UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ
BACHARELADO EM SISTEMAS DE INFORMAÇÃO

WILLIAM ROSENDO SOARES

DESENVOLVIMENTO DE UM REPOSITÓRIO DE MEDIDAS DE TESTE DE SOFTWARE ADERENTE AO MPT.BR

QUIXADÁ
2016

WILLIAM ROSENDO SOARES

DESENVOLVIMENTO DE UM REPOSITÓRIO DE MEDIDAS DE TESTE DE SOFTWARE ADERENTE AO MPT.BR

Trabalho de Conclusão de Curso submetido à Coordenação do Curso Bacharelado em Sistemas de Informação da Universidade Federal do Ceará como requisito parcial para obtenção do grau de Bacharel.

Área de concentração: computação

Orientador Profª. Msc. Carla Ilane Moreira Bezerra

QUIXADÁ
2016

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca do Campus de Quixadá

S652d Soares, William Rosendo
Desenvolvimento de um repositório de medidas de teste de software aderente ao MPT.Br/ William Rosendo Soares. – 2016.
82 f. : il. color., enc. ; 30 cm.

Monografia (graduação) – Universidade Federal do Ceará, Campus de Quixadá, Curso de Bacharelado em Sistemas de Informação, Quixadá, 2016.
Orientação: Profa. Msc. Carla Ilane Moreira Bezerra
Área de concentração: Computação

1. Medidas 2. Software – Testes 3. Software – Desenvolvimento I. Título.

CDD 005.14

WILLIAM ROSENDO SOARES

DESENVOLVIMENTO DE UM REPOSITÓRIO DE MEDIDAS DE TESTE DE SOFTWARE ADERENTE AO MPT.BR

Trabalho de Conclusão de Curso submetido à Coordenação do Curso Bacharelado em Sistemas de Informação da Universidade Federal do Ceará como requisito parcial para obtenção do grau de Bacharel.

Área de concentração: computação

Aprovado em: 21 / janeiro / 2016.

BANCA EXAMINADORA

_____

Profª. MSc. Carla Ilane Moreira Bezerra (Orientador(a))
Universidade Federal do Ceará-UFC

_____

Profª. Dra. Paulyne Matthews Jucá
Universidade Federal do Ceará-UFC

_____

Profª. Dra. Ingrid Teixeira Monteiro
Universidade Federal do Ceará-UFC

Dedico esse trabalho a minha família, a minha namorada, a minha orientadora e a todos que me ajudaram de alguma forma. Dedico principalmente ao meu avô José Rosendo (*In Memoriam*), por sempre ter sonhado com esse momento.

AGRADECIMENTOS

Primeiramente a Deus que permitiu que tudo isso acontecesse, ao longo de minha vida, e não somente nesses anos de universitário, mas em que todos os momentos é o maior mestre que alguém pode conhecer e usufruir de todo o conhecimento.

Aos meus pais Nascimento e Hozana, pelo incentivo, amor, carinho e por todos os esforços que fizeram para que eu chegasse até aqui. Aos meus irmãos, Marcus, Marcelo, Ullysses, Vitória por sempre estarem ao meu lado sempre que precisei. À minha irmã de sangue Andresa que sempre me incentivou a continuar caminhando e à minha sobrinha amada que alegrou meus dias de cansaço.

Agradeço especialmente a minha namorada Ana Klyssia, que caminhou sempre comigo me dando amor, carinho, força, que, eu, apesar de às vezes fraquejar, ela se manteve firme para me apoiar.

A esta Universidade, seu corpo docente, direção e administração que oportunizaram a janela de onde hoje vislumbro um horizonte superior, eivado pela acendrada confiança no mérito e na ética presente.

Agradeço especialmente a minha orientadora Carla Ilane pela oportunidade de elaboração desse trabalho, pelos incentivos e correções.

E mais uma vez agradeço ao saudoso Pai Celestial que me deu força e perseverança para continuar o desenvolvimento desse trabalho, mesmo muitas vezes fraquejando, e principalmente por ter colocado pessoas para me ajudar a completar essa jornada.

Obrigado!

"O sucesso é ir de fracasso em fracasso sem perder entusiasmo."
(Winston Churchill)

RESUMO

Atualmente as empresas de desenvolvimento de software tem buscado constante aprimoramento na qualidade dos produtos desenvolvidos, a fim de buscar competitividade no mercado. Teste de Software tem sido utilizado no intuito de obter uma visão da qualidade dos produtos. Durante a execução dos testes dentro do ciclo de desenvolvimento de software, um *pool* de informações muito grande é gerado conforme o aumento da complexidade dos sistemas desenvolvidos. Existem diversos modelos de maturidade para melhoria dos processos de software, um destes modelos é o MPT.Br, que tem por intuito selecionar as melhores práticas de teste baseados em modelos existentes, bem como estabelecer medidas de Teste de Software. Neste contexto, o objetivo deste trabalho é o desenvolvimento de um repositório genérico de medidas de Teste de Software que seja aderente ao MPT.Br. Para construção desse repositório, foi desenvolvido um *survey* para priorização e coleta de objetivos de medição de testes de software que foram identificados em trabalhos da literatura. Esse *survey* foi aplicado com especialistas de testes de vários estados brasileiros. Após a aplicação do *survey* foi realizada a análise do perfil dos participantes e a priorização dos objetivos de medição de testes de software com o maior grau de importância. Com base nos objetivos de medição priorizados, foi executada a fase de Definição da técnica GQM para identificação e especificação de medidas de Testes de Software. Para validação do repositório, foi realizada uma análise de aderência aos níveis 1, 2 e 3 do MPT.Br e posteriormente o repositório foi validado com uma aplicação em uma empresa de desenvolvimento de software.

Palavras-chave: Medidas, Teste de Software. MPT.Br. GQM. Medidas de Teste de Software.

ABSTRACT

Currently the software development enterprises have been constantly searching improvements in the quality of the developed products, in order to achieve market competitiveness. Software testing has been used in order to obtain a vision of the product quality. During the running of tests within the software development cycle, a very large information pool is generated with increasing complexity of the systems developed. There are several maturity models for improvement of software process, and one of them is the MPT.Br model, that is meant to select the best tests practices based on existent models, and establish metrics of software testing as well. In this context, the objective of this work is to develop a generic repository of metrics and measures of Software testing that are adherent to the MPT.Br. For the construction of this repository, it was developed a survey for priorization and collection of measurement goals of software testing that were identified in the literature works. This survey was applied by software testing specialists from various Brazilian states. After the application of the survey, an analysis of the profiles of the participants and the priorization of the measurement software testing goals was conducted with a higher level of importance. Based on the prioritized measuring goals, the definition of GQM technique for information and specification of software testing measures was executed. For the repository validation, an compliance analysis to the levels 1, 2 ,3 of the MPT.Br was performed, and later the repository was validated with the application in an development software company.

Keywords: Measures. Software Testing. MPT.Br. GQM. Software Test measures.

LISTA DE TABELAS

LISTA DE ILUSTRAÇÕES

SUMÁRIO

1 INTRODUÇÃO

O Teste de Software desempenha um papel vital no ciclo de desenvolvimento de Software (KHAN e KHAN, 2014). O objetivo das atividades de testes é fornecer uma visão da qualidade do software (ISO/IEC, 2013). A execução de testes durante o ciclo de desenvolvimento do software fornece um *pool* de informações que podem ser utilizadas no processo de tomada de decisão e na melhoria da qualidade do produto (HEIKKINEN, 2015). A quantidade de informações sobre a qualidade do software cresce conforme a evolução da complexidade dos sistemas testados, e para que os testes sejam produtivos e eficientes há necessidade de uma boa gestão e boas práticas de testes. Uma forma de gerir essas informações é através de medições (HEIKKINEN, 2015).

Segundo Fenton (1994), medição é o processo pelo qual números ou símbolos são associados a atributos e entidades no mundo real, com o objetivo de descrevê-las de acordo com um conjunto de regras claramente definidas, produzindo como resultado um conjunto de medidas.

No padrão IEEE (1998) o propósito de medidas de software é fazer a avaliação ao longo do ciclo de vida do software para validar se os requisitos de qualidade de software estão sendo atendidos. Medidas são ferramentas fundamentais para uma gestão de testes eficaz, bem como para extração de informações dos resultados dos testes para tomadas de decisões. Há duas frases que são frequentemente citadas no intuito de justificar o uso de medidas como ferramentas de controle. A primeira é “Você não pode controlar o que você não pode medir” e a outra é “Quando você medir o que está falando e expressar em números, algo você sabe sobre isso, mas quando você não pode expressar em números, o seu conhecimento é de um tipo frágil e insatisfatório” (HEIKKINEN, 2015). Agarwal, Tayal e Gupta (2010) afirmam que “a gestão baseada em medidas também é um componente-chave na estratégia de Engenharia de Software para atingir seus objetivos”.

Medidas são partes de modelos de maturidade (HEIKKINEN, 2015). Diante disso é possível afirmar que nos últimos anos, devido a grande demanda e exigência de qualidade do mercado de software e sabendo que a qualidade de software é diretamente relacionada à qualidade do seu processo de desenvolvimento, organizações desenvolvedoras de software têm dado cada vez mais atenção aos seus processos de software (FUGGETTA, 2000).

No intuito de promover a melhoria na qualidade dos processos, têm sido propostos modelos que avaliam e certificam o nível de maturidade de uma organização na execução de seus processos, como o CMMI (*Capability Maturity Model Integration*) (SEI, 2010) e o modelo de maturidade brasileiro MPS.BR (Melhoria de Processo de Software Brasileiro) (SOFTEX, 2012). De acordo com o CMMI, "*a qualidade de um sistema ou produto é amplamente influenciada pela qualidade do processo utilizado*", além de fornecer dados pertinentes à formação de uma base que servirá como insumo para maximização da produtividade das pessoas e o uso da tecnologia para se tornar mais competitivo no mercado (SEI, 2010).

Assim como existem modelos de melhoria de processo de software, existem modelos de melhoria de processos de teste, como o TMMi (*Test Maturity Model Integration*) (VEENENDAAL, 2012) e o modelo de maturidade brasileiro MPT.Br (Melhoria do Processo de Teste Brasileiro) (SOFTEXRECIFE, 2011). Em linhas gerais, o MPT.Br trata a melhoria do processo de teste através práticas relativas às atividades desenvolvidas durante o ciclo de vida de teste do produto (SOFTEXRECIFE, 2011).

O modelo de maturidade MPT.Br é composto por cinco níveis de maturidade, que vão do nível 1 ao nível 5. A área de Medição e Análise de testes de software é abordada nos níveis 1, 2 e 3 do MPT.Br, nas áreas de processo Gerência de Projetos de Teste – GPT e Medição e Análise de Teste - MAT. Duas práticas de GPT utilizam medidas de testes, são elas: Estabelecer indicadores de desempenho de teste do nível 1 (GPT13) e Monitorar defeitos do nível 2 (GPT24). A prática GPT13 tem por objetivo estabelecer um conjunto de indicadores do teste de software para que a gerência de projeto seja feita com base em dados objetivos, e a prática GPT24 tem por objetivo realizar um acompanhamento sistemático dos defeitos do produto, identificando tendências e tomando ações corretivas. Medidas de testes são utilizadas em todas as práticas de MAT, são elas: Definir objetivos de medição de teste (MAT1), Estabelecer e documentar medidas (MAT2), Especificar procedimentos de medição (MAT3), Coletar, analisar e comunicar dados de medição (MAT4) e Armazenar dados de medição (MAT5). A prática MAT1 tem por objetivo estabelecer e manter objetivos de medição de teste derivados de necessidades de informação. A prática MAT2 tem por objetivo estabelecer e manter medidas que satisfazem e estão diretamente associadas aos objetivos de medição. A prática MAT3 tem por objetivo definir os procedimentos operacionais para coleta, análise, verificação, comunicação e armazenamento dos dados de medição. A prática MAT4 tem por objetivo coletar e analisar os dados de medição de acordo com os procedimentos operacionais

de medição presentes nas especificações das medidas. A MAT5 tem por objetivo gerenciar e armazenar os dados de medição, especificações de medidas e análise de resultados (SOFTEXRECIFE, 2011).

Na literatura é possível encontrar trabalhos que definem medidas. Araújo (2008) utiliza uma técnica chamada de GQM (*Goal*/*Question*/*Metric*) para construção de repositório de medidas em Organizações de software. Sato (2007) propõe um modelo de uso de medidas em métodos ágeis de desenvolvimento de software. Heikkinen (2015) identifica medidas para apoiar o gerenciamento de testes em Organizações.

O objetivo principal deste trabalho é criar um repositório genérico de medidas de teste que atenda aos resultados esperados dos processos de Gerência de Projetos de Teste e Medição e Análise de Teste relacionados a medidas de teste nos níveis 1, 2 e 3 do MPT.Br baseado no contexto de empresas de desenvolvimento de software. Os resultados específicos são: definir um *survey* para coleta de metas de empresas de desenvolvimento de software, mapear as medidas de teste de acordo com as metas coletadas, avaliar o atendimento das medidas ao processo de Gerência de Projetos de Testes em uma empresa do estado do Ceará.

Este trabalho está dividido da seguinte forma: no capítulo 2 são apresentados os conceitos chaves relacionados a este trabalho. Os conceitos de Testes de Software, MPT.Br, Medidas, *Survey* e a técnica de GQM. Após apresentar os conceitos-chave, são apresentados no capítulo 3 os procedimentos que devem ser realizados para atingir os objetivos estabelecidos anteriormente. O capítulo 4 apresenta a construção de um *survey* para a coleta de objetivos de medição de Teste de Software de Organizações de desenvolvimento de software, bem como seus graus de importância. No capítulo 5 os dados coletados pelo *survey* são analisados e os objetivos de medição são selecionados. No capítulo 6 é realizada a execução da fase de Definição da Técnica de GQM, bem como uma verificação da aderência das medidas estabelecidas ao Modelo de Processo de Teste Brasileiro – MPT.Br. No capítulo 7 é realizada uma avaliação do repositório de medidas em uma empresa de desenvolvimento de software do Estado do Ceará. Após a avaliação do repositório de medidas, no capítulo 8, são apresentadas as considerações finais do trabalho.

2 REVISÃO BIBLIOGRÁFICA

Neste capítulo serão apresentados os conceitos importantes utilizados no trabalho. Cada conceito que o trabalho aborda será detalhado e definido como o mesmo será utilizado no trabalho. Na primeira seção será apresentado o conceito de testes de software. Teste de software pode ser definido como uma validação do correto funcionamento do software. Na segunda seção será apresentado o modelo de maturidade MPT.Br. Na terceira seção serão apresentados conceitos de Medição e Análise. E por fim, na quarta seção serão apresentados trabalhos que utilizam conceitos e técnicas similares as que serão utilizadas nesse.

2.1 Teste de Software

Segundo Bertolino (2007), Teste de Software é um termo circundado por um amplo espectro de diferentes atividades, desde o teste de pequenos pedaços de código (teste de unidade) até a validação do consumidor em grandes sistemas de informação (teste de aceitação).

Rios e Moreira (2013) citam algumas definições para teste de software, como:

- verificar se o software está fazendo o que deveria fazer, de acordo com os seus requisitos, e não o que não deveria fazer (RIOS e MOREIRA, 2013);
- processo de executar um programa ou sistema com a intenção de encontrar defeitos (MYERS; BADGETT; SANDLER, 2004); e
- qualquer atividade que a partir da avaliação de um atributo ou capacidade de um programa ou sistema seja possível determinar se ele alcança ou não os resultados esperados (HETZEL, 1988).

Vicente (2010) afirma que “área de teste de software tem evoluído, preocupando-se com a aplicação de testes em sistemas cada vez mais dinâmicos e reutilizáveis com propostas para o aumento de qualidade e produtividade de software”. O teste de software tem por objetivo revelar a presença de erros ou defeitos no produto e aumentar a confiança de que o produto esteja correto (MYERS; BADGETT; SANDLER, 2004).

De acordo com Myers, Badgett e Sandler (2004), a atividade de testes (conduzida de forma incremental) consiste em quatro etapas executadas após o planejamento dos testes:

- **Projeto dos casos de teste**: nessa etapa os casos de teste são planejados a partir dos requisitos de teste;

- **Preparação dos dados do teste**: nessa etapa o ambiente de teste é construído. São definidos quais dados são utilizados para execução dos testes;
- **Execução dos casos de teste**: nessa etapa os casos de teste são utilizados para execução; e
- **Avaliação dos testes**: a avaliação dos testes é realizada por meio da comparação do resultado dos testes com o resultado esperado.

A seguir são mostradas as fases de teste de software.

2.1.1 Fases de Teste de Software

Em projetos tradicionais de desenvolvimento de software a atividade de teste ocorre basicamente por meio de duas estratégias: o sistema é testado somente quando ele está totalmente construído ou utiliza-se a estratégia de teste incremental. A estratégia de teste incremental ocorre da seguinte forma (MYERS; BADGETT; SANDLER, 2004):

- **Teste de Unidade**: estágio mais baixo da escala de testes. Testes são aplicados nos menores componentes de código criados, visando garantir que eles atendam as especificações, em termos de características e funcionalidade;
- **Teste de Integração**: estes testes são executados em uma combinação de componentes para verificar se eles funcionam juntos da forma correta. Em outras palavras, assegurar que as interfaces e que os dados que estão sendo processados conforme as especificações. Componentes podem ser pedações de código, módulos, aplicações distintas, clientes e servidores etc. Estes testes podem ser feitos de forma incremental, em que cada componente é incluído de modo sequencial até que todos os casos de teste possam ser testados. Este nível de testes pode ser realizado por algumas estratégias, são elas (RIOS e MOREIRA, 2013):
  - *Botton-up*: os testes começam agrupando os componentes do mais baixo nível para formar novos módulos ou subsistemas em níveis superiores;
  - *Top-down*: o inverso da estratégia anterior. A vantagem dessa estratégia é que os resultados podem ser apresentados aos usuários antes de a construção de níveis mais baixos serem completada;
  - Fluxo de dados: a integração dos componentes é realizada a partir do desenho de fluxo de dados;
  - Funcional: a integração é feita baseada na junção de componentes que produzam um resultado funcional significativo para os usuários; e

- *Big-Bang*: combinam todos os componentes de uma só vez.

- **Teste de Sistema**: estes testes são realizados pela equipe de testes, visando à execução do sistema dentro de um ambiente organizacional controlado para validar a exatidão e perfeição na execução de suas funções. Neste estágio de testes a operação normal do sistema deve ser simulada, de forma que todas as suas funções seja executadas com maior proximidade possível de como serão executadas em produção; e
- **Teste de Regressão**: Visa garantir que o software permaneça intacto depois de novas mudanças serem realizadas. Um conjunto de dados e scripts deve ser mantido como *baseline*[1] e a fim de verificar que mudanças introduzidas não danificarão códigos aceitos.

Este trabalho abordará testes de software no intuito de criar um repositório de medidas que colete informações conforme a execução de testes de software.

2.2 O Modelo de Maturidade MPT.Br

O modelo de Melhoria de Processo de Teste Brasileiro (MPT.Br) trata a melhoria do processo de teste através das melhores práticas relativas às atividades desenvolvidas ao longo do ciclo de teste do produto (SOFTEXRECIFE, 2011). O MPT.Br tomou como base outros modelos de referência em teste de software e modelos de referência em melhoria de processo de software, tais como:

- *Testability Support Model* (TSM) (GELPERIN, 1996);
- *Testing Maturity Model* (TMM) (BURNSTEIN; SUWANNASART; CARLSON, 1996);
- *Test Process Improvement* (TPI) (KOOMEM e POL, 1999);
- *Test Organization Maturity* (TOM) (Systeme Evolutif, 1989);
- *Testing Assessement Program* (TAP) (IE Testing Consultancy e Software Futures, 1995);
- *Testing Improvement Model* (TIM) (ERICSON; SUBOTIC; URSING, 1997);
- *Testing Maturity Model Integration* (TMMi) (VEENENDAAL, 2012);

1 É um marco de referência no desenvolvimento de software. Caracterizado pela entrega de um ou mais itens de configuração e aprovação desses itens. Essa aprovação é obtida pela revisão técnica formal.

- *Maturity Model for Automated Software Testing* (KRAUSE, 1994);
- Modelo de Melhoria de Teste (MMT) (RIOS e MOREIRA, 2013);
- Capability Maturity Model Integration (CMMi) (SEI, 2010); e
- Melhoria de Processo de Software Brasileiro (MPS.BR) (SOFTEX, 2012).

O MPT.Br é dividido em dois componentes: Modelo de referência, documento em que são apresentados a estrutura, as áreas de processo e as práticas do modelo, e o Guia de Avaliação, que contém todo o processo de avaliação e um roteiro de avaliação de uma Organização com base no MPT.Br (SOFTEXRECIFE, 2011). Para este trabalho apenas o modelo de referência será abordado.

O MPT.Br tem como principais objetivos (SOFTEXRECIFE, 2011):

- Tornar-se um modelo de referência para definição, implantação e melhoria dos processos de teste;
- Abordar a melhoria contínua dos processos de teste conforme os objetivos organizacionais e nível de maturidade almejado;
- Fornecer uma base para avaliação e consequente identificação do grau de maturidade presente nas organizações; e
- Reunir as melhores práticas e estruturá-las segundo o grau de complexidade *versus* o nível de maturidade a que a organização estará relacionada.

O modelo de referência está dividido em cinco níveis de maturidade do MPT.Br, são eles:

1. **Parcialmente gerenciado** – este nível representa o patamar mais básico em que uma Organização deve demonstrar que testes são aplicados.
2. **Gerenciado** – este nível demonstra que o processo de testes possui maior visibilidade. O escopo do projeto passa a ser controlado pelo processo de gestão de mudanças, padrões são definidos e os processos são monitorados e controlados.
3. **Definido** – neste nível o teste se torna organizacional. Processos padrões de teste são adotados, a garantia da qualidade passa a auxiliar na definição

de processos, de responsabilidades para a organização do teste e a implantação de um programa de medição. O processo de teste passa a integrar o ciclo de desenvolvimento, o teste estático e de aceitação são formalizados e procedimentos sistemáticos são aplicados no fechamento do teste.

4. **Prevenção de defeitos** – neste nível um processo de gestão de defeitos existe na Organização, em que defeitos encontrados durante o ciclo de vida de desenvolvimento são acompanhados e ações proativas são tomadas para evitar novos defeitos. Análises de riscos funcionais e não-funcionais são realizadas para minimizar estes riscos, bem como análise para determinar a eficácia do teste e a determinação com dados objetivos do nível de qualidade do produto.
5. **Automação e Otimização** – este nível tem por objetivo estabelecer um processo de melhoria contínua e automação de testes. Uma abordagem sistemática para automação e execução dos testes é utilizada, um processo sistemático para seleção e adoção de ferramentas CASE é utilizado. O processo é controlado estatisticamente e está sob melhoria continua.

Um nível de maturidade é composto por um conjunto de áreas de processo e, uma área de processo é um agrupamento de práticas relacionadas que satisfazem um determinado objetivo. Cada nível de maturidade também é associado a um conjunto de práticas genéricas que devem ser aplicadas a cada área de processo que compõe o nível de maturidade almejado. Dessa forma, para uma Organização obter uma certificação em um nível de maturidade, ela deve demonstrar, através da avaliação, que o processo de teste aplicado em seus projetos atende a todas as áreas de processo daquele nível e que todos os níveis de maturidade anteriores também são atendidos, bem como demonstrar o atendimento às práticas genéricas associadas àquele nível de maturidade (SOFTEXRECIFE, 2011).

Para alcançar níveis mais altos de maturidade requer-se melhoria constante do processo fundamentada em dados quantitativos e qualitativos, portanto medição e análise dos resultados são requisitos essenciais para melhoria de qualidade.

A organização das áreas de processo MPT.Br é mostrada na tabela a seguir.

Tabela 1- Os Níveis do MPT.Br

Nível de Maturidade	**Áreas de Processo**	**Práticas Genéricas**
Nível 1	**GPT – Gerência de Projetos de Teste** (práticas específicas GPT1 a GPT20) PET – Projeto e Execução de Teste (práticas específicas PET1 a PET4)	PG1 a PG6
Nível 2	GRT – Gerência de Requisitos de Teste (práticas específicas GRT1 a GRT5) **GPT – Gerência de Projetos de Teste** (práticas específicas GPT21 a GPT25) PET – Projeto e Execução de Teste (práticas específicas PET5 e PET6)	PG7 a PG9
Nível 3	FDT – Fechamento do Teste (práticas específicas FDT1 a FDT4) GDQ – Garantia da Qualidade (práticas específicas GDQ1 a GDQ3) **MAT – Medição e Análise do Teste** (práticas específicas MAT1 a MAT5) OGT – Organização do Teste (práticas específicas OGT1 a OGT10) TDA – Teste de Aceitação (práticas específicas TDA1 a TDA7) TES – Teste Estático (práticas específicas TES1 a TES5) TER – Treinamento (TRE1 a TRE4) GPT – Gerência de Projetos de Teste (práticas específicas GPT26 a GPT28) PET – Projeto e Execução do Teste (prática específica PET7)	
Nível 4	AQP – Avaliação da Qualidade do Produto (práticas específicas AQP1 a AQP5) GDD – Gestão de Defeitos (práticas específicas GDD1 a GDD3) TNF – Teste Não-Funcional (práticas específicas TNF1 a TNF3) OGT – Organização do Teste (práticas específicas OGT11 e OGT12)	
Nível 5	AET – Automação da Execução do Teste (práticas específicas AET1 a AET6) CEP – Controle Estatístico do Processo (práticas específicas CEP1 a CEP5) GDF – Gestão de Ferramentas (práticas específicas GDF1 a GDF6)	

Fonte: Guia de Referência do Modelo – MPT.Br (SOFTEXRECIFE, 2011)

Este trabalho abordará duas práticas de GPT e duas práticas de MAT que utilizam medidas de testes, que são elas: GPT13 – Estabelecer indicadores de desempenho de testes no

nível, GPT24 – Monitorar defeitos no nível 2, MAT1 - Definir objetivos de medição de teste e MAT2 – Estabelecer e documentar medidas.

A prática GPT13 tem por objetivo o estabelecimento de indicadores do teste de software para embasamento da gerência do projeto para tomada de decisão, bem como acompanhamento da evolução do projeto. De acordo com a ISO/IEC 15939, um indicador é uma medida que fornece uma estimativa ou avaliação de atributos específicos derivados de um modelo relativo a determinadas necessidades de informação. Diante disso, os indicadores são a base para a análise e tomada de decisão. Os indicadores permitem medir o progresso com relação a alguma finalidade, logo devem refletir os objetivos e valores do negócio de teste. A prática GPT24 tem por objetivo o monitoramento de defeitos encontrados durante a execução do Projeto de Testes. Diante disso, medidas relacionadas a defeitos devem ser utilizadas para realização de um acompanhamento sistemático dos defeitos do produto, identificando tendências e tomando ações corretivas. A prática MAT1 tem por objetivo estabelecer e manter os objetivos de medição de teste derivados das necessidades de informação. Uma Organização possui um conjunto de necessidade de informação de teste que necessitam ser endereçadas. Estas necessidades de informação devem ser derivadas em objetivos de medição de teste para que estas possam ser gerenciadas e monitoradas. A prática MAT2 tem por objetivo estabelecer e manter as medidas que satisfazem e estão diretamente associados aos objetivos de medição. Após a definição dos objetivos de medição é necessário identificar um conjunto de medidas que possibilitem a sua obtenção.

2.3 Medição e Análise

À medida que a complexidade do desenvolvimento e manutenção de software aumenta, ferramentas e metodologias de desenvolvimento passaram a coexistir, competindo entre si. Dessa forma, toda Organização desenvolvedora de software precisa compreender o seu processo, produtos e projetos de software, deve ser capaz de fazer previsões e continuamente implantar melhorias de forma sistemática e controlada. Diante disso, a medição pode ajudar a fornecer informações que servirão de base para a tomada de decisão (VON WANGENHEIM; VON WANGENHEIM; LINO, 2012).

Fenton (1994) define medição como o processo pelo qual números ou símbolos são atribuídos aos atributos de entidades do mundo real, descrevendo-os de acordo com as regras claramente definidas.

Uma **entidade** pode ser:

- um produto (qualquer artefato produzido ou modificado durante o processo de software);
- um processo (qualquer processo/atividade no processo de software);
- um recurso utilizado no processo de software.

O **atributo** é uma característica ou uma propriedade da entidade.

A **medida** é uma variável a qual é atribuído um valor como resultado de medição (ISO/IEC, 2007). O termo “medida” foi adotado neste trabalho de acordo com a norma ISO/IEC 15939 como um termo mais abrangente que o termo “métrica”.

Dentre as razões para realizar medições em processos, produtos e recursos, pode-se citar o fato de que através da análise de seus resultados, elas fornecem subsídios para (PARK et al., 1996):

- **Caracterizar** processos, produtos e recursos para melhor conhecê-los e estabelecer *baselines* para comparações;
- **Monitorar** a execução do processo e assim verificar sua aderência em relação ao seu planejamento, possibilitando tomada de ações para controle de custos, prazos e redução de riscos;
- **Avaliar** o alcance de metas de qualidade e o impacto de ações de melhorias tecnológicas e de processo;
- **Prever** e planejar, baseando-se em dados históricos, obtendo conhecimento para estimar com apoio de evidências; e
- **Melhorar**, identificando, através de informações quantitativas ineficiências e oportunidades de melhoria, propiciando melhor escolha dos métodos, técnicas e ferramentas.

Sato (2007) apresenta algumas das possíveis classificações para medidas:

- **Objetiva**: uma medida objetiva depende somente do objeto que está sendo avaliado e não do ponto de vista de quem está interpretando;
- **Subjetiva**: uma medida subjetiva depende do objeto em questão e também do ponto de vista de quem está interpretando;
- **Quantitativa**: medidas quantitativas pertencem a um intervalo de uma magnitude e geralmente são calculadas por um número; e

- **Qualitativa**: medidas qualitativas são representadas por valores representados por palavras, símbolos ou figuras ao invés de números.

De acordo com Park et al.(1996), o desafio da medição é identificar quais atributos devem ser coletados para gerar visões úteis das entidades que necessitam ser utilizadas.

2.3.1 Modelo de Especificação de Medidas

Através de um trabalho de revisão, pelo grupo de trabalho WG6 do Subcomitê de Sistemas e Software (SC7) da ISO/IEC, das normas das séries ISO/IEC 9126 (ISO/IEC, 2003) e ISO/IEC 14598 (ISO/IEC, 1997) de especificação e avaliação da qualidade de produto de software um novo modelo foi criado: o modelo *SQuaRE*, que é um acrônimo de *Software Quality Requirements and Evaluation* (SOFTEX, 2013).

O núcleo principal do SQuaRE é composto por quatro divisões de normas e uma sequência prevista para extensão do modelo (ISO/IEC, 2011a):

- ISO/IEC 2500n – Divisão Gestão da Qualidade;
- ISO/IEC 2501n – Divisão Modelo de Qualidade;
- ISO/IEC 2502n – Divisão Medição de Qualidade;
- ISO/IEC 2503n – Divisão Requisitos de Qualidade;
- ISO/IEC 2504n – Divisão Avaliação da Qualidade; e
- ISO/IEC 2505n - ISO/IEC 25099 – Extensão do SQuaRE.

Essas divisões são compostas de normas, harmonicamente integradas, que detalham os tópicos relacionados à especificação e avaliação da qualidade de produtos de software. Do modelo SQuaRE, destacamos a divisão Medição de Qualidade que apresenta elementos de medida de qualidade, medida de qualidade interna externa e de qualidade em uso e um modelo de referência. Além disso, fornece orientações aos usuários para selecionar ou desenvolver e aplicar medidas de qualidade de produto de software.

A norma ISO/IEC 25023 (2011a) pretende definir medidas de qualidade de sistemas e de produto de software, segundo o modelo de qualidade da ISO/IEC 25010 (ISO/IEC, 2011b). Um modelo de especificação de medidas é definido e utilizado nos campos de aplicação de qualidade de abordadas pela norma. Os campos são:

- Medidas de adequação funcional
- Medidas de eficiência de desempenho
- Medidas de compatibilidade

- Medidas de usabilidade
- Medidas de confiabilidade
- Medidas de segurança
- Medidas de manutenção
- Medidas de portabilidade

O modelo de especificação de medidas segue o modelo de qualidade da ISO/IEC 25010 (ISO/IEC, 2011b) e deve conter as seguintes informações:

- Nome da medida de qualidade: o nome da medida deve ser único, caso haja necessidade um número serial pode ser definido;
- Objetivo da medida: que questão a medida deve responder;
- QMEs (*Quality Measures Element*): os elementos utilizados no cálculo da medida;
- Fórmula de cálculo: é a fórmula utilizada no cálculo da medida; e
- Foco da medida: determina a quem a informação é necessária, se é interna a Organização ou externa.

Tabela 2 - Modelo de Especificação de Medidas

Nome da Medida	**Descrição/Objetivo**	**QMEs**	**Fórmula de Cálculo**	**Foco da medida**
Running Tested Features (RTF)	Monitorar funcionalidades entregues com testes	RTF são *features* testadas que foram entregues em um projeto	∑ RTF	Interno/Externo

Fonte: Adaptado de ISO/IEC 25023 (2011)

A Tabela 2 mostra o modelo de especificação de medidas da ISO/IEC 25023, que será utilizado nesse trabalho.

Existem diferentes técnicas que podem ser adotadas para auxiliar esta etapa, assim como todo o processo de medição. Podemos destacar a técnica GQM (*Goal-Question-Metric*) proposta por Basili et al. (1994) apresentada a seguir.

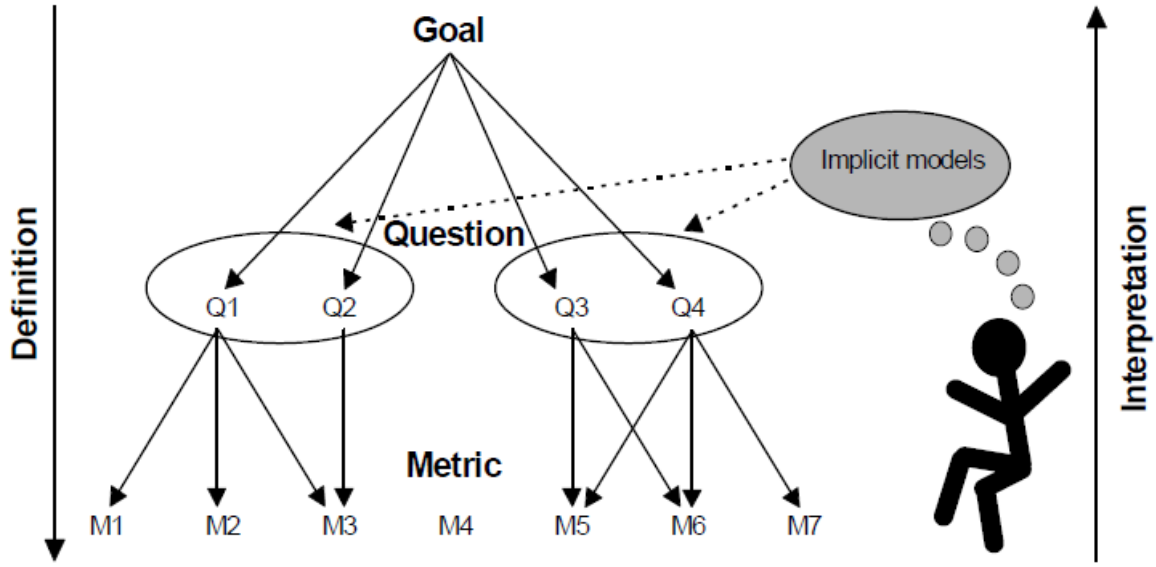
2.3.2 GQM (*GOAL/QUESTION/METRIC*)

O GQM é uma abordagem sistemática para integrar objetivos aos modelos dos processos de software, produtos e perspectivas de qualidade de interesse com base nas necessidades de informação do projeto e da Organização (Basili et al, 1994). O paradigma

GQM é embasado no requisito de que a medição deve ser orientada a objetivos. O GQM utiliza uma abordagem *top-down* que pode ser resumida nos seguintes passos:

- Objetivos (*GOAL*) – definir objetivos para a organização;
- Questões (*QUESTION*) – gerar um conjunto de questões que definam os objetivos mediante aspectos quantitativos;
- Medidas (*METRIC*) – especificar um conjunto de medidas que precisam ser coletadas para responder às questões geradas.

Figura 1 – A técnica GQM



Fonte: The GQM Paradigm (Basili et al., 1994)

A Figura 1 mostra uma visão simplificada do fluxo de execução da técnica GQM. Inicialmente são definidos os objetivos da organização, em seguida são definidas questões que respondam aos objetivos de forma quantitativa e, por fim, especificar um conjunto de medidas para coleta de dados.

O paradigma GQM é amplamente utilizado e citado na literatura, podendo ser empregado isoladamente ou no contexto de uma abordagem de melhoria de processo. Dentre os resultados de sua utilização, estão: identificação de medidas relevantes; definição de um contexto para análise e interpretação dos dados obtidos com a rastreabilidade dos objetivos às medidas; e documentação do refinamento dos objetivos em medidas, que possibilita uma avaliação da validade das medidas para a organização e para os projetos (ARAÚJO, 2008).

O paradigma GQM está definido em quatro fases (Basili et al, 1994):

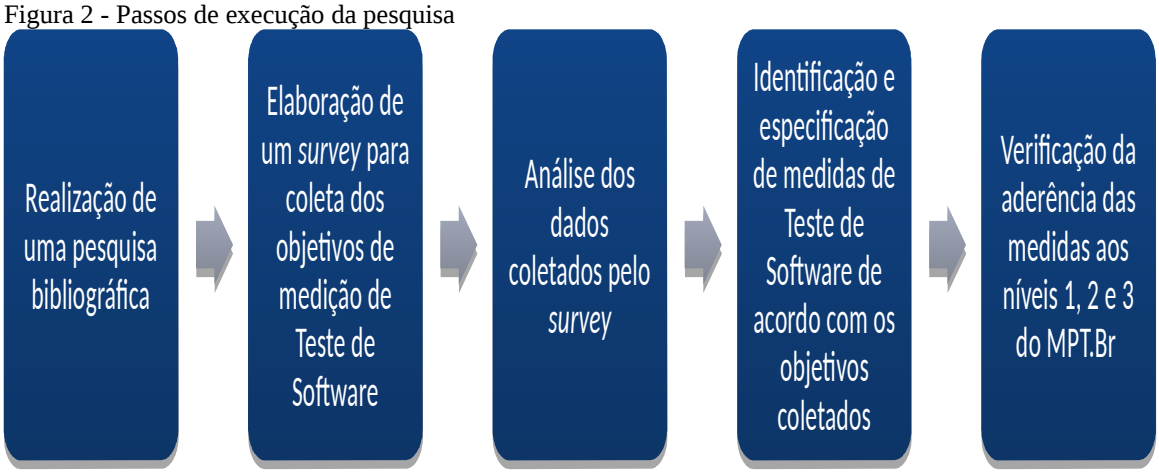
- Planejamento: os principais objetivos dessa fase são coletar informações fundamentais para introdução do programa, bem como motivar os membros da organização para o programa de medição. Nessa fase um plano de projeto é definido para execução da técnica.
- Definição: nessa fase, o programa de medição é definido formalmente. Nessa fase são produzidos três artefatos:
    - Plano de GQM: nesse artefato os objetivos do programa de medição são definidos, bem como contém todas as informações para execução do mesmo.
    - Plano de Medição: nesse artefato são definidas as questões e hipóteses do programa de medição.
    - Plano de Análise: nesse artefato as medidas são definidas.
- Coleta de Dados: nessa fase acontece a coleta de dados para validação da aplicação da técnica de GQM.
- Interpretação: nessa fase os objetivos, respostas e medidas são validados.

Para este trabalho será utilizada a fase de Definição do GQM. As fases de Planejamento, Coleta e Interpretação não serão necessárias, pois o objetivo do trabalho é definir um repositório genérico baseado em objetivos de Organizações de desenvolvimento, e para que fossem utilizadas essas fases seria necessária a execução em todas as Organizações. Dessa forma o objetivo do trabalho seria prejudicado, pois ao final da execução seria definido um repositório para cada Organização, e não um repositório genérico.

3 PROCEDIMENTOS METODOLÓGICOS

Com a utilização de Teste de Software na busca por qualidade nos produtos desenvolvidos e uma maior competitividade no mercado, uma grande quantidade de dados é gerada. A quantidade de informações sobre a qualidade cresce conforme a evolução da complexidade dos sistemas testados, e por consequência há necessidade de uma boa gestão e boas práticas de testes. No padrão IEEE (1998) o propósito das medidas de software é fazer a avaliação ao longo do ciclo de vida do software para validar se os requisitos de qualidade de software estão sendo atendidos. Com base na necessidade de medição de atividades de Teste de Software, este trabalho propõe a criação de um repositório genérico de medidas de Teste de Software que sejam aderentes aos níveis 1, 2 e 3 do MPT.Br. Os passos para a execução desse trabalho estão descritos de acordo com a Figura 2.

Figura 2 - Passos de execução da pesquisa



Fonte: Elaborada pelo autor.

Os passos apresentados envolvem a realização de uma revisão bibliográfica, elaboração de um *survey* para coleta dos objetivos de medição de Teste de Software, análise dos dados coletados pelo *survey*, identificação e especificação de medidas de Teste de Software de acordo com os objetivos coletados e verificação da aderência das medidas aos níveis 1, 2 e 3 do MPT.Br.

3.1 Pesquisa bibliográfica

Nessa etapa, uma pesquisa foi realizada na literatura com o intuito de selecionar objetivos de medição de teste de software e classifica-los de acordo com sua relevância. Essa pesquisa foi realizada em repositórios nacionais e internacionais. Para sua realização foram definidas strings de buscas, bem como os repositórios em que foi realizada a busca.

3.2 Elaboração de um *survey* para coleta dos objetivos de medição de Teste de Software

Nessa etapa, um *survey* foi desenvolvido para coleta da relevância dos objetivos de medições de teste de software. Os objetivos foram obtidos a partir de uma pesquisa bibliográfica. Para classificar os objetivos foi utilizada uma escala de *Likert*. Na escala de *Likert*, as respostas para cada item variam segundo o grau de intensidade (ALEXANDRE et al., 2003). Para este trabalho a escala foi definida com os seguintes graus de intensidade: sem importância, pouco importante, imparcial, muito importante e extremamente importante.

O *survey* foi construído utilizando a ferramenta *LimeSurvey*, que é uma ferramenta livre de criação, publicação e coleta de dados de questionários (SCHMITZ, 2003).

3.3 Análise dos dados coletados pelo *survey*

Nessa etapa, os dados coletados pelo *survey* foram analisados para fins de seleção dos objetivos de medição para execução da fase de Definição da técnica GQM. Para seleção das respostas a serem analisadas foram levados em consideração os seguintes critérios: perfis que afirmaram que não tinham experiência com Teste de Software, respostas que não condizem com o objetivo da pesquisa, respostas idênticas para todos os itens da pesquisa e respostas incompletas.

Para seleção dos objetivos de medição foi estabelecido um critério de seleção que consiste em selecionar os objetivos a partir do somatório de percentuais conforme apresentado na seção 5.2.1.

3.4 Identificação e especificação de medidas de testes de software de acordo com os objetivos coletados utilizando GQM

Nessa etapa, foi realizado o mapeamento das medidas de acordo com os objetivos coletados utilizando a técnica GQM. Para este trabalho apenas a fase de Definição da técnica GQM foi utilizada, visto que somente essa fase foi suficiente para especificação das medidas. Após a seleção dos objetivos são levantados questionamentos sobre a necessidade da informação e por fim, as medidas são estabelecidas. A aplicação da fase de Definição do GQM é apresentada na seção 6.

3.5 Verificação da aderência das medidas aos níveis 1, 2 e 3 do MPT.Br

Nessa etapa, o repositório foi avaliado a fim de verificar sua aderência aos níveis 1, 2 e 3 do MPT.Br. As práticas abordadas durante a avaliação são: GPT13 – Estabelecer

indicadores de desempenho de testes no nível, GPT24 – Monitorar defeitos no nível 2, MAT1 - Definir objetivos de medição de teste e MAT2 – Estabelecer e documentar medidas.

4 ELABORAÇÃO DE UM *SURVEY* PARA ANÁLISE DE OBJETIVOS DE MEDIÇÃO DE TESTES DE SOFTWARE

Nesta seção o processo de construção do *survey* para coleta de objetivos é apresentado, desde a realização de uma pesquisa bibliográfica até a sua aplicação.

Partindo do princípio de que a presente proposta tem como resultado final um repositório de medidas genérico e que utiliza a técnica de especificação de medidas GQM, que tem como insumo para especificação de objetivos de medição, uma pesquisa bibliográfica foi realizada no intuito de coletar medidas de Teste de Software e, consequentemente, seus objetivos.

A pesquisa foi realizada em repositórios nacionais e internacionais de trabalhos publicados com temas relacionados a teste de software, medidas de teste de software, medidas de software e medidas de desenvolvimento de software ágil. São eles:

- *IEEE Xplore Digital Library*
- *ScienceDirect*
- *Scopus*
- Universidade Federal de Pernambuco
- Universidade de Fortaleza
- Google Acadêmico

Foram definidas *strings* de busca, tais como:

- Métricas de teste de software
- Teste de software
- Métricas
- Objetivos de medição
- *Software testing*
- *Software testing measures*
- *Measures*
- *Measure goals*

Foram selecionados trabalhos que tivessem objetivos de medição, bem como medidas de teste de software. Após a seleção dos trabalhos foi realizado um cruzamento entre as medidas obtidas nos trabalhos para remoção de inconsistências, tais como: medidas idênticas e medidas incompletas (sem descrição, fórmulas de cálculo). Ao final dessa etapa foram selecionados 9 trabalhos, são eles:

1. Uso eficaz de métricas em métodos ágeis de desenvolvimento de software (SATO, 2007). Este trabalho investiga o uso de medidas de

acompanhamento de projetos utilizando Métodos Ágeis de desenvolvimento de software.

2. Uso de métricas no Teste de Software (TRODO, 2009). Este trabalho apresenta fundamentos de medidas de Teste de Software e realiza um diagnóstico da situação das empresas de TI do Rio Grande do Sul.
3. *Agile requirements prioritization in large-scale outsourced system projects: An empirical Study* (DANEVA et al., 2012). Este trabalho propõe a descoberta dos conceitos que os profissionais de uma grande organização usam no processo de priorização de software e suas boas práticas.
4. *Appropriate Agile Measurement: Using metrics and diagnostics to deliver business value*. (HARTMANN e DYMOND, 2006). Este trabalho analisa medidas ágeis adequadas ao desenvolvimento de software ágil e propõe o uso de ferramentas para equipes ou Organizações.
5. *A replicated survey of software testing practices in the Canadian province of Alberta: Whats has changed from 2004 to 2009?* (GAROUSI e FARMA, 2010). Este trabalho foi realizado através da aplicação de um *survey* a fim de descobrir informações sobre as práticas de Teste de Software em Alberta – Canadá.
6. *Cost effetive software Test Metrics* (LAZIC e MASTORAKIS, 2008). Este trabalho propõe uma discussão sobre medidas de Teste de Software e sua capacidade de demonstrar evidências objetivas para melhoria de processos em uma Organização de desenvolvimento de software.
7. Definição e gerenciamento de métricas de teste no contexto de métodos ágeis (VICENTE, 2010). Este trabalho caracteriza a atividade de Teste de Software aplicada dentro de métodos de desenvolvimento ágil no intuito de eliminar aspectos de teste não produtivos, identificar boas práticas e criar formas de acompanhamento e melhorar continuamente a condução da atividade de teste.
8. *Toward a software testing and reliability early warning metric suite* (NAGAPPAN, 2004). Este trabalho propõe um conjunto de métricas de Teste de Software e Alertas precoces de confiabilidade para linguagens orientadas a objetos.
9. *Enhancing quality in Scrum Software Projects* (MUKKER, MISHRA e SINGH, 2014). Este trabalho utiliza diferentes ferramentas de monitoramento e representação para otimizar o desempenho no processo

de desenvolvimento de software. Medidas de qualidade são utilizadas para avaliar o desempenho do desenvolvimento.

Foi verificado que algumas medidas não tinham o seu objetivo de medição ou justificativa da informação definido. A partir daí foi necessário um trabalho de identificação dos objetivos das medidas. Durante essa etapa foram definidos objetivos de medição baseados na experiência do autor do trabalho, tais como:

- Monitorar os testes de aceitação
- Monitorar os testes de unidade

A Tabela 3 contém todos os objetivos identificados. Após a identificação de objetivos de medidas de software, deu-se início a criação do *survey* utilizando a ferramenta *LimeSurvey*. O *survey* foi definido a partir de duas perguntas base, de acordo com a estrutura fornecida pela ferramenta: Perfil dos participantes e objetivos de medição de Teste de Software e seus graus de importância.

Tabela 3 - Objetivos de medição de Teste de Software

ID	Objetivos	Descrição	Referência
SQ01	Acompanhar a eficácia dos testes	Esse objetivo tem por intuito garantir que os testes executados foram corretos.	(TRODO, 2009)
SQ02	Acompanhar a eficiência dos testes	Esse objetivo tem por intuito garantir que os testes estão sendo executados corretamente.	(TRODO, 2009) (LAZIC e MASTORAKIS, 2008).
SQ03	Acompanhar a resolução dos defeitos encontrados	Esse objetivo tem por intuito garantir uma visão geral da resolução dos defeitos encontrados.	(TRODO, 2009) (HARTMANN e DYMOND, 2006).
SQ04	Acompanhar o percentual de defeitos por severidade	Esse objetivo tem por intuito acompanhar o grau de severidade dos defeitos encontrados e permitir um mapeamento das funcionalidades críticas do produto.	(TRODO, 2009)
SQ05	Acompanhar o percentual de retrabalho	Esse objetivo tem por intuito fornecer o percentual de retrabalho do desenvolvimento.	(LAZIC e MASTORAKIS, 2008).
SQ06	Avaliar a conformidade do produto com os requisitos de teste	Esse objetivo tem por intuito garantir a	(VICENTE, 2010) (NAGAPPAN, 2004).

ID	Objetivos	Descrição	Referência
		conformidade do produto com os requisitos do sistema	
SQ07	Avaliar a qualidade do teste	Esse objetivo tem por intuito avaliar a qualidade dos testes	(MUKKER, MISHRA e SINGH, 2014)
SQ08	Avaliar o impacto do defeito	Esse objetivo tem por intuito avaliar o impacto dos defeitos encontrados	(MUKKER, MISHRA e SINGH, 2014)
SQ09	Monitorar a cobertura de testes	Esse objetivo tem por intuito permitir um monitoramento da cobertura dos testes	(HARTMANN e DYMOND, 2006).
SQ10	Monitorar a criação de códigos de teste	Esse objetivo tem por intuito monitorar a criação de códigos de teste	(TRODO, 2009) (SATO, 2007)
SQ11	Monitorar a estabilidade do escopo dos testes	Esse objetivo tem por intuito fornecer uma visão da estabilidade do escopo de testes.	(TRODO, 2009)
SQ12	Monitorar a execução dos testes	Esse objetivo tem por intuito fornecer informações sobre as execuções dos testes	(TRODO, 2009) (VICENTE, 2010) (LAZIC e MASTORAKIS, 2008). (GAROUSI e FARMA, 2010).
SQ13	Monitorar a quantidade de *builds* por iteração	Esse objetivo permite monitorar a quantidade de builds gerados para execução de testes.	(HARTMANN e DYMOND, 2006).
SQ14	Monitorar as funcionalidades críticas do produto	Esse objetivo permite selecionar as funcionalidades que devem ter maior prioridade de execução de testes.	(TRODO, 2009)
SQ15	Monitorar funcionalidades entregues com testes	Esse objetivo permite visualizar as funcionalidades que foram entregues com testes.	(SATO, 2007)
SQ16	Monitorar o planejamento dos testes	Esse objetivo permite visualizar informações sobre a assertividade do planejamento dos testes	(DANEVA et al., 2012)
SQ1	Monitorar os custos dos testes	Esse objetivo permite	(TRODO, 2009)

ID	Objetivos	Descrição	Referência
7		visualizar os custos de testes durante o desenvolvimento de software	
SQ18	Monitorar os defeitos do produto em produção	Esse objetivo permite visualizar a estabilidade do produto em produção.	(TRODO, 2009)
SQ19	Monitorar os testes de aceitação	Esse objetivo tem por intuito monitorar a assertividade do produto em relação aos seus requisitos de aceitação.	Elaborado pelo autor
SQ20	Monitorar os testes de unidade	Esse objetivo tem por intuito monitorar a execução de testes unitários.	Elaborado pelo autor

Fonte: Elaborada pelo autor.

A seção Perfil dos Participantes é composta por 9 questões. Essas questões abordam desde a identificação dos participantes até o estado (UF) em que atua na área de Teste de Software. A seção Objetivos de medição de Teste de Software e seus graus de importância é composto pelos objetivos selecionados durante a pesquisa bibliográfica e os graus de importância são definidos por uma escala de *Likert*, com os seguintes graus: Sem importância, Pouco importante, Imparcial, Muito importante e Extremamente importante. O *survey* permitia ao participante que informasse objetivos específicos da sua Organização que não estavam listados, bem como informar seu grau de importância. O *survey* está disponível no Apêndice B.

Durante a construção do *survey* uma etapa importante é o pré-teste (BABBIE, 1999). Quanto ao pré-teste, Babbie (1999) destaca que: a) O objetivo é melhorar o instrumento de pesquisa; b) O instrumento deve ser pré-testado da mesma forma como será aplicado na pesquisa; e c) A seleção de quem irá participar do pré-teste é flexível, entretanto recomenda-se que as pessoas sejam razoavelmente adequadas para as perguntas.

Após a elaboração do *survey* um pré-teste foi realizado resultando em alterações de questões e modificações em questões já existentes.

5 ANÁLISE DOS DADOS COLETADOS

Nesta seção, são expostos os resultados obtidos por meio das respostas dos participantes ao questionário submetido, disponível no apêndice B, e também são feitas algumas análises e considerações em relação aos dados obtidos.

Para melhor entendimento dos resultados foram utilizadas seções conforme foi dividido o questionário, iniciando na seção Perfil demográfico dos participantes, seguido dos Objetivos de Medição de Teste de Software e sua importância.

A pesquisa foi realizada do dia 01/12/2015 até o dia 15/12/2015. Disponibilizado através de e-mails e grupos em redes sociais, tais como:

- Grupo de Teste de Software – CE
- Teste de Software Ceará
- Teste e Qualidade de Software
- Qualidade Manaus
- Teste de Software de Belo Horizonte
- Grupo de Teste de Software – BA
- Grupo de Teste de Software – SP
- Grupo de Teste de Software - RS

Em todo o processo de aplicação do *survey* foram obtidas 56 respostas que foram analisadas de acordo com os critérios listados abaixo.

- Perfis que afirmaram que não tinham experiência com Teste de Software;
- Respostas que não condizem com o objetivo da pesquisa;
- Respostas idênticas para todos os itens da pesquisa; e
- Respostas incompletas.

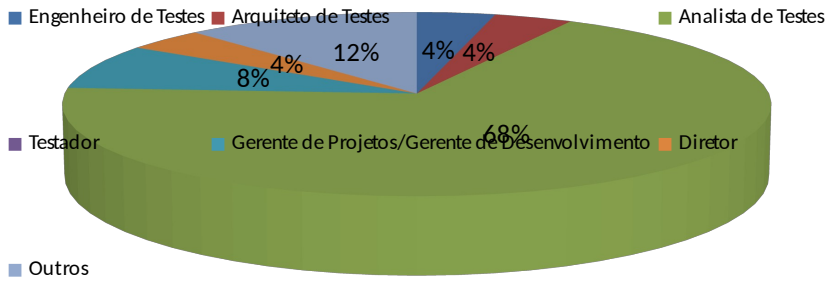
De um total de 56 respostas apenas 32 estavam completas. De 32 respostas completas 2 participantes afirmaram não ter experiência em Teste de Software, 3 participantes informaram as mesmas respostas para todos os itens e 2 participantes responderam com dados inconsistentes. Ao final do processo, de um total de 56 respostas apenas 25 respostas foram consideradas para esse texto, de acordo com os critérios acima.

5.1 Perfil demográfico dos participantes

Nesta seção poderemos verificar de forma clara os resultados da pesquisa relacionados aos perfis dos participantes, por meio de gráficos e comentários ao longo da seção. As informações sobre o perfil do participante é fundamental para avaliar a maturidade das respostas coletadas. Nessa análise, uma resposta corresponde a 4% do total.

A Figura 3 demonstra as respostas referentes às funções dos participantes na Organização em que atuam. Cerca de 68% dos participantes exercem a função de Analista de Testes que está ligada diretamente a especificação e execução de testes, 12% exercem outras funções, 8% exercem a função de Gerente de Projetos/Gerente de Desenvolvimento que está relacionada a coleta de dados a partir de medidas, as funções de Diretor, Arquiteto de Teste e Engenheiro de Testes obtiveram percentual de 4% cada. A função de testador não obteve percentual. Diante disso, é provável afirmar a seriedade das respostas, pois o Analista de Testes detém os dados de execução dos testes.

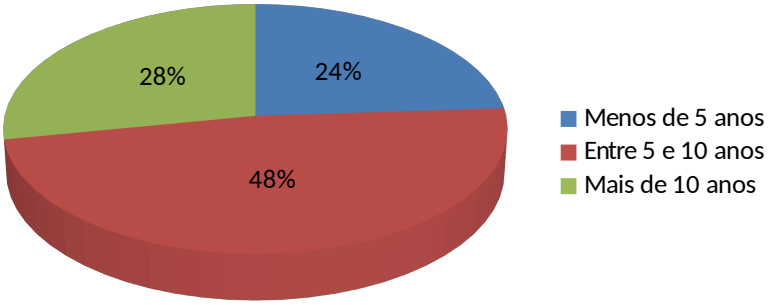
Figura 3 - Qual o seu papel atualmente na Organização?



Fonte: Elaborada pelo autor.

Pode-se perceber através da Figura 4, que faz referência ao tempo de experiência em desenvolvimento de software, que 48% dos participantes possuem entre 5 e 10 anos de experiência, 28% dos participantes possuem mais de 10 anos de experiência e apenas 24% possuem experiência menor que 5 anos. Pode-se constatar, a partir da experiência dos participantes que as respostas possuem um alto nível, bem como a amplitude de visão do assunto em suas devidas realidades.

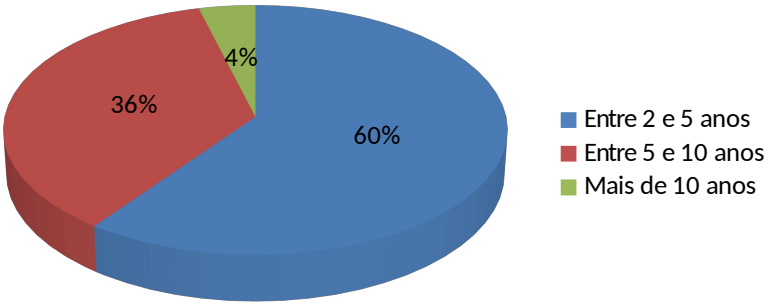
Figura 4 - Quanto tempo (em anos) você trabalha com Tecnologia da Informação e desenvolvimento de Software?



Fonte: Elaborada pelo autor.

Na Figura 5 é possível perceber o tempo de experiência dos participantes especificamente em Teste de Software, cerca de 60% dos participantes possuem experiência de 2 até 5 anos, 36% possuem experiência entre 5 e 10 anos e 4% possuem mais de 10 anos de experiência. Foram consideradas respostas de participantes de tinham experiência superior a 1 ano.

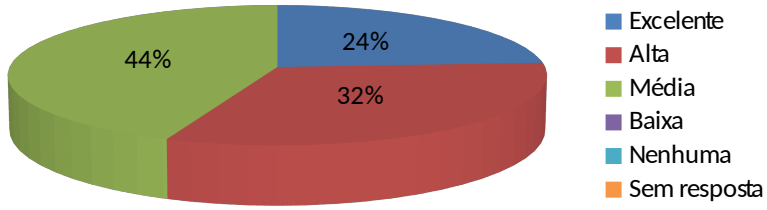
Figura 5 - Quanto tempo (em anos) você trabalha especificamente com Teste de Software?



Fonte: Elaborada pelo autor.

A Figura 6 diz respeito ao nível de experiência dos participantes em relação à área de Teste de Software. 24% dos participantes afirmaram possuir um grau excelente de experiência, 44% dos participantes afirmaram possuir um grau médio de experiência e 32% afirmaram possuir um grau baixo de experiência.

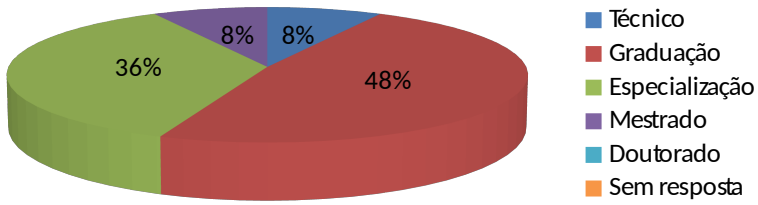
Figura 6 - Como você considera ser sua experiência em Teste de Software?



Fonte: Elaborada pelo autor.

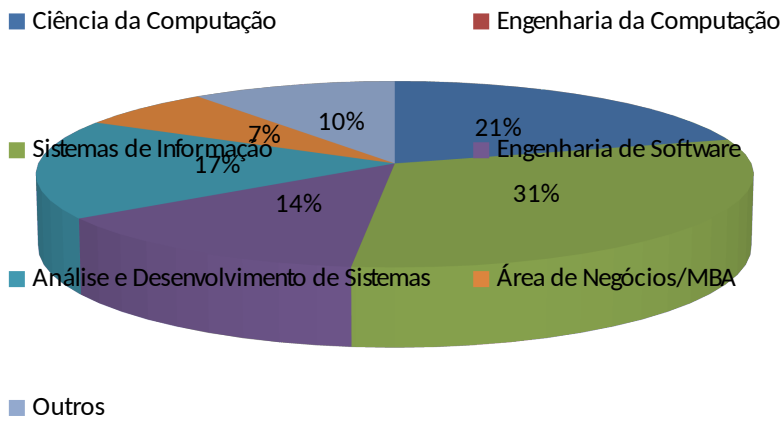
Pela Figura 7, podemos constatar que a maioria dos participantes possui nível de graduação, o que se torna um dos fatores de confiança, no sentido de fidelidade com as respostas, reafirmado pelo alto índice de participantes com formação na área de TI exposto na Figura 8.

Figura 7 - Qual sua formação acadêmica?



Fonte: Elaborada pelo autor.

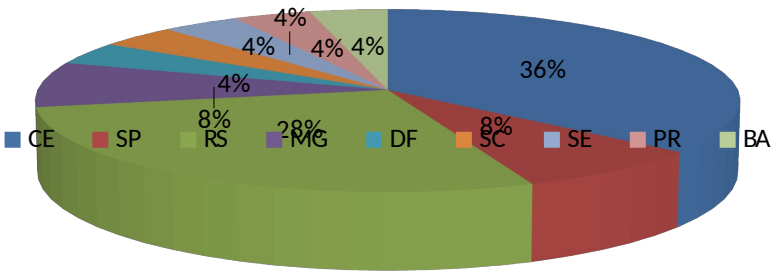
Figura 8 - Qual a(s) sua(s) área(s) de formação acadêmica?



Fonte: Elaborada pelo autor.

Na Figura 9 é possível visualizar o alcance da aplicação do questionário, concentrando-se nas regiões Nordeste, Sudeste e Sul. 36% dos participantes eram do Ceará, 28% do Rio Grande do Sul, 8% de São Paulo, 8% de Minas Gerais e os demais, Distrito Federal, Paraná, Santa Catarina, Sergipe e Bahia com 4% cada.

Figura 9 - Em que estado (UF) você executa suas atividades de desenvolvimento e Teste de Software?



Fonte: Elaborada pelo autor.

5.2 Objetivos de Medição de Teste de Software e sua importância

Nesta seção, são priorizados os objetivos de medição de Teste de Software, bem como seus graus de importância. O *survey* utilizado permitia aos respondentes informar objetivos de medição de Teste de Software, bem como classificar seu grau de importância para a Organização. Para esse item apenas um participante informou um objetivo, porém de

forma inconsistente, sem informar seu grau de importância, diante disso a resposta foi desconsiderada.

Para todos os objetivos submetidos à pesquisa foi utilizada a escala de *Likert* com os graus: Sem importância, Pouco importante, Imparcial, Muito importante e Extremamente importante. Para que um objetivo fosse considerado válido foram definidos os seguintes critérios:

- O somatório dos percentuais dos graus Muito importante e Extremamente importante maiores que o somatório dos percentuais dos graus Sem importância e Pouco importante; e
- O grau de importância Imparcial representa um elemento neutro, ou seja, que não influencia negativamente e nem positivamente.

De acordo com os critérios acima mencionados e com os graus de importância dos objetivos de medição, na Figura 10, é possível selecionar quais objetivos irão ser utilizados nessa pesquisa.

Como exemplo de seleção, podem-se citar as questões SQ01, SQ04, SQ10 e SQ15.

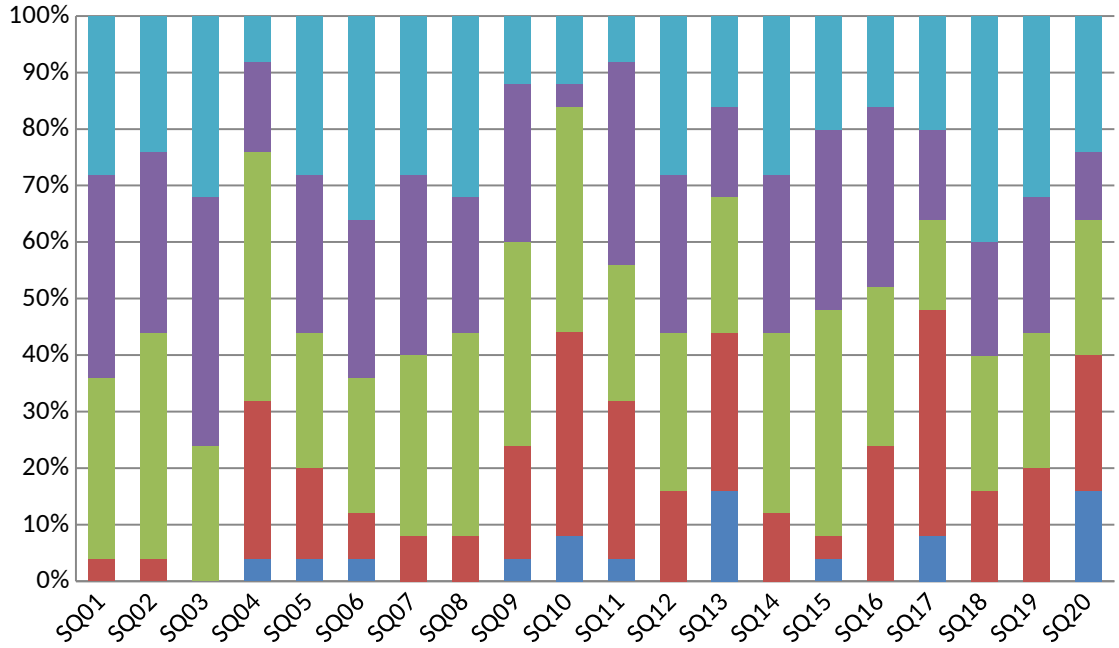
- A SQ01 atingiu somatório dos graus Sem importância e Pouco importante a 4% e o somatório dos graus Muito importante e Extremamente importante é igual a 64%, então se tem que o somatório dos graus Muito importante e Extremamente importante é maior que o somatório dos graus Sem importância e Pouco importante. Diante disso a SQ01 foi selecionada para a pesquisa;
- A SQ04 atingiu somatório dos graus Sem importância e Pouco importante a 32% e o somatório dos graus Muito importante e Extremamente importante é igual a 24%, então se tem que o somatório dos graus Muito importante e Extremamente importante é menor que o somatório dos graus Sem importância e Pouco importante. Diante disso a SQ04 não foi selecionada para a pesquisa;
- A SQ10 atingiu somatório dos graus Sem importância e Pouco importante a 44% e o somatório dos graus Muito importante e Extremamente importante é igual a 16%, então se tem que o somatório dos graus Muito importante e Extremamente importante é menor que o somatório dos graus Sem importância e Pouco importante. Diante disso a SQ10 não foi selecionada para a pesquisa; e

- A SQ15 atingiu somatório dos graus Sem importância e Pouco importante a 8% e o somatório dos graus Muito importante e Extremamente importante é igual a 52%, então se tem que o somatório dos graus Muito importante e Extremamente importante é maior que o somatório dos graus Sem importância e Pouco importante. Diante disso a SQ15 foi selecionada para a pesquisa.

Seguindo o critério acima, foram selecionados os seguintes objetivos de medição para execução da fase de Definição da técnica GQM:

1. SQ01 - Acompanhar a eficácia dos testes
2. SQ02 - Acompanhar a eficiência dos testes
3. SQ03 - Acompanhar a resolução dos defeitos encontrados
4. SQ05 - Acompanhar o percentual de retrabalho
5. SQ06 - Acompanhar a conformidade do produto com os requisitos de teste
6. SQ07 - Avaliar a qualidade do teste
7. SQ08 - Avaliar o impacto do defeito
8. SQ09 - Monitorar a cobertura de testes
9. SQ11 - Monitorar a estabilidade do escopo de testes
10. SQ12 - Monitorar a execução dos testes
11. SQ14 - Monitorar funcionalidades críticas do produto
12. SQ15 - Monitorar as funcionalidades entregues com testes
13. SQ16 - Monitorar o planejamento dos testes
14. SQ18 - Monitorar os defeitos do produto em produção
15. SQ19 - Monitorar os testes de aceitação

Figura 10 - Objetivos de medição de Teste de Software



Fonte: Elaborada pelo autor.

.

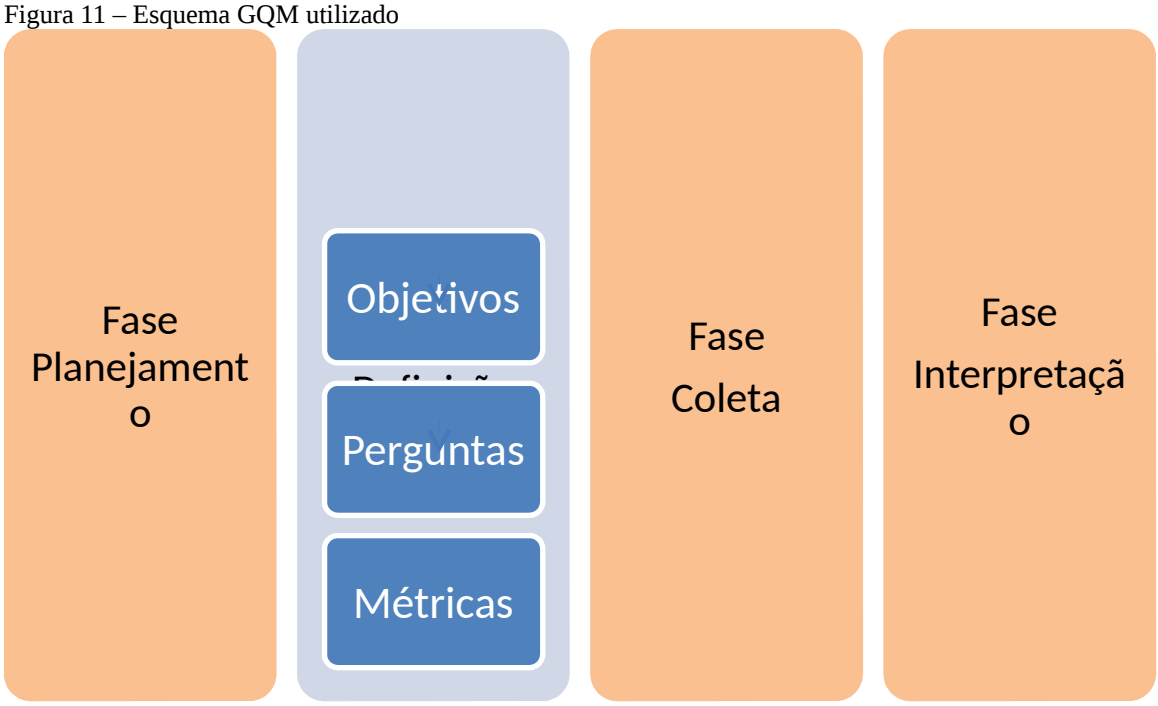
6 REPOSITÓRIO DE MEDIDAS

Nesta seção o repositório de medidas é estabelecido. A seção 6.1 aborda a execução da fase de Definição da técnica de GQM e a seção 6.2 aborda a verificação de aderência do repositório estabelecido aos níveis 1, 2 e 3 do MPT.Br.

6.1 Execução da fase de Definição do GQM

Nesta seção será apresentado como foi realizada a especificação das medidas de Teste de Software. Conforme apresentado na seção 2.3.2 apenas a fase de Definição do GQM foi executada, pois é possível definir as medidas de Teste de Software ao final de sua execução. Basicamente essa fase é executada em três passos: definição de objetivos, definição de perguntas a fim de atender aos objetivos e, por fim, definir as medidas.

Figura 11 – Esquema GQM utilizado



Fonte: Adaptado de Solingem Apud Covatti (2007, p.55).

Ao iniciar o processo, primeiramente foram identificados os objetivos de medição de Teste de Software através da aplicação de um *survey* conforme mostrado na seção 5. Durante a realização da pesquisa bibliográfica foi possível coletar medidas de teste de software. Diante disso, na etapa de execução do GQM foi necessário criar o relacionamento dos objetivos com as medidas, estabelecendo perguntas.

Como exemplo de um objetivo pode-se usar: “**Acompanhar a eficácia dos testes**”. Ao efetuar a pergunta para o objetivo anteriormente descrito, pode-se utilizar: “**Quantos defeitos foram encontrados no produto?**”, “**Qual a criticidade dos defeitos encontrados no produto?**”.

Para responder as perguntas elaboradas e conseguir atingir o objetivo proposto, escolhem-se duas medidas:

- Quantidade de falhas encontradas no produto. É obtida a partir do somatório de defeitos encontrados no produto após sua liberação. Sua fórmula de cálculo é **∑D**, **D** representa os defeitos encontrados em produção.
- Densidade dos defeitos residuais. É uma estimativa do número de defeitos não resolvidos que foram para produção. Essa medida permite alcançar um nível de defeitos que seja aceitável para o cliente.

As especificações das medidas estabelecidas estão disponíveis no Apêndice C.

Tabela 4 - Medidas propostas

OBJETIVO	QUESTÕES	MEDIDAS
Acompanhar a eficácia dos testes	Quantos defeitos foram encontrados no produto?	Quantidade de falhas encontradas no produto
	Qual a criticidade dos defeitos encontrados no produto?	Densidade dos defeitos residuais
OBJETIVO	**QUESTÕES**	**MEDIDAS**
Acompanhar a eficiência dos testes	Qual a taxa de ocorrências válidas encontradas na execução dos testes?	Taxa de ocorrências válidas
	Qual a taxa de problemas encontrados no reteste?	Taxa de problemas encontrados na correção dos defeitos
	Qual o percentual de eficiência dos testes?	Percentual de eficiência de testes
OBJETIVO	**QUESTÕES**	**MEDIDAS**
Acompanhar a resolução dos defeitos encontrados	Qual o status dos defeitos encontrados?	Status das ocorrências
	Qual o tempo de correção de um defeito?	Tempo de correção de um defeito
	Qual a diferença entre a quantidade de defeitos em aberto e a quantidade de defeitos corrigidos?	Quantidade de defeitos abertos X Quantidade de defeitos corrigidos
	Qual a quantidade de defeitos em aberto?	Ocorrências pendentes de correção

	Quantos defeitos não foram corrigidos na iteração em que foram descobertos?	Quantidade de defeitos enviados para a próxima iteração
OBJETIVO	**QUESTÕES**	**MEDIDAS**
Acompanhar o percentual de retrabalho	Qual o percentual de retrabalho?	Percentual de retrabalho
OBJETIVO	**QUESTÕES**	**MEDIDAS**
Acompanhar a conformidade do produto com os requisitos de teste	Qual o percentual de aceitação do produto?	Percentual de assertivas de testes de aceitação passando
	Qual o percentual de falha nos testes de aceitação?	Percentual de assertivas de testes de aceitação falhando
OBJETIVO	**QUESTÕES**	**MEDIDAS**
Avaliar a qualidade do teste	Qual o grau de qualidade dos testes executados?	Qualidade do teste
OBJETIVO	**QUESTÕES**	**MEDIDAS**
Avaliar o impacto do defeito	Qual o impacto dos defeitos encontrados	Densidade do defeito
OBJETIVO	**QUESTÕES**	**MEDIDAS**
Monitorar a cobertura de testes	Qual a quantidade de testes funcionais por *user story*?	Quantidade de testes funcionais por *user story*
	Qual o percentual de cobertura de testes?	Percentual de cobertura de testes
OBJETIVO	**QUESTÕES**	**MEDIDAS**
Monitorar a estabilidade do escopo de testes	Qual o percentual de mudança de escopo de testes?	Mudanças de escopo
OBJETIVO	**QUESTÕES**	**MEDIDAS**
Monitorar a execução dos testes	Qual o tempo médio para encontrar um defeito?	Tempo médio para encontrar um defeito
	Qual o percentual de efetividade dos casos de teste?	Efetividade do caso de testes
	Os testes foram adequados? Qual o percentual de qualidade do conjunto de testes?	Efetividade e eficiência dos testes
	Qual a taxa de falhas na primeira execução dos casos de teste?	Taxa de falhas na primeira execução dos casos de teste
	Quantas ocorrências foram corrigidas, mas não foram testadas?	Ocorrências resolvidas que ainda não foram testadas
	Quais as fases do ciclo de vida de desenvolvimento de software que mais se encontram defeitos? Qual a fase que o defeito foi encontrado?	Fase em que o defeito foi encontrado
	Qual o percentual de testes de aceitação que passaram?	Testes de aceitação que passaram
	Qual o número total de casos	Número de casos de teste

	de teste?	
	Qual o número de casos de teste executados?	Número de casos de teste executados
	Qual o número dos casos de teste que passaram?	Número de casos de teste que passaram
	Qual o número dos casos de teste que falharam?	Número de casos de teste que falharam
	Qual o número de casos de teste que estão sob investigação?	Número de casos de teste que estão sob investigação
	Qual o número de casos de teste que estão bloqueados?	Número de casos de teste bloqueados
	Quantos casos de teste foram reexecutados?	Número de casos de teste reexecutados
	Quantos casos de teste falharam em primeira execução?	Número de casos de teste que falharam em primeira execução
	Qual o número total de execuções de testes?	Número total de execuções
	Qual o tempo de execução por caso de teste?	Tempo de execução por caso de teste
	Qual o tempo total de execução dos testes?	Tempo de execução do teste
	Qual o percentual dos casos de teste que passaram?	Percentual dos casos de teste que passaram
	Qual o percentual dos casos de teste bloqueados?	Percentual dos casos de teste bloqueados
	Qual a taxa de falhas de execução dos casos de teste em primeira execução?	Taxa de falhas em primeira execução
	Qual a taxa de falhas de execução dos casos de teste em geral?	Taxa de falhas em geral
	Qual a taxa de descoberta de defeitos?	Taxa de descoberta de defeitos
	Qual o percentual de testes de unidade que passaram?	Percentual de assertivas de Teste de Unidade que passaram
	Qual o percentual de testes de unidade que falharam?	Percentual de assertivas de Teste de Unidade que falharam
OBJETIVO	**QUESTÕES**	**MEDIDAS**
Monitorar funcionalidades críticas do produto	Quais as funcionalidades críticas do sistema?	Densidade de defeitos encontrados
OBJETIVO	**QUESTÕES**	**MEDIDAS**
Monitorar as funcionalidades entregues com testes	Quais features foram entregues com testes?	Running Tested Features
OBJETIVO	**QUESTÕES**	**MEDIDAS**
Monitorar o planejamento	Qual o tempo de teste	Tempo de teste estimado X

dos testes	estimado? Qual o tempo de teste utilizado?	Tempo de teste utilizado
	Qual o fator de segurança na estimativa dos testes?	Fator de Segurança
	Qual a taxa de esforço na execução dos testes?	Taxa de esforço do teste
	Qual o número de cenários de teste?	Número de cenários de teste
	Qual o tempo de especificação de casos de teste por pessoa/dia?	Tempo necessário para elaboração de casos de teste por pessoa/dia
OBJETIVO	**QUESTÕES**	**MEDIDAS**
Monitorar os defeitos do produto em produção	Quais clientes possuem mais defeitos?	Defeitos por unidade organizacional
	Qual a quantidade de defeitos encontrados em produção?	Quantidade de defeitos encontrados em produção
OBJETIVO	**QUESTÕES**	**MEDIDAS**
Monitorar os testes de aceitação	Qual a quantidade de testes de aceitação por funcionalidade?	Quantidade de testes de aceitação por funcionalidade

Fonte: Elaborada pelo autor.

6.2 Verificação da aderência ao MPT.Br

Nesta seção será demonstrado o processo de verificação de aderência do repositório de medidas ao MPT.Br. Os níveis em que o repositório será avaliado são: 1, 2 e 3, especificamente em práticas que tratem de medidas de Teste de Software.

Primeiramente será realizada uma verificação com a prática GPT13 – Estabelecer indicadores de desempenho de testes do nível 1, em seguida com o GPT24 – Monitorar defeitos no nível 2 e, por fim, abordará duas práticas do nível 3, que são: MAT1 – Definir objetivos de medição de teste e MAT2 – Estabelecer e documentar medidas.

Inicialmente o objetivo da prática GPT13 foi verificado, que consiste em **estabelecer um conjunto de indicadores do teste de software para que a gerência do projeto seja feita com base em dados objetivos**. Baseado no texto fornecido pelo MPT.Br, um indicador é uma medida que fornece uma estimativa ou uma avaliação de atributos específicos derivados de um modelo relativo a determinadas necessidades de informação (SOFTEXRECIFE, 2011). A aderência do repositório ao GPT13 foi verificada na avaliação do final do repositório na seção 7, em que uma Organização de desenvolvimento de software que estava implantando o nível 1 do MPT.Br, e que utilizou o repositório para selecionar indicadores dos testes de software.

O objetivo da prática GPT24 consiste em **realizar um acompanhamento sistemático dos defeitos do produto, identificando tendências e tomando ações corretivas**. Baseado no texto fornecido pelo MPT.Br, os defeitos encontrados devem ser monitorados. Na definição dos objetivos de medição de Teste de Software foram levantados objetivos específicos de acompanhamento de defeitos, tais como:

- Monitorar os defeitos do produto em produção
- Monitorar a execução dos testes
- Acompanhar a resolução dos defeitos encontrados
- Avaliar o impacto do defeito

No repositório (Apêndice C), é possível visualizar medidas que foram especificadas a partir dos objetivos listados acima. Dessa forma, o repositório é aderente ao GPT24 pois propõe medidas de monitoramento, acompanhamento e avaliação de defeitos encontrados.

O objetivo da prática MAT1 consiste em **estabelecer e manter objetivos de medição de Teste de Software derivados de necessidades de informação**. Baseado no texto fornecido pelo MPT.Br objetivos devem ser estabelecidos e mantidos, dessa forma o repositório é aderente pelo fato de possuir objetivos de medição que foram selecionados de acordo com as respostas informadas no *survey* conforme seção 5.

E por fim, o objetivo MAT2 consiste em **estabelecer e manter medidas que satisfazem e estão diretamente associados aos objetivos de medição**. Baseado no texto fornecido pelo MPT.Br é necessário que haja uma rastreabilidade entre objetivos de medição e medidas que endereçam esses objetivos, dessa forma o repositório é aderente pois mantém os objetivos relacionados as medidas do repositório. O MPT.Br não define um modelo de especificação de medidas, diante disso não há impacto negativo utilizar o modelo fornecido pela ISO/IEC 25010.

Pelo exposto acima o repositório de medidas estabelecido neste trabalho é aderente às práticas GPT13, GPT24, MAT1 e MAT2 dos níveis 1, 2 e 3 do MPT.Br.

7 AVALIAÇÃO FINAL DO REPOSITÓRIO

Nesta seção será descrita a forma de avaliação do repositório de medidas. A avaliação do repositório foi realizada em uma Organização de desenvolvimento de software que tem suas atividades concentradas no estado do Ceará que está implantando o modelo de processo de testes MPT.Br. Por questões de sigilo em relação as informações abordadas não utilizamos o nome da empresa, mas somente Empresa.

A avaliação do repositório deu-se em 4 etapas:

- Visão geral do trabalho
- Disponibilização do repositório para a Empresa
- Seleção de medidas que atendem as necessidades da Empresa
- E por fim, um relato da percepção da Empresa.

7.1 Visão geral do trabalho

A primeira etapa para avaliação do repositório genérico de medidas de testes de software foi realizar uma apresentação para o gerente da Empresa participante, no intuito de deixar a Organização ciente dos objetivos e da sua relevância. Essa apresentação ocorreu nas dependências da Empresa. Inicialmente houve uma apresentação do autor da pesquisa e os passos que foram executados para especificar as medidas. Em seguida, o objetivo geral da pesquisa foi apresentado para fins de norteamento dos passos posteriores.

7.2 Disponibilização do repositório de medidas para a empresa

O repositório foi disponibilizado para o gerente da Empresa no intuito de efetuar uma análise das medidas e as suas possíveis utilizações. A estava em processo de implantação do nível 1 do MPT.Br e existia a necessidade de medidas que atendessem ao GPT13, que consiste na definição de indicadores de desempenho de testes.

7.3 Seleção de medidas para a empresa

Para seleção das medidas que seriam utilizadas pela Empresa foi realizada uma reunião com o autor do trabalho, um diretor, um gerente de desenvolvimento e um analista de qualidade. Nessa reunião, foi definido por consenso dos participantes que seriam selecionadas uma quantidade pequena de medidas, pois estavam na implantação do nível 1 do MPT.Br e queriam garantir que tudo seria executado. Conforme houvesse o amadurecimento do processo, vão selecionar mais medidas. Diante disso, foram selecionadas 4 medidas do repositório (Apêndice C), são elas:

Tabela 5 - Medidas selecionadas na avaliação do repositório

Objetivo	Medida
Acompanhar a eficácia dos testes	Quantidade de falhas encontradas no produto em produção
Monitorar a execução dos testes	Cobertura de teste baseada em código
Monitorar a execução dos testes	Tempo de teste estimado X Tempo de teste utilizado
Acompanhar o percentual de retrabalho	Percentual de retrabalho

Fonte: Elaborada pelo autor.

Essas medidas foram utilizadas em três projetos nessa empresa. O projeto A consiste no desenvolvimento de um módulo de sincronização de um produto mantido pela Empresa. O projeto B consiste na manutenção dos sistemas da Empresa. Essa manutenção é oriunda da abertura de atividades dos clientes da Empresa. O projeto C consiste no desenvolvimento de uma interface para utilização de cartão de credito no sistema.

A Tabela 6 contém os valores coletados com as medidas nos projetos A, B e C.

Tabela 6 - Medidas coletadas

Projeto	Medida	Valor
Projeto A	Quantidade de falhas encontradas no produto em produção	**Total: zero**
	Cobertura de teste baseada em código	**0%**
	Tempo de teste estimado X Tempo de teste utilizado	**Os tempos estão conforme o estimado**
	Percentual de retrabalho	**7,27%**
Projeto B	Quantidade de falhas encontradas no produto em produção	**Total: 3 erros**
	Cobertura de teste baseada em código	**2,7%**
	Tempo de teste estimado X Tempo de teste utilizado	**Os tempos estão conforme o estimado**
	Percentual de retrabalho	**2,8%**
Projeto C	Quantidade de falhas encontradas no produto em produção	**Total: zero**
	Cobertura de teste baseada em código	**71,28%**
	Tempo de teste estimado X Tempo de teste utilizado	**Os tempos estão conforme o estimado**
	Percentual de retrabalho	**0%**

Fonte: Elaborada pelo autor.

Devido os procedimentos da Empresa, os valores das medidas são contabilizados no final do projeto, como forma de obter uma visão geral do projeto ao invés de uma visão simplificada durante as *sprints*. A partir da Tabela 6 é possível perceber a estabilidade alcançada no produto a partir das falhas encontradas no produto em produção, o crescimento do percentual de cobertura de testes que indicam o aumento da maturidade do time de

desenvolvimento e consequentemente a qualidade do produto, a assertividade da equipe nas estimativas em projetos e a diminuição do percentual de retrabalho.

7.4 Relato da empresa

De acordo com o gerente da Empresa a medida **Quantidade de falhas encontradas no produto** é um indicador importante para mensurar a qualidade do sistema entregue aos clientes da empresa. Com ele também é possível verificar pontos do sistema que o cliente usa e que não são cobertos pela rotina de testes. A medida **Cobertura de teste baseada em código** possibilita um alinhamento do conhecimento do time a nível de código e aumentar o nível de qualidade do que é produzido aos clientes. Com esse dado, é possível notar a diminuição de retornos do time de testes para o time de desenvolvimento. A medida **Tempo de teste estimado X Tempo de teste utilizado** essa medida é utilizada para medir o conhecimento e domínio do time de testes em relação ao produto desenvolvido. A assertividade mostra como as definições e conhecimentos da ferramenta estão progredindo. Quando muito discrepante a relação entre o tempo estimado e o tempo utilizado é possível perceber falta de conhecimento na tecnologia utilizada, não entendimento do que foi solicitado ou até mesmo falta de domínio sobre o produto. A medida **Percentual de retrabalho** não avalia somente os retornos oriundos do cliente ou do time de testes, visa também às atividades urgentes com prazo e escopo fechados que podem gerar retrabalhos. Destacou também que, com a utilização de medidas, foi possível estabelecer metas mais concisas para o time de desenvolvimento e auxiliando na avaliação constante do time. Outro fator destacado pelo gerente da Empresa foi a utilização do repositório de medidas na implantação do nível 1 do MPT.Br, permitindo atingir os requisitos do GPT13 bem como outras práticas do MPT.Br na Organização.

8 CONSIDERAÇÕES FINAIS

Com a necessidade da boa gestão dos dados gerados pela execução dos testes durante o ciclo de vida de desenvolvimento de software e consequente aumento da qualidade dos produtos desenvolvidos, medidas de Teste de Software são ferramentas poderosas para o refinamento de dados obtidos. Medidas são comumente utilizadas em Modelos de Processos de desenvolvimento, bem como em Modelos de Processos de Teste de Software – MPT.Br voltado para a realidade brasileira.

Diversos trabalhos na literatura abordam medidas de Teste de Software, porém não tinham como objetivo construir um repositório baseado no contexto de Organizações de desenvolvimento de Software. O objetivo geral desse é o desenvolvimento de um repositório de medidas de Teste de Software baseado no contexto de Organizações de desenvolvimento e que seja aderente ao Modelo de Processo de Teste de Software - MPT.Br.

Inicialmente foi realizada uma busca na literatura por medidas de Teste de Software existentes e consequentemente pelos seus objetivos. Após o refinamento das medidas, os objetivos foram levantados no intuito de criar um *survey* para coleta de objetivos de medição de Teste de Software e seus graus de importância para a Organização. Para criação do *survey* foi utilizada a ferramenta *LimeSurvey*, que é uma ferramenta de criação, publicação e coleta de dados, e para definição do grau de importância foi utilizada uma escala de *Likert* com os seguintes graus: sem importância, pouco importante, imparcial, muito importante e extremamente importante. Durante a construção do *survey* um pré-teste foi realizado para validar a coerência da estrutura do questionário, bem como modificações nas questões. Em seguida o questionário foi aplicado do 01/12/2015 até o dia 15/12/2015.

Após a coleta de dados do *survey* foi realizada uma análise do perfil do participante e dos objetivos com seus graus de importância. Essa análise permitiu selecionar respostas baseadas na experiência dos participantes, bem como selecionar os objetivos de medição.

Com os objetivos selecionados foi possível executar a fase de Definição da técnica de GQM, que consiste na seleção de objetivos de medição, perguntas que respondam a esses objetivos e a especificação da medida que atende as perguntas. Uma verificação de aderência do repositório de medidas ao MPT.Br foi realizada para constatar o produto final da pesquisa.

Uma avaliação do repositório foi realizada em uma empresa de desenvolvimento de software que estava implantando o nível 1 do MPT.Br. Essa avaliação permitiu validar os

benefícios do repositório e sua possibilidade de utilização em diversas Organizações de software, visto que o mesmo foi construído de acordo com os graus de importância de diversas Organizações.

Com o trabalho foi possível perceber a contribuição de trabalhos acadêmicos em Organizações de desenvolvimento de software, pois a empresa selecionou todos os seus indicadores a partir do repositório de medidas. Porém o trabalho focou apenas na criação de um repositório de medidas a partir de objetivos de Organizações de desenvolvimento de software que fosse aderente apenas ao modelo MPT.Br, o mesmo não levou em consideração outros modelos como o MPS.BR que também trata de medidas. Além disso, o repositório foi aplicado em apenas uma organização que utilizou um escopo reduzido de medidas.

Como trabalhos futuros, temos:

1. Pretende-se validar o repositório em outras organizações de software, visando aplicar as outras medidas que foram especificadas.
2. Verificar a aderência do repositório a outros modelos de processos de testes.
3. Desenvolvimento de um sistema para gerenciamento das medidas, bem como um tutorial para utilizar o repositório.
4. Fazer uma revisão por pares no repositório.

REFERÊNCIAS

Agarwal, B. B., Tayal, S. P. & Gupta, M. **Software engineering and testing.** Jones and Bartlett publishers, LLC. 516 pp. 2010.

ALEXANDRE, João Welliandre Carneiro et al. Análise do número de categorias da escala de *Likert* aplicada à gestão pela qualidade total através da teoria da resposta ao item. In: ENCONTRO NACIONAL DE ENGENHARIA DE PRODUÇÃO, 23, 2003, Ouro Preto. **ENEGEP.** Ouro Preto: Enegep, 2003. p. 1 - 8.

ARAÚJO, Solange Alcântara. **Uma abordagem para construção do repositório de medidas em Organizações de software.** 2008. 84 f. Dissertação (Mestrado) - Curso de Informática Aplicada, Universidade de Fortaleza, Fortaleza, 2008.

Associação para Promoção da Excelência do Software Brasileiro – Softex. **MPS.Br – Guia Geral**, 2012. Disponível em:<http:// www.softex.br>. Acesso em 17 de setembro.

Associação para Promoção da Excelência do Software Brasileiro – Softex. **MPS.Br – Guia de Aquisição**, 2013. Disponível em:<http:// www.softex.br>. Acesso em 17 de setembro.

Associação para Promoção da Excelência do Software Brasileiro – Softex Recife. **MPT.Br – Guia de Referência do Modelo**. 2012. Disponível em: < http://mpt.org.br/>. Acesso em 17 de setembro.

BABBIE, Earl. ***Survey* Research Methods.** International Thomson Publishing Inc., 1997.

Basili, V., Caldieira, G., Rombach, H. D. Goal Question Metric Paradigm. Encyclopedia of Software Engineering. John Wiley & Sons, Inc., 1994,.p. 528-532.

BASTOS, Aderson et al. **Base de Conhecimento em Teste de Software.** São Paulo: Martins Fontes, 2012. 263 p.

BERTOLINO, Antonia. Software Testing Research: Achievements, Challenges, Dreams. **Future Of Software Engineering (fose '07),** [s.l.], n. 7, p.85-103, maio 2007. Institute of Electrical & Electronics Engineers (IEEE). DOI: 10.1109/fose.2007.25.

BURNSTEIN, Ilene; SUWANNASART, Taratip; CARLSON, C. R.. Developing a Testing Maturity Model for software test process evaluation and improvement. **International Test Conference.** Washington, Dc, p. 581-589. out. 1996.

CMMI. Product Team. **CMMI for development**. Version 1.2. Software Engineering Institute. Agosto. 2006.

COVATTI, Andressa. **MIBCIS: método de integração entre o BSC, CMMI E Six Sigma utilizando GQM no suporte a definição de métricas.** 2007. 368 f. Dissertação (Mestrado) - Curso de Ciência da Computação, Pontifícia Universidade Católica do Rio Grande do Sul, Rio Grande do Sul, 2007.

DANEVA, Maya et al. Agile requirements prioritization in large-scale outsourced system projects: An empirical study.Elsevier: The journal od Systems and Software. Enschede, vol 86 p. 1333-1354. 29 dez. 2012.

ERICSON, Thomas; SUBOTIC, Anders; URSING, Stig. TIM - Test Improvement Model. **Software: Testing, Verification e Reliability.** Suécia, p. 229-246. dez. 1997.

FENTON, N. E. **Software Measurement: A Necessary Scientific Basis**. IEEE Transactions on Software Engineering, vol 20 n. 3, 1994, p. 199-206.

FUGGETTA, Alfonso. **Software Process: A Roadmap.** In: Proceedings of the Conference of Future of Software Engineering, ICSE'200, Limerick, Ireland, 2000.

GAROUSI, Vahid; FARMA, Tan. A replicated *survey* of software testing practices in the Canadian province of Alberta: Whats has changed from 2004 to 2009?. Elsevier: The journal od Systems and Software. Interplay between Usability Evaluation and Software Development. . Alberta, Canada, vol 83 p. 2251-2262. 15 jul. 2010.

GELPERIN, David. TSM – Testability Support Model. 1996.

HARTMANN, Deborah; DYMOND, Robin. Appropriate Agile Measurement: Using metrics and diagnostics to deliver business value. Proceedings Of Agile Conference. Minneapolis, Mn, p. 6-134. 23 jul. 2006.

HEIKKINEN, Elli. **Testing metrics in a software development company.** 2015. 76 f. Dissertação (Mestrado) - Curso de Mestre em Ciências, Tampere University Of Technology, Tampere, 2015.

HETZEL, Bill. **The complete guide to Software Testing.** 2. ed. Wellesley: Book, 1988.

ISO/IEC. **ISO/IEC 9126: Software systems engineering - Product Quality**. 2003.

ISO/IEC. **ISO/IEC 14598: Software systems engineering - Software Product Evaluation**. 2007.

ISO/IEC. **ISO/IEC 15939: Software systems engineering - Measurement process**. 2007.

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

ISO/IEC. **ISO/IEC 25023: Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE) – Measurement of system and software product quality**. 2011a.

ISO/IEC. **ISO/IEC/IEEE 29119-1: Software systems engineering – software testing**. 2013.

IEEE. **IEEE Standard for a Software Quality Metrics Methodology**. Software Engineering Standards Committee of the IEEE Computer Society. Vol 1. p 1. Dezembro. 1998.

IE Test Consultancy LTD, Software Futures. **Testing Assessment Programme – TAP.** 1995.

KHAN, Ehmer e KHAN, Farmeena. **Importance of Software Testing in Software Development Life**. IJCSI - International Journal of Computer Science Issues. Vol 11 Issue 2 n. 2. Março. 2014.

KOOMEN, Tim; POL, Martin. **Test Process Improvement: A Step-by-step Guide to Structured Testing.** Uk: Addison Wesley, 1999. 240 p.

KRAUSE, Mitchel. **Maturity Model for Automated Software Testing – MMAST.** 1994.

LAZIC, Ljubomir; MASTORAKIS, Nikos. Cost effetive software Test Metrics. Wseas Transactions On Computers. Piraeu, Grécia, p. 599-619. jun. 2008.

MALDONADO, José Carlos. **Critérios Potenciais Usos: Uma Contribuição ao Teste Estrutural de Software**. PhD thesis. DCA/FEE/UNICAMP. Campinas-SP. Julho. 1991.

MUKKER, Amrita Raj; MISHRA, Anil Kumar; SINGH, Latika. Enhancing quality in Scrum Software Projects. International Journal Of Science And Research (ijsr). Gurgaon, India, p. 1-7. abr. 2014.

NAGAPPAN, Nachiappan. Toward a software testing and reliability early warning metric suite. In: 26th International Conference on Software Engineering, 2004 (ICSE 2004). Edinburgh, United Kingdom, IEEE Computer Society, 2004, p. 60-62.

NAGAPPAN, Nachiappan et al. Providing test quality feedback using static source code and automatic test suite metrics. In: 16th IEEE International Symposium on Software Reliability Engineering, 2005 (ISSRE 2005). Chicago, IL, USA, IEEE Computer Society, 2005, p. 85-94. 01 nov. 2005.

SCHMITZ, Carsten et al. **Lime*Survey*.** Fevereiro. 2003. Disponível em: < https://www.lime*survey*.org/en/>. Acesso em 28 de dezembro de 2015.

MYERS, Glenford; BADGETT, Tom; SANDLER, Corey. **The Art of Software Testing.** 3. ed. Hoboken, New Jersey: Wiley, 2012. 240 p.

Park, R.E., Goethert, W.B., FLORAC, W.A. **Goal-Driven Software Measurement – A Guidebook**. Handbook CMU/SEI-96-HB-002, Pittsburgh, Software Engineering Institute, Carnegie Mellon University, 1996. Disponível em: <http://www.sei.cmu.edu>.

RIOS, Emerson; MOREIRA, Trayahú. **Teste de Software.** 3. ed. Brasil: Alta Books, 2013. 305 p.

SATO, Danilo Toshiaki. **Uso eficaz de métricas em métodos ágeis de desenvolvimento de software.** 2007. 155 f. Dissertação (Mestrado) - Curso de Ciências da Computação, Universidade de São Paulo - USP, São Paulo, 2007.

SEI, S. E. I.-. **CMMI for Development, Version 1.3**. Software Engineering Institute - SEI. [S.l.]. 2010.

Systeme Evolutif. **Test Organization Maturity – TOM**. 1989.

TRODO, Lia Degrazia. Uso de Métricas nos Teste de Software. 2009. Universidade Federal do Rio Grande do Sul.

VEENENDAAL, Erik Van. **Test Maturity Model Integration – TMMI**. TMMI Foundation. Release. Irlanda. 2012. Disponível em: < http://www.tmmi.org>. Acesso em 17 de setembro de 2015.

VICENTE, André Abe. Definição e gerenciamento de métricas de teste no contexto de métodos ágeis. 2010. 158 f. **Dissertação (Mestrado) - Curso de Mestre em Ciências - Ciências da Computação e Matemática Computacional, Instituto de Ciências**

Matemáticas e de Computação - ICMC, Universidade de São Paulo - USP, São Carlos, 2010.
VON WANGENHEIM, Christiane Gresse; VON WANGENHEIM, Aldo; LINO, Juliana Izabel. **Medição de Software:** Guia Prático. Florianópolis: Bookess, 2012.

APÊNDICES

APÊNDICE A – MODELO DE EMAIL ENVIADO AOS PARTICIPANTES DO QUESTIONÁRIO

Prezados,

Eu sou William Rosendo, graduando do curso de Sistemas de Informação pela Universidade Federal do Ceará - Campus Quixadá. Atualmente trabalho em uma empresa de desenvolvimento de software, mais especificamente na área de Teste de Software.

Como pré-requisito da obtenção do título em bacharel em Sistemas de Informação, estou desenvolvendo uma pesquisa no TCC. Esta pesquisa tem por intuito construir um repositório genérico de medidas de Teste de Software.

As medidas de testes serão formalizadas a partir dos objetivos de Teste de Software de Organizações. Os objetivos de Teste de Software são importantes por nortearem quais informações sobre as atividades de testes executadas são importantes para as Organizações de desenvolvimento de software.

O questionário visa coletar os principais objetivos de Teste de Software para a Organização de desenvolvimento de software, bem como o seu grau de importância.

Para participar, por favor, utilize o link:
http://ger.quixada.ufc.br/pesquisas/index.php/275569/lang-pt-BR
Atenciosamente,
William Rosendo Soares
Bacharelando em Sistemas de Informação
Universidade Federal do Ceará

APÊNDICE B – QUESTIONÁRIO DISPONIBILIZADO AOS PARTICIPANTES

Estrutura do questionário:

- S1 Perfil dos participantes
- S2 Objetivos de medição de teste de software e seu grau de importância na Organização

Questões	Grupo	Pergunta	Tipo de Resposta	Justificativa/Referência
Perguntas de Contexto				
CONTEXTO	S1	Nome do Participante	Pergunta aberta	Opcional
CONTEXTO	S1	Qual o seu papel atualmente na Organização?	( ) Engenheiro de Testes ( ) Arquiteto de Testes ( ) Analista de Testes ( ) Testador ( ) Gerente de Projeto/Gerente de Desenvolvimento ( ) Diretor ( ) Outros	É necessário saber qual o cargo ou função do respondente, identificando por meio desta, se tem alguma relação entre o cargo e possíveis respostas.
CONTEXTO	S1	Quanto tempo (em anos) você trabalha com Tecnologia da Informação e desenvolvimento de software?	Pergunta aberta	É necessário saber o tempo de experiência dos participantes para possíveis cruzamentos de informações entre as respostas obtidas com determinados perfis de experiência.
CONTEXTO	S1	Quanto tempo (em anos) você trabalha especificamente com Teste de Software?	Pergunta aberta	É necessário saber o tempo de experiência dos participantes para possíveis cruzamentos de informações entre as respostas obtidas com determinados perfis de experiência.
CONTEXTO	S1	Como você considera ser sua experiência em Teste de Software?	( ) Excelente ( ) Alta ( ) Média ( ) Baixa ( ) Nenhuma	É necessário saber a experiência dos participantes para possíveis cruzamentos de informações entre as respostas obtidas com determinados perfis de experiência.
CONTEXTO	S1	Qual sua formação acadêmica?	( ) Técnico ( ) Graduação ( ) Especialização	É necessário saber qual o nível de formação do respondente,

Questões	Grupo	Pergunta	Tipo de Resposta	Justificativa/Referência
			( ) Mestrado ( ) Doutorado	identificando por meio desta, qual a formação de quem respondeu ao questionário e possibilitando um delinear para comparativos futuros entre as respostas sobre engenharia de requisitos.
CONTEXTO	S1	Qual a(s) sua(s) área(s) de formação acadêmica?	( ) Ciência da Computação ( ) Engenharia da Computação ( ) Sistemas de Informação ( ) Engenharia de Software ( ) Análise e Desenvolvimento de Sistemas ( ) Área de Negócios / MBA	É necessário saber qual a área de formação, pois nem sempre pessoas ligadas a requisitos são da área da Computação e cursos afins, possibilitando alguns cruzamentos entre as respostas obtidas por perfis diferentes.
CONTEXTO	S1	Em que estado (UF) você executa suas atividades de desenvolvimento e Teste de Software?	Pergunta aberta	É necessário para validar o público alvo da pesquisa.
Questões de Pesquisa – Objetivos de medição de Teste de Software				
APLICAÇÃO	S2	Acompanhar a eficácia dos testes	( ) Sem importância ( ) Pouco importante ( ) Nem importante/ Nem sem importância ( ) Muito importante ( ) Extremamente importante	Verificar o grau de importância dos objetivos de medição de Teste de Software.
APLICAÇÃO	S2	Acompanhar a eficiência dos testes	( ) Sem importância ( ) Pouco importante ( ) Nem importante/ Nem sem importância ( ) Muito importante ( ) Extremamente importante	Verificar o grau de importância dos objetivos de medição de Teste de Software.
APLICAÇÃO	S2	Acompanhar a resolução de defeitos encontrados	( ) Sem importância ( ) Pouco importante ( ) Nem importante/ Nem sem	Verificar o grau de importância dos objetivos de medição de Teste de Software.

Questões	Grupo	Pergunta	Tipo de Resposta	Justificativa/Referência
			importância ( ) Muito importante ( ) Extremamente importante	
APLICAÇÃO	S2	Acompanhar o percentual de defeitos por severidade	( ) Sem importância ( ) Pouco importante ( ) Nem importante/ Nem sem importância ( ) Muito importante ( ) Extremamente importante	Verificar o grau de importância dos objetivos de medição de Teste de Software.
APLICAÇÃO	S2	Acompanhar o percentual de retrabalho	( ) Sem importância ( ) Pouco importante ( ) Nem importante/ Nem sem importância ( ) Muito importante ( ) Extremamente importante	Verificar o grau de importância dos objetivos de medição de Teste de Software.
APLICAÇÃO	S2	Avaliar a conformidade do produto com os requisitos de Teste de Software	( ) Sem importância ( ) Pouco importante ( ) Nem importante/ Nem sem importância ( ) Muito importante ( ) Extremamente importante	Verificar o grau de importância dos objetivos de medição de Teste de Software.
APLICAÇÃO	S2	Avaliar a qualidade dos testes	( ) Sem importância ( ) Pouco importante ( ) Nem importante/ Nem sem importância ( ) Muito importante ( ) Extremamente importante	Verificar o grau de importância dos objetivos de medição de Teste de Software.
APLICAÇÃO	S2	Avaliar o impacto do defeito	( ) Sem importância ( ) Pouco importante ( ) Nem importante/ Nem sem importância ( ) Muito importante ( ) Extremamente importante	Verificar o grau de importância dos objetivos de medição de Teste de Software.
APLICAÇÃO	S2	Monitorar a cobertura de testes	( ) Sem importância ( ) Pouco importante ( ) Nem importante/ Nem sem importância	Verificar o grau de importância dos objetivos de medição de Teste de Software.

Questões	Grupo	Pergunta	Tipo de Resposta	Justificativa/Referência
			( ) Muito importante ( ) Extremamente importante	
APLICAÇÃO	S2	Monitorar a criação de códigos de teste	( ) Sem importância ( ) Pouco importante ( ) Nem importante/ Nem sem importância ( ) Muito importante ( ) Extremamente importante	Verificar o grau de importância dos objetivos de medição de Teste de Software.
APLICAÇÃO	S2	Monitorar a estabilidade do escopo de testes	( ) Sem importância ( ) Pouco importante ( ) Nem importante/ Nem sem importância ( ) Muito importante ( ) Extremamente importante	Verificar o grau de importância dos objetivos de medição de Teste de Software.
APLICAÇÃO	S2	Monitorar a execução dos testes	( ) Sem importância ( ) Pouco importante ( ) Nem importante/ Nem sem importância ( ) Muito importante ( ) Extremamente importante	Verificar o grau de importância dos objetivos de medição de Teste de Software.
APLICAÇÃO	S2	Monitorar a quantidade de builds por iteração	( ) Sem importância ( ) Pouco importante ( ) Nem importante/ Nem sem importância ( ) Muito importante ( ) Extremamente importante	Verificar o grau de importância dos objetivos de medição de Teste de Software.
APLICAÇÃO	S2	Monitorar funcionalidades críticas do produto	( ) Sem importância ( ) Pouco importante ( ) Nem importante/ Nem sem importância ( ) Muito importante ( ) Extremamente importante	Verificar o grau de importância dos objetivos de medição de Teste de Software.
APLICAÇÃO	S2	Monitorar funcionalidades entregues com testes	( ) Sem importância ( ) Pouco importante ( ) Nem importante/ Nem sem importância ( ) Muito importante	Verificar o grau de importância dos objetivos de medição de Teste de Software.

Questões	Grupo	Pergunta	Tipo de Resposta	Justificativa/Referência
			( ) Extremamente importante	
APLICAÇÃO	S2	Monitorar o planejamento dos testes	( ) Sem importância ( ) Pouco importante ( ) Nem importante/ Nem sem importância ( ) Muito importante ( ) Extremamente importante	Verificar o grau de importância dos objetivos de medição de Teste de Software.
APLICAÇÃO	S2	Monitorar os custos dos testes	( ) Sem importância ( ) Pouco importante ( ) Nem importante/ Nem sem importância ( ) Muito importante ( ) Extremamente importante	Verificar o grau de importância dos objetivos de medição de Teste de Software.
APLICAÇÃO	S2	Monitorar os defeitos do produto em produção	( ) Sem importância ( ) Pouco importante ( ) Nem importante/ Nem sem importância ( ) Muito importante ( ) Extremamente importante	Verificar o grau de importância dos objetivos de medição de Teste de Software.
APLICAÇÃO	S2	Monitorar os testes de aceitação	( ) Sem importância ( ) Pouco importante ( ) Nem importante/ Nem sem importância ( ) Muito importante ( ) Extremamente importante	Verificar o grau de importância dos objetivos de medição de Teste de Software.
APLICAÇÃO	S2	Monitorar os testes de unidade	( ) Sem importância ( ) Pouco importante ( ) Nem importante/ Nem sem importância ( ) Muito importante ( ) Extremamente importante	Verificar o grau de importância dos objetivos de medição de Teste de Software.
APLICAÇÃO	S2	Caso existam outros objetivos importantes para sua Organização que não estejam listados acima, favor especificá-	Pergunta aberta	Opcional

Questões	Grupo	Pergunta	Tipo de Resposta	Justificativa/Referência
		los abaixo com seus respectivos graus de importância.		

APÊNDICE C – REPOSITÓRIO DE MEDIDAS DE TESTE DE SOFTWARE

Nome da Medida	Descrição/Objetivo	QMEs	Fórmula de Cálculo	Interpretação	Tipo de Escala	Foco da medida
Quantidade de falhas encontradas no produto em produção	Acompanhar a eficácia dos testes	Essa medida tem por intuito mostrar a quantidade de defeitos reportados em produção	**∑QtdP** **QtdP** representa os defeitos encontrados em produção	A quantidade ideal de defeitos encontrados em produção é zero. Porém a Organização pode definir metas até atingir o valor ideal.	Absoluta	Interno/Externo
Densidade dos defeitos residuais	Acompanhar a eficácia dos testes	É uma estimativa do número de defeitos não resolvidos que foram para produção. Essa medida permite alcançar um nível de defeitos que seja aceitável para o cliente.	Estimativa	Estimativa definida pela Organização de acordo com as suas necessidades.	Absoluta	Interno/Externo
Taxa de ocorrências válidas	Acompanhar a eficiência dos testes	Essa medida mostra o percentual de ocorrências válidas em um certo período de tempo, indicando a eficiência do teste.	**TOV = NOV/NOE** **NOV** é o número de ocorrências válidas **NOE** é o total de ocorrências encontradas.	**TOV** representa a taxa de ocorrências válidas encontradas pelos testes. Quanto mais próximo de 1, mais adequado.	Taxa	Interno
Taxa de problemas encontrados na correção de defeitos	Acompanhar a eficiência dos testes	Essa medida mostra o percentual de defeitos corrigidos que resultaram em novas ocorrências.	**TPCD = NDNO/NTDC** **NDNO** é o número de defeitos que geraram novas ocorrências	**TPCD** representa a taxa de ocorrências válidas encontradas pelos testes.	Taxa	Interno

Nome da Medida	Descrição/Objetivo	QMEs	Fórmula de Cálculo	Interpretação	Tipo de Escala	Foco da medida
			NTDC é o número total de defeitos corrigidos.	Quanto mais próximo de 0, mais adequado.		
Percentual de eficiência de testes	Acompanhar a eficiência dos testes	O objetivo dessa medida é mapear o percentual de eficiência de testes durante o ciclo de desenvolvimento	**EF = (QDP/QDFT)*100** **QDP** é a quantidade de defeitos encontrados em produção **QDFT** é a quantidade de defeitos encontrados em fase de testes	**EF** representa o percentual de eficiência nos testes. Quanto mais próximo de 100%, mais adequado.	Percentual	Interno
Status das ocorrências	Acompanhar a resolução de defeitos encontrados	Essa medida tem por intuito identificar o número de ocorrências em cada status	**∑SDO** **SDO** representa os status distintos de ocorrências	**SDO** representa os status das ocorrências. Essa medida varia de acordo com a Organização, porém o ideal é que todas as ocorrências sejam resolvidas.	Absoluta	Interno
Tempo de correção de um defeito	Acompanhar a resolução de defeitos encontrados	O objetivo dessa medida é apresentar um índice de manutenibilidade da ferramenta.	**∑HCD** **HCD** é o número de horas utilizadas para correção de defeitos.	**HCD** representa o tempo de correção de um defeito. Essa medida varia de acordo com a	Absoluta	Interno/Externo

Nome da Medida	Descrição/Objetivo	QMEs	Fórmula de Cálculo	Interpretação	Tipo de Escala	Foco da medida
				Organização, porém quanto menor o tempo de correção, mais adequado.		
Quantidade de defeitos abertos X Quantidade de defeitos corrigidos	Acompanhar a resolução de defeitos encontrados	Essa medida mostra a quantidade de defeitos que vão para produção. O objetivo dessa medida é mostrar os problemas que não foram possíveis reproduzir e que foram para produção.	Um gráfico é construído mostrando a quantidade de defeitos abertos e a quantidade de defeitos corrigidos.	Essa medida depende da boa gerência da Organização.	Absoluta	Interno
Ocorrências pendentes de correção	Acompanhar a resolução de defeitos encontrados	Essa medida é baseada no número de ocorrências que ainda estão pendentes de correção pela equipe de desenvolvimento.	**∑NDC** **NDC** é o número de defeitos por corrigir.	**NDC** represente as ocorrências pendentes de correção. Essa medida depende da visão da Organização. De modo geral quanto menos ocorrências com status pendente, mais adequado.	Absoluta	Interno/Externo
Quantidade de defeitos enviados para a próxima iteração	Acompanhar a resolução de defeitos encontrados	O objetivo dessa medida é mapear a quantidade de erros que passam para	Medida baseada na quantidade de erros que não resolvidos na iteração a qual são	**ENRi** representa a quantidade de defeitos enviados para a próxima iteração.	Absoluta	Interno

Nome da Medida	Descrição/Objetivo	QMEs	Fórmula de Cálculo	Interpretação	Tipo de Escala	Foco da medida
		outras iterações.	oriundos. **∑ENRi** **ENRi** são os erros não resolvidos na iteração i.	Essa medida depende da visão da Organização. De modo geral, o mais adequado é que todas as pendências fossem resolvidas na iteração em que são descobertas.		
Percentual de retrabalho	Acompanhar o percentual de retrabalho	O objetivo dessa medida é mapear o percentual de retrabalho	Medida baseada na quantidade de atividades que retornaram ao desenvolvimento **PT = (QAR/QTA)*100** **QAR** é a quantidade de atividades que retornaram **QAT** é a quantidade total de atividades	**PT** representa o percentual de retrabalho. Quanto menor o percentual de retrabalho, mais adequado.	Percentual	Interno
Quantidade de Testes de aceitação por funcionalidade	Avaliar a conformidade do produto com os requisitos de testes	Acompanhar a evolução da quantidade de testes de aceitação produzidos.	**TTAi = TCTi - TCTi-1**. Base de medição: **TTAi** = número de casos de teste de aceitação na iteração i.	Essa medida depende da visão estratégica da Organização.	Absoluta	Interno/Externo

Nome da Medida	Descrição/Objetivo	QMEs	Fórmula de Cálculo	Interpretação	Tipo de Escala	Foco da medida
			TCTi = total de casos de teste de aceitação na iteração i. **TCTi** = total de casos de testes de aceitação na iteração anterior (i-1).			
Porcentagem de assertivas de teste de aceitação passando	Avaliar a conformidade do produto com os requisitos de testes	Verificar qual a porcentagem de assertivas de testes de aceitação que estão passando.	**AP(i)** % = **NAPi/NTA** **NAPi** é o número de assertivas passando na iteração i. **NTA** é o número total de assertivas	**AP** representa o percentual de assertivas de testes de aceitação que passaram. Quanto mais próximo de 100%, mais adequado.	Percentual	Interno/Externo
Porcentagem de assertivas de teste de aceitação falhando	Avaliar a conformidade do produto com os requisitos de testes	Verificar qual a porcentagem de assertivas de testes de aceitação estão falhando	**AF(i)** % = **NAFi/NTA** **NAFi** é o número de assertivas falhando na iteração i. **NTA** é o número total de assertivas	**AF** representa o percentual de assertivas de testes de aceitação que falharam. Quanto mais próximo de 0%, mais adequado.	Percentual	Interno/Externo
Qualidade do teste	Avaliar a qualidade do teste	Verificar a qualidade dos testes executados	**QT = QtdD/(QtdD+QtdP)*100** **QT** representa a qualidade do teste **QtdD** é o número de	**QT** representa a qualidade do teste.	Percentual	Interno

Nome da Medida	Descrição/Objetivo	QMEs	Fórmula de Cálculo	Interpretação	Tipo de Escala	Foco da medida
			defeitos encontrados **QtdP** é número de defeitos encontrados em produção	Quanto mais próximo de 100%, mais adequado.		
Eficácia dos testes	Avaliar a qualidade do teste	Verificar a eficácia dos testes	**EF = t/(t+UAT)** em que: **EF** representa a eficácia dos testes **t** = número total de defeitos reportados durante o teste **UAT** = número total de defeitos que são reportados durante o teste de aceitação de usuário.	**EF** representa a eficácia dos testes. Quanto mais próximo de 1, mais adequado.	Taxa	Interno
Densidade do defeito	Avaliar o impacto do defeito	Verificar a densidade dos defeitos encontrados	**DD = NDE/TD** **NDE** é o número de defeitos encontrados **TD** é o tamanho do defeito	Esse cálculo depende da estimativa do tamanho do defeito. Esse valor varia de acordo com a Organização.	Taxa	Interno
Quantidade de testes unitários por user history	Monitorar a cobertura de testes	O objetivo dessa medida é obter a média de testes unitários por user history	Medida baseada na quantidade de user history dividido pela quantidade de testes unitários. **QTUH = QUH/QTU**	**QTUH** representa a quantidade de testes unitários por user	Taxa	Interno

Nome da Medida	Descrição/Objetivo	QMEs	Fórmula de Cálculo	Interpretação	Tipo de Escala	Foco da medida
			QUH é a quantidade de user history **QTU** é a quantidade de testes unitários	history Quanto mais próximo de 1, mais adequado.		
Quantidade de testes funcionais por user story	Monitorar a cobertura de testes	O objetivo dessa medida é obter a quantidade média de testes funcionais por user history	Medida baseada na quantidade de user history dividido pela quantidade de testes unitários. **QTFUH = QUH/QTF** **QUH** é a quantidade de user history **QTF** é a quantidade de testes funcionais	**QTFUH** representa a quantidade de testes funcionais por user history Quanto mais próximo de 1, mais adequado.	Absoluta	Interno
Percentual de cobertura de testes	Monitorar a cobertura de testes	O objetivo dessa medida é mapear a cobertura de testes por módulo	**PCT = NCTM/NCT** **NCTM** é o número de casos de teste por módulo **NCT** é o número total de casos de teste	**PCT** representa o percentual de cobertura de testes. Quanto mais próximo de 100%, mais adequado.	Percentual	Interno
Mudanças de escopo	Monitorar a estabilidade do escopo dos testes	Essa medida tem por intuito mostrar o número de alterações feitas no escopo de teste e indica a estabilidade ou a	**ME = NIAE/NTI** **NIAE** é o número de itens alterados no escopo **NTI** é o número total de itens.	**ME** representa as mudanças de escopo. Quanto mais próximo de 0%, mais	Percentual	Interno/Externo

Nome da Medida	Descrição/Objetivo	QMEs	Fórmula de Cálculo	Interpretação	Tipo de Escala	Foco da medida
		volatilidade dos requisitos e do processo como um todo.		adequado.		
Tempo médio para encontrar um defeito	Monitorar a execução dos testes	Essa medida tem por intuito mostrar o esforço necessário para encontrar um defeito.	**TMD = HET/QtdD** **TMD** é o tempo médio para encontrar um defeito **HET** é a soma das horas gastas para execução de testes **QtdD** é a quantidade de erros encontrados	**TMD** representa o tempo médio para encontrar um defeito. Quanto mais próximo de zero, mais adequado.	Absoluta	Interno
Cobertura de teste baseada em código	Monitorar a execução dos testes	A medida de cobertura de teste baseada em código é efetuada a partir da quantidade de código executado durante o período de testes em comparação à quantidade total de código pendente de execução.	**Cobertura de teste baseada em código = Ie / Tlic.** **Ie** representa número de itens executados(instruções de código, ramificações de código, caminhos de código, pontos de decisão do estado de dados ou nomes de elementos de dados). **Tlic** é o número total de itens do código.	**Cobertura de teste baseada em código**. Quanto mais próximo de 100% mais adequado.	Percentual	Interno
Efetividade do caso	Monitorar a	Essa medida permite	**EFT = NCD/NCT**	**EFT** representa a		Interno

Nome da Medida	Descrição/Objetivo	QMEs	Fórmula de Cálculo	Interpretação	Tipo de Escala	Foco da medida
de teste	execução dos testes	verificar se os casos de teste encontram defeitos, podendo indicar a utilidade e a estabilidade do software.	**EFT** representa a efetividade do caso de teste **NCD** é o número de casos de teste que resultaram em ocorrências registradas **NCT** é o número total de casos de teste.	efetividade do caso de teste. Quanto mais próximo de 0, mais adequado.	Taxa	
Efetividade e eficiência dos testes	Monitorar a execução dos testes	Essa medida possibilita verificar o quão adequado foram os testes, a habilidade do conjunto de testes em encontrar defeitos, ou seja, a qualidade do conjunto de testes.	**EF = NDT /NDP** **NDT** é o número de defeitos encontrados pelo conjunto de testes **NDP** é o número de defeitos encontrados no produto em produção.	**EF** representa a efetividade do caso de teste. Quanto maior a taxa, mais adequado.	Taxa	Interno
Percentual de falhas na primeira execução dos casos de teste	Monitorar a execução dos testes	Essa medida é usada para determinar a efetividade dos processos de análise e desenvolvimento.	**TFCT = (CTFP/QtdCT)*100** **CTFP** representa os casos de testes que falharam em primeira execução **QtdCT** representa a quantidade total dos casos de teste	**TFCT** representa o percentual de falhas na primeira execução dos casos de teste. Quanto mais próximo de zero, mais adequado.	Percentual	Interno
Ocorrências resolvidas que ainda	Monitorar a execução dos	Essa medida tem por objetivo mostrar o	**∑DCNT** **DCNT** é o número de	**DCNT** representa as ocorrências resolvidas		Interno

Nome da Medida	Descrição/Objetivo	QMEs	Fórmula de Cálculo	Interpretação	Tipo de Escala	Foco da medida
não foram retestadas	testes	número de defeitos que foram corrigidos e ainda não foram testados pela equipe de testes.	ocorrências que foram resolvidas e que ainda não foram testadas.	que ainda não foram retestadas. A interpretação dessa medida depende da boa gerência da Organização.	Absoluta	
Fase em que o defeito foi encontrado	Monitorar a execução dos testes	Essa medida tem por intuito mostrar se o defeito foi encontrado na fase correta.	**PDF = NDF/NDT** **NDF** é o número de defeitos encontrados em cada fase **NDT** é o número total de defeitos encontrados.	**PDF** representa a fase em que o defeito foi encontrado. Quanto mais próximo de 0, mais adequado.	Taxa	Interno
Tempo de teste estimado X Tempo de teste utilizado	Monitorar a execução dos testes	Essa medida tem por intuito mostrar se os testes foram planejados corretamente, ou seja, fornece indicadores da performance do planejamento dos testes.	Um gráfico é construído mostrando a quantidade de tempo estimada e a quantidade de tempo de testes utilizada	Quanto menor a diferença entre os valores dos tempos melhor.	Gráfico	Interno
Fator de segurança	Monitorar a execução dos testes	O fator de segurança é uma medida que mede a eficácia de uma estimativa efetuada.	O fator de segurança pode ser calculado a partir da relação entre o tempo gasto nos testes e o tempo estimado.	**FS** representa o fator de segurança. Quanto mais próximo de 1, mais adequado.	Taxa	Interno

Nome da Medida	Descrição/Objetivo	QMEs	Fórmula de Cálculo	Interpretação	Tipo de Escala	Foco da medida
			FS = TGT/TTE **TGT** é o tempo gasto nos testes **TTE** é o tempo estimado para testes			
Taxa de esforço de teste	Monitorar a execução dos testes	O objetivo da captura dessa medida é obter indicações no nível de investimentos necessários para os testes.	O cálculo é efetuado a partir da divisão do esforço total do teste pelo esforço total do projeto. **TET = ETE/ETP** **ETE** é o esforço total do teste **ETP** é o esforço total do projeto	**TET** representa a taxa de esforço de teste. Essa interpretação depende da Organização.	Taxa	Interno
Testes de aceitação que passaram	Monitorar a execução dos testes	O objetivo dessa medida é mostrar o grau de aceitação do produto.	Medida baseada na quantidade de testes de aceitação divididos pela quantidade de testes de aceitação que passaram. **TAP = (TAP/TA)*100** **TAP** é a quantidade de testes de aceitação que passaram **TA** é a quantidade total de testes de aceitação	**TAP** representa os testes de aceitação que passaram. Quanto mais próximo de 100%, mais adequado.	Percentual	Interno/Externo
Número de casos de	Monitorar a	O objetivo dessa	**∑CDT**	**CDT** representa o		Interno

Nome da Medida	Descrição/Objetivo	QMEs	Fórmula de Cálculo	Interpretação	Tipo de Escala	Foco da medida
teste	execução dos testes	medida é mapear a quantidade de casos de teste	**CDT** significa casos de teste. Essa medida fornece a quantidade total de casos de teste.	número de casos de teste. Essa interpretação depende da Organização.	Absoluta	
Número de casos de teste executados	Monitorar a execução dos testes	O objetivo dessa medida é mapear a quantidade de casos de teste executados	**∑CDTE** **CDTE** significa casos de teste que foram executados. Essa medida fornece a quantidade total de casos de teste que foram executados.	**CDTE** representa o número de casos de teste executados. Essa interpretação depende da Organização.	Absoluta	Interno
Número de casos de teste que passaram	Monitorar a execução dos testes	O objetivo dessa medida é mapear a quantidade de casos de teste que passaram	**∑CDTP** **CDTP** significa casos de teste que passaram. Essa medida fornece a quantidade total de casos de teste que passaram.	**CDTP** representa o número de casos de teste que passaram. Essa interpretação depende da Organização.	Absoluta	Interno
Número de casos de teste falharam	Monitorar a execução dos testes	O objetivo dessa medida é mapear a quantidade de casos de teste que falharam	**∑CDTF** **CDTF** significa casos de teste que falharam. Essa medida fornece a quantidade total de casos de teste que falharam.	**CDTF** representa o número de casos de teste que passaram. Essa interpretação depende da Organização.	Absoluta	Interno
Número de casos de	Monitorar a	O objetivo dessa	**∑CDTI**	**CDTI** representa o		Interno

Nome da Medida	Descrição/Objetivo	QMEs	Fórmula de Cálculo	Interpretação	Tipo de Escala	Foco da medida
teste que estão sob investigação	execução dos testes	medida é mapear a quantidade de casos de teste que estão sob investigação	**CDTI** significa casos de teste que estão sob investigação. Essa medida fornece a quantidade total de casos de teste que estão sob investigação.	número de casos de teste que estão sob investigação. Essa interpretação depende da Organização.	Absoluta	
Número de casos de teste que estão bloqueados	Monitorar a execução dos testes	O objetivo dessa medida é mapear a quantidade de casos de teste que estão bloqueados	**∑CDTB** **CDTB** significa casos de teste que estão bloqueados. Essa medida fornece a quantidade total de casos de teste que estão bloqueados	**CDTB** representa o número de casos de teste que estão bloqueados. Essa interpretação depende da Organização.	Absoluta	Interno
Número de casos de teste que foram reexecutados	Monitorar a execução dos testes	O objetivo dessa medida é mapear a quantidade de casos de teste que foram reexecutados	**∑CDTR** **CDTR** significa casos de teste que foram reexecutados Essa medida fornece a quantidade total de casos de teste que foram reexecutados.	**CDTR** representa o número de casos de teste que foram reexecutados. Essa interpretação depende da Organização.	Absoluta	Interno
Número de casos de teste que falharam na primeira execução	Monitorar a execução dos testes	O objetivo dessa medida é mapear a quantidade de casos de teste que falharam em primeira execução	**∑CDTFP** **CDTFP** significa casos de teste que falharam em primeira execução	**CDTFP** representa o número de casos de teste que falharam em primeira execução.	Absoluta	Interno

Nome da Medida	Descrição/Objetivo	QMEs	Fórmula de Cálculo	Interpretação	Tipo de Escala	Foco da medida
			Essa medida fornece a quantidade total de casos de teste que falharam em primeira execução.	Essa interpretação depende da Organização.		
Número total de execuções	Monitorar a execução dos testes	O objetivo dessa medida é mapear a quantidade de execuções	**∑E** **E** significa execução de testes Essa medida fornece a quantidade total de execuções de testes	**E** representa o total de execução de testes Essa interpretação depende da Organização.	Absoluta	Interno
Tempo de execução por caso de teste	Monitorar a execução dos testes	O objetivo dessa medida é mapear o tempo de execução por caso de teste	Medida baseada no tempo de execução por caso de teste **TECT** é tempo de execução de cada caso de teste.	**TECT** representa o tempo de execução por caso de teste. Essa interpretação depende da Organização.	Absoluta	Interno
Tempo de execução do teste	Monitorar a execução dos testes	O objetivo dessa medida é mapear o tempo total de execução dos testes	Medida baseada no tempo total de execução dos testes **∑TECT** **TECT** é o tempo de execução de cada caso de teste	**TECT** representa o tempo de execução do teste. Essa interpretação depende da Organização.	Absoluta	Interno
Percentual de casos de teste de teste que passaram	Monitorar a execução dos testes	O objetivo dessa medida é mapear o percentual de casos de testes que passaram	**PCT = (NCTM/NCT)*100** **NCTP** é o número de casos de teste que	**PCT** representa o percentual de casos de teste que passaram.	Percentual	Interno

Nome da Medida	Descrição/Objetivo	QMEs	Fórmula de Cálculo	Interpretação	Tipo de Escala	Foco da medida
			passaram **NCT** é o número total de casos de teste	Quanto mais próximo de 100%, mais adequado.		
Percentual de casos de teste que foram bloqueados	Monitorar a execução dos testes	O objetivo dessa medida é mapear o percentual de casos de teste que foram bloqueados	**PCT = (NCTM/NCT)*100** **NCTB** é o número de casos de teste bloqueados **NCT** é o número total de casos de teste	**PCT** representa o percentual de casos de teste que foram bloqueados. Quanto mais próximo de 0%, mais adequado.	Percentual	Interno
Taxa de falhas em primeira execução	Monitorar a execução dos testes	O objetivo dessa medida é mapear a taxa de falhas em primeira execução	**PCT = (NCTM/NCT)*100** **NCTFP** é o número de casos de teste que falharam em primeira execução **NCT** é o número total de casos de teste	**PCT** representa o percentual de casos de teste que falharam em primeira execução. Quanto mais próximo de 0%, mais adequado.	Percentual	Interno
Percentual de falhas em geral	Monitorar a execução dos testes	O objetivo dessa medida é mapear a taxa de falhas em geral	**TFG = NCTF/QCD** **NCTF** é a quantidade de casos de teste que falharam **QCD** é a quantidade de casos de teste.	**TFG** representa o percentual de falhas em geral. Quanto mais próximo de 0%, mais adequado.	Percentual	Interno
Percentual de descoberta de defeitos	Monitorar a execução dos testes	O objetivo dessa medida é mapear a taxa de descoberta de	O cálculo dessa medida pode ser efetuado dividindo a	Quanto mais próximo de 100%, mais adequado.		Interno

Nome da Medida	Descrição/Objetivo	QMEs	Fórmula de Cálculo	Interpretação	Tipo de Escala	Foco da medida
		defeitos.	quantidade de defeitos encontrados durante a fase de testes com a quantidade de defeitos encontrados.		Percentual	
Percentual de assertivas de Teste de Unidade que passaram	Monitorar a execução dos testes	Verificar qual a porcentagem de assertivas dos casos de unidade que estão passando	A porcentagem de assertivas passando por iteração(i) será dada por: Assertivas passando(i) % = Número de assertivas passando(i)/Número total de assertivas	Quanto mais próximo de 100%, mais adequado.	Percentual	Interno
Percentual de assertivas de Teste de Unidade que falharam	Monitorar a execução dos testes	Verificar qual a porcentagem de assertivas dos casos de unidade que estão falhando	A porcentagem de assertivas falhando por iteração(i) será dada por: Assertivas passando(i) % = Número de assertivas falhando(i)/Número total de assertivas	Quanto mais próximo de 0%, mais adequado.	Percentual	Interno
Densidade de defeitos encontrados	Monitorar funcionalidades críticas do produto	Essa medida possibilita identificar a funcionalidade/unidade do software mais problemática.	**∑NDF** **NDF** é a quantidade de defeitos encontrados em cada unidade/funcionalidade do software.	**NDF** representa a densidade dos defeitos encontrados. Quanto mais próximo de zero, mais adequado.	Absoluta	Interno
Running Tested	Monitorar	Definição a partir de	**∑RTF**	**RTF** são features		Interno/Ex

Nome da Medida	Descrição/Objetivo	QMEs	Fórmula de Cálculo	Interpretação	Tipo de Escala	Foco da medida
Features	funcionalidades entregues com testes	funcionalidades entregues em um projeto	**RTF** são features testadas que foram entregues em um projeto	testadas que foram entregues em um projeto. Essa interpretação depende da visão da Organização.	Absoluta	terno
Número de cenários de testes	Monitorar o planejamento dos testes	O objetivo dessa medida é mapear os cenários de testes	Essa medida tem por intuito mapear os cenários de teste. **∑CT** **CT** significa cenário de teste	**CT** representa os cenários de teste. Essa interpretação depende da visão da Organização.	Absoluta	Interno
Tempo necessário para elaboração de casos de testes por pessoa/dias	Monitorar o planejamento dos testes	O objetivo dessa medida é obter um mapeamento do tempo gasto para elaboração de casos de teste.	**∑NDP** **NDP** significa o número de dias utilizados para elaboração de casos de teste.	**NDP** representa o número de dias utilizados para elaboração de casos de teste. Essa interpretação depende da visão da Organização.	Absoluta	Interno
Número de ocorrências	Monitorar os defeitos do produto	Número total de ocorrências encontradas em um determinado período ou fase de teste.	**∑OPD** **ODP** representa as ocorrências em um determinado período	**ODP** representa o número de ocorrências encontradas no produto em produção. Quanto mais próximo	Absoluta	Interno

Nome da Medida	Descrição/Objetivo	QMEs	Fórmula de Cálculo	Interpretação	Tipo de Escala	Foco da medida
				de 0, mais adequado.		
Defeitos por módulo	Monitorar os defeitos do produto	Quantidade de defeitos por módulo	**∑DM** **DM** são os defeitos encontrados por módulo	**DM** representa os defeitos por módulo. Quanto mais próximo de 0, mais adequado.	Absoluta	Interno
Defeitos por tecnologia utilizada	Monitorar os defeitos do produto	Quantidade de defeitos por tecnologia utilizada.	**∑DT** **DT** são os defeitos oriundos da tecnologia utilizada	**DT** representa os defeitos oriundos da tecnologia utilizada. Quanto mais próximo de 0, mais adequado.	Absoluta	Interno
Defeitos por unidade organizacional	Monitorar os defeitos do produto	Quantidade de defeitos encontrados por clientes em produção	**∑DU** **DU** são os defeitos oriundos de clientes de produção	A quantidade ideal de defeitos por unidade organizacional é zero. Porém, a Organização pode definir metas até atingir o ponto ideal.	Absoluta	Interno/Externo
Defeitos por funcionário	Monitorar os defeitos do produto	Quantidade de defeitos gerados por funcionário	**∑DF** **DF** são os defeitos por funcionário	A quantidade ideal de defeitos por funcionário é zero. Porém a Organização pode definir metas até atingir o ideal.	Absoluta	Interno
Percentual de aceitação	Monitorar os testes de aceitação	Essa medida define o grau de aceitação do produto	**PA = QTAP/QTA * 100** **QTAP** é quantidade de testes de aceitação que passaram	O percentual ideal para os testes de aceitação é 100%. Porém a Organização pode definir metas até	Percentual	Interno/Externo

Nome da Medida	Descrição/Objetivo	QMEs	Fórmula de Cálculo	Interpretação	Tipo de Escala	Foco da medida
			QTA é quantidade de testes de aceitação	atingir esse percentual.		