste, mas também observando as propriedades esperadas de um sistema.

1.2 MOTIVAÇÃO

A Lei Federal de nº. 12.572/2011 Brasil (2011), conhecida como **Lei de acesso à informação** (LAI), trata de assuntos de interesse da União, dos Estados, do Distrito Federal e também dos Municípios. Como a própria Constituição Federal prevê, todos têm direito a receber dos órgãos públicos tanto informações de seu interesse particular, quanto de interesse coletivo ou geral.

É dever dos órgãos e entidades públicas incentivar, independentemente de requerimentos, a divulgação, em local de fácil acesso, das informações de interesse coletivo ou geral. Cada órgão deve publicar anualmente em um site apropriado na internet (**Portal da transparência**). Essas informações devem ser publicadas e disponibilizadas nas sede dos órgãos e entidades para consulta pública, que também manterão extrato com a lista de informações classificadas, acompanhadas da data, do grau de sigilo e dos fundamentos da classificação.

Entretanto, para manter a integridade dos dados e para que o acesso a esses dados pelos cidadãos seja efetivo, é necessário que esses sistemas sejam bem estruturados e que ofereçam recursos que facilitem a busca por informações e desempenhem suas funções de forma rápida, eficiente e transparente, sem apresentar falhas que comprometam seu uso. Logo, as atividades de **teste de *software*** compõem uma abordagem comum para garantia e verificação de qualidade de *software*.

Porém, alguns sistemas de transparência não possuem código aberto. Consequentemente, com a indisponibilidade do código fonte destes sistemas, desenvolvedores interessados em verificar sua qualidade ficam impossibilitados de realizar alguns tipos de testes, como por exemplo Testes de Caixa Branca, os quais identificam defeitos na estrutura interna do produto. Testes de sistema de forma geral, como os **Teste de Caixa Preta** são possíveis nesse caso, pois uma vez que tenha acesso a usabilidade do sistema, é possível criar casos de teste baseado nos resultados esperados a partir de entradas fornecidas.

Segundo Ammann e Offutt (2016), o teste de Caixa Preta e o teste de Caixa Branca são termos amplamente usados em teste de *software*. No teste de Caixa Preta, derivamos os testes de descrições externas do *software*, incluindo especificações, requisitos e *design*. No teste de caixa branca, por outro lado, derivamos testes das partes internas do código-fonte do *software*, especificamente incluindo ramificações, condições individuais e instruções.

O **Teste Metamórfico** é uma técnica adequada para gerar casos de teste quando não é possível determinar com certeza os resultados esperados, a partir de um conjunto de dados de entrada. Segundo Segura et al. (2020), as falhas não são reveladas pela verificação de saídas individuais, mas pela verificação das relações esperadas entre várias execuções do programa em teste. Desde sua introdução por Chen et al. (1998), a literatura sobre Testes Metamórficos cresceu de forma impressionante, e muitas aplicações bem-sucedidas da técnica surgiram e têm se mostrado eficazes em diversos domínios, incluindo programas numéricos, teoria dos grafos, aplicações orientadas a serviços, sistemas baseados em aprendizagem de máquina ou até mesmo de sistemas embarcados, o que evidencia a relevância de explorar essa técnica.

1.3 PROBLEMÁTICA

Como citado anteriormente, acompanhar a gestão do dinheiro público em todos os níveis da administração é um direito garantido por lei. De acordo com a LAI , o acesso às informações públicas pode ser garantido por meio da criação de serviço de informações ao cidadão nos órgãos e entidades do poder público, em local com condições apropriadas para atender e orientar o público. Desse modo, gestores devem direcionar esforços para que os

dados de interesse público estejam disponibilizados de maneira fácil e rápida, utilizando TICs para atingir esses objetivos.

O descumprimento dessa lei, pode acarretar bloqueio de verbas para o município, comprometendo projetos em andamento, a boa prestação de serviços e, consequentemente, a qualidade de vida dos cidadãos. Porém, é comum encontrar observações dessa natureza. Uma rápida pesquisa em sites de notícias evidencia que o objetivo da transparência ainda enfrenta diversos obstáculos.

É evidente nos sistemas de transparência que links contendo erros, falta de informações ou sistemas com inconsistências estão entre os problemas mais comuns. Problemas dessa natureza poderiam ter sido antecipados com a aplicação de atividades de Testes de Software, antes que fosse demonstrado durante o uso do sistema pelo usuário final [1].

Uma vez que não se tem acesso ao código ou especificações dos sistemas de transparência, porém, possui empiricamente o conhecimento de como esses sistemas deveriam se comportar, o problema a ser abordado neste trabalho está associado a responder a seguinte pergunta de pesquisa: "**Como demonstrar a presença de erros em sistemas de transparência com código fechado?**", para que seja possível reportar esses erros de maneira clara e objetiva para os responsáveis por seu desenvolvimento.

Para simplificar o entendimento do problema, o termo sistemas de transparência pode ser identificado pelos sistemas que os órgãos e entidades públicas disponibilizam informações acerca de gastos públicos.

1.4 OBJETIVOS

O objetivo geral deste trabalho é **explorar o uso de Testes Metamórficos aplicados a sistemas de transparência**. Para atingir esse objetivo, é proposto neste trabalho os seguintes objetivos específicos:

- Verificar a viabilidade do uso de Relações Metamórficas para testes de Caixa Preta;
- Selecionar um conjunto de sistemas de transparência relevantes para realização dos testes propostos;
- Identificar Relações Metamórficas na literatura para derivar ou usar nos sistemas propostos;

[1] Portais da transparência têm falta de informações e dados desatualizados - <http://glo.bo/ZhFloI>. Acesso em 20/11/2020

- Realizar um experimento com sistemas de transparência utilizando as Relações Metamórficas levantadas posteriormente e validar os resultados obtidos;
- Avaliar a eficácia do conjunto de casos de testes propostos neste trabalho.

1.5 CONTRIBUIÇÕES

No início do trabalho, esperava-se identificar falhas em sistemas de transparência a partir de uma nova abordagem, o que seria de grande relevância, caso estas existissem, no qual o intuito principal era notificá-las aos desenvolvedores do sistema. Diante dos objetivos propostos, foi explorado a utilização das técnicas de Testes Metamórficos a partir das Relações Metamórficas levantadas. Além deste trabalho contribuir no destaque da utilização de sistemas de transparência como forma de conscientizar a população sobre os gastos públicos, a abordagem utilizada permitiu explorar uma área que é inovadora em Testes de Software e que se mostrou interessante para a geração de casos de teste. As equipes de desenvolvimento envolvidas nos referidos sistemas podem se utilizar das mesmas ideias para derivar novos casos de teste e explorar cenários a partir das relações metamórficas expostas neste trabalho.

1.6 ESTRUTURA DO TRABALHO

Além deste capítulo de Introdução, o restante do trabalho está organizado da seguinte maneira: no Capítulo 2, é apresentado o **Referencial Teórico**, expondo a Lei de Acesso à informação e abordando os conceitos de Testes, desde os fundamentos até as definições necessárias para o entendimento de Testes Metamórficos; no Capítulo 3, a **Metodologia** do trabalho será descrita e em seguida as ativiades executadas neste trabalho; no Capítulo 4, é apresentado o único **cliclo de *design*** produzido, na qual é descrito as etapas que foram executadas durante o desenvolvimento dos testes, a análise dos resultados obtidos e o relato das dificuldades encontradas durante a produção deste trabalho. Por fim, no Capítulo 5 é apresentada a **Conclusão** deste trabalho, relatando as considerações finais e ponderações sobre trabalhos futuros.

2 REFERENCIAL TEÓRICO

Este capítulo apresenta a fundamentação teórica que embasa esta pesquisa. Na seção será apresentada A **Lei de Acesso à Informação** de forma geral. Na seção 2.2 serão apresentados os conceitos básicos de **Testes de Software**, assim como alguns termos que serão usados frequentemente neste trabalho, que é um desafio relevante para o entendimento deste trabalho. Por fim, na seção 2.3 são apresentados os conceitos e cenários de uso para **Testes Metamórficos**, como também a apresentação do termo **Relações Metamórficas**.

2.1 LEI DE ACESSO À INFORMAÇÃO

Como citado anteriormente, em 2011, o Senado Federal aprovou o projeto que deu origem a chamada **Lei de Acesso à Informação** (LAI), Lei Federal de nº. 12.572/2011 (BRASIL, 2011). Cumprindo mandamento constitucional, todos têm direito a receber dos órgãos públicos tanto informações de seu interesse particular, quanto de interesse coletivo ou geral, lembrando-se sempre que algumas exceções existem para a própria segurança da sociedade e do Estado. Importante também lembrar que esta Lei inclui toda a administração direta e indireta, considerando também as entidades controladas direta ou indiretamente pelos Municípios.

Nossa Carta Magna é suficiente ao proteger o acesso à informação por parte dos seus cidadãos ao prolatar que todos têm direito de receber dos órgãos públicos informações de interesse particular, coletivo ou geral, uma vez que todos os atos e fatos da Administração Pública devem ser pautados no princípio Constitucional da Publicidade, na qual todos os atos ou fatos que advirem desta deverão ser públicos, ressalvados aqueles que possuem a classificação de sigilosos. Como a LAI não é absoluta, e não se pode divulgar algumas informações, como hipóteses de sigilo previstas em outras leis, que inclui sigilo fiscal e bancário, segredos de justiça e industrial, é dever dos órgãos e entidades públicas garantirem que essas informações sejam protegidas, preservando a sua veracidade, integridade e divulgação (BRASIL, 2011).

Segundo a TRANSPARÊNCIA (2020), o acesso às informações públicas pode ser garantido por meio da criação de serviço de informações ao cidadão nos órgãos e entidades do poder público, em local com condições apropriadas para atender e orientar o público. Deve-se também informar sobre a tramitação de documentos nas respectivas unidades, além de protocolizar documentos e requerimentos de acesso a informações. Estados, Distrito Federal e municípios, em legislação própria, obedecida a LAI, devem constantemente

definir regras específicas a suas realidades. O regulamento deve conter o tratamento dado à informação e organização do portal da transparência, incluindo a forma de divulgação do relatório anual e a criação de serviço de informações ao cidadão.

Órgãos e entidades públicas deverão utilizar todos os meios e instrumentos legítimos disponíveis. A divulgação pela internet é obrigatória, então estes sistemas deverão seguir alguns requisitos, como conter ferramenta de pesquisa de conteúdo, possibilitar o acesso automatizado por sistemas externos em formatos abertos, estruturados e possíveis de serem lidos por máquina, divulgar em detalhes os formatos utilizados, garantir autenticidade e integridade das informações, manter atualizadas as informações, indicar local e instruções que permitam ao interessado comunicar-se por via eletrônica ou telefônica, com o órgão ou entidade detentora do sítio e adotar medidas necessárias para garantir a acessibilidade de conteúdo para pessoas com deficiência (BRASIL, 2011).

Municípios com população de até 10.000 habitantes não precisam divulgar obrigatoriamente pela internet, porém devem manter a obrigatoriedade de divulgação de informações relativas à execução orçamentária e financeira (TRANSPARÊNCIA, 2020).

Todos os órgãos públicos devem obedecer a LAI, para assim poder conseguir seus benefícios como órgão público e obter maior eficiência na gestão, permitir que o cidadão acompanhe todos os gastos públicos e criar uma confiança da administração pública.

2.2 TESTES DE SOFTWARE

Testes de *software* se tornaram elementos essenciais de garantia de qualidade de *software*. Segundo a norma da IEEE (1990) [1], um *software* com qualidade é o grau com que um sistema, *software*, componente ou processo cumpre os requisitos especificados e as necessidades ou expectativas do cliente. Conforme Sommerville (2016), a preocupação com o gerenciamento de qualidade do produto fornece uma verificação precisa e independente do processo de desenvolvimento de *software*. O gerenciamento de qualidade é caracterizado por realizar a verificação de quais funcionalidades do projeto já estão prontas, e com isso verificar se as mesmas estão de acordo com os requisitos e padrões solicitados.

Então, testes de *software* desempenham um papel importante na garantia e na avaliação de qualidade de um sistema. Segundo Myers et al. (2012), teste é o processo de avaliar um programa com a intenção de encontrar erros. O conceito de teste de *software*, segundo Bourque e Fairley (2014), descreve como sendo a verificação dinâmica do funcionamento de um programa utilizando um conjunto finito de casos de teste, adequadamente

[1] Apesar da data de citação da norma da IEEE, esta é uma fonte de informações confiáveis e revisadas por vários profissionais do termo qualidade de *software*.

escolhido dentro de um domínio de execução infinito, contra o seu comportamento esperado. A atividade de teste é realizada para avaliar a qualidade do produto de *software* e para melhorá-lo por meio da identificação de falhas.

Para planejar a atividade de teste, deve ser definido um modelo para o processo, ou seja, um conjunto de passos no qual é possível incluir geração de casos de teste e a utilização de critérios de teste específicos (PRESSMAN, 2015). Atividades de teste sem dúvida contribuem para a melhoria da qualidade do *software*, porém pode ter um custo significativo. Segundo Pressman (2015), o destaque crescente do *software* como elemento de sistemas e os "custos" envolvidos associados às falhas de *software* são forças propulsoras para que a atividade de teste seja cuidadosa e bem planejada, tornando comum encontrar organizações que gastem pelo menos 40% do esforço total do projeto em testes.

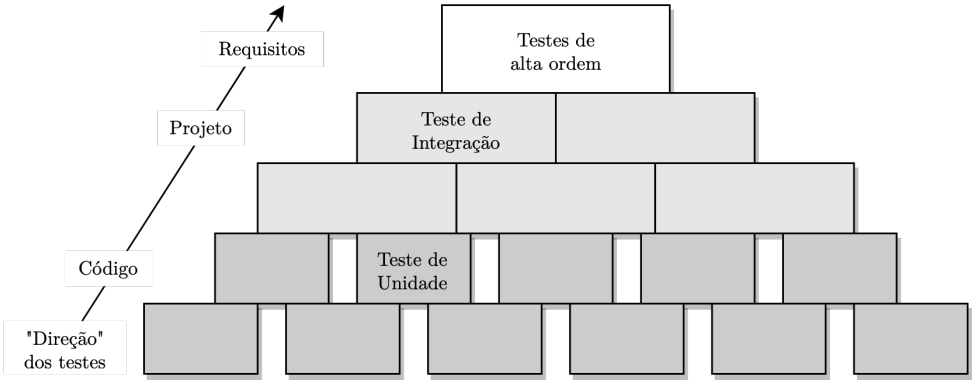
Segundo Bastos Rios (2007), com a execução do processo de testes ocorrendo de maneira paralela ao de desenvolvimento, os custos necessários para a correção de defeitos que seriam detectados em produção futuramente caem drasticamente. O custo do desenvolvimento do *software*, que é baseado em seu desenvolvimento e manutenção, será relativamente menor quando a aplicação é testada de maneira correta.

Em algum momento do processo de desenvolvimento de produtos de *software*, as falhas podem fazer o projeto ficar financeiramente mais caro, além de trazer um maior esforço da equipe de desenvolvimento ou até mesmo tornar o projeto inviável de ser finalizado. Desse modo, as prioridades de uma equipe de desenvolvimento deve ser em entregar *software* confiável e com a menor quantidade de bugs possível (HAYES, 2004). Muitos dos esforços dedicados a Testes de software visam automatizar o máximo possível desse processo para torná-lo mais barato, rápido e confiável (BARR et al., 2014).

2.2.1 Tipos e níveis de teste

Devido à diversidade dos fatores que podem colaborar para a ocorrência de erros, a atividade de teste se torna extremamente complexa. Levando-se em consideração o alvo de um teste, o teste de *software* pode ser classificado em diferentes níveis: testes de unidade, testes de integração e testes de sistema (BOURQUE; FAIRLEY, 2014). Em seu trabalho, Moreira (2020) propõe uma visão procedural em 3 partes dos níveis de teste, demonstrado na Figura 2.1.

Figura 2.1 – Passos do teste de *software*



Fonte: Adaptado de Pressman (2015)

O **teste de unidade** tem foco na verificação da menor unidade de um projeto de *software*, por exemplo, componentes, módulos ou classes (PRESSMAN, 2015). O objetivo principal é identificar os erros diretamente relacionados a algoritmos incorretos ou mal interpretados, estruturas de dados incorretas, ou simples erros de programação (DELAMARO et al., 2016). Essa atividade faz uso de técnicas que executam caminhos específicos do código-fonte, com o intuito de garantir que os testes cubram completamente (ou boa parte) das possibilidades de execução, provendo assim uma taxa de detecção de erros maior (PRESSMAN, 2015).

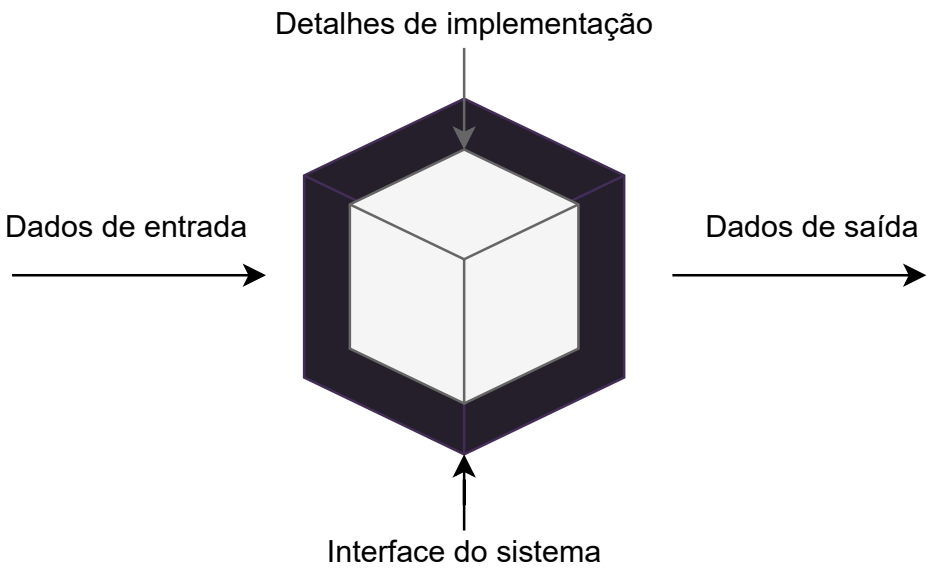
Os **testes de integração** têm por objetivo verificar o funcionamento do *software* após a integração das unidades, ou seja, o objetivo é descobrir erros associados às interfaces entre os módulos ou componentes de um sistema, quando esses elementos são integrados na construção da arquitetura do *software* que foi planejada (PRESSMAN, 2015). À medida que as diversas partes do *software* são colocadas para trabalhar juntas, se torna necessário verificar se a interação entre elas funciona de maneira adequada (DELAMARO et al., 2016).

Por fim, o **teste de sistema** tem a finalidade encontrar falhas no *software* por meio da utilização do *software* como um todo, verificando se todas as funcionalidades foram implementadas de acordo com a especificação do sistema. O teste de sistema também pode ser definido como uma série de diferentes testes cuja finalidade é executar totalmente o sistema, sob diferentes perspectivas, visando garantir seu adequado funcionamento sob todas essas visões diferentes (PRESSMAN, 2015).

2.2.2 Testes Funcionais (Caixa Preta)

A técnica de **Caixa Preta** é muito utilizada para projetar casos de teste considerando o sistema como se fosse uma caixa preta, ou seja, não é considerando os detalhes da construção do sistema, mas visando principalmente a usabilidade do ponto de vista do usuário. Para testá-lo são fornecidas entradas e avaliadas as saídas, verificando se estas estão em conformidade com os objetivos especificados. A principal característica desta técnica é desconsiderar os detalhes de implementação avaliando o *software*, segundo o ponto de vista do usuário e suas funcionalidades (DELAMARO et al., 2016). Porém, segundo Myers et al. (2012), a técnica funcional muitas vezes é inviabilizada em função das infinitas possibilidades combinatórias dos dados de entrada.

Figura 2.2 – Teste de Caixa Preta



Fonte: Elaborado pelo Autor (2021)

Segundo a figura 2.2, o teste de **Caixa Preta** se baseia a partir da observação do resultado da saida do teste conforme os dados de entrada, derivando seus requisitos de teste a partir dos requisitos funcionais do sistema testado, desconsiderando detalhes de implementação do sistema, apenas a sua interface.

2.3 TESTES METAMÓRFICOS

Como citado anteriormente, o teste de *software* é uma abordagem comum para garantia e verificação de qualidade de *software*, porém, esses testes podem apenas mostrar a presença de erros, não a sua ausência. Em muitos cenários, casos de teste que revelam erros são considerados úteis, enquanto os casos de teste bem sucedidos (ou seja, os que não

detectaram nenhum defeito) são considerados como inúteis e então descartados ou retidos apenas para fins de testes de regressão (SEGURA et al., 2020). Partindo dessa premissa, **Testes Metamórficos** (TM) fornecem uma alternativa para testar um programa quando a saída esperada de um caso de teste é desconhecida ou difícil de comparar com a saída real.

O TM difere das técnicas de teste convencionais, pois não se concentra na verificação de cada saída individual do *software* em teste (KANEWALA; CHEN, 2019). Segundo Chen (2010), seu conceito é simples e sua automação é fácil. O TM envolve várias execuções do programa em teste, em que é verificado se essas execuções do programa em teste cumprem certas propriedades necessárias, que são as **Relações Metamórficas**. Se essas entradas múltiplas e suas saídas não satisfizerem a Relação Metamórfica, isso implica uma implementação incorreta. Uma Relação Metamórfica é uma propriedade necessária da funcionalidade do programa pretendido que relaciona dois ou mais dados de entrada e suas saídas esperadas (ZHOU et al., 2016).

Segundo Chen et al. (1998), que conceberam o termo Teste Metamórfico, a proposta inicial do seu trabalho era gerar casos de teste a partir de outros casos previamente executados e bem sucedidos, ou seja, que não revelaram nenhum defeito, embora atualmente, saiba-se que pode ser aplicada também em casos de testes que revelaram erros.

Desde a primeira publicação sobre esse tema por (CHEN et al., 1998), vários estudos foram conduzidos sobre vários aspectos em relação a TMs, pois essa abordagem se mostrou muito eficaz na detecção de falhas, mesmo para programas estudados de forma muito extensiva. Por exemplo, um estudo detectou três bugs até então desconhecidos em três dos sete programas da suíte Siemens, um *benchmark* amplamente utilizado pela comunidade de pesquisa para a avaliação de várias técnicas de Teste de software. Dado que os programas da Siemens são pequenos e foram testados extensivamente por diferentes grupos de pesquisa nas duas décadas anteriores, a detecção de bugs anteriormente desconhecidos demonstra claramente que Testes Metamórficos complementa as estratégias de teste de *software* existente (SEGURA; ZHOU, 2018).

Segundo Chen et al. (2018), um fator importante que afeta os custos é o problema de escalabilidade, na qual o número necessário de casos de teste ou esforços de teste necessários cresce exponencialmente de acordo com o tamanho do programa em teste. Então, TMs podem ser aplicados com um conjunto de testes de qualquer tamanho, independente do tamanho e da complexidade do programa em teste. Isso se torna uma vantagem, pois, o quão grande ou pequeno é o conjunto de testes, não afeta a implementação do TM.

Para simplificar o entendimento durante o trabalho, a partir deste ponto serão utilizados os seguintes termos, traduzidos previamente por Moreira (2020):

Caso de Teste Original: Um caso de teste pré-existente ou que foi gerado e que foi bem-sucedido, ou seja, que não detectou nenhum defeito. Segundo Moreira (2020), esse termo foi adaptado a partir da literatura existente, que utiliza o nome de *Source Test Case* e não existem traduções conhecidas no nosso conhecimento.

Caso de Teste Complementar: Um caso de teste gerado a partir de uma Relação Metamórfica e tendo como base o caso de teste original. Segundo Moreira (2020), esse termo foi adaptado a partir da literatura existente, que utiliza o nome de *Follow-up Test Case* e não existem traduções conhecidas no nosso conhecimento.

2.3.1 Relações metamórficas

Como citado anteriormente, um elemento crucial dos TMs, é um conjunto de **Relações Metamórficas** (RM), que são propriedades necessárias do sistema em relação a suas entradas e suas saídas esperadas. Ao fazer a implementação desse tipo de abordagem de casos de teste, com base nas entradas do programa, são geradas o **Caso de Teste Original**, e com base nos quais uma RM pode então ser usada para gerar novas entradas, é gerado o **Caso de Teste Complementar**.

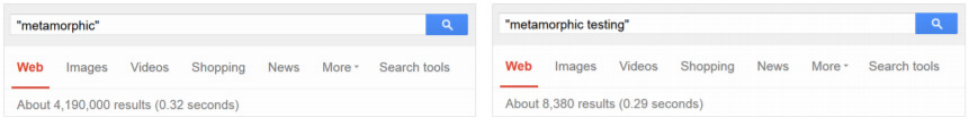
Ao contrário da maneira tradicional de verificar o resultado do teste de cada caso de teste individual, o TM verifica a origem e os casos de teste complementar, bem como suas saídas em relação a RM correspondente. Então, se o sistema em teste funcionar corretamente, as RMs serão satisfeitas para todos os casos de teste e, se o resultado do teste não satisfizer a RM, é possível que haja um defeito no sistema (CHEN et al., 2018). Ou seja, uma **Relação Metamórfica**, é uma propriedade necessária da funcionalidade do sistema, que relaciona dois ou mais dados de entrada e suas saídas esperadas, que podem ser usadas como critério de correção do sistema.

As RMs desempenham um papel crucial tanto na geração de Casos de Teste de Complementar quanto na verificação dos resultados dos testes. A eficácia dos TMs é fortemente influenciada pelas RMs usadas e, portanto, identificar RMs eficazes é uma etapa crítica. A identificação de RMs é uma tarefa que requer criatividade e domínio de conhecimento, mas existem pistas que podem auxiliar no processo. Uma abordagem comum para identificar RMs é verificar a especificação do programa ou a documentação, e considerar como as entradas do programa podem ser modificadas para que possam produzir uma mudança previsível na saída (CHEN et al., 2016). Normalmente, um algoritmo tem

muitas RMs. Portanto, a seleção de Relações Metamórficas é uma questão chave na aplicação de Testes Metamórficos (CHEN, 2010).

Para um melhor entendimento, considere testar um mecanismo de busca online como o da Google. Seja *Count* (Q) o número de resultados retornados para uma consulta de pesquisa. Intuitivamente, o número de resultados retornados para Q deve ser maior ou igual ao obtido ao refinar a pesquisa com outra palavra-chave (K). Isso pode ser expresso como a seguinte RM: Contagem (Q) $\geq$ Contagem (Q + K), onde + denota a concatenação de duas palavras-chave. A Figura 2.3 ilustra a aplicação dessa relação metamórfica no Google. Considere um Caso de Teste Original que consiste em uma busca pela palavra-chave "*metamorphic*", resultando em "Cerca de" 4,2 milhões de resultados. Suponha que um Caso de Teste Complementar seja construído pesquisando as palavras-chave "*metamorphic testing*": Isso leva a 8.380 resultados, que é menor do que o resultado de "*metamorphic*" e, portanto, satisfaz a relação. Se mais resultados fossem encontrados, isso violaria a RM, revelando um bug no sistema.

Figura 2.3 – Teste Metamórfico no motor de busca *online Google* verificando a relação Contagem (Q) $\geq$ Contagem (Q + K)



Fonte: Elaborada pelo autor (2020)

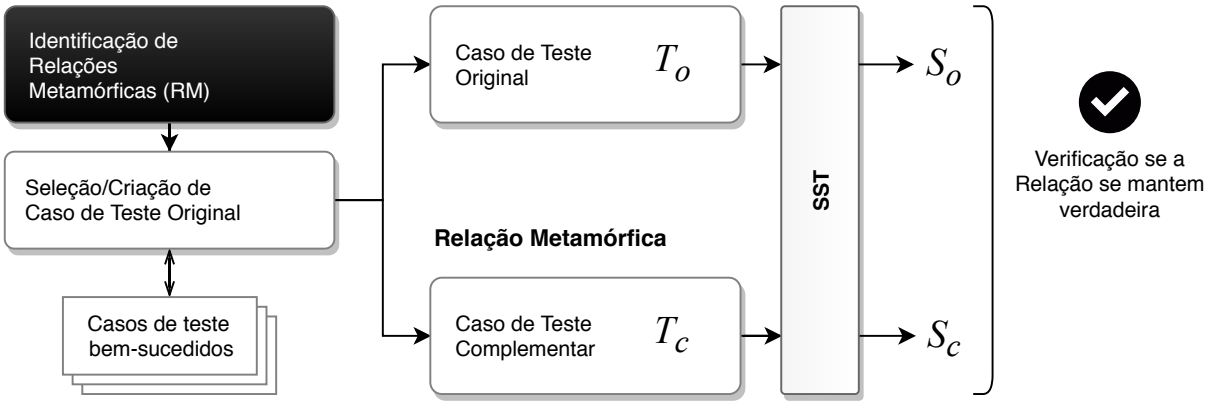
De acordo com tudo o que foi descrito, é correto afirmar que as RMs desempenham um papel crucial na estrutura proposta, pois são essenciais tanto na geração de Casos de Teste Complementar quanto na verificação dos resultados dos testes.

2.3.2 Processo de Testes Metamórficos

Segundo Kanewala e Chen (2019), o processo básico para a aplicação do Teste Metamórfico pode ser resumido da seguinte forma:

1. **Identificação de um conjunto de Relações Metamórficas**, que devem ser satisfeitas pelo sistema sob teste. Essa identificação geralmente é manual e demanda conhecimento do domínio do sistema e pode ser feito a partir das especificações de um sistema ou de um algoritmo Barr et al. (2014) Segura et al. (2016a) Chen et al. (2018). Segundo Segura et al. (2016b), deve ser identificado as propriedades necessárias do programa em teste e represente-as como RMs entre várias entradas

Figura 2.4 – Processo de Testes Metamórficos



Fonte: Moreira (2020)

de caso de teste e suas saídas esperadas, juntamente com algum método para gerar um **Caso de teste Complementar** com base em um **Caso de teste Original**.

2. **Criação e Execução de Casos de Teste Originais**, em que este pode ser um caso de teste já existente ou pode ser gerado a partir de alguma técnica de geração de casos de teste (MOREIRA, 2020).

3. **Criação de Casos de Teste Complementares**, nas quais as RMs encontradas na etapa de Identificação de um conjunto de Relações Metamórficas irão servir para transformar o Caso de teste Original em um Caso de Teste Complementar.

4. **Execução de Casos de Teste Complementares**, na qual serve para executar o Caso de Teste Complementar e comparar os resultados dos Caso de Teste Original e Caso de Teste Complementar para verificar se as RMs correspondentes estão satisfeitas. Se as saídas de um Caso de Teste Original e seu Caso de Teste Complementar violarem a RM, o caso de Teste Metamórfico terá falhado, indicando que o programa em teste contém um bug Segura et al. (2016b).

Estes processos podem ser melhor entendidos também a partir da visualização da Figura 2.4.

2.3.3 Padrões de saída de Relação Metamórfica

A identificação de RMs é uma tarefa manual que requer criatividade e um bom conhecimento do programa em teste. Para facilitar esse trabalho, nesta seção será apresentado o termo **Padrões de Saída de Relação Metamórfica** (PSRM), baseado no trabalho de Segura e Zhou (2018). Segundo Segura e Zhou (2018), para cada PSRM, pode haver uma variedade de relações de entrada que satisfazem a relação de saída definida

pelo padrão. A relação de entrada pode depender do sistema em teste, o que normalmente inclui adicionar, remover ou alterar parâmetros de entrada e recursos de entrada de uma determinada maneira. Quando combinado com relações de entrada apropriadas, cada PSRM pode ser instanciado em uma ou mais RMs concretas em cada sistema em teste. Então, estes padrões que poderão ser identificados durante a execução dos testes neste trabalho:

1. ***Equivalence* (Equivalência):** Esse padrão representa as relações em que as saídas Originais e Complementares são equivalentes. Pode ser definido duas ou mais saídas como equivalentes se incluírem os mesmos itens, embora não necessariamente na mesma ordem. Por exemplo, ordenar os resultados de uma consulta por critérios diferentes deve produzir resultados equivalentes.

2. ***Equality* (Igualdade):** Esse padrão representa as relações em que as saídas de Originais e Complementares devem conter os mesmos itens e na mesma ordem. Por exemplo, não importa se os valores padrão dos parâmetros de consulta são especificados ou não. Este padrão pode ser generalizado para representar as RMs em que a saída de Original deve ser igual a pelo menos uma saída Complementar.

3. ***Subset* (Subconjunto):** Este padrão agrupa as relações onde as saídas Complementares devem ser subconjuntos da saída Original e subconjuntos entre eles. Esse padrão pode ser instanciado com qualquer parâmetro que filtre os resultados de uma consulta, ou em operações de consulta em que a maioria dos parâmetros são filtros. Resumindo, a saída Complementar deve ser um subconjunto da saída Original.

4. ***Complete* (Completo):** Esse padrão inclui aquelas relações em que a união das saídas de Acompanhamento deve conter os mesmos itens que a saída Original. Esse padrão é normalmente aplicável quando os resultados da pesquisa, por exemplo, podem ser divididos em partições separadas e completas. Esse padrão envolve pelo menos três entradas Complementares ao contrário da maioria dos outros padrões, em que uma única entrada Complementar é normalmente usada.

5. ***Disjoint* (Disjunto):** Este padrão representa aquelas relações onde a interseção entre as saídas Originais e Complementares deve ser vazia. Esse padrão é aplicável quando os resultados da consulta podem ser divididos em partições separadas, ou seja, a saída de ambos os casos de teste não deve ter itens em comum.

2.4 CONSIDERAÇÕES FINAIS

Este capítulo apresentou a fundamentação teórica necessária para entender o desenvolvimento deste trabalho. Foi realizada uma introdução sobre a **Lei de Acesso a**

informação, sobre **Testes de software**, tipos e níveis de teste e testes funcionais, que é a base para compreensão do problema desta pesquisa. Além disso, foram apresentados os conceitos de **Testes Metamórficos**, **Relações Metamórficas**, todo o Processo de Teste Metamórfico e por fim, os **Padrões de saída de Relação Metamórfica**.

3 METODOLOGIA

A abordagem metodológica utilizada neste trabalho é conhecida como ***Design Science Research*** (DSR), que é voltada a orientar pesquisas que pretendem justamente resolver problemas ou buscar soluções para tais problemas. Segundo Dresch et al. (2015), DSR é uma forma de produção de conhecimento científico que envolve o desenvolvimento de uma inovação, com a intenção de resolver problemas do mundo real e, ao mesmo tempo, fazer uma contribuição de caráter prescritivo.

As seções seguintes irão fundamentar e apresentar a metodologia deste trabalho. Na seção 3.1 é apresentada a fundamentação teórica sobre ***Design Science Research***, que foi a abordagem metodológica utilizada na condução dessa pesquisa. Na seção 3.2 é apresentado as **Atividades** que foram desenvolvidas neste trabalho.

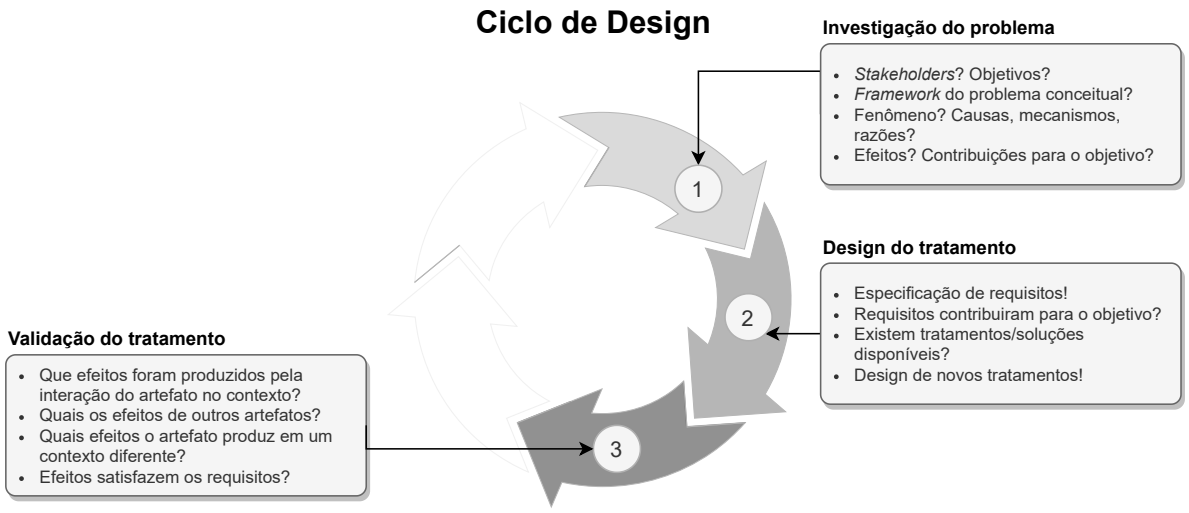
3.1 *DESIGN SCIENCE*

O método DSR busca a geração de conhecimento no processo de criação de artefatos, com base na compreensão de um determinado problema. A partir do entendimento do problema, estas saídas são elaboradas e avaliadas, com o intuito de transformar estes contextos para estados desejados e/ou melhores (DRESCH et al., 2015).

Segundo (BAX, 2014), DSR envolve construir, investigar, validar e avaliar artefatos, tais como construtos, arcabouços, modelos, métodos e instâncias de sistema de informações, a fim de resolver novos problemas práticos. Ou seja, um problema é o responsável por conduzir uma pesquisa, e a partir disso, poderá surgir outros problemas práticos e questões que irão agregar conhecimento. Para Wieringa (2014), problemas e questões encadeiam um verdadeiro ciclo, que consiste em 3 etapas e este pode ser exemplificado de acordo com o trabalho de Moreira (2020), como demonstrado na Figura 3.1.

- O ciclo se inicia na etapa "**Investigação do problema**"(1), na qual o foco principal é buscar a compreensão sobre o problema antes de propor um artefato como a solução. Ou seja, a ideia principal nessa etapa é identificar, descrever, explicar e avaliar os problemas que poderão ser tratados nas próximas etapas.
- A etapa seguinte é o "***Design* do tratamento**"(2), que verifica com base em todos os artefatos que foram gerados durante a investigação do problema, se tem o que é necessário para a solução do problema. O objetivo principal é criar uma melhor situação para os *stakeholders* quando for aplicada a solução. Um *stakeholder* de um problema no contexto de *design science* pode ser um indivíduo, um grupo de

Figura 3.1 – Ciclo de *design*



Fonte: Moreira (2020).

indivíduos, ou instituições que podem ser afetadas pelo tratamento do problema (WIERINGA, 2014).

- Ao fim do Ciclo de Design, a "**Validação do tratamento**"(3) tendo como base tudo o que foi produzido anteriormente, tem como objetivo justificar se a contribuição que foi feita durante todo o processo realmente irá atender aos objetivos dos *stakeholders*, caso venha acontecer uma implementação. A validação é feita utilizando um contexto real para antever o que pode acontecer ao aplicar de fato o tratamento.

Alguns termos serão explicados para serem usados como base de conhecimento, com o propósito de melhorar entendimento do conteúdo que for apresentado neste trabalho. As explicações dos termos apresentados a seguir, foram baseados de acordo com o estudo apresentado no trabalho de Moreira (2020).

- **Design:** Decisão sobre o que fazer;
- **Tratamento:** Substitui o termo "solução", pois é possível que o artefato solucione apenas alguma parte do problema ou não solucionar o problema;
- **Artefatos:** Tudo o que foi produzido durante os ciclos de *design*;
- **Implementação:** Consiste na execução da proposta de tratamento que foi planejada no contexto original do problema.

3.2 ATIVIDADES

Para o desenvolvimento deste trabalho, com o intuito principal de atingir os objetivos citados anteriormente, foram feitas as atividades elencadas a seguir. As atividades foram desenvolvidas entre os meses de setembro do ano de 2020 e junho do ano 2021.

- **Atividade 1 - Realizar a implementação dos testes:** Após ter sido feito um estudo sobre todos os sistemas de transparência a serem testados, foi desenvolvida a implementação dos casos de teste de acordo com as Relações Metamórficas que foram identificadas durante as avaliações feitas;
- **Atividade 2 - Realizar avaliação dos experimentos:** Após a atividade anterior ter sido feita, os resultados de todos os testes foram analisados e posteriormente avaliados, com o intuito de verificar se existem problemas nos sistemas de transparência testados.
- **Atividade 3 - Submeter escrita no documento com a descrição da realização e avaliação dos testes:** Na última etapa, foi submetido neste documento a análise de todas as atividades que foram descritas, listando todas as etapas de implementação e testes que foram realizados. Em seguida foi descrito como foram feitas as avaliações com base em todos os casos de teste, validando e descrevendo se existem problemas nos sistemas de transparência testados.

4 CICLOS DE DESIGN

Este capítulo apresenta a proposta e execução do ciclo de *design* deste trabalho, como evidenciado no capítulo 3.

4.1 INVESTIGAÇÃO DO PROBLEMA

A investigação do problema busca mapear os fenômenos existentes dentro de um contexto e como uma proposta de melhoria pode ser aplicada, bem como os *stakeholders* e seus objetivos dentro de um ciclo de *design*. Nessa etapa do projeto, os artefatos ainda não existem ou carecem de melhorias.

O contexto abordado nesse trabalho será sobre os **sistemas de transparência**, que são evidenciados no Apêndice A. Prováveis problemas que poderão ser identificados nesses sistemas, estão detalhados na seção de problemática deste trabalho. Os *stakeholders* são **gestores públicos**, **população** interessada nas informações das cidades envolvidas nessa proposta, *desenvolvedores e empresas* envolvidas na produção de *softwares* de gestão pública, principalmente os que serão observados nessa proposta.

Buscando novamente responder a pergunta de pesquisa: "**Como demonstrar a presença de erros em sistemas de transparência com código fechado?**", esse ciclo vai buscar realizar uma **proposta de solução** para que, ao final, seja possível ter uma ideia do que pode ser realizado, as limitações da abordagem, e os resultados obtidos.

Nesta etapa de investigação do problema do **ciclo 1 de *design***, foram analisados os sistemas de transparência providos pelas cidades da Região Metropolitana de Cajazeiras (MACEDO, 2015). O levantamento desses sistemas está no Apêndice A, evidenciando quais opções para busca e filtragem são disponibilizados pelos sistemas, para executar consultas a partir das **Relações Metamórficas** posteriormente encontradas.

Uma vez que a maioria dos sistemas compartilham das mesmas características por sua natureza baseada em legislação, é esperado que a maioria desses sistemas apresentem interfaces de interação compatíveis. A escolha pelos sistemas da região metropolitana de Cajazeiras no primeiro ciclo, se dá arbitrariamente por questões de limitação de escopo. A intenção é que ciclos futuros possam abordar outras esferas da administração pública ou até um número maior de sistemas de transparência.

4.2 DESIGN DE TRATAMENTO

A tratamento da etapa de *design* de tratamento do ciclo 1, é baseada em **Testes Metamórficos** (TM), que como citado na seção 2, fornece uma alternativa para testar um sistema quando as saídas esperadas são desconhecidas ou difíceis de serem comparadas, e após executar cada um dos casos de teste, verificar se várias dessas saídas cumprem propriedades chamadas de **Relações Metamórficas** (RM). As seguintes RMs foram levantadas para esse ciclo, de acordo com os sistemas de transparência elencados na investigação do problema, evidênciados no Apêndice A.

As propostas de RMs apresentadas são baseadas na ideia de **Padrões de saída de Relação Metamórfica** (PSRM), apresentada no trabalho de Segura e Zhou (2018). Foram levantadas inicialmente cinco RMs como propostas, detalhadas na subseção a seguir. Uma vez que estas RMs são ideias abstratas, fica mais fácil utilizá-las para domínios distintos. Outros trabalhos apresentam RMs semelhantes, porém muito específicas quanto ao domínio, o que as torna muito difíceis de serem aplicadas.

4.2.1 Relações Metamórficas propostas

RM - 1 - Período: Dado um período de consulta P que representa a **quantidade de meses** que estão incluídos em uma consulta de despesas/receitas em um portal de transparência, o sistema retorna um resultado R que representa os itens dos meses informados. Ao diminuir o valor de P em um caso de teste complementar, o resultado total da quantidade de itens despesas/receitas deve ser $\leq$ do que R. Essa relação se baseia nos PSRM ***Equivalence***, ***Equality*** e ***Subset*** citados no trabalho de Segura e Zhou (2018).

RM - 2 - Subconjunto em órgão/unidade gestora/orçamentária: Ao originar uma consulta sem nenhum parâmetro para despesas/receitas em um portal de transparência, o sistema irá retornar um resultado R que representa todas as despesas feitas a partir do ano fiscal consultado. Ao incluir um parâmetro que pode ser referido por órgão/unidade gestora/unidade orçamentária P em um caso de teste complementar, o resultado total da quantidade de itens despesas/receitas deve ser $\leq$ do que R. Essa relação se baseia nos PSRM ***Equivalence***, ***Equality*** e ***Subset*** citados no trabalho de Segura e Zhou (2018).

RM - 3 - Consulta por CPF/CNPJ: Ao originar uma consulta sem nenhum parâmetro para despesas/receitas em um portal de transparência, o sistema irá retornar um resultado R que representa todas as despesas feitas a partir do ano fiscal consultado. Ao incluir um parâmetro P que pode ser referido por CPF/CNPJ em um caso de

teste complementar, o resultado total da quantidade de itens despesas/receitas deve ser $\leq$ do que R. Essa relação se baseia nos PSRM ***Equivalence***, ***Equality*** e ***Subset*** citados no trabalho de Segura e Zhou (2018).

RM - 4 - Consulta em Conjunto por órgão/unidade gestora/orçamentária: Ao originar uma consulta sem nenhum parâmetro para despesas/receitas em um portal de transparência, o sistema irá retornar um resultado R que representa todas as despesas feitas a partir do ano fiscal consultado. Ao incluir um parâmetro P que pode ser referido por órgão/unidade gestora/unidade orçamentária em um caso de teste complementar, o resultado total da quantidade de itens despesas/receitas deve ser $\leq$ do que R. Ao se criar então mais dois casos testes complementares, cada teste com um(a) órgão/unidade gestora/unidade orçamentária diferente do primeiro caso de teste complementar citado anteriormente, cada um desses três casos de teste criados terão que retornar valores diferentes entre os seus resultados, em que o conjunto de todos desses valores, seja igual (=) ao resultado R do caso de teste Original. Nesse caso, essa relação se baseia no PSRM ***Complete***, citada no trabalho de Segura e Zhou (2018).

RM - 5 - Consulta Disjunta por orgão/unidade gestora/orçamentária: Ao originar uma consulta com um parâmetro P1 que representa um órgão/unidade gestora/unidade orçamentária em um portal de transparência, o sistema irá retornar um resultado R1 que representa todas as despesas feitas por tal órgão/unidade gestora/unidade orçamentária incluído na consulta, a partir do ano fiscal consultado. Ao incluir um parâmetro P2 em um Caso de Teste Complementar, que pode ser referido por um órgão/unidade gestora/unidade orçamentária distinto do parâmetro p1 do Caso de Teste Original, o resultado R2 deve ser o total de despesas feitas por tal órgão/unidade gestora/unidade orçamentária incluído na consulta, a partir do ano fiscal consultado e também o resultado R2 deve ser $\neq$ do resultado R1, ou seja, ambos os casos de teste não deve ter itens em comum. Nesse caso, essa relação se baseia no PSRM ***Disjoint***, citada no trabalho de Segura e Zhou (2018).

4.2.2 Ferramentas

Serão executados testes nos sistemas de transparência elencados por meio das ferramentas **Selenium**[1] e **JUnit**[2] em dias distintos, em horários alternados, sendo a ordem dos casos de teste escolhidos arbitrariamente. Essa decisão por executar os casos de teste em momentos distintos acontece por não termos a garantia de disponibilidade dos serviços em que os sistemas estão hospedados.

[1] <https://www.selenium.dev/>
[2] <https://junit.org/junit5//>

O Selenium é um conjunto de ferramentas *open source* (código aberto) multiplataforma, que é usado para testar aplicações web de forma automatizada a partir da localização de elementos *DOM* (*Document Object Model*) de uma página. A ferramenta permite criar *scripts* de testes funcionais automatizados para aplicações *web*, sendo que esses *scripts* reproduzem o ambiente real de navegação da aplicação sem que o testador tenha a necessidade de ter que aprender uma linguagem de *scripts* de teste. A finalidade do Selenium resulta em várias funções de teste que são voltadas para qualquer tipo de necessidade em aplicações *web*, que realiza testes flexíveis e que permite localizar qualquer elemento *HTML* dentro da interface do sistema e comparar resultados esperados e obtidos, tendo estimativas do comportamento da aplicação. Os testes podem ser executados em qualquer navegador que possuir suporte a *JavaScript*, por função da integração dessa ferramenta com o navegador.

JUnit é um *Framework open source* que facilita a criação e execução de testes unitários na linguagem de programação ***Java***, além de permitir a escrita testes que retenham seu valor ao longo do tempo, ou seja, que possam ser reutilizáveis. O JUnit possibilita a criação das classes de testes sendo utilizado principalmente para verificar se os resultados gerados pelos métodos são os esperados. Caso não sejam de acordo com o esperado, o JUnit exibe os possíveis erros que estão ocorrendo nos métodos. Essa verificação é chamada de teste unitário ou teste de unidade. O teste de unidade testa o menor dos componentes de um sistema de maneira isolada. Com JUnit, o testador tem a possibilidade de usar esta ferramenta para criar um modelo padrão de testes de forma automatizada.

As ferramentas foram escolhidas pela natureza do processo de experimento, além da familiaridade dos autores com as mesmas. É oportuno ressaltar que outras ferramentas poderiam ser utilizadas para produzir os mesmos resultados, tais como a ferramenta de automação de testes ***TestComplete***[3], que permite a utilização de diversas linguagens, como ***JavaScript***, e ***Python*** e que possibilita a criação de testes automatizados para aplicativos ***Web*** e das plataformas ***iOS***, ***Microsoft Windows*** e ***Android***.

4.2.3 Planejamento do processo de testes

O planejamento do processo para execução dos Testes Metamórficos transcorre pelos seguintes passos:

1. **Acessar o website do sistema de transparência sob teste:** O teste automatizado com a ferramenta Selenium executará uma instância do *browser*, com o intuito de descobrir a disponibilidade do sistema, onde deve ser inserido a *URL* desse sis-

[3] <https://smartbear.com/product/testcomplete/overview///>

tema que será feito o experimento deste trabalho, como evidenciado na tabela A do Apêndice A).

2. **Buscar os campos envolvidos no teste:** Ao acessar o sistema sob teste, é preciso investigar todos os elementos ***HTML*** disponíveis da página e onde estão dispostos, pois é a partir dessa investigação que se torna viável a realização de consultas no sistema para a execução dos experimentos.

3. **Executar a consulta:** A partir da busca dos campos envolvidos no teste e por meio das Relações Metamórficas identificadas posteriormente, é executada a consulta no sistema, produzindo o Caso de Teste Original. O resultado desse caso de teste é armazenado para que possa ser avaliado na etapa 5 deste processo.

4. **Alterar os valores de entrada de acordo com a Relação Metamórfica:** Ao criar um caso de teste original executando uma consulta no sistema testado, uma nova consulta poderá ser feita como sendo referente ao Caso de Teste Complementar, inserindo novos parâmetros na consulta de acordo com as Relações Metamórficas encontradas durante o estudo desse experimento.

5. **Comparação dos resultados:** Após a execução de todas as etapas anteriores, caso os resultados de ambos os testes demonstrem que o sistema não conseguiu manter a propriedade, ou seja, não satisfaça a Relação Metamórfica, o teste automatizado aponta uma possível falha do sistema.

É ressaltado que cada página *web* dos sistemas testados, possuem um *DOM* diferente, e é por meio desta que a ferramenta **Selenium** localiza todos os elementos *HTML* que serão utilizados para a execução dos testes. Apesar de ser possível generalizar cenários para diferentes páginas *web* com o uso dessa ferramenta, **a implementação de cada teste será feita de uma forma diferente**, por conta da diferenciação da disponibilização dos itens de cada sistema, dando assim integridade aos testes executados.

O fato dessas ferramentas atuarem sobre o *DOM* da página *web*, traz algumas limitações e ameaças a validade, explicitadas na seção 4.6. É importante destacar que qualquer **teste de caixa preta sofre com efeitos colaterais das mudanças do sistema sob teste**, uma vez que é independente de sua implementação.

Ao final da execução dos testes, é possível assegurar o mapeamento dos **Padrões de Relação Metamórfica**, que constituem "ideias reusáveis" do que pode ser testado e como esse teste será planejado, deixando para que cada equipe de desenvolvimento implemente esses testes de sua maneira.

4.3 VALIDAÇÃO DO TRATAMENTO

Nesta seção, serão descritas as avaliações que foram feitas utilizando as **Relações Metamórficas** descritas anteriormente e a eficácia da metodologia proposta, que tem como objetivo principal revelar falhas nos sistemas de transparência testados. Em particular, o interesse principal deste trabalho é responder a seguinte pergunta de pesquisa: **"Como demonstrar a presença de erros em sistemas de transparência com código fechado?"** Portanto, a resposta será buscada pela análise dos resultados oriundos dos testes no 15 sistemas de transparência das cidades pertencentes à **Região Metropolitana de Cajazeiras** (MACEDO, 2015), evidênciado no Apêndice A.

O teste automatizado é iniciado ao acessar o endereço *url* em que o sistema se encontra hospedado. Para as RMs **1 - Período**, **2 - Subconjunto em órgão/unidade gestora/orçamentária**, **3 - Consulta por CPF/CNPJ** e **4 - Consulta em Conjunto por órgão/unidade gestora/orçamentária**, o valor do **Caso de Teste Original** (CTO) foi capturado a partir do acesso do teste automatizado a *interface* do sistema, sendo que este é referente a quantidade de todos os itens cadastrados no período entre o primeiro dia do ano fiscal consultado até a data onde o caso de teste foi executado.

Para a Relação Metamórfica **5 - Consulta Disjunta por órgão/unidade gestora/orçamentária**, o valor do CTO foi capturado a partir do acesso do teste automatizado a *interface* do sistema, na qual é selecionada uma unidade gestora de forma aleatória e posteriormente são capturados todos os identificadores referentes a cada despesa listada na página que foi cadastrado no sistema entre o primeiro dia do ano fiscal consultado até a data em que o caso de teste foi executado e referente a unidade gestora selecionada.

Para cada uma das RMs propostas no trabalho, o valor do **Caso de Teste Complementar**(CTC) foi capturado da seguinte forma:

- **Relação Metamórfica 1 - Período:** A quantidade de valores referentes ao **CTC** foi capturada ao inserir um período de tempo menor do que o período inserido no **CTO**.
- **Relação Metamórfica 2 - Subconjunto em órgão/unidade gestora/orçamentária:** A quantidade de valores referentes ao **CTc** foi capturada ao inserir como parâmetro da consulta uma unidade gestora identificada no sistema, na qual o sistema apenas poderia retornar os valores cadastrados referentes a esta unidade gestora no mesmo ano fiscal consultado.
- **Relação Metamórfica 3 - Consulta por CPF/CNPJ:** A quantidade de valores referentes ao **CTC** foi capturada ao inserir como parâmetro da consulta um número

de CPF/CNPJ identificado posteriormente em uma consulta manual no sistema, na qual o sistema apenas poderia retornar os valores cadastrados referentes a este CPF/CNPJ no mesmo ano fiscal consultado.

- **Relação Metamórfica 4 - Consulta em Conjunto por órgão/unidade gestora/orçamentária:** A quantidade de valores referentes ao **CTC** foi capturada ao inserir como parâmetro da consulta cada uma das unidades gestoras disponibilizadas no sistema, em que este valor pode ser representado pela somatória quantidade de gastos cadastrados referentes a cada unidade gestora consultada.
- **Relação Metamórfica 5 - Consulta Disjunta por orgão/unidade gestora/orçamentária:** O resultado do **CTC** foi capturado ao inserir como parâmetro da consulta uma unidade gestora de forma aleatória e diferente da inserida como parâmetro no **CTO**. Posteriormente os valores obtidos do **CTO** e do **CTC** foram comparados com o intuito de descobrir se existiam valores iguais entre os dois resultados. Esta abordagem foi repetida 30 vezes, com o intuito de buscar resultados idênticos em órgãos distintos e validar o caso de teste executado.

Os dados que são armazenados e disponibilizados por esses sistemas de transparência são atualizados com frequência, o que pode levar a inconsistências entre os valores obtidos para os **CTO** e para os **CTC**. Para resolver este problema, seguido a cada execução de um caso de teste, foi executado o mesmo caso de teste. Caso os valores obtidos a partir de cada caso de teste fossem distintos, a avaliação seria reiniciada.

Não foi possível executar os testes utilizando todas as RMs propostas neste trabalho em alguns dos sistemas elencados. Em determinados sistemas, foram possíveis apenas a execução dos casos de teste entre períodos de data, sendo assim possível apenas verificar a **Relação Metamórfica 1 - Período**. Em outros sistemas não foi possível executar os casos de teste referentes a **Relação Metamórfica 5 - Consulta Disjunta por órgão/unidade gestora/orçamentária**, pois os identificadores de cada um dos itens capturado eram dinâmicos, impossibilitando a execução de um teste automatizado. Também houveram exemplos de sistemas em que foram identificados apenas uma unidade gestora, assim não sendo possível fazer comparação entre resultados de unidades gestoras diferentes.

Ao longo desta seção, será detalhado os resultados para cada uma das RMs identificadas. A Tabela 4.1 mostra os resultados gerais das RMs aplicadas aos sistemas de transparência testados, em que **CTO** é referente ao Caso de Teste Original, **CTC** é referente ao Caso de Teste Complementar e **VI** é referente ao Valor Inserido. O detalha-

mento sobre os resultados e como os testes foram implementados em todos os sistemas, estarão especificados nas subseções seguintes.

Tabela 4.1 – Resultados dos testes executados

Cidade	RM1			RM2			RM3			RM4			RM5		
	CTO	CTC	VI	CTO	CTC	VI	CTO	CTC	VI	CTO	CTC	VI	CTO	CTC	VI
Bernardino Batista	1614	1347	01/02/2021	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Bom Jesus	1272	1138	01/02/2021	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Bonito de Santa Fé	1366	1151	01/02/2021	1366	59	Gabinete do Prefeito	1366	11	00000000 120766	1366	1366	Unidades Gestoras	N/A	N/A	N/A
Cachoeira dos Índios	4654	4318	28/02/2020	4654	96	Gabinete do Prefeito	4654	155	0000000 0009903	4654	4654	Unidades Gestoras	N/A	N/A	N/A
Cajazeiras	2280	1939	01/02/2021	2280	104	Secretaria de Saúde	2280	73	0912365 4000187	2280	2280	Unidades Gestoras	0	0	Unidades Gestoras
Carrapateira	104	83	01/02/2021	N/A	N/A	Gabinete do Prefeito	N/A	N/A	0000000 0116572	N/A	N/A	Unidades Gestoras	N/A	N/A	N/A
Joca Claudino	1202	1045	01/02/2021	1202	24	N/A	1202	164	N/A	1202	1202	N/A	N/A	N/A	N/A
Monte Horebe	758	758	01/02/2021	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Poço Dantas	173	157	01/02/2018	173	173	Câmara Municipal	173	37	0000000 0124168	173	173	Unidades Gestoras	N/A	N/A	N/A
Poço José de Moura	4693	3331	01/05/2020 e 01/12/2020	4693	519	Secretaria de educação	4693	354	0000000 0144940	4693	4693	Unidades Gestoras	0	0	Unidades Gestoras
Santa Helena	1693	1347	01/02/2021	1693	83	Fundo Municipal de saúde	1693	8	0000000 0087459	1693	1693	Unidades Gestoras	N/A	N/A	N/A
São João do Rio do Peixe	91	79	01/02/2021	91	91	Câmara municipal de SJRP	91	9	0000000 0000191	91	91	Unidades Gestoras	N/A	N/A	N/A
São José de Piranhas	5008	3756	01/02/2021	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Triunfo	2007	1822	01/02/2021	2007	2007	Prefeitura Municipal de Triunfo - PB	2007	132	0000000 0164119	2007	2007	Unidades Gestoras	N/A	N/A	N/A
Uiraúna	2704	2283	01/02/2021	2704	89	Governo e Articulação Politica	2704	227	0000000 0124168	2704	2704	Unidades Gestoras	N/A	N/A	N/A

Fonte: Elaborado pelo Autor (2021)

4.3.1 Testes realizados

Nesta seção serão melhor detalhados os resultados da execução de cada caso de teste referente às **Relações Metamórficas** (RMs) propostas neste trabalho.

4.3.1.1 RM - 1 - Período

Em todos os sistemas testados foi possível executar os testes utilizando a **RM 1 - Período**, pois todos possuem consulta por período. Com exceção do sistema de transparência da cidade de **Monte Horebe**, em todos os outros sistemas, a quantidade de itens encontrados referente ao **Caso de Teste Original** foi maior do que a quantidade de itens encontrados referente ao **Caso de Teste Complementar**. Na execução do teste no sistema de transparência da cidade de **Monte Horebe**, a quantidade de itens referente ao **Caso de Teste Original** foi igual a quantidade de itens referente **Caso de Teste Complementar**, concluindo assim que os resultados dos testes em todos os sistemas de transparência testados satisfizeram a **RM** baseada nos **Padrões de saída de Relação Metamórfica** (PSRM) ***Equivalence***, ***Equality*** e ***Subset*** citados no trabalho de Segura e Zhou (2018).

4.3.1.2 RM - 2 - Subconjunto em órgão/unidade gestora/orçamentária

Com exceção dos sistemas de transparência das cidades de **Bernardino Batista**, **Bom Jesus**, **Carrapateira**, **Monte Horebe** e **São José de Piranhas**, em todos os outros sistemas foi possível executar os testes utilizando a **RM 2 - Subconjunto em órgão/unidade gestora/orçamentária**, pois estes possuíam a consulta incluindo como parâmetro um órgão/unidade gestora/orçamentária. Nos testes executados em sistemas de transparência das cidades de **Poço Dantas**, **São João do Rio do Peixe** e **Triunfo**, a quantidade de itens referente ao **Caso de Teste Original** foi igual a quantidade de itens referente **Caso de Teste Complementar**. No restante, a quantidade de itens encontrados referente ao **Caso de Teste Original** foi maior do que a quantidade de itens encontrados referente ao **Caso de Teste Complementar**, concluindo que os resultados dos testes em todos os sistemas de transparência testados satisfizeram a **RM** baseada nos PSRM ***Equivalence***, ***Equality*** e ***Subset*** citados no trabalho de Segura e Zhou (2018).

4.3.1.3 RM - 3 - Consulta por CPF/CNPJ

Com exceção dos sistemas de transparência das cidades de **Bernardino Batista, Bom Jesus**, **Carrapateira**, **Monte Horebe** e **São José de Piranhas**, em todos os outros foi possível executar os testes utilizando a **RM 3 - Consulta por CPF/CNPJ**, pois estes possuíam a consulta incluindo como parâmetro um CNPJ/CPF. Em todos os sistemas de transparência em que foi possível a execução dos testes, a quantidade de itens encontrados referente ao **Caso de Teste Original** foi maior do que a quantidade de itens

encontrados referente ao **Caso de Teste Complementar**, concluindo que os resultados dos testes em todos os sistemas de transparência testados satisfizeram a **RM** baseada nos PSRM ***Equivalence***, ***Equality*** e ***Subset*** citados no trabalho de Segura e Zhou (2018).

4.3.1.4 RM - 4 - Consulta em Conjunto por órgão/unidade gestora/orçamentária

Com exceção dos sistemas de transparência das cidades de **Bernardino Batista**, **Bom Jesus**, **Carrapateira**, **Monte Horebe** e **São José de Piranhas**, em todos os outros sistemas foi possível executar os testes utilizando a **RM 4 - Consulta em Conjunto por órgão/unidade gestora/orçamentária**, pois estes possuíam a consulta incluindo como parâmetro um um órgão/unidade gestora/orçamentária. Em todos os sistemas de transparência em que foi possível a execução dos testes, a quantidade de itens encontrados referente ao **Caso de Teste Original** foi igual a quantidade de itens encontrados referente ao **Caso de Teste Complementar**, concluindo que os resultados dos testes em todos os sistemas de transparência testados satisfizeram a **RM** baseada nos PSRM ***Complete*** citada no trabalho de Segura e Zhou (2018).

4.3.1.5 RM - 5 - Consulta Disjunta por orgão/unidade gestora/orçamentária

Apenas nos sistemas de transparência das cidades de **Cajazeiras** e **Poço José de Moura** foi possível executar os testes utilizando a **RM 5 - Consulta Disjunta por orgão/unidade gestora/orçamentária**, pois estes possuíam a consulta incluindo como parâmetro um um órgão/unidade gestora/orçamentária além de ser possível a cada caso de teste identificar e capturar cada um dos elementos disponibilizados e comparar os resultados obtidos. Nestes sistemas de transparência em que foi possível a execução dos testes, em todas as repetições comparando os resultados obtidos nos **Caso de Teste Original** e **Caso de Teste Complementar** não foram encontrados valores em comum, satisfazendo a **RM** baseada nos PSRM ***Disjoint***, citada no trabalho de Segura e Zhou (2018).

4.4 CICLOS EXECUTADOS

Neste trabalho foi instanciado apenas um ciclo de *design*, executando casos de teste utilizando as Relações Metamórficas identificadas anteriormente. Algumas dificuldades influenciaram na produção e planejamento de novos ciclos de *design*. Uma dificuldade importante que ocorreu duranto a execução do ciclo de *design* que pode ser destacada, é a curta quantidade de tempo que foi disponibilizada para a produção deste trabalho. Apesar disso, novos ciclos de design serão sugeridos na seção de **Trabalhos futuros** (Seção 5.2). Outro problema se trata sobre a aprendizagem e experiência no uso das ferramentas utilizadas neste trabalho (notadamente Selenium), pois, sem ter um conhecimento prévio

destas ferramentas, foi necessário um treinamento anterior à criação e execução dos casos de teste, o que teve um custo de tempo considerável. Por último, a própria implementação de cada um dos testes, demandaram tempo e esforço considerável para serem implementados, já que os sistemas testados possuem interfaces distintas, e, antes de elaborar o caso de teste, é necessário identificar previamente todos os componentes presentes na página.

4.5 ANÁLISE DOS RESULTADOS

Apesar dos resultados obtidos indicarem a ausência de violações das **Relações Metamórficas** (RMs), isso não demonstra necessariamente a corretude dos sistemas, apenas que as RMs levantadas neste trabalho não foram capazes de demonstrar erros no conjunto selecionado. Isso pode indicar duas possibilidades:

1. As RMs levantadas não foram capazes de revelar falhas nos sistemas de transparência levantados neste trabalho, embora possam ser úteis para outros sistemas, vide como exemplo o trabalho de (SEGURA et al., 2019), na qual também foram levantadas RMs acerca dos sistemas a serem testados e os resultados a partir de cada caso de teste executado, foram avaliados a partir dos PSRM.
2. O conjunto de sistemas testados é ínfimo dado o total de sistemas do mesmo tipo que pode ser encontrado a nível estadual ou federal.

Os testes que foram executados nesse trabalho estão atados ao período de produção e execução deste trabalho, uma vez que as interfaces de interação podem mudar a qualquer momento (elementos trocando de lugar, rótulos diferentes, e seletores/identificadores alterados), causando efeitos colaterais no teste já implementados, fazendo com que os mesmos deixem até de funcionar por completo. Esse é uma das inconveniências de se utilizar testes de Caixa Preta, porém, como citado na **Problemática** (Seção 1.3), esse é um obstáculo que infelizmente não pôde ser ultrapassado dentro do nosso contexto.

4.6 DIFICULDADES ENCONTRADAS

Uma das dificuldades encontradas ocorreu durante as capturas dos resultados referentes a cada caso de teste, pois em alguns dos sistemas de transparência testados não era disponibilizada a quantificação dos resultados obtidos após ser feita a consulta nos testes. Como demonstrado na Figura 4.1, um exemplo que pode ser comentado é sobre o sistema de transparência da cidade de **Poço Dantas**, sendo necessário utilizar outra abordagem para a captura dos resultados. O valor **Total da Página** e **Total Geral** explicitam uma soma dos valores referentes aos gastos públicos consultados, não fazendo

qualquer indicação sobre a quantidade de despesas que estão na *interface* do sistema de transparência testado.

Figura 4.1 – Problema encontrado no sistema de transparência de Poço Dantas

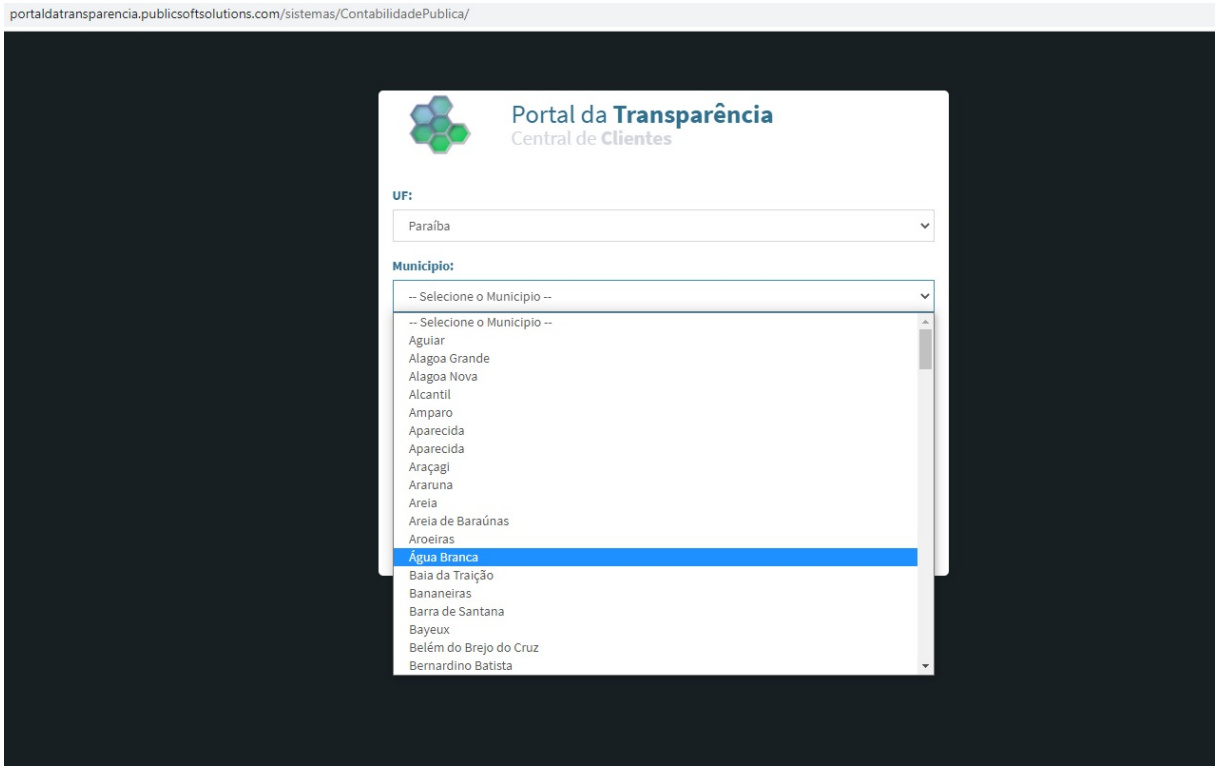
Fonte: Elaborado pelo Autor (2021)

Outra dificuldade encontrada é referente a sistemas que se encontravam indisponíveis no momento da execução dos casos de teste, impossibilitando o acesso ao sistema e posteriormente a captura dos resultados. Um exemplo que vale a pena destacar é o do sistema de transparência da cidade de **São João do Rio do Peixe**, o qual na data da execução dos casos de teste estava indísponível, sendo necessário utilizar outra abordagem para a captura dos dados referentes a este sistema.

A abordagem utilizada para capturar os dados referentes a estes dois sistemas de transparência, foi executar os casos testes na plataforma da ***Publicsoft*** [4], pois foi identificado que a administração das cidades em que tiveram seus sistemas de transparência testados neste trabalho, a utiliza para armazenam dados referentes aos seus respectivos gastos públicos. Para executar os casos testes usando esta plataforma como sistema sob teste, é necessário que a ferramenta de teste automatizado selecione anteriormente o sistema de transparência que se identifica com o nome da cidade a ser testada, como demonstrado na Figura 4.2.

[4] <https://portaldatransparencia.publicsoftsolutions.com/sistemas/ContabilidadePublica/>

Figura 4.2 – Plataforma da *Publicsoft*



Fonte: Elaborado pelo Autor (2021)

Um problema a se destacar é que em vários sistemas não foi possível fazer os testes utilizando todas as RMs propostas neste trabalho, pois estas possuíam apenas consultas entre períodos de data.

Outro problema, este de menor importância, que também foi identificado é que em alguns sistemas de transparência, os dados não são atualizados com frequência, fazendo com que os testes fossem executados utilizando anos fiscais anteriores aos anos fiscais de 2020 e 2021, anos em que este trabalho foi desenvolvido. O fato desses sistemas estarem desatualizados não compromete os objetivos do teste, que são focados em **como o sistema se comporta** e não analisam os dados ou a qualidade dos mesmos.

E por último, durante a execução de alguns casos testes foi detectado alterações nos identificadores de elementos nas páginas *web* dos sistemas de transparência elencados, sendo necessário refatorar a codificação do teste automatizado cada momento que fosse detectado uma alteração na *DOM* do sistema. Os testes propostos neste trabalho são fortemente acoplados ao *DOM* da página, sofrendo de efeitos colaterais toda vez que estas sofrem alguma alteração ou atualização.

5 CONCLUSÕES

Este capítulo apresenta as considerações finais do trabalho e sugestões para novos trabalhos acerca do tema proposto.

5.1 CONSIDERAÇÕES FINAIS

O presente trabalho teve como objetivo principal **buscar erros em sistemas de transparência com código fechado da Região Metropolitana de Cajazeiras**. Primeiramente foi realizada uma pesquisa para identificar todos os sistemas de transparência de todas as cidades elencadas posteriormente e a viabilidade da execução dos casos de teste. Em seguida foi feito um levantamento de todas as **Relações Metamórficas** (RMs) propostas. Por último, foi feita a execução de todos os casos de testes utilizando as RMs propostas e em seguida a avaliação de todos os resultados obtidos.

Diante dos objetivos propostos neste trabalho, foi explorado a utilização de técnicas de **Testes Metamórficos**. As RMs levantadas anteriormente, foram implementadas em cada caso de teste. Todos os sistemas de transparência testados possuem interface e consultas com objetivos em comum, assim permitindo utilizar as mesmas RMs propostas para maioria dos sistemas de transparência elencados anteriormente, com o propósito de validar esta pesquisa. Durante a execução do **ciclo de *design***, foram utilizados a motivo de comparação os **Padrões de Saída de Relação Metamórfica** (PSRM), que já foram apresentados em outros trabalhos publicados, com o intuito de garantir a coerência da avaliação dos resultados obtidos neste trabalho.

É perceptível que a disponibilização de dados nestes sistemas não é um processo simples e exige muito esforço dos colaboradores, pois devem ser continuamente aprimorados e monitorados, para evitar que fiquem com informações desatualizadas. Em um contexto diferente do proposto neste trabalho, é observado que ainda faltam políticas que incentivem a utilização destes sistemas por parte da população.

Em suma, uma das contribuições importantes deste trabalho é destacar que a utilização de sistemas de transparência é uma forma de conscientizar a sociedade sobre os gastos públicos e demonstrar os benefícios da utilização dos dados que são disponibilizados. O desenvolvimento deste trabalho também permitiu explorar uma área inovadora em **Testes de *Software*** e que tende a ganhar mais destaque. **Testes Metamórficos** se apresentam como uma técnica interessante para para geração de casos de teste, verificação e validação, além de outros aspectos que poderão surgir à medida que seus conceitos serão utilizados para se ter uma garantia da qualidade do *software* testado.

Ao fim do ciclo de design, com as experiências e os resultados obtidos durante a execução dos casos de teste, podemos retornar a pergunta de pesquisa para tentar responder: **Como demonstrar a presença de erros em sistemas de transparência com código fechado?** Dessa forma, apesar dos resultados obtidos indicarem a ausência de violações das **Relações Metamórficas**, os principais objetivos deste trabalho podem ser considerados atingidos.

A utilização de RMs com o objetivo de criar casos de testes, verificar e validar sistemas aproveitando-se principalmente da abordagem dos PSRM, trazem vantagens para o ciclo de desenvolvimento, analisando aspectos que muitas vezes poderiam ser negligenciados com as abordagens de criação de casos de teste tradicionais baseados em requisitos. As RMs derivadas neste trabalho são apenas um pequeno conjunto de possibilidades de relações concretas que podem ser criadas a partir dos PSRM, o que poderia estender esse estudo para outras perspectivas dentro do mesmo contexto. Além disso, o trabalho traz questionamentos e possibilidades para pesquisas futuras utilizarem a mesma abordagem que foi utilizada neste trabalho.

5.2 TRABALHOS FUTUROS

Para trabalhos futuros, novos **ciclos de design** podem ser produzidos para expandir as ideias deste trabalho, como propõe a abordagem do *design science*. Foi organizado um entendimento do que poderá ser feito para além deste trabalho, mas não se limitando apenas nisso, uma vez que a abordagem proposta neste trabalho é promissora e pode ser derivada para inúmeros contextos:

1. Utilizar outros conjuntos de **sistemas de transparência de código fechado**, seja a nível municipal, estadual ou federal, com o intuito de comprovar as ideias supracitadas;

2. Levantamento de novas **Relações Metamórficas** além destas que foram propostas neste trabalho, com o mesmo intuito de demonstrar erros em sistemas de transparência. Tomando como base os conceitos de Padrões de Saída de Relações Metamórficas, um conjunto quase infinito de novas Relações Metamórficas podem ser derivadas;

3. Visto que, este estudo limita-se apenas a **análise dos sistemas de transparência referentes à Região Metropolitana de Cajazeiras**, poderia se utilizar as mesmas **Relações Metamórficas** propostas neste trabalho para executar esta mesma abordagem em diferentes tipos de sistemas em **outras áreas**. Para exemplificar, além do contexto de sistemas de transparência, poderia se utilizar a mesma abordagem utilizada neste trabalho em sistemas de **busca parametrizada**, por exemplo, os

sistemas Spotify[1] e Youtube[2] ou até mesmo redes sociais, como Twitter[3] ou Facebook[4], tendo como base o trabalho de Segura et al. (2019), sendo antes necessário verificar anteriormente as possibilidades oferecidas a partir dos parâmetros utilizados nas consultas e das Relações Metamórficas que podem ser levantadas.

[1] <https://spotify.com>
[2] <https://www.youtube.com/>
[3] <https://twitter.com>
[4] <https://www.facebook.com/>

REFERÊNCIAS

AMMANN, P.; OFFUTT, J. **Introduction to Software Testing**. Cambridge University Press, 2016. Disponível em: <https://doi.org/10.1017/9781316771273>.

BARR, E. T.; HARMAN, M.; MCMINN, P.; SHAHBAZ, M.; YOO, S. The Oracle Problem in Software Testing : A Survey. p. 1–30, 2014.

BASTOS RIOS, C. M. T. **Base De Conhecimento - Em Teste De Software**. MARTINS EDITORA, 2007. ISBN 9788580630534. Disponível em: <https://books.google.com.br/books?id=z_lsLwEACAAJ>.

BAX, M. Design science: filosofia da pesquisa em ciência da informação e tecnologia. **XV Encontro Nacional de Pesquisa em Ciência da Informação – ENANCIB 2014**, v. 42, p. 3883–3903, 01 2014.

BOURQUE, P.; FAIRLEY, R. E. **SWEBOK v.3 - Guide to the Software Engineering - Body of Knowledge.** [s.n.], 2014. 346 p. ISSN 07407459. ISBN 0-7695-2330-7. Disponível em: <www.swebok.org>.

BRASIL. **Lei Federal de n.12.572/2011**. Brasília, DF, 2011. Disponível em: <http://www.planalto.gov.br/ccivil_03/_ato2011-2014/2011/lei/l12527.htm>.

BRITO, K. dos S.; NETO, M. dos S.; COSTA, M. A. da S.; GARCIA, V. C.; MEIRA, S. R. de L. Using parliamentary brazilian open data to improve transparency and public participation in brazil. In: **Proceedings of the 15th Annual International Conference on Digital Government Research - dg.o 14**. ACM Press, 2014. Disponível em: <https://doi.org/10.1145/2612733.2612769>.

CHEN, T.; CHEUNG, S.; YIU, S. Metamorphic testing: a new approach for generating next test cases. **Dep. Comput. Sci. Hong Kong Univ. Sci. Technol.**, p. 1–11, 1998. Disponível em: <https://www.cse.ust.hk/{~}scc/publ/CS98-01-metamorphictesting.p>.

CHEN, T. Y. Metamorphic testing: A simple approach to alleviate the oracle problem. In: **2010 Fifth IEEE International Symposium on Service Oriented System Engineering**. IEEE, 2010. Disponível em: <https://doi.org/10.1109/sose.2010.31>.

CHEN, T. Y.; KUO, F.-C.; LIU, H.; POON, P.-L.; TOWEY, D.; TSE, T. H.; ZHOU, Z. Q. Metamorphic testing. **ACM Computing Surveys**, Association for Computing Machinery (ACM), v. 51, n. 1, p. 1–27, abr. 2018. Disponível em: <https://doi.org/10.1145/3143561>.

CHEN, T. Y.; POON, P.-L.; XIE, X. METRIC: METamorphic relation identification based on the category-choice framework. **Journal of Systems and Software**, Elsevier BV, v. 116, p. 177–190, jun. 2016. Disponível em: <https://doi.org/10.1016/j.jss.2015.07.037>.

DELAMARO, M. E.; MALDONADO, J. C.; JINO, M. **Introdução ao teste de software**. [S.l.]: Elsevier, 2016.

DRESCH, A.; LACERDA, D. P.; Antunes Jr, J. A. V. **Design Science Research**. Cham: Springer International Publishing, 2015. ISSN 1063-8016. ISBN 978-3-319-07373-6. Disponível em: <http://link.springer.com/10.1007/978-3-319-07374-3>.

HAYES, L. G. **Automated Testing Handbook**. [S.l.]: Software Testing Inst, 2004. ISBN 0970746504.

HORCH, J. W. **Practical Guide to Software Quality Management (Artech House Computing Library)**. [S.l.]: Artech House; 2nd Edition (January 31, 2003), 2003.

IEEE. **IEEE Standard Glossary of Software Engineering Terminology**. 1990. 1-84 p.

KANEWALA, U.; CHEN, T. Y. Metamorphic testing: A simple yet effective approach for testing scientific software. **Computing in Science & Engineering**, Institute of Electrical and Electronics Engineers (IEEE), v. 21, n. 1, p. 66–72, jan. 2019. Disponível em: <https://doi.org/10.1109/mcse.2018.2875368>.

MACEDO, R. Noberto de. RegiÃo metropolitana de cajazeiras – pb: Dos limites institucionalizados aos limites da coesÃo interna metropolitana. **Programa Institucional de Bolsas de Iniciação Científica – PIBIC (Edital PROPEX04/2015 PIBIC/CNPq UFCG)**, p. 1–11, 2015.

MOREIRA, D. D. **Testes de sistemas de classificação de cenas acústicas utilizando relações metamórficas**. Dissertação (Mestrado) — CESAR School, Recife, Brasil, 2020.

MYERS, G.; BADGETT, T.; THOMAS, T.; SANDLER, C. **The Art of Software Testing**. [S.l.]: John Wiley & Sons, 2012. (Business Data Processing: a Wiley Series). ISBN 9780471469124.

PRESSMAN, R. S. **Software engineering: A practitioner's approach**. [S.l.]: McGraw-Hill Education, 2015.

SEGURA, S.; DURAN, A.; TROYA, J.; RUIZ-CORTES, A. Metamorphic relation patterns for query-based systems. In: **2019 IEEE/ACM 4th International Workshop on Metamorphic Testing (MET)**. IEEE, 2019. Disponível em: <https://doi.org/10.1109/met.2019.00012>.

SEGURA, S.; FRASER, G.; SANCHEZ, A. B.; RUIZ-CORTES, A. A Survey on Metamorphic Testing. **IEEE Transactions on Software Engineering**, v. 42, n. 9, p. 805–824, 2016. ISSN 00985589.

SEGURA, S.; FRASER, G.; SÁNCHEZ, A. B.; CORTÉS, A. R. **Metamorphic Testing: A Literature Review v1.2**. Seville, Spain, 2016. Version 1.2.

SEGURA, S.; TOWEY, D.; ZHOU, Z. Q.; CHEN, T. Y. Metamorphic testing: Testing the untestable. **IEEE Software**, Institute of Electrical and Electronics Engineers (IEEE), v. 37, n. 3, p. 46–53, maio 2020. Disponível em: <https://doi.org/10.1109/ms.2018.2875968>.

SEGURA, S.; ZHOU, Z. Q. Metamorphic testing 20 years later. In: **Proceedings of the 40th International Conference on Software Engineering: Companion Proceeedings**. ACM, 2018. Disponível em: <https://doi.org/10.1145/3183440.3183468>.

SOMMERVILLE, I. **Software engineering**. [S.l.]: Pearson education, 2016.

TRANSPARÊNCIA, F. E. C.-G. D. U. M. D. **Aplicação da lei de acesso à informação na administração pública federal**. 2020. Disponível em: <https://www.gov.br/acessoainformacao/pt-br/central-de-conteudo/publicacoes/arquivos/aplicacao_lai_2edicao.pdf>.

WIERINGA, R. J. **Design Science Methodology for Information Systems and Software Engineering**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014. 1–332 p. ISBN 978-3-662-43838-1. Disponível em: <http://link.springer.com/10.1007/978-3-662-43839-8>.

ZHOU, Z. Q.; XIANG, S.; CHEN, T. Y. Metamorphic testing for software quality assessment: A study of search engines. **IEEE Transactions on Software Engineering**, Institute of Electrical and Electronics Engineers (IEEE), v. 42, n. 3, p. 264–284, mar. 2016. Disponível em: <https://doi.org/10.1109/tse.2015.2478001>.

APÊNDICE A – SISTEMAS LEVANTADOS PARA A PROPOSTA

Quadro A.1 – Sistemas levantados para a proposta

Cidade	Endereço URL	Consultas efetuadas
Bernardino Batista	<http://portal.prefbernardinobatista-pb.agilicloud.com.br/Cidadao/ConsultasReceitas.aspx>	Entre Períodos
Bom Jesus	<https://transparencia.elmartecnologia.com.br/Contab/Despesas?e=201031&ctx=201031&Tab=6>	Entre períodos
Bonito de Santa Fé	<https://portaldatransparencia.publicsoftsolutions.com/sistemas/ContabilidadePublica/views>	Entre períodos Cpf/Cnpj Unidade orçamentária
Cachoeira dos Índios	<https://portaldatransparencia.publicsoftsolutions.com/sistemas/ContabilidadePublica/views>	Entre períodos Cpf/Cnpj Unidade orçamentária
Cajazeiras	<https://siteseticons.com.br/portal/ws/empresa/MDg5MjM5NzEwMDAxMTU=/empenhos#>	Entre períodos Cpf/Cnpj Unidade orçamentária
Carrapateira	<https://transparencia.elmartecnologia.com.br/Contab/Despesas/Empenhos?ctx=201054>	Entre períodos Cpf/Cnpj Unidade orçamentária

Joca Claudino	<https://portaldatransparencia.publicsoftsolutions.com/sistemas/ContabilidadePublica/views>	Entre períodos Cpf/Cnpj Unidade orçamentária
Monte Horebe	<https://transparencia.elmartecnologia.com.br/Contab/Despesas?e=201122&menu=off&isModal=False>	Entre períodos
Poço Dantas	<https://portaldatransparencia.publicsoftsolutions.com/sistemas/ContabilidadePublica/views>	Entre períodos Cpf/Cnpj Unidade orçamentária
Poço José de Moura	<http://siteseticons.com.br/portal/ws/empresa/MDE2MTU3ODQwMDAxMjU=/inicio>	Entre períodos Cpf/Cnpj Unidade orçamentária
Santa Helena	<https://portaldatransparencia.publicsoftsolutions.com/sistemas/ContabilidadePublica/views>	Entre períodos Cpf/Cnpj Unidade orçamentária
São João do Rio do Peixe	<https://portaldatransparencia.publicsoftsolutions.com/sistemas/ContabilidadePublica/views>	Entre períodos Cpf/Cnpj Unidade orçamentária
São José de Piranhas	<https://transparencia.elmartecnologia.com.br/Contab/Despesas/Extra?e=201188&tabs=off&Tab=1>	Entre períodos

Triunfo	<http://portal.preftriunfo-pb.agilicloud.com.br/Cidadao/ConsultaDiarioDespesas.aspx>	Entre períodos Cpf/Cnpj Unidade orçamentária
Uiraúna	<https://portaldatransparencia.publicsoft.com.br/sistemas/ContabilidadePublica/NoM3/despesa-orcamentaria#>	Entre períodos Cpf/Cnpj Unidade orçamentária

Fonte: Elaborado pelo Autor (2021)

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DA PARAÍBA
Campus Cajazeiras
Rua José Antônio da Silva, 300 - Bairro Jardim Oásis, CEP 58.900-000, Cajazeiras (PB)
CNPJ: 10.783.898/0005-07 - Telefone: (83) 3532-4100

Documento Digitalizado Restrito

Entrega de TCC - 2

Assunto: Entrega de TCC - 2
Assinado por: Romulo Pereira
Tipo do Documento: Anexo
Situação: Finalizado
Nível de Acesso: Restrito
Hipótese Legal: Informação Pessoal (Art. 31 da Lei no 12.527/2011)
Tipo do Conferência: Cópia Simples

Documento assinado eletronicamente por:

- **Romulo Pereira Dantas Ferreira**, **ALUNO (201622010060) DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS - CAJAZEIRAS**, em 29/06/2021 11:00:50.

Este documento foi armazenado no SUAP em 29/06/2021. Para comprovar sua integridade, faça a leitura do QRCode ao lado ou acesse https://suap.ifpb.edu.br/verificar-documento-externo/ e forneça os dados abaixo:

Código Verificador: 264203
Código de Autenticação: 3107dc8031