SERVIÇO NACIONAL DE APRENDIZAGEM INDUSTRIAL

FACULDADE DE TECNOLOGIA SENAI/SC FLORIANÓPOLIS

CURSO TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

HELDER BRANDÃO VIETRO

TESTES DE SOFTWARE:

PROCESSOS DE TESTE E A NORMA ISO-15504

Florianópolis (SC)

2013

HÉLDER BRANDÃO VIETRO

TESTES DE SOFTWARE:

PROCESSOS DE TESTE E A NORMA ISO-15504

Trabalho de Conclusão de Curso apresentado à banca examinadora do Curso de Tecnologia em Análise e Desenvolvimento de Sistemas da Faculdade de Tecnologia do SENAI/SC Florianópolis como requisito parcial para obtenção do Grau de Tecnólogo em Análise de Sistemas sob a orientação do Professor João Carlos Testi Ferreira.

FLORIANÓPOLIS (SC)

2013

HÉLDER BRANDÃO VIETRO

TESTES DE SOFTWARE:

PROCESSOS DE TESTE E A NORMA ISO-15504

Trabalho de Conclusão de Curso apresentado à Banca Examinadora do Curso de Tecnologia em Análise e Desenvolvimento de Sistemas da Faculdade de Tecnologia SENAI Florianópolis em cumprimento a requisito parcial para obtenção do título de Tecnólogo/Especialista em Análise de Sistemas.

APROVADA PELA **COMISSÃO EXAMINADORA**

EM FLORIANÓPOLIS, ____ DE _____ DE _____

Prof. Nome Completo do(a) Coordenador(a), Título abreviado,
(SENAI/SC) - Coordenador(a) do Curso

Prof. Nome Completo do(a) Coordenador(a), Título abreviado,
(SENAI/SC) - Coordenador(a) de TCC

Prof. Nome Completo do(a) Professor(a), Título abreviado,
(SENAI/SC) - Orientador(a)

Prof. Nome Completo do(a) Professor(a), Título abreviado,
(SENAI/SC) - Examinador

Prof. Nome Completo do(a) Professor(a), Título abreviado,
(SENAI/SC) - Examinador

AGRADECIMENTOS

Agradeço meus pais, Araci e Celso, meus irmãos, Vinícius e Ainá, meu afilhado Pedro Henrique e minha noiva Samira, pelo apoio dado em todos os momentos desta graduação.

“The world is full of kings and queens, who blind your eyes and steal your dreams. It´s Heaven and Hell” (Ronnie James Dio **m**/)

VIETRO, Hélder Brandão. **Testes de software:** processo de testes e a norma ISO- 15504**.** Florianópolis, 2013. 48f. Trabalho de Conclusão de Curso (Graduação Tecnológica) - Curso de Tecnologia em Análise e Desenvolvimento de Sistemas. Faculdade de Tecnologia do SENAI, Florianópolis, 2013.

RESUMO

Normalmente as atividades de teste de software são utilizadas com o intuito de validar a correta implementação dos requisitos de um sistema. Porém, uma importante função dos testes é a de medir a qualidade e agregar valor a um sistema de software. Com a maturidade das equipes de teste, é possível começar a organizar suas práticas em processos específicos de teste, com planejamento e atendimento dos objetivos do negócio que está sendo explorado. O objetivo deste trabalho é evidenciar de que maneira a norma ISO-15504, uma das bases para a criação do MPS.BR, fornece suporte suficiente para se avaliar um processo de testes e melhorá-lo. Para a exploração deste tema, foram feitas pesquisas bibliográficas com objetivos explicativos e descritivos. A análise efetuada mostrou que, em seus seis níveis de maturidade, a norma aborda práticas e estrutura atividades que permitem a melhoria do processo de testes de software.

Palavras-chave: Testes. Processos. Qualidade. Melhoria de processos.

VIETRO, Hélder Brandão. **Testes de software:** processo de testes e a norma ISO- 15504**.** Florianópolis, 2013. 48f. Trabalho de Conclusão de Curso (Graduação Tecnológica) - Curso de Tecnologia em Análise e Desenvolvimento de Sistemas. Faculdade de Tecnologia do SENAI, Florianópolis, 2013.

ABSTRACT

Software testing activities exists to validate the correct implementation of system requirements. However, the importance of software testing is to measure the quality and add value to software systems. With the maturity of the testing teams, it is possible to organize specific test procedures, by planning and meeting business objectives. The objective of this work is to show how the ISO-15504 standard, one of the bases for the creation of MPS.BR, provides sufficient evaluation bases for supporting and improving software testing process. To explore this theme, literature searches were performed using descriptive and explanatory purposes. The analysis performed showed that, in its six levels of maturity, the standard practices and structure activities provides enough bases for the improvement of the software testing process.

Key words: Tests, Processes, Quality, Process Improvement.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ABNT – Associação Brasileira de Normas Técnicas

ASQC – American Society for Quality Control

CMMI – Capability Maturity Model – Integration

IEEE – Institute of Electrical and Electronics Engineering

ISO – International Organization for Standardization

MCP – Melhoria Contínua do Processo

MCTI – Ministério da Ciência, Tecnologia e Inovação

MPS.BR – Melhoria do Processo de Software Brasileiro

OTAN – Organização do Tratado do Atlântico Norte

SEI – Software Engineering Institute

SOFTEX – Associação para Promoção da Excelência do Software Brasileiro

TAP – Testing Assessment Program

TI – Tecnologia da Informação

TIM – Testing Improvement Model

TMM – Testing Maturity Model

TPI – Test Process Improvement

TSM – Testability Support Model

SUMÁRIO

1. INTRODUÇÃO

Desenvolver sistemas de software com qualidade é um dos principais desafios da contemporaneidade. Isso porque de uma maneira ou de outra, sistemas de software estão presentes em todos os lugares do mundo, até onde não se vê.[1] O impacto dos sistemas de computação, atualmente, ocorre em diferentes setores da sociedade, como por exemplo economia, segurança (pública e privada) e saúde.

"Qualidade" é um conceito intuitivo[2], de fácil compreensão, mas de difícil explicação, construído sobre noções específicas e complexas. No que diz respeito à história dos produtos gerados pelo trabalho humano, é notado, desde o advento da revolução industrial[3], que a concorrência entre as empresas desencadeou ações para que o produto gerado fosse gradativamente tendo suas características aprimoradas.

No assunto "sistemas de software", é visível como estas "ações" – ou processos – por vezes, se tornam particulares de cada organização que explora o mercado de Tecnologia da Informação. O processo de desenvolvimento de sistemas é algo complexo e muito particular de cada empresa, e para evitar desequilíbrio entre as diferentes concepções sobre processos de desenvolvimento de software, instituições tornaram-se responsáveis por criar e padronizar modelos de processos de desenvolvimento de software e qualidade: American Society for Quality Control (ASQC), Associação para Promoção da Excelência do Software Brasileiro (SOFTEX), Associação Brasileira de Normas Técnicas (ABNT), International Organization for Standardization (ISO), Software Engineering Institute (SEI), etc.[4]

Dentro da perspectiva dos processos de desenvolvimento, ou da "Garantia de Qualidade", muitos são os modelos de processos definidos, com seus níveis de maturidade e organização: processos de desenvolvimento, processos de qualidade, etc. Um ponto muito importante e constantemente desprezado pelas empresas de desenvolvimento de sistemas computacionais é o de adotar um processo bem definido de testes. Por motivos de cronograma

---

[1] NAIK, Kshirasagar. Software Testing and Quality Assurance: Theory and Practice. Waterloo: Wiley, 2008.

[2] KOSCIANSKI, André. Qualidade de Software. São Paulo: Novatec Editora, 2012. p. 17.

[3] ibidem, p.18.

[4] ibidem, p.19.

(muitas entregas em pouco tempo), economia dos projetos (investimento maior em implementadores) ou até mesmo, e principalmente, falta de mão de obra especializada, os processos de teste acabam sendo englobados pelas etapas do desenvolvimento e é comum exemplos de cortes no cronograma dos testes ou da diminuição da importância dos testes em um determinado momento do projeto de um sistema.

Os testes de um sistema em criação, ou de sistemas legado que outrora eram feitos somente pelos codificadores, podem ser bem mais explorados por empresas que nutrem o desejo de alcançar garantia de qualidade certificada ou concorrer a licitações governamentais e novos projetos. Ter processo de testes bem definido significa ter verificações acontecendo desde o início, nas fases iniciais do projeto – design, onde as modelagens / especificações de regras iniciais estão sendo executadas – até o momento de entregas, com testes automatizados, testes unitários, testes exploratório, etc. e principalmente planejamento e especificação dos testes que acontecerão ao longo de todo o projeto de desenvolvimento.

Assim como acontece com o desenvolvimento de software, os testes de software também podem ser encarados como um processo válido dentro de um projeto de desenvolvimento de sistemas, produzindo artefatos e contribuindo para o controle e garantia de qualidade do sistema a ser desenvolvido.

O presente trabalho pretende explorar a temática do estudo dos processos de qualidade, em específico dos processos de testes, e como a adoção destes pode impactar de maneira positiva na melhoria contínua dos processos de qualidade; a chamada garantia de qualidade.

A melhoria do processo de testes será abordada com base na norma ISO-15504, de modo a se entender os testes como processos interdependentes ao processo de desenvolvimento, com planejamento, execução de atividades e papéis bem definidos. Esta abordagem permite, em última análise, gerar artefatos e indicadores suficientes, de modo a utilizá-los na análise da eficácia do próprio processo de testes e na crítica do processo, definindo pontos a serem alterados com objetivo da Melhoria Contínua do Processo (MCP).

1.1 Justificativa

O presente trabalho surge a partir de experiências observadas no dia a dia de uma equipe de testes. Notou-se que a criação da equipe em questão se deu de forma não planejada, com processos pouco definidos e heterogêneos, onde cada componente da equipe executa, reativamente, uma gama muito grande de atividades não definidas, quando comparado com outras equipes da mesma organização / unidade de negócio.

Na empresa mencionada, os processos de desenvolvimento, testes, suporte, etc., de um modo geral, estão ainda em construção, sendo que no caso específico da equipe de testes, ainda existe muita improvisação em sua atuação. Nesse sentido, entender as consequências das ações dos testadores/analistas de teste ainda é uma atividade complexa e que apresenta vários obstáculos. O próprio sistema que gerencia os defeitos e novas atividades não foi construído com a intenção de fornecer dados necessários a esta tarefa. Neste sistema, constantemente são feitas alterações com o objetivo de melhorar relatórios para tomadas de decisão sobre os processos supracitados.

A criação de uma equipe de testes mostra a necessidade de alcançar qualidade no produto gerado pela empresa. Porém, entende-se que a criação de processos definidos de testes deve ser acompanhada de medidas permitam o aperfeiçoamento deste processo, à medida que este se desenvolve e vai gerando artefatos, sendo esta a razão principal pela escolha do tema em questão.

1.2 Objetivo Geral

Mostrar a importância da utilização dos processos de testes e como a avaliação de sua execução permite alcançar a melhoria do processo**.**

1.3 Objetivos Específicos

a) Diferenciar “garantia de qualidade” do conceito “controle de qualidade”

b) Relacionar processos de teste com processos de qualidade

c) Elencar artefatos do processo de teste

d) Avaliar processos de teste de acordo com norma ISO-15504

1.4 Metodologia do Trabalho

Para a proposta adotada no presente trabalho, executou-se uma pesquisa / revisão bibliográfica, com discussão dos conceitos apresentados. Nesse sentido, a pesquisa apresenta objetivos descritivos (os processos do desenvolvimento de software e suas comparações) e explicativos (conceitos, como por exemplo qualidade, testes, melhoria de processo, etc.).

Assim, foi utilizada intensa pesquisa bibliográfica sobre o tema e como ele é discutido pelos mais diferentes autores, confrontando ideias e, se possível, exemplificando com dados reais para compor uma discussão mais próxima do que pode ser entendido como “mundo real”.

2. REVISÃO DE LITERATURA

Na revisão literária serão discutidos conceitos básicos relativos à garantia de qualidade. Os assuntos aqui abordados mostrarão de que maneira os testes de software podem ser analisados no âmbito da qualidade de software, iniciando por conceituação e relação entre qualidade e software e passando por uma rápida discussão sobre processos de desenvolvimento de software.

2.1 Qualidade

Talvez um empreendimento tão antigo quanto a vontade humana de construir grandes artefatos de massa, a busca por qualidade certamente não surgiu com o software e nem é um fenômeno restrito à atualidade[5]. Dentre os mais diferentes significados encontrados para o termo, encontra-se no dicionário o seguinte resultado: "Propriedade, atributo ou condição das coisas ou das pessoas capaz de distingui-las das outras e de lhes determinar a natureza."[6].

"Qualidade" define, então, as características de um artefato produzido pelo homem e que do seu trabalho é fruto direto ou indireto. É a qualidade que, a partir de uma escala de valores, permitirá avaliar, aprovar, aceitar ou recusar qualquer coisa. Porém, são necessários critérios para julgar quais são as características positivas de um produto, a fim de considerá-lo como sendo de qualidade. Em alguns produtos, tais critérios são simples de identificar. Conforme Koscianski:

> Em domínios como engenharia elétrica ou mecânica, as informações necessárias [para medição da qualidade] são obtidas em função da finalidade a que se destina um determinado produto. Para dispositivos simples, como um fusível ou uma engrenagem, não é difícil enumerar algumas características que provavelmente são relevantes: ponto de fusão, condutância térmica, resistência a cisalhamento ou dimensões físicas. Passando para objetos mais complexos, como um transistor ou amortecedor, a complexidade e a quantidade de requisitos tendem a aumentar[7].

[5] NAIK, op.cit., p.1.
[6] HOLANDA, Aurélio Buarque de. Novo Dicionário da Língua Portuguesa. Rio de Janeiro: Editora Nova Fronteira, s/d. p.1175.
[7] KOSCIANSKI, op.cit., p.25 et seq.

Conforme explicitado, analisando produtos mais complexos, a quantidade de requisitos utilizados para avaliar a qualidade tende a aumentar. Utilizando um computador pessoal como exemplo, nota-se que existe um sistema – o produto "computador" – composto por diferentes subsistemas: disco rígido, placa mãe, memória, processador, etc. Neste caso, cada subsistema possui sua própria série de especificações, e é comum concluir o nível de qualidade apenas pela análise das especificações de um produto, ou como definiu Crosby[8] "A qualidade é conformidade aos requisitos".

2.2 Os Fatores de Qualidade de Mccall

Em sistemas de software, o assunto qualidade tem sido amplamente discutido principalmente a partir do final da década de 1960. O próprio termo "Engenharia de software" provavelmente foi utilizado pela primeira vez na Alemanha, no ano de 1968, em uma conferência do Comitê de Ciência da Organização do Tratado do Atlântico Norte (OTAN). O relatório desta e de outras conferências da mesma época[9] apontam os principais problemas enfrentados na construção e utilização de software:

- Cronogramas não observados;
- Projetos com tantas dificuldades que são abandonados;
- Módulos que não operam corretamente quando combinados;
- Programas que não fazem exatamente o que era esperado;
- Programas descartados de tão difíceis de usar;
- Programas que simplesmente param de funcionar.

[8] CROSBY PB. Quality is Free: the Art of Making Quality Certain. Denver: Mentor Book, 1992.
[9] KOSCIANSKI, op.cit., p.21.

Foi a partir da metade da década de 1970 que estudos foram criados para entender a qualidade de software em termos de quantidade mensurável. Neste sentido, um dos principais estudos existentes é o de Jim McCall, intitulado “fatores na qualidade de software - conceitos e definições da qualidade de software”[10].

O referido artigo foi um estudo completo sobre “qualidade de software”, as relações com os requisitos especificados para o sistema em desenvolvimento e a medição de como estes requisitos estão sendo satisfeitos à medida que o sistema vai evoluindo. Em suas palavras:

> Enquanto o funcionamento, custo e cronogramas de desenvolvimento podem ser objetivamente definidos, medidos e avaliados no momento do desenvolvimento do sistema, a qualidade desejada foi sendo definida, historicamente, apenas em termos subjetivos. Isso ocorre principalmente devido ao fato de o Escritório de Programação de Sistemas (SPO) não ter critérios quantificáveis para julgar a qualidade do software até que o mesmo comece a usar o sistema já em condições de operação[11].

Neste sentido, julgar a qualidade de um sistema computacional era possível apenas após o sistema estar “pronto”, ou conforme o texto: “em condições de operação”. A idéia principal, perseguida por McCall, era quantificar a qualidade do software em desenvolvimento utilizando os chamados “fatores de qualidade”.

Conforme McCall, fator de qualidade significa:

> Uma condição ou característica que contribui ativamente para a qualidade software. Para propósitos de padronização, todos os fatores serão relacionados com um custo normalizado para: ou executar uma atividade caracterizada com o fator, ou operar com aquele grau de qualidade. Por exemplo, manutenibilidade é um esforço que requer localizar e corrigir um erro em um programa em operação. Este esforço requerido pode ser demonstrado em unidades como “tempo”, “dólares” ou “mão de obra”. As seguintes regras foram usadas para determinar o conjunto inicial de fatores de qualidade:
>
> - Uma condição ou característica que contribui para a qualidade do software;

[10] MCCALL, Jim A. Factors In Software Quality: Concept and Definitions of Software Quality. Nova Iorque: General Electric, 1977.
[11] O artigo foi fruto de uma tarefa solicitada pela Força Aérea Estadunidense para definir padrões e guias técnicos para os gerentes de desenvolvimento de software.

- Uma característica relacionada ao usuário;
- Custo para executar uma atividade caracterizada pela função, ou para operar com tal grau de qualidade[12].

Assim, McCall elencou os onze fatores e os organizou baseado em três tipos de atividades[13]:

1 - Operação do produto:

- Correção: Medida na qual um programa satisfaz suas especificações e cumpre com os objetivos do usuário.
- Confiança: Medida na qual um programa executa com precisão o que é esperado.
- Eficiência: A quantidade de recursos computacionais e de codificação necessárias para um programa executar uma função.
- Integridade: Medida na qual se controla o acesso ao software ou aos dados por pessoas não autorizadas.
- Usabilidade: Esforço necessário para aprender, operar, preparar as entradas e interpretar as saídas de um programa.

2 - Revisão do produto:

- Manutenibilidade: Esforço necessário para localizar e corrigir erros em um programa em operação.
- Testabilidade: esforço necessário para testar um programa e assegurar que ele executa suas funções especificadas.
- Flexibilidade: Esforço necessário para modificar um programa em operação.

3 - Transição do produto:

- Portabilidade: esforço necessário para transferir um programa de uma configuração de hardware e/ou ambiente de sistema para outro.

[12] MCCALL, Jim A., op.cit., p.2-1.
[13] Ibidem, p.3-1 et.seq.

- Reusabilidade: Medida na qual um programa pode ser usado em outras aplicações - relacionada com os pacotes e escopo das funções que o programa executa.
- Interoperabilidade: Esforço necessário para acoplar um sistema em outro.

Porém, para alcançar os referidos fatores de qualidade, McCall definiu critérios passíveis de serem mensuráveis. O nível de implementação de um fator de qualidade no software é, então, julgado ou definido pelos critérios ou atributos do software. Neste sentido, critérios são orientados ao software, enquanto os fatores são orientados ao usuário, e um critério pode afetar mais de um fator.[14].

A tabela 1 relaciona alguns fatores de qualidade com alguns critérios. Um critério pode impactar em mais de um fator de qualidade, e o aprimoramento de um fator pode degradar ou melhorar outro fator. Exemplo: ao aprimorar um fator de testabilidade se degrada a eficiência do sistema, pois mais códigos serão enxertados no software; assim como um esforço para melhorar a exatidão pode aprimorar a confiança[15].

Tabela 1 – Relação entre critérios de qualidade e os fatores de qualidade

Fatores de qualidade	Critérios de qualidade
Correção	Rastreabilidade
	Completude
	Consistência
Confiabilidade	Consistência
	Precisão
	Tolerância a erros
	Simplicidade
Eficiência	Eficiência na execução
	Eficiência no armazenamento
Integridade	Controle de acesso
	Auditoria de acesso
Usabilidade	Operabilidade
	Treinamento
	Comunicabilidade
Manutenibilidade	Consistência
	Simplicidade
	Concisão
	Autodescrição
	Modularidade

[14] Ibidem, p.2-2.
[15] Ibidem, p.527.

Testabilidade	Simplicidade
	Instrumentação
	Autodescrição
	Modularidade
Portabilidade	Autodescrição
	Modularidade
	Independência de software
	Independência de máquina

Fonte: NAIK, Kshirasagar. Software Testing and Quality Assurance: Theory and Practice. Waterloo: Wiley, 2008 p.527.

Como explicado anteriormente, a qualidade foi "quantificada" através dos fatores e critérios de qualidade. Uma questão importante, e básica para a discussão aqui apresentada, é a seguinte: "Qual seria a necessidade de medir qualidade de um sistema de software?"

Para Koscianski[16], os métodos e ferramentas de engenharia de software servem para obter o objetivo de ter qualidade nos programas, mas sem definir qual seria precisamente este objetivo (qual fator / atributos de qualidade seguir), o uso do arsenal de engenharia de software pode se revelar menos eficaz.

De uma maneira mais prática, medir qualidade permite criar uma visão quantitativa do conceito de qualidade. Por exemplo, medir qualidade permite dar embasamento ao desenvolvedor sobre o conceito de qualidade, criando noções mais específicas do que se espera de suas entregas.

Do ponto de vista do produto, uma das mais importantes finalidades de se medir qualidade, com certeza, é a de se identificar o ponto do produto gerado que deve ser melhorado, ou em outras palavras, "o nível de qualidade de um produto deve ser avaliado para que a necessidade de melhorias possa ser investigada" [17].

A importância de medir qualidade pode também ser compreendida a partir dos processos de qualidade. Do ponto de vista gerencial, o investimento em processos deve ser feito com base em objetivos esperados, ou seja, empresas que investem em melhoria de processos devem ter noção do quanto a qualidade foi aprimorada para decidirem sobre o nível dos futuros investimentos em melhoria dos processos.

[16] KOSCIANSKI, op.cit., p.24.
[17] NAIK, op.cit., p.521

2.3 Qualidade e Processos de Desenvolvimento

"A noção de processos possui importante papel no desenvolvimento de software. Sem processos repetíveis, o único resultado a se repetir seria o de produzir erros"[18]. Esta afirmação aponta uma realidade cada vez mais comum nas empresas que desenvolvem sistemas de software: a necessidade de se adotar processos de desenvolvimento e processos de garantia qualidade reside, muitas vezes, na própria sobrevivência da empresa frente ao mercado. Num ambiente cada vez mais hostil e competitivo, produzir software em pouco tempo e com qualidade é sim uma questão de sobrevivência[19].

A adoção de processos de desenvolvimento permite uma divisão cada vez maior das tarefas necessárias para a entrega do produto final, onde:

> no contexto da Engenharia de Software, um processo compreende um conjunto de atividades que são executadas para o desenvolvimento de produtos de software. As atividades são expressas na forma de métodos, técnicas, estratégias, procedimentos e práticas. Tais atividades se baseiam em repositórios de informações como documentos padronizados e políticas da empresa. Processos diferentes são conduzidos por metas diferentes e disponibilidade de recursos[20].

Nesse sentido,

> implantar um processo de qualidade de software é estabelecer um processo que garanta e gerencie o nível de qualidade do produto e do processo de desenvolvimento. [...] Fabricar softwares não adequados [...] aumenta significativamente os custos totais de desenvolvimento [...], consome a rentabilidade dos projetos de software, além de ampliar os riscos de insucesso dos projetos existentes[21].

A algumas páginas atrás foi informada uma lista com as principais dificuldades encontradas por empresas internacionais no que diz respeito ao desenvolvimento de software, na década de 1960. Estatísticas[22] atuais de empresas estadunidenses permitem concluir que muitas das dificuldades encontradas naquela época ainda existem:

---

[18] Ibidem, p.546.
[19] BARTIE, Alexandre. Garantia de Qualidade de Software. Rio de Janeiro: Elsevir Editora, 2012. p.4 e 5.
[20] NAIK, Kshirasagar, op.cit., p.546.
[21] BARTIÉ, op.cit., p.8.
[22] Ibidem, p.6.

- Mais de 30% dos projetos são cancelados antes de serem finalizados
- Mais de 70% dos projetos falham nas entregas das funcionalidades esperadas
- Os custos extrapolam em mais de 180% os valores originalmente previstos
- Os prazos excedem em mais de 200% os cronogramas originais

A ideia de adotar processos surge justamente para organizar e racionalizar as atividades corporativas. O objetivo principal é obter maior controle e qualidade das atividades executadas nas diferentes fases do processo utilizado. Não é à toa que "Softwares não "adequados" são sintomas da falta de controle do processo de desenvolvimento".[23] Ter processo definido significa adotar "um processo que é gerenciado – planejado, monitorado e ajustado – e adaptado de um conjunto de processos-padrão de acordo com os guias de adaptação da organização"[24].

E aqui podemos entender de maneira prática a diferença entre controle de qualidade e garantia de qualidade. De acordo com a norma Institute of Electrical and Electronics Engineering (IEEE) 610[25], controle de qualidade é o conjunto de atividades projetadas para avaliar a qualidade de desenvolvimento ou manufatura dos produtos. Atividades de controle de qualidade agem diretamente sobre o produto gerado, com o objetivo de avaliar, naquele item selecionado, se os fatores / atributos de qualidade planejados tornaram-se realidade.

Já as atividades de garantia de qualidade atuam no controle dos processos que geram os produtos, avaliando suas atividades através do artefato gerado em cada atividade / ciclo de vida. Para que a garantia de qualidade seja realidade em uma empresa, um grupo de pessoas deve ser indicado como responsável pela análise e avaliação das atividades do processo. Na norma IEEE 610[26], garantia da qualidade é a padronização sistemática e planejada de todas as ações necessárias para gerar confiança que um produto está em conformidade com os requisitos técnicos. E ainda: um conjunto de atividades projetadas para avaliar o processo onde produtos são desenvolvidos e manufaturados.

---

[23] Ibidem, p.8.
[24] **MPS.BR Guia Geral MPS de Software 2012**. – APUD ISO 2004. SOFITEX, 2012.
[25] NAIK, Kshirasagar, op.cit., p.500.
[26] Ibidem, p.499.

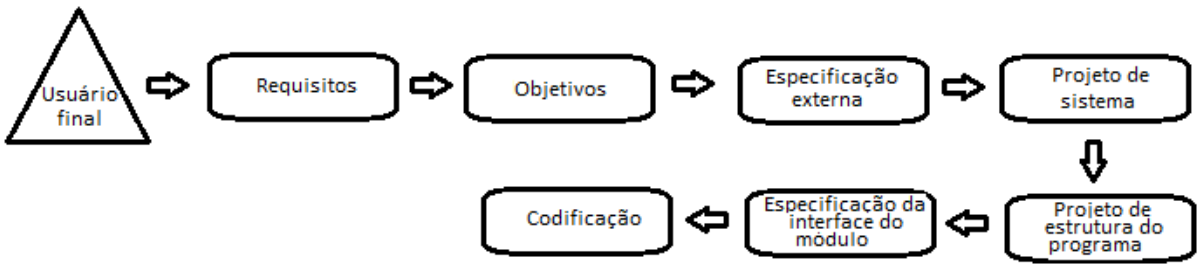
Apesar de muitos autores entenderem as atividades de teste como pertencendo ao grupo de conceitos da garantia de qualidade de software, é possível diferenciar processos de teste e suas avaliações e análises (garantia de qualidade) das atividades de teste em si (teste de conformidade, teste de segurança, teste de regressão), mais vinculadas aos fatores de qualidade e à conformidade dos requisitos de sistema.

Mesmo um processo de desenvolvimento de software que possua atividades de planejamento e alto nível de gerenciamento não está livre de erros e incontingências. Isso porque o processo de desenvolver sistemas

> "é largamente um processo de comunicar informações sobre o eventual programa e a tradução desta informação de uma forma para outra. Por esta razão, a maioria de defeitos de software pode ser atribuída a falhas, erros e ruídos durante a comunicação e tradução das informações."[27]

As palavras de Myers são simples, porém profundamente esclarecedoras. A grande questão envolvendo os problemas encontrados em software, fruto dos processos de desenvolvimento, tem forte ligação com a comunicação entre as diferentes atividades executadas. Para entender como o ruído de comunicação pode afetar profundamente a qualidade de um sistema, basta analisarmos um processo de desenvolvimento genérico[28], conforme a figura 1:

Figura 1 – Um processo de desenvolvimento de sistemas genérico



Fonte: MYERS, Glenford J. The Art of Software Testing. EUA: Wiley, 2004. p.123.

Passo 01 "Usuário final => Requisitos": As necessidades de usuário do programa são traduzidas em um conjunto de requisitos escritos. Estas serão as metas do produto.

[27] MYERS, Glenford J. The Art of Software Testing. EUA: Wiley, 2004. p.123.
[28] Ibidem, p.123.

Passo 02 “Requisitos => Objetivos”: Os requisitos são traduzidos em objetivos específicos através da avaliação de viabilidade e custo, resolvendo os requisitos conflitantes e estabelecendo prioridades e trocas.

Passo 03 “Objetivos => Especificação externa”: Os objetivos são traduzidos em especificações de produto, enxergando o produto como uma caixa preta e considerando apenas as interfaces e as interações com o usuário final. Esta é a chamada especificação externa.

Passo 04 “Especificação externa => Projeto de sistema”: Se o produto for um sistema como um sistema operacional, sistema de controle aéreo, gerenciador de banco de dados ou sistema pessoal do empregado, e não um programa (compilador, programa de pagamentos, processador de texto), o próximo passo é o projeto de sistema. Este passo divide o sistema em programas individuais, componentes, ou subsistemas, e define suas interfaces.

Passo 05 “Projeto de sistema => Projeto de estrutura do programa”: A estrutura do programa ou programas é projetada através da especificação da função de cada módulo, a estrutura hierárquica dos módulos, e da interface entre os módulos.

Passo 06 “Projeto de estrutura do programa => Especificação da interface do módulo”: Uma especificação precisa é desenvolvida para que defina a interface e função de cada módulo.

Passo 07 “Especificação da interface do módulo => Codificação”: Através de um ou mais subpassos, a especificação da interface do módulo é traduzida para o algoritmo e código fonte de cada módulo.

É perceptível como comunicação, interpretação e tradução de ideias são primordiais para a melhor qualidade de cada entrega, num processo de desenvolvimento. E é justamente a qualidade das entregas que se torna possível mensurar, conforme explicado por McCall, uma vez que se tenha em mente “o que” se pretende melhorar, e “quanto” se pretende melhorar.

Uma das maneiras de se mensurar o resultado de uma entrega é a adoção de passos separados que verifiquem o produto que está sendo gerando, no que diz respeito a artefatos, entregáveis, etc. e encontrem o maior número possível de defeitos. Seriam passos externos às atividades do processo do desenvolvimento, atividades que testassem de uma maneira ou de outra, o que já foi produzido e o que há de ser produzido.

Nesta perspectiva, utilizar testes, e mais ainda, processos específicos de teste, permite análise prática do que está sendo desenvolvido. Permite, ainda, comparar o que se produz com os fatores de qualidade que se pretende alcançar, e agir na melhoria dos processos de desenvolvimento e qualidade. Nas próximas páginas, serão feitas análises do conceito de testes e de como um processo de testes pode trazer qualidade a cada etapa do processo de desenvolvimento, do ponto de vista da garantia de qualidade.

3. TESTES

Nesta tópico, serão abordados assuntos que relacionam os conceitos de qualidade de software, apresentados até aqui, com atividades de testes de software. O objetivo é analisar como o assunto testes de software pode ser discutido e aplicado do ponto de vista da qualidade, dando exemplo de técnicas de teste e processos de teste.

3.1 Testes e Qualidade

> Os testes possuem um papel importante em atingir e avaliar a qualidade de um produto de software. Por um lado, podemos aprimorar a qualidade dos produtos enquanto repetimos o ciclo testar => achar defeitos => consertar, durante o desenvolvimento. Por outro lado, avaliamos o quão bom é o nosso sistema quando executamos testes de sistema antes de liberar um produto.[29]

Como garantir que todo o software produzido executa exatamente o que foi projetado para executar e, o mais importante, não executa o que não deve executar? [30] As palavras contidas no clássico livro de Glenford Myers ainda são profundamente atuais, e mesmo passados mais de trinta anos desde a sua primeira publicação, empresas de desenvolvimento de software enfrentam grandes dificuldades no que diz respeito a testes de software.

Parte disso se dá pela maneira como se aborda todo o conceito de testes, dentro das empresas. É comum encontrarmos as seguintes noções de testes de software: “Teste é o

[29] NAIK, op.cit., p.7.
[30] MYERS, op.cit., p.XI.

processo de demonstrar que os erros não estão presentes"; ou ainda: "O propósito dos testes é mostrar que um programa executa corretamente suas funções pretendidas"[31].

Que fique claro: o ato de testar um programa gera, de uma maneira ou de outra, mais valor a este programa. Gerar valor através do teste significa, em última instância, aumentar a qualidade, confiança **do** e confiança **no** programa. Logo, para gerar mais valor ao programa, deve-se encontrar e corrigir mais erros: é uma inferência simples. Partindo para uma segunda conclusão, e já criando uma quebra nos paradigmas do conceito de testes: "Testar é um processo de executar um programa com a intenção de encontrar erros"[32].

Para Koscianski:

> O objetivo do teste é encontrar defeitos, revelando que o funcionamento do software em uma determinada situação não está de acordo com o esperado. Um teste bem sucedido identifica defeitos que ainda não foram descobertos e que podem ser, então, corrigidos pelo programador. Quando a atividade de testes é planejada de maneira sistemática e rigorosa, pode ser utilizada como um dos parâmetros para estimar a confiabilidade e a qualidade do software construído[33].

Apesar de se entender que a função dos testes é, primariamente, a de buscar falhas, Cristiano Caetano [34] afirma que os testes podem se tornar atividades estruturadas e sistemáticas, baseadas em técnicas, ferramentas e processos formais.

Porém, as atividades de teste são comumente adotadas como etapas dentro do desenvolvimento de software, quando não são apenas executados pelos próprios desenvolvedores e pelos usuários do sistema. Nesse sentido, o senso comum, encontrado nas diferentes empresas de sistemas de software, é o de utilizar os testes manuais para "garantir" que especificações ou requisitos de negócio foram implementados[35].

Retomando a análise do processo genérico de desenvolvimento, no fim do capítulo anterior, podemos concluir que o processo apresentado pode ser dividido em dois momentos

---

[31] Ibidem, p.6.
[32] Ibidem, p.6.
[33] KOSCIANSKI, op.cit.,p.337.
[34] CAETANO, Cristiano. **Gestão de testes:** Ferramentas Open Source e melhores práticas na gestão de testes. Engenharia de Software Magazine. Retirado de http://www.devmedia.com.br/. p.58.
[35] BASTOS, Anderson. Base de conhecimento em testes de software. São Paulo: Martins Fontes, 2007. p.17.

distintos, mas que se completam: o primeiro é **a fase de coleta de informações do negócio e o planejamento da arquitetura do software**. Como mencionado, é notável o esforço feito para traduzir as mesmas informações para as mais diferentes etapas: neste momento, componentes tecnológicos não são implementados; existem "apenas" criações intelectuais que vão gerar, em última instância, uma maquete do projeto de software. Destaca-se, nestas etapas, uma forma de testes não muito aplicada nas empresas de TI (Tecnologia da Informação): testes de verificação.

A segunda parte do processo caracteriza-se pela **existência ou criação de um ou vários componentes computacionais**, onde os testes podem ser diretamente executados para avaliar se o que foi produzido condiz com os requisitos levantados e especificados na primeira fase.[36] É na segunda fase que uma forma muito comum de testes de software é aplicada: testes de validação.

É possível entender um descompasso envolvendo o momento da aplicabilidade dos testes no processo de desenvolvimento:

> "apesar de ser muito positivo algumas empresas construírem ambientes para a realização dos testes de validação, estas não estão dirigindo seus esforços para as fases iniciais do processo de software, portanto os custos de correções dessas organizações e o numero de incidência de erros nos softwares desenvolvidos serão muito altos nas fases de teste de validação."[37]

O ideal é que as atividades de testes sejam empregadas em ambos os momentos do processo de desenvolvimento. Testes de verificação e testes de validação são atividades complementares entre si, que de modo algum se anulam. O conceito destes dois tipos de testes está intimamente conectado com duas atividades intrínsecas aos processos de qualidade: **verificação** e **validação**.

3.2 Verificação e Validação

As atividades de validação e verificação servem para assegurar que o software funcione de acordo com o que foi especificado e atenda aos requisitos especificados[38]. Claro que o objetivo é sempre construir corretamente na primeira tentativa, mas a natureza

[36] BARTIÉ, Op.Cit.p.35 et seq.
[37] Ibidem, p.37.
[38] KOSCIANSKI, op.cit., p.332.

extremamente intelectual, marca registrada do desenvolvimento de um sistema de software, afasta esta possibilidade.

Utilizar o método escrever – testar – modificar até o momento de se obter o resultado desejado, é extremamente moroso e possui riscos de insucessos muito grandes. Portanto, utilizar maneiras de certificar que as próximas fases do processo possam ser executadas se traduz nas atividades de verificação e validação. Em outras palavras: "um bom processo de qualidade de software deverá potencializar essas duas formas de testes, de modo que os esforços sejam minimizados e os resultados sejam os mais positivos possíveis".[39]

Atividades de verificação ajudam na avaliação de um sistema de software, determinando se o produto de uma determinada fase de desenvolvimento satisfaz as exigências estabelecidas antes do início desta fase. Pode-se notar que um produto pode ser um produto intermediário, como especificação de requisitos, especificação de projetos, codificação, manual, ou até mesmo o produto final, entregável: são atividades que verificam a procedência de uma fase de desenvolvimento[40]. Classicamente, a resposta à seguinte pergunta define se as atividades executadas são de verificação: "O sistema foi construído corretamente"?

Seguem algumas atividades de verificação, na construção de um sistema de software[41]:

- Revisões de requisitos
- Revisões de modelos
- Inspeções de código
- Revisões e inspeções técnicas em geral

Atividades de validação ajudam na confirmação de que um produto cumpre as metas de sua utilização. Atividades de validação visam confirmar que um produto atende às expectativas de seus clientes. Em outras palavras, as atividades de validação concentram-se no produto final, que é extensivamente testado do ponto de vista do cliente. A validação estabelece se o produto atende às expectativas gerais dos usuários[42]. Classicamente, a resposta à seguinte pergunta define se as atividades executadas são de validação: "O sistema correto foi construído"?

---

[39] BARTIÉ, op. cit., p.37.
[40] NAIK, op.cit., p.8.
[41] BASTOS, op.cit., p.30.
[42] NAIK, op.cit., p.8.

Seguem algumas atividades de validação[43]:

- Teste unitário: nesta fase são testadas as menores unidades do software desenvolvidas. Ex.: métodos de uma classe.
- Teste de integração: nesta fase é testada a integração entre os componentes do sistema. Ex.: classes, módulos, subsistemas, etc.
- Teste de sistema: Nesta fase o sistema é testado como um todo com o objetivo de encontrar discordâncias entre o que foi implementado e o comportamento descrito nos requisitos.
- Teste de aceitação: Nesta fase o sistema é testado como um todo com o objetivo de encontrar discordâncias entre o que foi implementado e o comportamento descrito nos requisitos, sob o ponto de vista das necessidades do usuário final.

3.3 Atividades do Processo de Teste

> "O ciclo de vida de testes pressupõe que sejam realizados testes ao longo de todo o processo de desenvolvimento. Em determinados pontos, os produtos intermediários do ciclo de desenvolvimento devem ser revisados com objetivo de criar as condições necessárias para uma correta implementação, procurando-se identificar defeitos o mais cedo possível"[44].

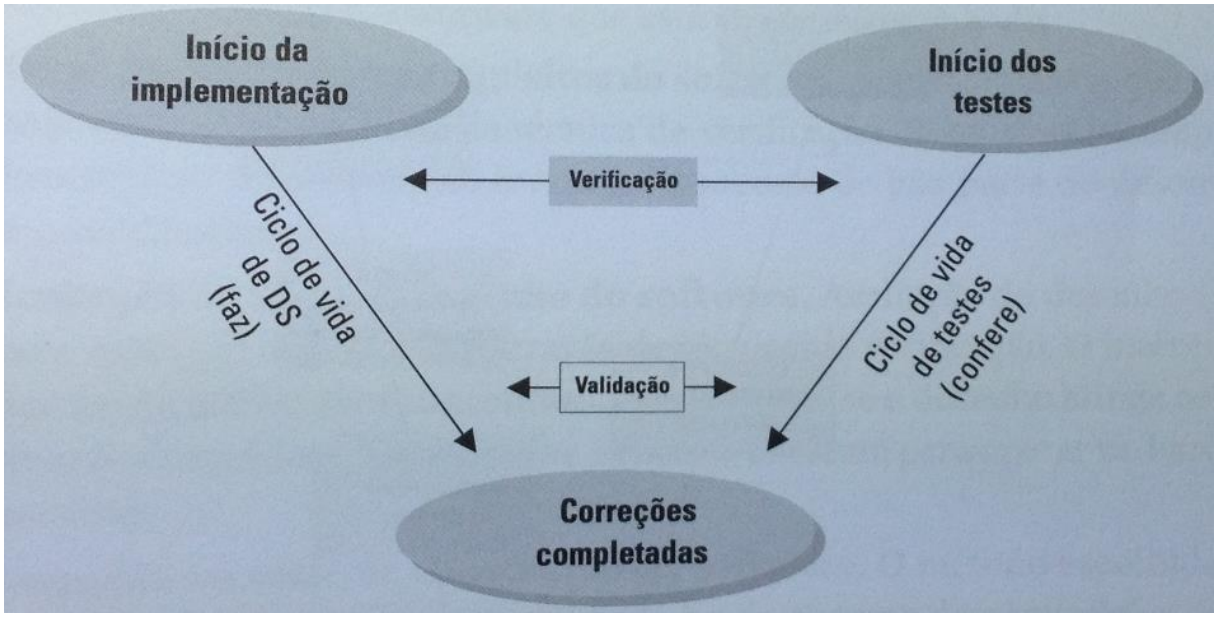
A ideia de adotar um processo de testes significa, de maneira interdependente com o processo de desenvolvimento, avaliar as mais diferentes atividades e produtos do processo de desenvolvimento de software, desde seu momento inicial até as atividades de implementação e, porque não, depois dela. Para Bastos[45], uma forte dependência do ciclo de vida de testes, com relação ao desenvolvimento, é que os testes necessitam da conclusão de atividades de certos produtos do desenvolvimento. Por outro lado, pode-se entender que existe uma interdependência entre os ciclos, já que a aplicação (testes e desenvolvimento), com a consequente geração de produtos, é um pré-requisito para viabilizar os processos de desenvolvimento de software.

Testes de verificação e testes de validação ocupam quase a totalidade das atividades existentes dentro do ciclo de vida dos testes, elas convergem-se até o final do projeto. A figura 02 ilustra um processo de teste e desenvolvimento caminhando lado a lado, porém com atividades complementares bem distintas: o chamado processo em "V".

---

[43] CAETANO, op.cit., p.59.
[44] BASTOS, op.cit., p.40.
[45] Ibidem, p.40.

Figura 2 – Processo de testes, desenvolvimento e atividades de verificação e validação.



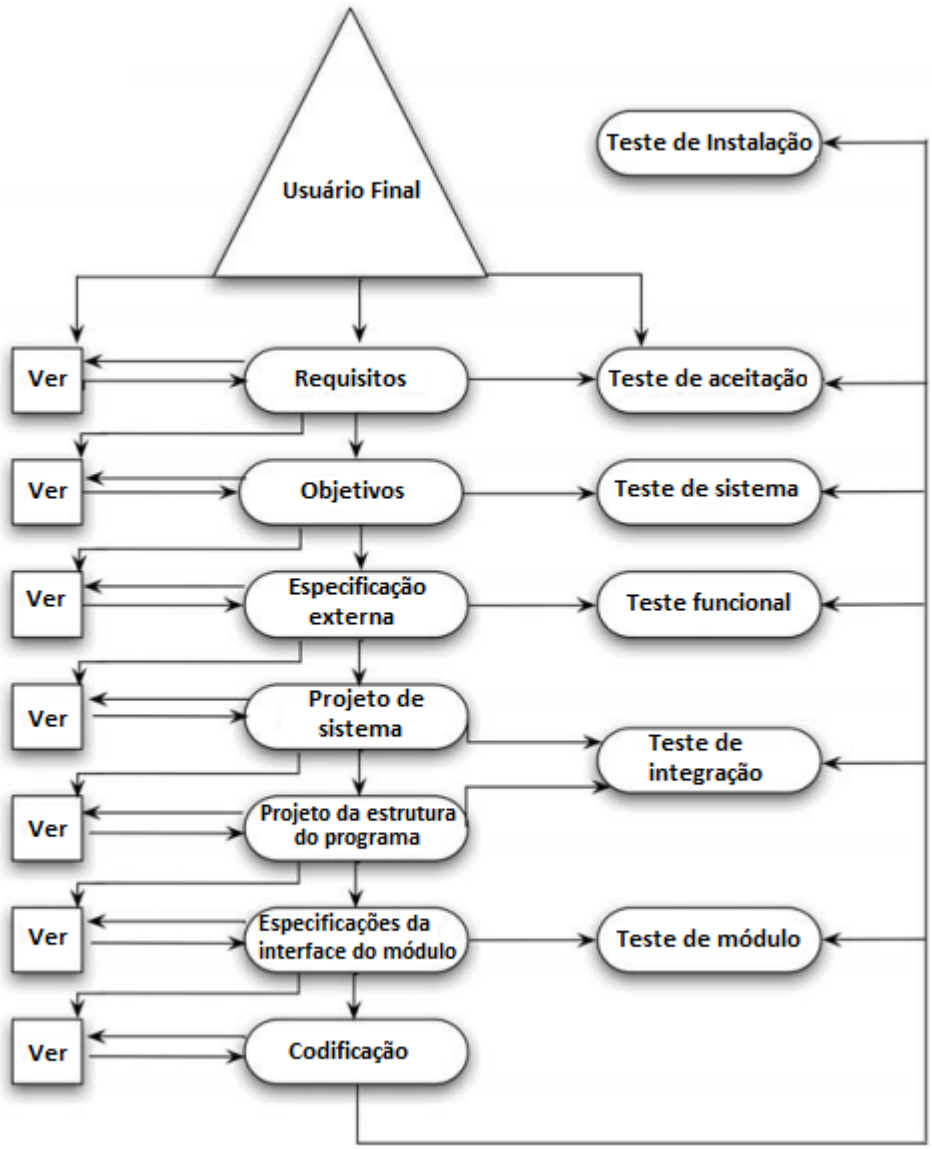
Fonte: BASTOS, Anderson. Base de conhecimento em testes de software. São Paulo: Martins Fontes, 2007, p.41.

Utilizar processos com início simultâneos pressupõe maturidade de processos. Isso porque o inicio do processo de testes é marcado por atividades de planejamento de testes do sistema a ser criado. De fato, os requisitos e documentações capturados pela equipe de desenvolvimento serão utilizados pela equipe de teste para a especificação e testes do sistema[46].

A figura 3[47] exibe em detalhes as atividades do processo de desenvolvimento e dos testes, com suas interdependências:

[46] Ibidem, p.40.
[47] MYERS, op.cit., p.127.

Figura 3: processo de desenvolvimento no centro e testes à esquerda e à direita



Fonte: MYERS, Glenford J. The Art of Software Testing. EUA: Wiley, 2004. p.127.

A figura 3 exibe alguns dos níveis de testes: estes variam igualmente com o nível do sistema que está sendo analisado. Do baixo nível para o mais alto nível temos um grande leque de possibilidades de teste: teste de unidade, teste de integração, teste de sistema e teste de aceitação são só alguns exemplos, além das atividades de verificação.

Assim, é comum a separação das estratégias de testes entre duas técnicas de teste: funcionais e estruturais. Partindo da ideia de Myers, testes funcionais pretendem mostrar que o sistema **não atende** aos requisitos, ou seja, que existem incongruências entre os requisitos e

o que foi codificado[48]: são os "Testes de Caixa Preta" que procuram avaliar aderência / conformidade com o que foi especificado nos requisitos.

Já os testes estruturais possuem na coesão e solidez seus principais objetivos: garantem que o programa funcione no contexto técnico onde serão instalados[49]: técnicas também conhecidas como "Testes de Caixa Branca", onde certos critérios são utilizados para especificar casos de teste que identifiquem defeitos nas estruturas internas do software[50].

Para cada técnica de teste podemos aplicar uma infinidade de tipos de teste. Criar uma estratégia de testes significa aliar os níveis do sistema a serem testados com os tipos de testes mais apropriados e a qualidade almejada. A busca por melhores resultados, no que tange a qualidade de um sistema, pode ser mais bem sucedida quando se planeja quais características, como os "fatores de McCall", por exemplo, da qualidade que se pretende alcançar no sistema. Assim, "os tipos de teste normalmente são definidos em função das características ou da dimensão da qualidade"[51]. A Tabela 2 exibe alguns tipos de teste aplicáveis a um projeto de testes de software, sua definição e os fatores de qualidade vinculados:

Tabela 2: Alguns tipos de teste e suas definições.

Tipos de teste	Definição
Teste de Estresse	Avalia o desempenho do sistema com um volume de acesso/transações acima da média esperada e em condições extremas de uso. Fator/Característica de qualidade: **confiabilidade, eficiência**.
Teste de Execução	Avalia se o sistema atende os requisitos de performance com um volume de acesso/transações dentro do esperado. Fator/Característica de qualidade: **eficiência**, **integridade**.
Teste de Contingência	Avalia se o sistema retorna a um status operacional após uma falha. Fator/Característica de qualidade: **confiabilidade, eficiência**, **integridade**.
Teste de Operação	Avalia se o sistema (aplicação, pessoal, procedimentos e manuais) pode ser executado corretamente em ambiente de pré-produção. Fator/Característica de qualidade: **usabilidade**
Teste de Conformidade	Avalia se o sistema foi desenvolvido em consonância com os padrões e metodologia estabelecidos no projeto. Fator/Característica de qualidade: **funcionalidade**
Teste de Segurança	Avalia se o sistema foi desenvolvido em consonância com os padrões de segurança da organização. Fator/Característica de qualidade: **confiabilidade**.

[48] Ibidem, p.128.
[49] BASTOS, op.cit., p. 48.
[50] CAETANO, op.cit, p.60.
[51] Ibidem, p.59.

Teste de Regressão	Avalia por meio do re-teste se uma funcionalidade que estava funcionando ainda funciona após uma modificação no sistema. Fator/Característica de qualidade: **testabilidade**.
Teste de Integração	Avalia se a interconexão entre as aplicações funciona corretamente. Fator/Característica de qualidade: **confiabilidade**.

Fonte: CAETANO, Cristiano. **Gestão de testes:** Ferramentas Open Source e melhores práticas na gestão de testes. Engenharia de Software Magazine. Retirado de http://www.devmedia.com.br/. p.60.

Além das técnicas, níveis e tipos de teste, um ciclo de vida de testes é amparado por documentos que fornecem, além de outras coisas, informações importantíssimas acerca do próprio processo de testes. Seguem alguns documentos[52] comuns ao processo de testes:

- Plano de testes: define o planejamento para execução do teste, incluindo a abrangência, abordagem, recursos e cronograma das atividades de teste. Identifica os itens e funcionalidades a serem testados, as características dos itens a serem testados, as tarefas a serem realizadas e os riscos associados com a atividade de teste.
- Especificação do projeto de teste: refina a abordagem apresentada no Plano de Teste e identifica as funcionalidades e características a serem testadas pelo projeto e pelos seus testes associados. Também identifica os casos e os procedimentos de teste, se existirem, e apresenta os critérios de aprovação para esses elementos.
- Especificação dos Casos de Teste: define os casos de teste, incluindo as pré-condições, passos, resultados esperados e condições gerais para a execução do teste.
- Especificação dos Procedimentos de Teste: define a sequência de ações necessárias para executar um conjunto de casos de teste.
- Diário de Testes: documenta os registros cronológicos dos fatos relevantes durante a execução dos testes.
- Relatório de Incidente de Testes: Documenta qualquer evento que ocorra durante a atividade de teste e que necessite de alguma análise posterior.
- Relatório Resumo de Testes: Documenta de forma resumida os resultados das atividades de teste associadas com uma ou mais especificações de projeto de teste e fornece avaliações baseadas nesses resultados.

Contudo, as atividades de testes podem, e devem, ser gerenciadas. O gerenciamento das atividades de teste significa identificar, acompanhar e medir o quanto de cada etapa está em execução / já foi executada, além de gerar outras conclusões sobre o processo de testes. Isso porque, a partir do momento em que os testes passam a ser executados por especialistas,

[52] Ibidem, p.61.

as empresas adotam processos de teste com o objetivo de melhorar a qualidade dos sistemas de software e, consequentemente, a qualidade do próprio processo de testes entra na discussão.

Mesmo que o objetivo da adoção de processos de teste seja, num último momento, a diminuição dos custos, dado que erros encontrados no ambiente de produção custam dez vezes mais para serem consertados, inicialmente, o investimento ainda é muito alto. Portanto, "não adianta apenas testar, deve-se testar bem"[53].

Além do controle das atividades, gerenciar processos de teste ajuda a responder perguntas chave sobre o processo em questão; dentre elas: "O quanto podemos melhorar do processo de testes"?

3.4 Norma ISO 15504 e os Níveis de Capacidade de Processos

> "Para o desenvolvimento de software com qualidade, dentro de prazos e custos controlados e compatíveis com o mercado, é fundamental a melhoria dos processos da engenharia de software"[54].

Aplicar atividades de testes no processo de desenvolvimento, e evoluir estas atividades para processos planejados, estabelecidos e melhorados, é um investimento que requer maturidade e planejamento, por parte da empresa e da equipe de qualidade.

Historicamente, a implantação de áreas de testes nas empresas de sistema de software, na década de 1990, levou alguns especialistas em processos a se preocupar com modelos de processos que permitissem sua melhoria[55]. De toda esta necessidade, foram surgindo, ao longo dos anos, inúmeros modelos de maturidade de teste de software, que se denominam *frameworks* para processos de testes. Seguem alguns exemplos: Testability Support Model (TSM), Testing Maturity Model (TMM), Test Process Improvement (TPI), Testing Assessment Program (TAP) e Testing Improvement Model (TIM).

No Brasil, a criação de um programa que estruture e de padronização a melhoria dos processos de software foi iniciativa da SOFTEX, com apoio do Ministério da Ciência,

---

[53] RIOS, Emerson: Usando o MPS.BR para amadurecimento das equipes de teste de software. Retirado de http://www.iteste.com.br.
[54] CRESPO, Adalberto Nobiato. **Uma Metodologia para Testes de Software no Contexto da Melhoria de Processo.** Campinas: UNICAMP, p.3.
[55] RIOS, op.cit., p.2.

Tecnologia e Inovação (MCTI) e outras instituições, com a Melhoria do Processo de Software Brasileiro (MPS.BR).[56]

A MPS.BR surge em 2003 para suprir a necessidade das pequenas e médias empresas de software brasileiras, que é a mudança paulatina da visão tradicional baseada em áreas funcionais para as redes de processos centrados nos clientes[57].

O conjunto de processos, documentos e níveis de maturidade existentes no MPS.BR foram baseados nas normas internacionais ISO-12207:2008, ISO-15504:2003 e o modelo Capability Maturity Model – Integration (CMMI)[58]. É válido lembrar que estas normas identificam processos fundamentais para a engenharia de software, e que todas elas identificam, direta ou indiretamente, teste de software como um destes processos[59]. Em especial, a norma ISO-15504 apresenta, em seu corpo documental, ferramentas suficientes para servir de base para a implantação, análise e melhoria dos processos de software.

A norma ISO-15504 é arquitetada em duas partes – **dimensão da capacidade de processo** e **dimensão de processo**. Na dimensão da capacidade de processo, são definidos seis níveis de capacidade, sequenciais e cumulativos, utilizados para avaliação de processos. Cada nível apresenta nome, definição e atributos. A tabela 3 identifica os níveis de capacidade e seus atributos.

Tabela 3: Os níveis de capacidade e atributos de processo da ISO/IEC 15504

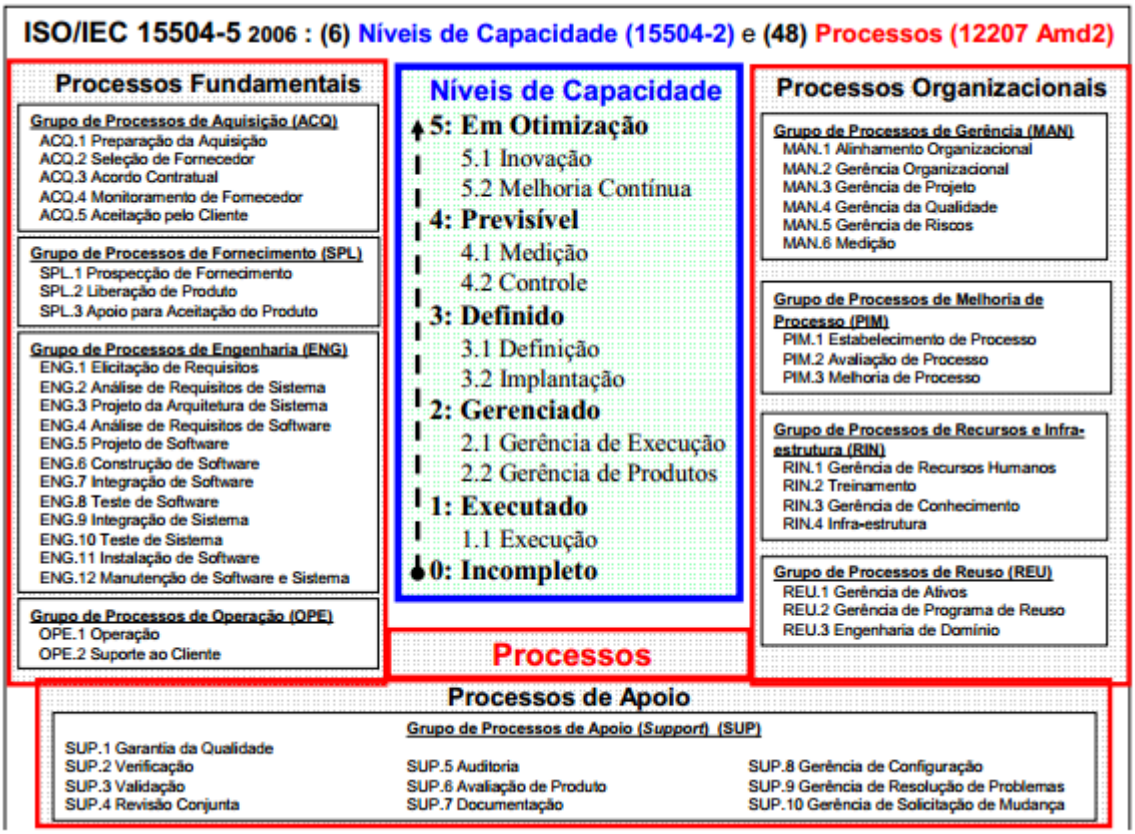
Nível de capacidade	Atributos do processo (PA)
Nível 0 – Incompleto	Não existem atributos
Nível 1 – Executado	PA 1.1 – Atributo de execução do processo
Nível 2 – Gerenciado	PA 2.1 – Atributo de Gerência de Configuração PA 2.2 – Atributo de Gerência de Produto de Trabalho
Nível 3 – Estabelecido	PA 3.1 – Atributo de Definição de Processo PA 3.2 – Atributo de Implementação de Processo
Nível 4 – Previsível	PA 4.1 – Atributo de Medição de Processo PA 4.2 – Atributo de Controle de Processo
Nível 5 – Em otimização	PA 5.1 – Atributo de Inovação de Processo PA 5.2 – Atributo de Otimização de Processo

Fonte: SALVIANO, Clênio Figueredo. **Melhoria e Avaliação de Processo de Software com o Modelo ISO/IEC 15504-5:2006**. Lavras: Centro de Editoração Quality Group. p.64.

[56] SOFITEX, op.cit., p.4.
[57] Idem, p.6.
[58] Idem, p.6.
[59] CRESPO, op.cit., p.3.

Na dimensão de processos, a norma apresenta um total de 48 processos universais e fundamentais para a Engenharia de Software. Os processos são organizados em três níveis: processos fundamentais, processos organizacionais e processos de apoio. A figura 4 ilustra os processos especificados pela norma: nota-se que o Grupo de Processos de Engenharia – **ENG.8 Teste de Software** apresenta um conjunto de boas práticas e um processo de teste de software.

Figura 4: Processos fornecidos pela norma ISO 15504 organizados em níveis



Fonte: CRESPO, Adalberto Nobiato. **Uma Metodologia para Testes de Software no Contexto da Melhoria de Processo.** Campinas: UNICAMP, p.83.

É possível aliar a análise de um processo de testes utilizando o conjunto de níveis de capacidade da norma ISO-15504, e prospectar sua melhoria aplicando atividades e artefatos que mensurem o processo. Esta será a discussão apresentada no capítulo “Resultados e Discussões”.

4. PROCEDIMENTOS METODOLÓGICOS

Pesquisa científica é um processo reflexivo sistemático, controlado e crítico, que permite descobrir novos fatos ou dados, relações ou leis, em qualquer campo do conhecimento[60].

Para explorar a temática "qualidade de sistemas de software", e mais especificamente, "testes de software", foi utilizada bibliografia base que pudesse contextualizar e definir conceitos importantes. Desta maneira, destaca-se o procedimento da pesquisa a partir da metodologia da pesquisa bibliográfica. Assim, de acordo com Maria de Carvalho[61]:

> "A pesquisa bibliográfica é a realizada por meio da identificação, localização e compilação dos dados escritos em livros, artigos de revistas especializadas, publicações de órgão oficiais, bases de dados, etc., sendo necessária a qualquer trabalho de pesquisa científica."

Contudo, ao analisar um ciclo de vida de testes e efetuar comparações com normas de qualidade, notou-se a necessidade de conhecer exemplos e conceitos de como este processo foi executado, dos fatores que contribuem para a melhoria do processo e conceitos relacionados às práticas. Considerar a melhoria do processo de testes como um fenômeno ocorrido a partir de ações humanas, entender quais ações são necessárias para seu acontecimento e explicar tais ações, são objetivos do trabalho em questão. Estas características são, ao mesmo tempo, características de pesquisas científica e explicativas, onde:

> "As pesquisas explicativas visam identificar os fatores que determinam ou contribuem para a ocorrência dos fenômenos. Por explicar a razão e o porquê das coisas, esse é o tipo de pesquisa que mais aprofunda o conhecimento da realidade"[62].

Ao mesmo tempo, existe um caráter descritivo na pesquisa, pois processos de teste e qualidade são descritos, além das suas atividades serem especificadas. Uma vez compreendida a racionalidade na exploração do assunto, foi possível efetuar uma simulação de como um

---

[60] LAKATOS, Eva Maria. **Fundamentos da metodologia científica.** São Paulo: Atlas, 2001. p.155.
[61] CARVALHO, Cecília M. **Construindo o saber.** Metodologia científica, fundamentos e técnicas. Campinas: Ed. Papirus, 2010. p.193.
[62] RAUPP, Maury Fabiano. **Como elaborar trabalhos monográficos em contabilidade.** São Paulo: Atlas, 2006. p.82.

processo de testes se comporta, à medida que o nível de maturidade da norma vai aumentando. A descrição destas ocorrências nos permite graduar a atual pesquisa, igualmente, como descritiva, pois:

> "a pesquisa descritiva configura-se como um estudo intermediário entre a pesquisa exploratória e a explicativa, ou seja, não é tão preliminar como a primeira nem tão aprofundada como a segunda. Nesse contexto, descrever significa identificar, relatar, comparar, entre outros aspectos"[63].

Sendo esta uma pesquisa bibliográfica, foi feito um levantamento de fontes secundárias que discutem o assunto abordado – qualidade, testes de software e processos de teste – para, num segundo momento, propor maneiras de evoluir um processo de testes de acordo com normas internacionais de qualidade.

5. RESULTADOS E DISCUSSÃO

Especificar um processo de testes é uma tarefa que depende das necessidades e situação / maturidade de cada equipe. Não cabe aqui definir todas as tarefas de testes possíveis, entendendo que as "atividades de testes requerem conhecimento, planejamento, projeto, execução, acompanhamento, recurso e também, e principalmente, alto nível de interação entre as equipes."[64]

Um processo de testes deve ser criado com base em um planejamento, onde algumas das estratégias de teste são definidas:

Técnica de teste: existem basicamente duas técnicas de testes: testes estruturais, que vão direcionar a criação dos casos de teste para identificação de defeitos internos do software, no nível do código, e funcionais, que vão direcionar a criação de casos de testes que exercitem os requisitos especificados. A decisão sobre quais técnicas utilizar vai depender da necessidade da nova implementação/novo projeto, isso porque a técnica vai influenciar toda a gama de testes especificados.

Nível de teste: atividades cuja execução dependerá da fase do desenvolvimento em que será aplicada. Ex.: testes de unidade (teste do módulo implementado, no nível do código), testes de integração (teste da integração entre os módulos), teste de sistema (testes para

[63] Ibidem, p.81.
[64] CRESPO, op.cit., p.3.

verificar atendimento aos requisitos funcionais e não funcionais), testes de aceitação (aceitação do sistema pelo usuário) e testes de regressão (testes da parte legada do sistema). Novamente a escolha o nível do teste, dentro de um processo, depende da empresa e das necessidades do que se está entregando.

Critério de teste: o critério vai definir os elementos do software que serão testados, com base na técnica escolhida: linhas de código, funções implementadas, variáveis definidas, requisitos de software, etc. É o critério que orienta o analista de testes, na hora de projetar e especificar os casos de teste.

Tipo de teste: Os tipos de testes definem quais características do software que serão testadas (teste de funcionalidade, teste de interface, teste de desempenho, teste de carga, teste de usabilidade, teste de segurança). Aqui é feita a relação entre o sistema e os fatores de qualidade esperados de cada teste, como, por exemplo, os fatores definidos por McCall.

As atividades de especificação e execução de testes compõem parte importante de um processo de testes, mas não são únicas. Planejamento, automatização de testes, preparação de ambiente, gerência dos testes, além dos testes estáticos, são algumas destas atividades.

Quanto mais planejadas estas atividades forem, mais maturidade possui o processo de testes. Para avaliar o nível de maturidade do ciclo de vida dos testes, é possível efetuar análise a partir da norma ISO-15504, através dos artefatos gerados pelos testes.

Para cada atividade de teste existem modelos de documento que podem ser utilizados como base para sua especificação/execução. Além desta importância fundamental dos documentos, os mesmos podem ser utilizados para tarefas de gerenciamento do processo de testes.

A norma IEE 829-1998, por exemplo, define alguns modelos de documentos a serem utilizados num processo de testes. Os seguintes modelos cobrem tarefas de planejamento, especificação e relato de testes: plano de testes, especificações (especificação do projeto de teste, especificação de casos de teste, especificação de procedimentos de teste), e relatórios (diário de testes, relatório de incidentes de teste, relatório-resumo de teste e relatório de encaminhamento de item de teste).

O propósito do processo de testes especificado pela norma ISO-15504, grupo de processos de engenharia de software – **ENG.8 Teste de Software**, é o de "confirmar que o produto integrado de software satisfaça seus requisitos definidos". Seguem os resultados que

devem ser observados, na execução de um processo de teste que esteja em concordância com a ISO-15504:

R1) Implementação de um software integrado demonstra conformidade com os requisitos especificados;

R2) O software integrado é verificado utilizando critérios definidos;

R3) Resultado dos testes são registrados;

R4) Estratégia de regressão é desenvolvida e aplicada para o reteste do software integrado, quando uma mudança no sistema é feita.

A seguir serão elencadas práticas de teste com base na norma ISO-15504. O intuito é demonstrar, a partir da comparação com os níveis de maturidade da ISO-15504, como um processo de testes se comportaria nos diferentes níveis de maturidade da norma em questão. Serão elencados artefatos, atividades ou exemplos que demonstrem a maturidade do nível em análise.

Nível 0 – Incompleto: Atividades de testes inexistentes ou sem nenhuma expressão no processo de desenvolvimento.

Nível 1 – Executado: Para que o "Nível 1" seja alcançado, basta que sejam demonstradas as práticas e resultados dispostos na "**ENG.8 Teste de Software**". Um exemplo aplicável seria a execução de testes utilizando casos de testes projetados com base nos requisitos especificados para uma mudança/evolução do sistema. Os resultados da execução devem ser registrados para futuras correções, além da execução de atividades de testes regressivos.

Documento: Casos de teste

Atividades:

- Especificação de casos de teste.

- Execução dos testes progressivos com base nos casos de teste.

- Execução dos testes regressivos.

Nível 2 – Gerenciado: Além das práticas acima serem demonstradas, deve existir o planejamento e acompanhamento das atividades de teste: gerência de testes. Os produtos do trabalho do teste são identificados. Ainda não existem fluxos propriamente ditos, apenas

atividades de testes executadas, apesar de que informações sobre a execução destas atividades permitem gerar algum tipo de indicador sobre os resultados da execução dos testes.

PA 2.1 - Atributo de Gerência de Configuração: As atividades de teste começam a ser planejadas, acompanhadas e registradas. Começa-se a pensar a partir das futuras novas funcionalidades, especificação de casos de teste, execução de testes em um backlog. Existe a identificação do responsável pela criação dos casos de teste. A utilização de ferramentas de gerência de testes como Testlink ou TestManager pode ajudar muito para colher os indicadores que os testes podem fornecer sobre processos.

PA 2.2 - Atributo de Gerência de Produto de Trabalho: Existem artefatos que servem de insumos para a especificação de testes e para a execução de testes. São definidos os requisitos de documentação e controle dos casos de testes, com registro e versionamento.

Documento: Plano de testes, a partir do plano de projeto, por exemplo.

Atividades: Planejamento de testes, análise do resultado dos testes executados.

Nível 3 – Processo estabelecido: Além das práticas acima demonstradas, existe a criação de processos padrões que serão adotados pela organização, e a definição de processos específicos de teste, com base no processo padrão, para os diferentes projetos. Os objetivos dos processos devem ser alcançados e demonstrados, além da identificação dos recursos humanos e infraestrutura. No fluxo do processo de especificação de casos e procedimentos de teste, por exemplo, poderiam ser definidos os papéis envolvidos (analista de teste, testador) e as atividades: especificação de testes, testes estáticos dos documentos gerados, execução dos casos de teste e geração dos relatórios das atividades de teste. O processo de especificação de teste pode conviver com outros processos de teste, como processo de criação de ambiente de testes, processo de implementação de testes automatizados, entre outros, servindo de insumo para os outros ciclos de vida. Papéis são definidos, como analista de testes, analista de criação de ambiente de testes, automatizador de testes, testador, líder de teste, além da infraestrutura: bases de teste, ambientes de testes de carga, etc.

PA 3.1 – Atributo de Definição de Processo: Definem-se processos de teste genéricos, relatando os fluxos de teste nos diferentes níveis (testes manuais, automatizados, testes de implementadores / testadores, testes estáticos, etc.). Este processo é adaptado para cada utilização dentro da organização, onde cada nova versão ou nova implementação vai customizar o uso das atividades do processo.

PA 3.2 – Atributo de Implementação de Processo: Os papéis, responsabilidades e autorizações para execução das tarefas, do ponto de vista do processo de testes padrão, são definidos e comunicados: analista de teste, testadores, analista de ambiente, automatizador de testes, líder de testes, etc.

Nível 4 – Processo previsível: Além das práticas demonstradas, as medições quantitativas do processo começam a acontecer de maneira mais enfática. A previsibilidade reside no apoio ao atendimento dos objetivos do negócio. Além disso, as capacitações que uma determinada função deve possuir são definidas e podem até serem oferecidas pela instituição.

PA4.1 – Atributo de Medição de Processo: São traçados objetivos que o processo específico de testes deve alcançar, do ponto de vista do negócio, normalmente objetivos ligados à requisitos funcionais/não funcionais e fatores de qualidade. O processo específico é medido para verificar se esses objetivos estão sendo alcançados e o escopo do processo é delimitado. Ex.: em uma nova funcionalidade, planeja-se mais atenção aos testes automatizados, e menos esforços na execução dos casos de testes manuais. Então as atividades de testes manuais serão diminuídas, e as execuções de testes automatizados devem ser planejadas com base nos requisitos de qualidade da nova funcionalidade - priorizando, por exemplo, testes de integração e de carga.

PA4.2 – Atributo de Controle de Processo: O processo de testes é medido com regularidade para que se tenha um processo cada vez mais estável. São definidos os requisitos para a frequência da medição do processo; ex.: a cada entrega de versão são feitas avaliações dos processos de testes executados – processo de especificação de testes, processos de configuração de ambiente de teste, processos de testes automatizados – para entender o quanto de variação aconteceu na execução das atividades: o quanto foi executado, o quanto não foi executado, e assim por diante. Ações serão tomadas para correção das variações identificadas.

Medição do processo pode ser feita através de documentos como Relatório de Resumo de Teste, Diário de Teste, Relatório de Incidentes de Teste e Relatórios de desempenho (documentos de gerenciamento), além do sistema de gestão de defeitos ou sistema de gerenciamento de testes.

Nível 5 – Processos em Otimização:

PA5.1 – Atributo de Inovação de Processo: Análise da execução e consequências das atividades dos processos de teste. Ex.: Quantos erros foram encontrados ao executar um grupo de casos de teste? Rastreabilidade dos erros: erro oriundo de especificação? Erro oriundo de implementação? Erro oriundo de modelo de dados? Os casos de teste executados estão fazendo diferença na quantidade de erros encontrados pelos processos de teste? Etc... A partir desta análise, se identificam pontos dos processos de teste que devem variar para que o mesmo possa evoluir.

P55.2 – Atributo de Otimização de Processo: Já com as mudanças executadas, são feitas medições no processo melhorado para gerenciamento do impacto das mudanças propostas. A eficácia do novo processo é averiguada, também, a partir dos requisitos definidos para o produto e objetivos do processo.

Por último, temos a tabela 4, que exibe, de maneira resumida, as atividades e artefatos de cada nível da norma apresentada, para um processo de testes ideal:

Tabela 4: Atividades, práticas e documentos utilizados em cada nível de maturidade – ISO 15504

Nível de capacidade	Atividades / artefatos
Nível 0 – Incompleto	-
Nível 1 – Executado	Documento: • Casos de teste. Atividades / práticas: • Especificação dos casos de teste. • Execução dos casos de teste (teste progressivo). • Execução de testes regressivos.
Nível 2 – Gerenciado	Documento: • Plano de teste. • Estratégia de testes Atividades / práticas: • Planejamento de testes. • Análise dos resultados dos testes executados.
Nível 3 – Processo estabelecido	• Adoção de processos de teste específicos, com base em processos padrão definidos pela instituição. • Definição dos papeis responsáveis pelas execuções. • Artefatos de teste servem como insumo para entrada de outros processos. Atividades / práticas: • Testes estáticos dos documentos gerados, • Execução dos casos de teste. • Geração dos relatórios das atividades de teste.
Nível 4 – Processo previsível	Atividades / práticas:

	• Para nova versão / nova implementação são traçados objetivos baseados nas necessidades do negócio. • Medição quantitativa do processo executado, com relação aos objetivos do negócio. • Definição de uma frequência para medição da variação do processo. • Medição da variação da execução do processo (comparação com outras execuções). Documentos: • Relatório-resumo de teste. • Relatório de encaminhamento de item de teste. • Utilização de sistemas de gerenciamento de teste.
Nível 5 – Processo em otimização	Se no nível 4 a avaliação era feita sobre o **quanto** o processo variou, no nível 5 se avalia **o que** deve variar. Avaliação do processo de execução de testes no seguinte sentido: • Quantos erros foram encontrados ao executar um grupo de casos de teste? • É possível rastrear a origem do erro (erro oriundo da especificação? Erro oriundo da implementação? Erro oriundo da especificação do modelo de dados?)? • Os casos de teste executados estão fazendo diferença na quantidade de erros encontrados pelos processos de teste? Documentos: • Utilização dos documentos de relatórios das atividades de testes.

6. CONCLUSÃO

Adotar processos de teste é um passo importante em qualquer projeto de sistemas de software. A possibilidade de medir a qualidade de um sistema encontra, nos processos de teste, grande fonte de informação. Os relatórios de execução de teste, em qualquer nível que seja, permitem tirar conclusões não somente sobre o próprio processo de testes, mas também sobre as debilidades existentes em outros processos adotados.

Assim, as atividades de testes carregam, junto a si, dois conceitos importantes sobre qualidade de sistemas. Ao mesmo tempo em que executar um teste exercita o nível de

implementação de um determinado requisito de sistema (fator de qualidade “correção”, por exemplo), avaliar o conjunto de atividades de testes executadas permite entender até que ponto o processo de testes utilizado gera os fatores de qualidades planejados.

Por consequência, na mesma atividade “testar sistemas” pode-se entender o conceito de controle de qualidade, na ação de testar um produto ou parte de um produto para concluir se aquilo que foi codificado / testado possui qualidade, e o conceito de garantia de qualidade, se o processo de teste, com todas as atividades, está gerando a qualidade que se espera.

Contudo, deve-se percorrer um longo caminho entre o planejamento de simples atividades de teste, e a implantação de processos com maturidade e que possam utilizar a retroalimentação para gerar a melhoria do processo. Apesar disso, apenas a boa vontade de se implementar um processo, em qualquer área de desenvolvimento de software, por si só, não basta.

Muitas vezes as normas que especificam boas práticas podem se tornar estruturantes demais, com documentos que geram mais burocracia do que ajudam na execução das atividades. Este não foi o caso do tema aqui abordado, onde a norma ISO-15504, se analisada por si só, não fornece qualquer indicação de documentos que devem ser utilizados. Devem-se planejar as atividades com uma boa dosagem de documentos que deem suporte e ainda assim permitam gerar relatórios e informações para futuras tomadas de decisão, sem que o processo esteja em risco pelo excesso de informações desnecessárias. Nesse ponto, o uso de ferramentas de gerenciamento de testes possui importância tremenda: mas esta discussão deve ser direcionada para um trabalho mais específico.

A discussão aqui apresentada, sobre a norma ISO-15504, especifica atividades genéricas sobre um processo de testes, e como a partir da análise dos seus níveis de maturidade, é possível evoluir as atividades de testes para ciclos de vida complexos.

Os níveis de maturidade especificados pela norma permitem, desde o momento em que atividades de testes são planejadas, análise dos resultados dos testes executados. Apesar disso, só se entende que um processo de testes é definitivamente implantado, quando atividades de planejamento forem aliadas a um ou vários fluxos de trabalho (fluxo de especificação e execução de testes, fluxo de preparação de ambientes de teste, fluxo de automatização de testes, fluxo de verificação de documento de testes, etc.), e quando os

objetivos do processo se entrelaçam com os objetivos do negócio da empresa, aí sim, dando ênfase à qualidade que se pretende ter do produto gerado.

A utilização da norma como base para desenhar, planejar, implementar e definitivamente melhorar um processo de testes é algo exequível. A própria estrutura de processos, apresentado pela ISO-15504, permite analisa-la do ponto de vista das atividades de teste. Contudo, empresas menores se beneficiariam com uma implantação utilizando esta metodologia. Isso porque a norma dá mais importância às atividades e processos do que determinar documentos e estratégias rígidas de execução.

Entendendo que a aplicação de processos de teste deve ser uma atividade planejada e que caminha lado a lado com o nível de especificações de requisitos/regras do sistema, caso os analistas de sistemas não possuam experiência nas técnicas de análise ou nas regras de negócio, nem uma equipe de analistas de testes experientes conseguirá implementar processos maduros. E se por um lado a demanda por profissionais de teste é alta, por outro o mercado sofre com a falta de capacitação dos profissionais. Mas esta é outra discussão que deverá ser feita em momento e local mais oportuno.

REFERÊNCIAS

BARTIE, Alexandre. **Garantia de Qualidade de Software**. Rio de Janeiro: Elsevir Editora, 2012.

BASTOS, Anderson. **Base de conhecimento em testes de software**. São Paulo: Martins Fontes, 2007.

CAETANO, Cristiano. **Gestão de testes:** Ferramentas Open Source e melhores práticas na gestão de testes. Engenharia de Software Magazine. Retirado de http://www.devmedia.com.br/

CARVALHO, Cecília M. **Construindo o saber.** Metodologia científica, fundamentos e técnicas. Campinas: Ed. Papirus, 2010.

CRESPO, Adalberto Nobiato. **Uma Metodologia para Testes de Software no Contexto da Melhoria de Processo.** Campinas: UNICAMP

CROSBY PB. **Quality is Free:** the Art of Making Quality Certain. Denver: Mentor Book, 1992.

HOLANDA, Aurélio Buarque de. **Novo Dicionário da Língua Portuguesa**. Rio de Janeiro: Editora Nova Fronteira, s/d.

KOSCIANSKI, André. **Qualidade de Software**. São Paulo: Novatec Editora, 2012.

LAKATOS, Eva Maria. **Fundamentos da metodologia científica.** São Paulo: Atlas, 2001.

MCCALL, Jim A. **Factors In Software Quality:** Concept and Definitions of Software Quality. Nova Iorque: General Electric, 1977.

MYERS, Glenford J. **The Art of Software Testing**. EUA: Wiley, 2004.

NAIK, Kshirasagar. **Software Testing and Quality Assurance:** Theory and Practice. Waterloo: Wiley, 2008.

RAUPP, Maury Fabiano. **Como elaborar trabalhos monográficos em contabilidade.** São Paulo: Atlas, 2006.

RIOS, Emerson: **Usando o MPS.BR para amadurecimento das equipes de teste de software.** Retirado de http://www.iteste.com.br.

SALVIANO, Clênio Figueredo. **Melhoria e Avaliação de Processo de Software com o Modelo ISO/IEC 15504-5:2006**. Lavras: Centro de Editoração Quality Group.